A Model for Quantifying Information Leakage

Steven Euijong Whang, Hector Garcia-Molina

Computer Science Department, Stanford University
353 Serra Mall, Stanford, CA 94305, USA
{swhang, hector}@cs.stanford.edu

Abstract. We study data privacy in the context of information leakage. As more of our sensitive data gets exposed to merchants, health care providers, employers, social sites and so on, there is a higher chance that an adversary can "connect the dots" and piece together a lot of our information. The more complete the integrated information, the more our privacy is compromised. We present a model that captures this privacy loss (information leakage) relative to a target person, on a continuous scale from 0 (no information about the target is known by the adversary) to 1 (adversary knows everything about the target). The model takes into account the confidence the adversary has for the gathered information (leakage is less if the adversary is not confident), as well as incorrect information (leakage is less if the gathered information does not match the target's). We compare our information leakage model with existing privacy models, and we propose several interesting problems that can be formulated with our model. We also propose efficient algorithms for computing information leakage and evaluate their performance and scalability.

1 Introduction

In practice we are continually giving out sensitive information: we need to give out our credit card data in order to purchase something; we need to tell our drug store what drugs we need; we need to give our employer (and many others) our social security number; our airline needs our passport number, and so on. Each bit of information we release represents a loss of privacy, and we never know who may end up getting our information. For instance, our store may share our information with some advertiser; or our airline may give our passport number to some governments.

The separate information losses can become much more serious if some adversary is able to gather and piece together our information. Our goal is to quantify how leakage can increase (or decrease) as information is pieced together. We do not wish to view privacy as all-or-nothing; rather, we wish to view it as a continuous measure that can represent the severity of our information loss. And once we can quantify leakage, we can study strategies for reducing leakage (or increasing it if one wants to take the point of view of a law-enforcement "adversary" trying to learn about possible criminals).

As a motivating example, suppose that Alice has the following information: her name is Alice, her address is 123 Main, her phone number is 555, her credit card number is 999, her social security number is 000. We represent Alice's information as the record: $p = \{\langle \text{N, Alice}\rangle, \langle \text{A, 123 Main}\rangle, \langle \text{P, 555}\rangle, \langle \text{C, 999}\rangle, \langle \text{S, 000}\rangle\}$. Suppose now that Alice buys something on the Web and gives the vendor a subset of her information, say $r = \{\langle \text{N, Alice}\rangle, \langle \text{A, 123 Main}\rangle, \langle \text{C, 999}\rangle\}$. By doing so, Alice has already partially compromised her privacy.

We can quantify the information leakage by measuring how correct and complete the information in r is against p. In our example, 3 out of 3 of r's attributes were correct while 3 out of 5 of p's attributes were found in r. Uncertain and incorrect information are also important factors in measuring information leakage. If an adversary Eve is not sure about Alice's information, then although the information itself is correct, the leakage should be considered less than when Eve is absolutely confident about the data. Moreover, if Eve is absolutely sure about some incorrect information about Alice, then the information leakage should decrease in proportion to Eve's confidence. Returning to our example above, suppose that Alice also gives the same vendor the following information (through another purchase): $\{\langle \text{N, Alice}\rangle, \langle \text{A, 777 Main}\rangle, \langle \text{C, 999}\rangle, \langle \text{X, 111}\rangle\}$. As a result, the vendor may only be half certain about Alice's address. In addition, if $\langle \text{X, 111}\rangle$ contains false information, the vendor now has an incorrect attribute X. Both errors should be factored in when computing the leakage.

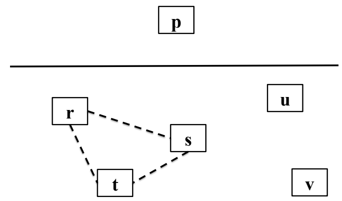


Fig. 1. Information Leakage with Entity Resolution

The information leakage may also be affected by any data analysis performed by adversary Eve. For example, Eve may run an entity resolution (ER) operation to identify which pieces of information refer to the same person. To illustrate, Figure 1 shows the records of five people: $r = \{\langle$N, Alice$\rangle$, $\langle$P, 123$\rangle\}$, $s = \{\langle$N, Alice$\rangle$, $\langle$C, 999$\rangle\}$, $t = \{\langle$N, Alice$\rangle$, $\langle$P, 987$\rangle\}$, $u = \{\langle$N, Bob$\rangle$, $\langle$P, 333$\rangle\}$, and $v = \{\langle$N, Carol$\rangle$, $\langle$S, 000$\rangle\}$. The record p above the line represents Alice's full information. Assuming that the name is a strong identifier for people, Eve may conclude that r, s, and t refer to the same person and merge their contents into $r + s + t$ $= \{\langle$N, Alice$\rangle$, $\langle$P, 123$\rangle$, $\langle$C, 999$\rangle$, $\langle$P, 987$\rangle\}$ (denoted as the dotted lines connecting r, s, and t). As a result, Eve may obtain better information about Alice. However, the analysis itself may be costly if the data to resolve is very large and Eve does not have sufficient resources to perform the analysis.

In summary, our contributions in this paper are as follows:

- We formalize information leakage as a general measure of privacy. Our measure reflects the following factors: the correctness and completeness of the leaked data, the adversary's confidence on the data, and the adversary's analysis on the data (Section 2).
- We compare our information leakage model with two related privacy models in data publishing: k-anonymity and l-diversity (Section 3).
- We formulate various problems for managing information leakage that can be solved using our framework (Section 4).
- We propose efficient exact and approximate algorithms for computing information leakage (Section 5).
- We experimentally evaluate leakage in a synthetic environment, both to check that the model matches our intuition, as well as to test the scalability of the exact and approximate leakage algorithms. (Section 6).

2 Information Leakage Measure

We consider a scenario where the adversary Eve has one record r pertaining to Alice, which could be a piece of information collected from a social network profile, a homepage, or even a tweet. Record r contains a set of attributes, and each attribute consists of a label and value. We do not assume a fixed schema to be able to represent data from different sources. As an example, the following record may represent Alice:

$$r = \{\langle \text{N}, \text{Alice}\rangle, \langle \text{A}, 20\rangle, \langle \text{A}, 30\rangle, \langle \text{Z}, 94305\rangle\}$$

Each attribute $a \in r$ is surrounded by angle brackets and consists of one label $a.l$ and one value $a.v$. Notice that there are two ages for Alice. We consider $\langle$A, 20$\rangle$ and $\langle$A, 30$\rangle$ to be two separate pieces of information, even if they have the same label. Multiple label-value pairs with identical labels can occur when two records merge and the label-value pairs are simply collected. In our example, Alice may have reported her age to be 20 in one case, but 30 in another. (Equivalently, year of birth can be used instead of age.) Although we cannot express the fact that Alice has only one age (either 20 or 30), the confidences we introduce in Section 2.3 can be used to indicate the likelihood of each value.

In addition, each attribute label l has an weight w_l that reflects the relative importance of an attribute with label l. These weights will be used below to compute leakage, so that attributes with highly weighted labels will contribute more than those with lower weights. In our example, we may give the credit card label

C a weight of $w_C = 3$ and the zip code label a weight of w_Z =1, to reflect that Alice considers her credit card number three times more important than her zip code. The absolute values of the weights are not important, only their relative sizes. Thus, if there are only three labels, giving them the weights 1, 2, and 3 is equivalent to giving them the weights 2, 4, and 6. We emphasize that the weights are assigned to labels and not on the individual attributes. We believe this simplification is useful, since giving weights to every possible attribute is not practical.

In our model we assume that different attributes are *not* correlated. However, in some cases the values for some attributes may depend on each other. For example, the birth date of a person depends on the age of a person because the birth date can be used to compute the age. In this case, if both the date of birth and the age are discovered by Eve, we do not want to account for the loss twice. We get around this problem by simply assuming that our model contains only one of the dependent attributes, e.g., either date of birth or age. In other cases, attributes may be correlated, but not equivalent. For example, phone number and address may be correlated: if we know the phone number we may be able to narrow down the possible addresses, and vice versa. We can model this situation by assuming there are three attributes: J contains the "joint information," A contains the remaining address information, and P the remaining phone information. If Eve discovers Alice's phone number, she has values for J and P; if she discovers the address, she gets J and A, and if she has both address and phone, Eve has J, A and P. Now we can provide weights for the J, A and P labels, and not double count the correlated knowledge.

We also assume a reference record p that contains Alice's complete information. An interesting question is how much of Alice's information has been revealed by exposing the record r? One might say that even if one attribute is leaked, a privacy breach has occurred, so Alice has no privacy. On the other extreme, however, one may say that, since not all of Alice's information has been leaked, the privacy has not been breached. In order to capture the amount of information that has been leaked, we define the *record leakage* of the record r as its similarity against the reference record p (see Definition 21). Also given a database R (which is a set of records), we define the *information leakage* of R to be its similarity against p after the "data analysis" of the adversary (see Definition 22).

In the following sections, we discuss the main components of our measure and formally define record leakage and information leakage.

2.1 Correctness

The correctness measure of a record r reflects the portion of r's information that is correct according to p. We adapt the definition of precision from the information retrieval literature [9] to define correctness. The precision of the record r against the reference p is defined as follows:

$$Pr(r,p) = \frac{\Sigma_{a \in r \cap p} w_{a.l}}{\Sigma_{a \in r} w_{a.l}}$$

If $\Sigma_{a \in r} w_{a.l} = 0$, we define Pr to be 0. Suppose that $p = \{\langle \text{N, Alice} \rangle, \langle \text{A, 20} \rangle, \langle \text{P, 123} \rangle, \langle \text{Z, 94305} \rangle\}$ and $r = \{\langle \text{N, Alice} \rangle, \langle \text{A, 20} \rangle, \langle \text{P, 111} \rangle\}$. Also say that $w_N = 2$ while the weights for all other labels are 1. Then the precision of r against p is $\frac{2+1}{2+1+1} = \frac{3}{4}$.

There are several ways to extend our definition of Pr. First, we can reflect the degree of error in computing information leakage where more correct information implies more leakage. For example, suppose that Alice is 30 years old. Then the information leakage when Eve guesses that Alice is 31 years old should be higher than the leakage when Eve suspects Alice is 80 years old. Second, we can take into account the statistical background knowledge of the adversary when measuring the leakage. For instance, if knowing that someone has an average age may be less leakage than knowing that someone has an exceptional age. We do not consider these extensions in this paper due to space limitations.

2.2 Completeness

Even if the correctness of r is very high, r may not be significant if only a small fraction of p has been discovered. Hence, we also define the notion of how much of p is found by r, which we call the completeness of r. This time, we adapt the definition of recall from the information retrieval literature. In general, one could use any other measure to capture how much correct information was found by the record r.

We define the recall of r against p as follows.

$$Re(r,p) = \frac{\Sigma_{a \in r \cap p} w_{a.l}}{\Sigma_{a \in p} w_{a.l}}$$

If $\Sigma_{a \in p} w_{a.l} = 0$, we define Re to be 0. In our example, the recall of r against p is $\frac{2+1}{2+1+1+1} = \frac{3}{5}$.

The information retrieval literature suggests the harmonic mean as one way of combining correctness and completeness. Given the correctness Pr and completeness Re, the weighted harmonic mean is defined as $F = \frac{1}{\alpha/Pr + (1-\alpha)/Re} = \frac{(\beta^2+1) \times Pr \times Re}{\beta^2 \times Pr + Re}$ where $\beta^2 = \frac{1-\alpha}{\alpha}$.

The F_1 measure sets β to 1 and thus gives equal weight to precision and recall. In information retrieval, the F_1 measure captures how relevant a search result is against a given query. In comparison, the information leakage measure quantifies the relevance of a record r against the correct answer p. We can combine the precision and recall to produce a single record leakage measure L^0 where the "0" superscript indicates the leakage computation without confidences.

$$L^0(r,p) = F_1(Pr(r,p), Re(r,p)) = \frac{2 \times Pr(r,p) \times Re(r,p)}{Pr(r,p) + Re(r,p)}$$

In our example, the F_1 value is $\frac{2 \times 3/4 \times 3/5}{3/4 + 3/5} = \frac{2}{3}$.

2.3 Adversary Confidence

As mentioned earlier, the confidence that adversary Eve has on her data plays a role in computing leakage. For example, Eve may have gotten some information of r from an unreliable website. Or Eve may have heard rumors of the subject indirectly from someone else. If Eve is not so confident about r, then even if r has a high accuracy against p, the information leakage should be less than when Eve is more confident. Also, if Eve is highly confident about information that is not accurate, then the information leakage should be considered less than when Eve is not so confident about the inaccurate information.

To reflect the confidence of Eve, we extend our data model to have per-attribute confidence values. As a result, a record r consists of a set of attributes, and each attribute contains a label, a value, and a confidence (from 0 to 1) that captures the uncertainty of the attribute from Eve's point of view (the more Eve knows about Alice, the higher the confidence values). Any attribute that does not exist in r is assumed to have a confidence of 0. As an example, the following record may represent Alice:

$$r = \{\langle \text{N}, \text{Alice}, 1\rangle, \langle \text{A}, 20, 0.5\rangle, \langle \text{A}, 30, 0.4\rangle, \langle \text{Z}, 94305, 0.3\rangle\}$$

That is, Eve is certain about Alice's name, but is only 50% confident about Alice being 30 years old, 40% confident in Alice being 30 years old, and 30% confident about Alice's zip code 94305. For each attribute $a \in r$, we can access a's label $a.l$, a single value $a.v$, and confidence $a.c$. We assume that attributes in the reference p always have a confidence of 1 and omit the confidence values. No two attributes in the same record may have the same label and value pair.

The confidences within the same record are independent of each other and reflect "alternate worlds" for Eve's belief of the correct information of Alice. For example, if we have $r = \{\langle$name, Alice, 1$\rangle$, $\langle$age, 20, 0.4$\rangle$, $\langle$phone, 123, 0.5$\rangle\}$, in Eve's view then there are four possible worlds: $\{\langle$name, Alice$\rangle$, $\langle$age, 20$\rangle$, $\langle$phone, 123$\rangle\}$ with probability $0.4 \times 0.5 = 0.2$, $\{\langle$name, Alice$\rangle$, $\langle$age, 20$\rangle\}$ with probability $0.4 \times (1-0.5) = 0.2$, $\{\langle$name, Alice$\rangle$, $\langle$phone, 123$\rangle\}$ with probability $(1-0.4) \times 0.5 = 0.3$, and $\{\langle$name, Alice$\rangle\}$ with probability $(1-0.4) \times (1-0.5) = 0.3$. We denote the possible worlds of a record r as the set of records without confidences

$$W(r) = \{\{\langle a.l, a.v\rangle | a \in r'\} | r' \in 2^r\}$$

where 2^r is the power set of r.

To help in our definition of leakage, we define the function $p(a, r)$ that simply gives the confidence of attribute a in record r:

$$p(a, r) = \begin{cases} b.c \ \exists b \in r \text{ s.t. } a.l = b.l \wedge a.v = b.v \\ 0 \quad o.w. \end{cases}$$

We now extend our information leakage measure in the previous section to use confidences. Since, the confidence values of the attributes in r are independent, we can define the record leakage of r against the reference p as follows. (In Section 5, we show how to compute the record leakage efficiently.)

Definition 21 *Given confidence values, the record leakage of record r against the reference p is*

$$\begin{aligned} L(r, p) &= E[L^0(\bar{r}, p)] \\ &= \sum_{r' \in W(r)} (\prod_{a \in r'} p(a, r))(\prod_{a \notin r'} 1 - p(a, r)) L^0(r', p) \end{aligned}$$

where $\bar{r}$ is a random variable of r's possible worlds.

That is, we are computing the expected value of the F_1 value between a possible world and p. For example, suppose that $p = \{\langle$N, Alice$\rangle$, $\langle$A, 20$\rangle$, $\langle$P, 123$\rangle\}$ and $r = \{\langle$N, Alice, 0.5$\rangle$, $\langle$A, 20, 1$\rangle\}$. Also say that $w_N = 2$ while all the other weights have a value of 1. There are two possible values for $\bar{r}$: $r_1 = \{\langle$A, 20$\rangle\}$ and $r_2 = \{\langle$N, Alice$\rangle$, $\langle$A, 20$\rangle\}$. We then compute $L^0(r_1, p) = \frac{2 \times 1/1 \times 1/3}{1/1 + 1/3} = \frac{1}{2}$ and $L^0(r_2, p) = \frac{2 \times 2/2 \times 2/3}{2/2 + 2/3} = \frac{4}{5}$. Thus $L(r, p) = \frac{1}{2} \times L^0(r_1, p) + \frac{1}{2} \times L^0(r_2, p) = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{4}{5} = \frac{13}{20}$. Notice that L receives a record with confidences as its first input and a record without confidences as its second input. One can also define the precision and recall metrics using confidences by replacing $L^0(r', p)$ in Definition 21 with $Pr(r', p)$ and $Re(r', p)$, respectively.

$L(r, p)$ quantifies leakage when Eve has a single record r in her possession. What happens if Eve has a set of records R? There is no simple answer in this case, but we take a "conservative" approach and define leakage as the worst leakage that may occur when Eve looks at any one of her R records. That is, $L^0(R, p) = max_{r \in R} L(r, p)$. Note that we are overloading the symbol L: if the first parameter is a set, it refers to set leakage; if the first parameter is a single record, then it is record leakage. We use the "0" superscript to distinguish this basic set leakage from leakage after the adversary analyzes and possibly combines records (next subsection).

2.4 Adversary Effort on Data Analysis

In order to increase the information leakage, the adversary Eve may further improve the quality of the database by fixing errors, adding more information, or removing duplicates. We illustrate three possible data analysis operations below.

- *Error Correction*: The adversary Eve identifies and corrects erroneous data. For example, Eve may fix misspellings of words in the database.
- *Augment Information*: Eve fills in missing data either by inferring the data or copying the data from other sources. For example, if Eve knows the addresses of people, then she could fill in their zip codes automatically.
- *Entity Resolution* [17, 4]: Eve can identify the records that refer to the same real-world entity and merge them into composite records. For example, if Eve has the three records r, s, and t in the database and know that r and s both refer to the same person, then she can merge r and s by combining the information of the two records into a single record $r + s$, resulting in a database with two records: $r + s$ and t. An ER operation can also use background information when resolving the records.

Among the operations, we will focus on the entity resolution (ER) operation. To illustrate how a data analysis operation can improve information leakage, say that we are running the ER function E on the database $R = \{r = \{\langle$N, Alice, 1$\rangle$, $\langle$P, 123, 1$\rangle\}$, $s = \{\langle$N, Alice, 1$\rangle$, $\langle$C, 999, 1$\rangle\}$, $t = \{\langle$N, Bob, 1$\rangle$, $\langle$P, 987, 1⟩}}. (Notice that we have set all the confidence values to 1 for simplicity.) Also suppose we have the reference record $p = \{\langle \text{N, Alice}\rangle, \langle \text{P, 123}\rangle, \langle \text{C, 999}\rangle, \langle \text{Z, 111}\rangle\}$. Before running E, the information leakage is $L^0(R,p) = max_{r\in R} L(r,p) = max\{\frac{2\times 2/2\times 2/4}{2/2+2/4}, \frac{2\times 2/2\times 2/4}{2/2+2/4}, 0\} = \frac{2}{3}$. While running E, suppose we conclude that r and s are the same person and produce the merged record $r+s = \{\langle \text{N, Alice, 1}\rangle, \langle \text{P, 123, 1}\rangle, \langle \text{C, 999, 1}\rangle\}$. Then the new database $E(R) = \{r+s, t\}$ has the information leakage $L^0(E(R),p) = max_{r\in E(R)} L(r,p)$ $= max\{\frac{2\times 3/3\times 3/4}{3/3+3/4}, 0\} = \frac{6}{7}$. Hence, by applying E, Eve has increased the information leakage of R from $\frac{2}{3}$ to $\frac{6}{7}$.

We can abstract any combination of data analysis operations as a function E, which receives the database R and returns another database $E(R)$ that may increase information leakage. For example, one can augment information to the database and then perform entity resolution.

While the above operations are powerful and may enhance information leakage, they require computation effort on Eve's side. For example, if a sophisticated ER algorithm takes quadratic time to run, then it may not be feasible to run the algorithm on all the hundreds of millions of people on the Web. If Eve uses a more relaxed and faster algorithm, then more records can be resolved.

To incorporate the adversary effort into our model, we define the cost function C that receives the adversary operation E and the database R, and returns the "cost" required to run E on R. The cost could be measured in computation steps, run time, or even in dollars. For instance, if has $O(n^2)$ complexity for resolving n records, then $C(E,R)$ could be $c \times |R|^2$ for some constant c.

Using the basic definition of information leakage and the data analysis operation E, we now define our information leakage measure as follows.

Definition 22 *Given an adversary operator E, the* information leakage *of R against p is $L(R,p,E) = L^0(E(R),p)$ with the cost $C(E,R)$.*

For example, say that there are 1000 records in the database R, and $L^0(R,p) = 0.3$. Also say that $C(E,R) = c \times |R|^2$ where $c = \frac{1}{1000}$. Now suppose that the operator E improves the information leakage where $L^0(E(R),p) = 0.9$. Hence, the data analysis using E has revealed an additional information of $0.9 - 0.3 = 0.6$ using a cost of $\frac{1}{1000} \times 1000^2 = 1000$. Notice that if E is an identity function where $E(R) = R$, then $L(R,p,E)$ reduces to the basic information leakage definition $L^0(R,p) = max_{r\in R} L(r,p)$.

We can extend our model to a scenario where Eve not just has a database of records R, but also has a "query" of interest. For example, Eve may be focusing on a person with name Alice who is 30 years old. In this case, Eve's query is $q = \{\langle \text{name, Alice, 1}\rangle, \langle \text{age, 30, 1}\rangle\}$. Eve can then look in R for records that are "related" to this Alice, thus expanding her information on this Alice. That is, starting with q, Eve may use ER to merge records in the database that refer to the same entity as q. Given an ER function E, we define the *dipping result* of q, $D(R,E,q)$, as the record $r \in E(R \cup \{q\})$ such that r is a merged result of q. For example, suppose we have the database $R = \{r = \{\langle \text{N, Alice, 1}\rangle, \langle \text{P, 123, 1}\rangle\}, s = \{\langle \text{N, Alice, 1}\rangle, \langle \text{C, 999, 1}\rangle\}, t = \{\langle \text{N, Bob, 1}\rangle, \langle \text{P, 987, 1}\rangle\}\}$. Also say that the ER function E merges all the records that have the same name. Given the query $q = \{\langle \text{N, Alice, 1}\rangle\}$, we then obtain a dipping result $r+s+q = \{\langle \text{N, Alice, 1}\rangle, \langle \text{C, 999, 1}\rangle, \langle \text{P, 123, 1}\rangle\}$ because both r and s have the same name as q. The information leakage of q can be defined as $L(D(R,E,q),p)$. (Reference [17] provides more details on ER and computing dipping results.)

3 Relationship to Other Measures

In this section, we compare our information leakage measure with two popular privacy models in data publishing. We first provide a detailed comparison of information leakage and the k-anonymity model [14]. Next, we briefly discuss how our measure relates to the l-diversity model [8]. Both models take an all-or-nothing approach where either all the records in a database are equally "safe" or none of the records are safe at all. In comparison, our information leakage model can be used to quantify various notions of privacy, e.g., the information leakage of an individual record within a database. Obviously, it is impossible to compare our model with every existing privacy model. For example, we do not directly compare our work with the t-closeness [7] or Differential Privacy [2] models. However, the same argument holds where information leakage

Name	Zip	Age	Disease
Alice	111	30	Heart
Bob	112	31	Breast
Carol	115	33	Cancer
Dave	222	50	Hair
Pat	299	70	Flu
Zoe	241	60	Flu

Table 1. Database of Patients (R)

Zip	Age	Disease
11*	3*	Heart
11*	3*	Breast
11*	3*	Cancer
2**	≥50	Hair
2**	≥50	Flu
2**	≥50	Flu

Table 2. 3-Anonymous Version of Table 1 (R_a)

can quantify various notions of privacy while the two existing models either deem the entire database safe or not safe.

3.1 k-anonymity

In data publishing, the k-anonymity model [14] prevents the identity disclosure of individuals within a database. More formally, a database R satisfies k-anonymity if for every record $r \in R$, there exist $k-1$ other records in R that have the same "quasi-identifiers." The quasi-identifers are attributes that can be linked with external data to uniquely identify individuals in the database. For example, suppose that all the records in R have three attributes: zip code, age, and disease. Given external information, if a person in R can be identified by looking at her zip code and age, then the quasi-identifiers are zip code and age. Also, a sensitive attribute is an attribute whose value for any particular individual must be kept secret from people who have no direct access to R. In our example, we consider the disease attribute as sensitive.

Table 1 illustrates a database R that contains the name, zip code, age, and disease information of patients. For example, patient 1 has the name Alice, zip code 111, an age of 30, and a heart disease. Table 2 shows the 3-anonymous version of R (called R_a) without the names. In order to anonymize a database, we "suppress" each quasi-identifier value by either replacing a number with a range of numbers or replacing one or more characters in a string with the same number of wild card characters. (A wild card character is denoted as '*' and represents any character.) For example, Dave's age 50 was suppressed to ≥ 50 while Alice's zip code 111 was suppressed to 1**. For any patient, there are two other patients that have the same zip code and age information. Each set of records that have the same quasi-identifiers form one equivalence class. We assume that the adversary Eve can view Table 2, but not Table 1.

The k-anonymity model takes an all-or-nothing approach where either all the records satisfy k-anonymity or not. That is, if each record is indistinguishable from $k-1$ other records in terms of quasi-identifiers, the released database is considered "safe." Otherwise, the database is not safe. In comparison, the information leakage model is more general and can quantify a wider range of privacy settings.

First, we can study the information leakage for individuals. For example, suppose that we compare the information leakage of Alice and Zoe. (For simplicity, we assume all attribute weights have the same value 1.) Say that we first run an ER algorithm E that merges all the records in Table 2 that have the same zip code and age. As a result, there are two merged records: r_1: {⟨Zip, 11*, 1⟩, ⟨Age, 3*, 1⟩, ⟨Disease, Heart, 1⟩, ⟨Disease, Breast, 1⟩, ⟨Disease, Cancer, 1⟩} and r_2: {⟨Zip, 2**, 1⟩, ⟨Age, ≥50, 1⟩, ⟨Disease, Hair, 1⟩, ⟨Disease, Flu, 1⟩}. If the reference record of Alice is p_a = {⟨Name, Alice⟩, ⟨Zip, 111⟩, ⟨Age, 30⟩, ⟨Disease, Heart⟩}, the

Name	Zip	Age
Alice	111	30

Table 3. Background Information (R_b)

information leakage of Alice is $max\{L(r_1, p_a), L(r_2, p_a)\} = max\{\frac{2\times 3/5\times 3/4}{3/5+3/4}, 0\} = \frac{2}{3}$. Here, we have made the simplification that a suppressed value (e.g., 1**) is equal to its non-suppressed version (e.g., 111). In practice, one could view a suppressed value as the original value with a reduced confidence value. If the reference record of Zoe is $p_b = \{\langle$Name, Zoe$\rangle$, $\langle$Zip, 241$\rangle$, $\langle$Age, 60$\rangle$, $\langle$Disease, Flu$\rangle\}$, then the information leakage of Zoe is $max\{L(r_1, p_b), L(r_2, p_b)\} = max\{0, \frac{2\times 3/4\times 3/4}{3/4+3/4}\} = \frac{3}{4}$. Again, we have simplified the comparison and considered a suppressed value (e.g, $\geq$50) to be equal to its unsuppressed version (e.g., 60). As a result, the information leakage of Zoe, $\frac{3}{4}$, is higher than that of Alice, $\frac{2}{3}$, although k-anonymity considers the records of both people to be equally safe.

Second, we can quantify the impact of background information on privacy. For example, say that the adversary knows additional information about Alice as shown in Table 2. The adversary can then combine the database R_b in Table 3 with Table 2 to measure Alice's information leakage. Using the same ER algorithm E as above, we now generate the two records r'_1: $\{\langle$Name, Alice, 1$\rangle$, $\langle$Zip, 11*, 1$\rangle$, $\langle$Age, 3*, 1$\rangle$, $\langle$Disease, Heart, 1$\rangle$, $\langle$Disease, Breast, 1$\rangle$, $\langle$Disease, Cancer, 1$\rangle\}$ and r_2: $\{\langle$Zip, 2**, 1$\rangle$, $\langle$Age, $\geq$50, 1$\rangle$, $\langle$Disease, Hair, 1$\rangle$, $\langle$Disease, Flu, 1$\rangle\}$. The information leakage of Alice is thus $max\{L(r'_1, p_a), L(r_2, p_a)\} = max\{\frac{2\times 4/6\times 4/4}{4/6+4/4}, 0\}$ $= \frac{4}{5}$. Hence, in the presence of the background information R_b, Alice's information leakage has increased from $\frac{2}{3}$ to $\frac{4}{5}$.

3.2 l-diversity

The l-diversity model [8] enhances the k-anonymity model by ensuring that the sensitive attributes of each equivalence class have at least l "well-represented" values. For example, in Table 2, the first equivalence class contains 3 distinct diseases while the second equivalence class has 2 distinct diseases. If $l = 3$, then we would like to enforce each equivalence class to have at least 3 distinct diseases. Suppose that we change Zoe's disease in Table 2 from Flu to Influenza. Then the modified database R'_a satisfies 3-diversity because each equivalence class has at least 3 different diseases.

Although R'_a is considered safe by l-diversity, the fact that Influenza is semantically similar to the Flu may result in less privacy for Zoe. We now illustrate how the information leakage model can quantify this change in privacy. First suppose that E considers the diseases Flu and Influenza to be different. Then using the ER algorithm E defined above, we generate the two records r_1: $\{\langle$Zip, 11*, 1$\rangle$, $\langle$Age, 3*, 1$\rangle$, $\langle$Disease, Heart, 1$\rangle$, $\langle$Disease, Breast, 1$\rangle$, $\langle$Disease, Cancer, 1$\rangle\}$ and r'_2: $\{\langle$Zip, 2**, 1$\rangle$, $\langle$Age, $\geq$50, 1$\rangle$, $\langle$Disease, Hair, 1$\rangle$, $\langle$Disease, Flu, 1$\rangle$, $\langle$Disease, Influenza, 1$\rangle\}$. Thus the information leakage of Zoe is $max\{L(r_1, p_b), L(r'_2, p_b)\}$ $= max\{0, \frac{2\times 3/5\times 3/4}{3/5+3/4}\} = \frac{2}{3}$. Now suppose that the operation E' is equivalent to E, but considers Influenza to be the same disease as the Flu and replaces all the occurrences of Influenza with Flu when merging records. In this case, the generated record r'_2 is now r''_2: $\{\langle$Zip, 2**, 1$\rangle$, $\langle$Age, $\geq$50, 1$\rangle$, $\langle$Disease, Hair, 1$\rangle$, $\langle$Disease, Flu, 1$\rangle\}$. As a result, the information leakage of Zoe becomes $max\{L(r_1, p_b), L(r''_2, p_b)\} = max\{0, \frac{2\times 3/4\times 3/4}{3/4+3/4}\}$ $= \frac{3}{4}$. Hence, by exploiting the application semantics, the information leakage of Zoe has increased from $\frac{2}{3}$ to $\frac{3}{4}$. Notice that this measurement could not be done using the l-diversity model, which cannot capture the usage of application semantics.

4 Applications

Our information leakage framework can be used to answer a variety of questions as we show in the following sections. As we use our framework, it is important to keep in mind "who knows what". In particular, if Alice is studying leakage of her information, she needs to make assumptions as to what her adversary Eve knows (database R) and how she operates (the data analysis function E Eve uses). These types of assumptions are common in privacy work, where one must guess the sophistication and compute power of Eve. On the other hand, if Eve is studying leakage she will not have Alice's reference information p. However, she may use a "training data set" for known individuals in order to tune her data analysis operations, or say estimate how much she really knows about Alice. In the following sections, we formalize problems in information leakage both in Alice's point of view and in Eve's point of view.

4.1 Releasing Critical Information

In this section, we formalize problems for managing Alice's information leakage. Suppose that Alice tracks R, the information she has given out in the past. She now wants to release a new record r (e.g., her credit card information) which may fall in the hands of the adversary who might use the ER function E to resolve other records with r. Alice can compute the direct leakage involved in releasing the record r, i.e., $L(R \cup \{r\}, p, E)$. However, we may want to capture the information leaked by r only instead of computing the entire leakage of the database. We thus define the incremental leakage of r as follows.

$$I(R, p, E, r) = L(R \cup \{r\}, p, E) - L(R, p, E)$$

Since r may make it possible for Eve to piece together big chunks of information about Alice, the incremental leakage may be large, even if r contains relatively little data.

To illustrate incremental leakage for a critical piece of information, say that Alice wants to purchase a cellphone app from an online store and is wondering which credit card c_1 or c_2 she uses will lead to a smaller loss of her privacy. Each purchase requires Alice to submit her name, credit card number, and phone number. Due to Alice's previous purchases, the store already has some information about Alice.

In particular:

- Alice's reference information is $p = \{\langle \text{N}, n_1\rangle, \langle \text{C}, c_1\rangle, \langle \text{C}, c_2\rangle, \langle \text{P}, p_1\rangle, \langle \text{A}, a_1\rangle\}$ where N stands for name, C for credit card number, P for phone, and A for address.
- The online store has two previous records $R = \{s = \{\langle \text{N}, n_1, 1\rangle, \langle \text{C}, c_1, 1\rangle, \langle \text{P}, p_1, 1\rangle\}, t = \{\langle \text{N}, n_1, 1\rangle, \langle \text{C}, c_2, 1\rangle\}\}$. (We omit the app information in any record for brevity.)
- The store accepts one of the two records $u = \{\langle \text{N}, n_1, 1\rangle, \langle \text{C}, c_1, 1\rangle, \langle \text{P}, p_1, 1\rangle\}$ or $v = \{\langle \text{N}, n_1, 1\rangle, \langle \text{C}, c_2, 1\rangle, \langle \text{P}, p_1, 1\rangle\}$ for the cellphone app purchase. Since Alice is purchasing an app, again no shipping address is required.

Suppose that two records refer to the same entity (or match) if their names and credit card numbers are the same or their names and phone numbers are the same, and that merging records simply performs a union of attributes. Also say that all weights w have the same value 1.

Then the information leakage of Alice before her purchase is $L(R, p) = max_{r \in E(R)} L(r, p) = max_{r \in \{s,t\}} L(r, p)$ $= max\{\frac{2\times 3/3 \times 3/5}{3/3+3/5}, \frac{2\times 2/2\times 2/5}{2/2+2/5}\} = max\{\frac{3}{4}, \frac{4}{7}\} = \frac{3}{4}$. If Alice uses c_1 and releases u to the store, then the information leakage is still $L(r, p) = \frac{2\times 3/3\times 3/5}{3/3+3/5} = \frac{3}{4}$ because u and s are identical and merge together, but not with t. If Alice uses c_2 and releases v instead, all three records merge together because v matches with both s and t. Hence the information leakage is $L(s + t + v, p) = \frac{2\times 4/4\times 4/5}{4/4+4/5} = \frac{8}{9}$. To compare Alice's two choices, we compute the incremental leakage values, i.e., the change in leakage values due to the app purchase. In our example, the incremental leakage of releasing u is $\frac{3}{4} - \frac{3}{4} = 0$ while the incremental leakage of releasing v is $\frac{8}{9} - \frac{3}{4} = \frac{5}{36}$. Thus, in this case Alice should use the credit card c_1 to buy her app because it preserves more of her privacy.

4.2 Releasing Disinformation

Releasing disinformation can be an effective way to reduce information leakage. Given previously released information R, Alice may want to release either a single record or multiple records that can decrease the information leakage. We call records that are used to decrease the leakage *disinformation* records. Of course,

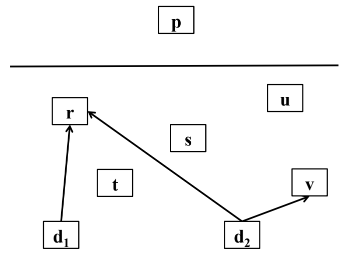


Fig. 2. Self and Linkage Disinformation

Alice can reduce the information leakage by releasing arbitrarily large disinformation. However, disinformation itself has a cost. For instance, adding a new social network profile would require the cost for registering information. As another example, longer records could require more cost and effort to construct. We use $C(r)$ to denote the entire cost of creating r.

We define the problem of minimizing the information leakage using one or more disinformation records. Given one data analysis operator E, a set of disinformation records S and a maximum budget of C_{max}, the optimal disinformation problem can be stated as follows:

$$\begin{aligned} &\text{minimize} \;\; L(R \cup S, p, E) \\ &\text{subject to} \; \Sigma_{r \in S} C(r) \leq C_{max} \end{aligned}$$

The set of records S that minimizes the information leakage within our budget C_{max} is called an optimal disinformation.

We study the problem of releasing disinformation. Figure 2 shows a database $R = \{r, s, t, u, v\}$ where r and s refer to the entity p while t, u, and v refer to an entity other than p. The disinformation record can reduce the database leakage in two ways. First, a disinformation record can perform *self disinformation* by acting as disinformation itself and add its irrelevant information to a correct record. In our example, the disinformation record d_1 snaps with the correct record r and adds its own information to r. Second, a disinformation record can perform *linkage disinformation* by reducing the database leakage by linking irrelevant records in R to a correct record. For example, the disinformation record d_2 is linking the irrelevant record v to the correct record r and thus adding v's information to r. Using these two basic disinformation strategies, one can perform a combination of self and linkage disinformation as well.

When creating a record, we use a user-defined function called $Create(S,\, L)$ that creates a new minimal record that has a size less or equal to L and is guaranteed to match all the records in the set S. If there is no record r such that $|r| \leq L$ and all records in S match with r, the *Create* function returns the empty record $\{\}$. A reasonable assumption is that the size of the record produced by *Create* is proportional to $|S|$ when $L > |S|$. We also assume a function called $Add(r)$ that appends a new attribute to r. The new attribute should be "incorrect but believable" (i.e., bogus) information. We assume that if two records r and s match, they will still match even if *Add* appends bogus attributes to either r or s. The *Create* function is assumed to have a time complexity of $O(|S|)$ while the *Add* function $O(|r|)$. Reference [17] provides more detail how to use the *Create* and *Add* functions to generate the optimal disinformation.

4.3 Enhancing a Composite Record

From the adversary Eve's point of view, there may also be interesting "optimization" questions to ask. Since Eve does not know Alice's full record p, the questions cannot be phrased in terms of p. Consider a composite record r_c that Eve has inferred from a set of facts in a set R. For whatever reason, Eve is very interested in r_c, but unfortunately there is some uncertainty in the attributes in r_c. We define r_p to be the same as r_c except that all confidences in r_c are set to 1 and omitted from the record. $L(r_c, r_p)$ is a measure of how certain r_c is: the closer $L(r_c, r_p)$ is to 1, the more certain Eve is of the information in r_c.

To improve $L(r_c, r_p)$ (i.e., make it closer to 1), Eve can try to increase her confidence in the attributes in R. For any given attribute $a = \langle l, v, c\rangle$ in some $r_i \in R$, Eve can improve the confidence of a by doing more research, bribing someone, issuing a subpoena, etc. The increase in the confidence of a will clearly have a cost associated with it. There are again many ways to model the cost, but for simplicity let us assume that the cost in changing the confidence from its current value of c to 1 is $C(a) = 1 - c$.

Now the question is, what is the most cost effective way to increase Eve's confidence in r_c. If Eve only wants to verify one attribute, then we want the one $a \in r_i$ that maximizes

$$\frac{L(r'_c, r_p) - L(r_c, r_p)}{C(a)}$$

where r'_c is the composite record Eve can infer when the confidence in a is increased to 1.

For example, suppose that we have the database $R = \{r_1 = \{\langle \text{N, Alice, } 1\rangle, \langle \text{A, } 20, 1\rangle\}, r_2 = \{\langle \text{N, Alice, } 0.9\rangle, \langle \text{P, } 123, 0.5\rangle, \langle \text{C, } 987, 1\rangle\}\}$ where N stands for name, A stands for age, P stands for phone, and C stands for credit card number. We assume that all weights w have the value 1. Suppose that r_1 and r_2 merges into $r_c = \{\langle \text{N, Alice, } 1\rangle, \langle \text{A, } 20, 1\rangle, \langle \text{P, } 123, 0.5\rangle, \langle \text{C, } 987, 1\rangle\}$ where we take the maximum confidence value when merging two attributes with the same label and value pair. Then $r_p = \{\langle \text{N, Alice}\rangle, \langle \text{A, } 20\rangle, \langle \text{P, } 123\rangle, \langle \text{C, } 987\rangle\}$. If we enhance the name in r_2 to have a confidence of 1, then $r'_c = \{\langle \text{N, Alice, } 1\rangle, \langle \text{A, } 20, 1\rangle, \langle \text{P, } 123, 0.5\rangle, \langle \text{C, } 987, 1\rangle\}$ (which is identical to r_c), and $C(\langle \text{N}, \text{Alice}, 0.9\rangle) = 1 - 0.9 = 0.1$. Then $\frac{L(r'_c, r_p) - L(r_c, r_p)}{C(\langle \text{N,Alice,0.9}\rangle)} = \frac{0}{0.1} = 0$. On the other hand, if we enhance the phone number of r_2, then $r'_c = \{\langle \text{N, Alice, } 1\rangle, \langle \text{A, } 20, 1\rangle, \langle \text{P, } 123, 1\rangle, \langle \text{C, } 987, 1\rangle\}$ with the cost $C(\langle \text{P}, 123, 0.5\rangle) = 1 - 0.5 = 0.5$. Then $\frac{L(r'_c, r_p) - L(r_c, r_p)}{C(\langle \text{P},123,0.5\rangle)} = (\frac{2\times 4/4 \times 4/4}{4/4+4/4} - (\frac{1}{2} \times \frac{2\times 4/4\times 4/4}{4/4+4/4} + \frac{1}{2} \times \frac{2\times 4/4 \times 3/4}{4/4+3/4}))/0.5 = \frac{1-(1/2+3/7)}{0.5} = \frac{1}{28}$. Hence, verifying the phone number in r_2 results in a better enhancement of r_c than verifying the name of r_2.

5 Computation

Computing information leakage efficiently is important because the amount of information (i.e., the number of attributes) within a record can be large in practice. Given a database R and a data analysis operation E, the information leakage (see Definition 22) can be computed by running $E(R)$, and then computing the maximum record leakage by iterating each record r in $E(R)$ and computing $L(r, p)$. (Since in general we do not have any $E(R)$ prior knowledge, the only strategy is to iterate through the resulting $E(R)$ records.) Given that computing the record leakage $L(r, p)$ takes $f(|r|, |p|)$ time and running E on R takes $g(|R|)$ time, the total complexity of computing the information leakage is $O(g(|R|) + \sum_{r\in E(R)} f(|r|, |p|))$.

A naïve approach for computing the record leakage is to iterate through all possible worlds of r and add the record leakage values (without confidences) multiplied by their probabilities as shown in Definition 21. This solution has an exponential complexity of $O(2^{|r|} \times |r|)$. In the following sections, we propose efficient solutions for computing the record leakage. We first propose a $O(|p| \times |r|^2)$-time algorithm that computes the exact value of information leakage and assumes constant weights. We then propose a $O(|p| \times |r|)$-time approximate solution that works for arbitrary weights.

5.1 Exact Solution using Constant Weights

We now show how to compute record leakage in polynomial time given that all the weights have a constant value w. We first re-write $L(r, p)$ as follows. First, equations (1) and (2) result from the definitions of L^0, Pr, and Re.

$$L(r,p) = E\left[\frac{2 \times Pr(\bar{r}, p) \times Re(\bar{r}, p)}{Pr(\bar{r}, p) + Re(\bar{r}, p)}\right] \tag{1}$$

$$= E\left[\frac{2 \times \sum_{a\in \bar{r}\cap p} w_{a.l}}{\sum_{a\in\bar{r}} w_{a.l} + \sum_{a\in p} w_{a.l}}\right] \tag{2}$$

Equation (3) uses the linearity of expectation and sums the record leakage for each attribute in p. The notation $\bar{r}\backslash\{b\}$ indicates the possible world of r without the attribute b.

$$= 2 \times \sum_{b \in p} p(b,r) \times \tag{3}$$
$$E\left[\frac{w_{b.l}}{\sum_{a \in \bar{r}\backslash\{b\}} w_{a.l} + w_{b.l} + \sum_{a \in p} w_{a.l}}\right]$$

Equation (4) simplifies equation (3) using the assumption that all the weights have an equal value.

$$= 2 \times \sum_{b \in p} p(b,r) \times E\left[\frac{1}{|\bar{r}\backslash\{b\}| + 1 + |p|}\right] \tag{4}$$

Equation (5) converts the expression $\frac{1}{X}$ to $\int_0^1 t^{X-1}dt$ and pushes the expectation operator within the integral.

$$= 2 \times \sum_{b \in p} p(b,r) \times \int_0^1 E\left[t^{|\bar{r}\backslash\{b\}|+|p|}\right] dt \tag{5}$$

Equation (6) uses the fact that all the attributes in r are independent and converts $E\left[t^{|\bar{r}\backslash\{b\}|}\right]$ into $\prod_{a \in z} E\left[t^{X_a}\right]$ where X_a is a random variable that is 1 if a appears in $\bar{r}$ and 0 otherwise. We also remove any attribute in r that has the same label and value as b by iterating the attributes in $z = \{\langle c.l, c.v\rangle | c \in r \wedge (c.l \neq b.l \vee c.v \neq b.v)\}$.

$$= 2 \times \sum_{b \in p} p(b,r) \times \int_0^1 t^{|p|} \prod_{a \in z} E\left[t^{X_a}\right] dt \tag{6}$$

Finally, equation (7) computes the expected value of t^{X_a} as $p(a,r) \times t + (1 - p(a,r))$.

$$= 2 \times \sum_{b \in p} p(b,r) \times \tag{7}$$
$$\int_0^1 t^{|p|} \prod_{a \in z} (p(a,r) \times t + (1 - p(a,r)))dt$$

Algorithm 1 numerically evaluates equation (7). Steps 3–12 convert the innermost product into an expression of the form $Y_0 \times t^n + Y_1 \times t^{n-1} + ...Y_n$, where n is the number of attributes in z. The coefficients $Y_0, ...Y_n$ are stored in list Y in the algorithm. (List Z is an auxiliary list used to compute the Y values.) Steps 13 and 14 evaluate equation (7). The integral is applied to each $p(b,r) \times t^{|p|} \times Y_x \times t^{n-x}$ term in turn, yielding $p(b,r) \times \frac{Y_x}{|p|+n-x+1}$. (Note that $|Y| = n + 1$.)

The complexity of the algorithm above is $O(|p| \times |r|^2)$ time because for each record in p, we dynamically construct the coefficients of t^X, which takes $O(|r|^2)$ time.

5.2 Approximation using Arbitrary Weights

We now show how to compute an approximation of $L(r,p)$ efficiently where the weights can be assigned different values. We can use the Taylor series of $F(X)$ about the point $E[X]$ to approximate $F(X)$ as follows.

$$F(X) = F(E[X]) + \frac{F'(E[X])}{1!} \times (X - E[X])$$
$$+ \frac{F''(E[X])}{2!} \times (X - E[X])^2 + \ldots$$

Algorithm 1: Record Leakage using Constant Weights

input : the records r, p
output: the record leakage $L(r,p)$

```
L ← 0;
for b ∈ p do
    Y ← (1.0);
    Z ← ();
    for a ∈ r do
        if a.l = b.l ∧ a.v = b.v then
            continue to next loop;
        Z.Add(Y.Get(0) × p(a, r));
        for x = 0, ..., |Y| − 1 do
            Z.Add(Y.Get(x) × (1 − p(a, r)) + Y.Get(x + 1) × p(a, r));
        Z.Add(Y.Get(|Y| − 1) × (1 − p(a, r)));
        Y ← Z;
    for x = 0, ..., |Y| − 1 do
        L ← L + 2 × p(b, r) × Y.Get(x) / (|p| + |Y| − x);
return L;
```

If we only take the second order approximation (i.e., the Taylor series visible in the above equation) and compute the expected value of $F(X)$, we have

$$E[F(X)] \approx F(E[X]) + \frac{F''(E[X])}{2!} \times Var[X]$$

Hence, we can derive the following approximation by starting from equation (3) and setting $Y = \sum_{a\in\bar{r}\setminus\{b\}} w_{a.l}$.

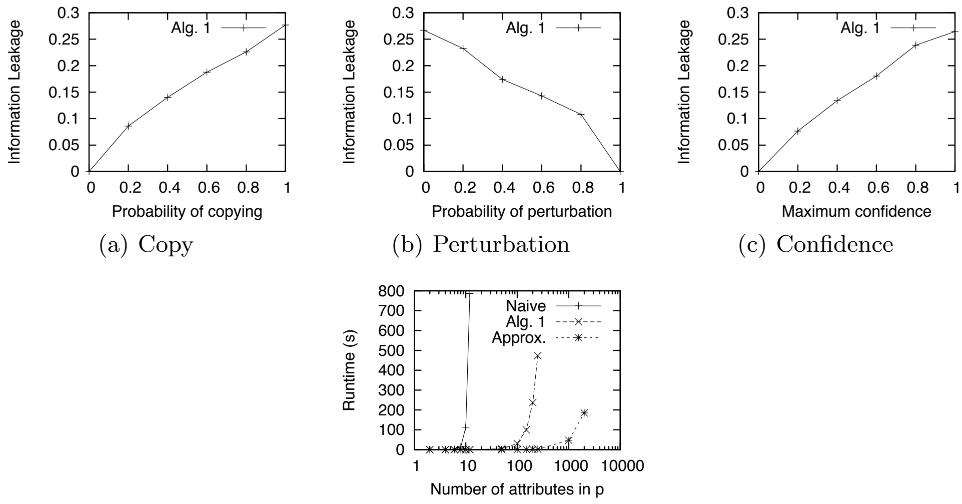
$$\begin{aligned} L(r,p) &= 2 \times \sum_{b\in p} p(b,r) \times E\left[\frac{w_{b.l}}{Y + w_{b.l} + \sum_{a\in p} w_{a.l}}\right] \\ &\quad (\text{where } Y = \sum_{a\in\bar{r}\setminus\{b\}} w_{a.l}) \\ &\approx 2 \times \sum_{b\in p} p(b,r) \times (\frac{w_{b.l}}{E[Y] + w_{b.l} + \sum_{a\in p} w_{a.l}} + \\ &\quad \frac{w_{b.l}}{(E[Y] + w_{b.l} + \sum_{a\in p} w_{a.l})^3} \times Var[Y]) \end{aligned}$$

We can also compute the expected value and variance of Y as follows.

$$\begin{aligned} E[Y] &= \sum_{a\in z} w_{a.l} \times a.c \\ Var[Y] &= \sum_{a\in z} w_{a.l}^2 \times a.c - (w_{a.l} \times a.c)^2 \end{aligned}$$

where $z = \{c | c \in r \wedge (c.l \neq b.l \vee c.v \neq b.v)\}$.

Hence, an approximation of the record leakage can be computed in $O(|p| \times |r|)$ time. One can extend the Taylor series above to produce an even more accurate solution. However, we show in Section 6.2 that our approximation based on the second order series is already quite accurate.



(a) Copy (b) Perturbation (c) Confidence

(d) Scalability

Fig. 3. Trends and Scalability

6 Experiments

We run experiments on synthetic data in order to observe trends and to study the scalability of our algorithms. Table 4 shows the configuration used. We first generate the reference record p by creating a set of n random attributes. We then generate a record $r \in R$ by iterating over each attribute in p and copying it with a probability p_c. However, each time there is a copy, we perturb the attribute with probability p_p into a new attribute. In addition, for each attribute in p we also add a new bogus attribute to r with probability p_b. The confidence value for each attribute generated was a random number between 0 and m, the maximum possible confidence. If $w = C$, then we set all the weights to 1, and if $w = R$, we randomly generated random real numbers between 0 and 1 for the weights. We repeated the generation of a record $|R|$ times. The last column of Table 4 shows the basic values of the parameters. Our base case does not represent any particular application or scenario; it is simply a convenient starting point from which to explore a wide range of parameter settings. Our algorithms were implemented in Java, and our experiments were run in memory on a 2.4GHz Intel(R) Core 2 processor with 4 GB of RAM.

Par.	**Description**	**Basic**
n	Size of the gold standard p	100
$\|R\|$	Number of records to generate	10,000
p_c	Probability of copying attribute from p to r	0.5
p_p	Probability of perturbing a copied attribute	0.5
p_b	Probability of adding bogus attribute to r	0.5
m	Maximum confidence value	0.5
w	Weights are constant (C) or random (R)	C

Table 4. Parameters for Data Generation

6.1 Trends

We plot the information leakage while varying the parameters p_c, p_p, and m. Any parameter that was not varied was set to its basic value in Table 4. Figure 3(a) shows the leakage when varying p_c from 0 to 1. As p_c

increases, more of p's attributes are copied to r, increasing the recall and thus the information leakage as well. Figure 3(b) shows the leakage when varying p_p from 0 to 1. This time, the more frequent the perturbation of an attributed being copied, the lower the precision and thus the information leakage becomes as well. Finally, Figure 3(c) shows the leakage when varying m from 0 to 1. As the average confidence increases, there are two competing factors that determine the information leakage: the higher confidence of correct information increases the leakage while the higher confidence of incorrect information decreases the leakage. In our setting, the correct information dominates and leakage increases as confidence increases.

6.2 Accuracy of Approximate Algorithm

We now evaluate the accuracy of the approximate algorithm in Section 5.2. Table 5 shows the leakage values for Algorithm 1 and the approximation algorithm while varying the parameters n, p_c, p_p, b, m, and w. If $w = C$, we generated $|R| = 10,000$ records with constant weights and ran Algorithm 1 to compute the exact leakage. If $w = R$, we can only compute the exact leakage with the naïve algorithm (which is not scalable as shown in Section 6.3). Thus, we limited ourselves to records with only 10 attributes ($|p| = 10$), and gave each attribute a random weight ranging from 0 to 1. As we can see in Table 5, in all scenarios the exact and approximate leakage values are nearly identical, with a maximum error rate of 0.006%. We conclude that our approximate algorithm is highly accurate.

n	p_c	p_p	b	m	w	**Exact**	**Approx.**
100	0.5	0.5	0.5	0.5	C	0.1740179	0.1740178
200	0.5	0.5	0.5	0.5	C	0.1374662	0.1374661
100	1.0	0.5	0.5	0.5	C	0.2752435	0.2752431
100	0.5	1.0	0.5	0.5	C	0.0	0.0
100	0.5	0.5	1.0	0.5	C	0.1581138	0.1581135
100	0.5	0.5	0.5	1.0	C	0.2625473	0.2625469
100	0.5	0.5	0.5	0.5	R	0.4047125	0.4046881

Table 5. Information Leakage Comparison

6.3 Runtime Performance

We compare the scalability of Algorithm 1 and the approximate algorithm against the naïve implementation defined in the beginning of Section 6. We varied the parameter n and used constant weights while using the basic values for the other parameters as defined in Table 4. Figure 3(d) shows that the naïve algorithm only handle up to 12 attributes. In comparison, Algorithm 1 scales to 250 attributes, and the approximation algorithm scales to more than 2,000 attributes, demonstrating the scalability of our algorithms.

7 Related Work

Many works have proposed privacy schemes [14, 8, 10, 2, 18]. The k-anonymity [14, 13] model guarantees that linkage attacks on certain attributes cannot succeed. Subsequent works such as the l-diversity [8] and t-closeness [7] models have improved the k-anonymity model. Rastogi et al. [10] shows the tradeoff between privacy and utility in data publishing in the context of maintaining the accuracy of counting queries. A recent line of works [2, 3] study differential privacy, which ensures that a removal or addition of a single database item does not (substantially) affect the outcome of any analysis on the database. In comparison, our information leakage measure focuses on quantifying the privacy of an individual against a given database and has the following features. First, we assume that some of our data is already public (i.e., out of our control) and that there can be a wide range of scenarios where we need to measure privacy. Second, our information leakage reflects the four different factors of privacy: the correctness and completeness of the leaked database, the adversary's confidence on the database, and the adversary's analysis on the database.

A closely-related framework to ours is P4P [1], which seeks to contain illegitimate use of personal information that has already been released to an external (possibly adversarial) entity. For different types of information, general-purpose mechanisms are proposed to retain control of the data. More recently, a startup called ReputationDefender [11] has started using disinformation techniques for managing the reputation of individuals focusing on improving search engine results (e.g., adding to the web positive or neutral information about its customers by either creating new web pages or by multiplying links to existing ones). The focus of ReputationDefender is to make one's correct information clearly visible. Hence, the disinformation is being used to maximize the information leakage. TrackMeNot [15] is a browser extension that helps protect web searchers from surveillance and data-profiling by search engines using noise and obfuscation. Finally, ICorrect [5] is a web site that allows one to clarify his/her misinformation on the Web. We believe that the works above show a clear need for using information leakage as a measure of privacy.

The information theoretic metric of entropy [12] is often used in the context of communication privacy and quantifies the amount of information an attacker is missing in verifying a hypothesis within a confidence interval. In comparison, the information leakage model focuses on capturing the intuitive notion of leakage: correctness, completeness, and the adversary confidence. In addition, our model incorporates the data analysis operations of the adversary and the costs for performing the operations.

Information retrieval [9] searches for relevant information within documents. Many different measures for evaluating the performance of information retrieval have been proposed. The notions of precision were first proposed by Kent et al.[6]. The F measure was introduced by van Rijsbergen [16]. Information leakage adopts these measures in a privacy setting. In addition, our measure reflects the adversary confidence and data analysis. Compared to probabilistic information retrieval [9] where documents are probabilistically ranked, our work probabilistically computes the information leakage itself using possible worlds semantics.

8 Conclusion

We have proposed a framework using information leakage as a measure for data privacy. In many applications, an important observation is that privacy is no longer an all-or-nothing concept because some of our data may inevitably become public through various interactions (e.g., buying a product from a vendor online). Our information leakage measure reflects four important factors of privacy: the correctness and completeness of the leaked data, the adversary's confidence on the data, and the adversary's data analysis. We have compared our information leakage model with the k-anonymity and l-diversity models. We have described several challenges in managing information leakage that can be posed by our framework. We have proposed efficient algorithms for computing the exact and approximate values of information leakage. Finally, we have shown through extensive experiments on synthetic data that the information leakage measure indeed captures the important factors of privacy, and that our information leakage algorithms can scale to large data.

References

1. G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, N. Mishra, R. Motwani, U. Srivastava, D. Thomas, J. Widom, and Y. X. 0002. Vision paper: Enabling privacy for the paranoids. In *VLDB*, pages 708–719, 2004.
2. C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.
3. C. Dwork. Differential privacy: A survey of results. In *TAMC*, pages 1–19, 2008.
4. A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Trans. Knowl. Data Eng.*, 19(1):1–16, 2007.
5. ICorrect. http://http://www.icorrect.com/.
6. A. Kent, M. M. Berry, F. U. Luehrs, Jr., and J. W. Perry. Machine literature searching VIII. Operational criteria for designing information retrieval systems. *American Documentation*, 6(2):93–101, 1955.

7. N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, pages 106–115, 2007.
8. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *ICDE*, page 24, 2006.
9. C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.
10. V. Rastogi, S. Hong, and D. Suciu. The boundary between privacy and utility in data publishing. In *VLDB*, pages 531–542, 2007.
11. ReputationDefender. http://www.reputationdefender.com.
12. F. M. Reza. *An Introduction to Information Theory*. Dover Publications, Sept. 1994.
13. L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):571–588, 2002.
14. L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
15. TrackMeNot. http://cs.nyu.edu/trackmenot.
16. C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2nd edition, 1979.
17. S. E. Whang and H. Garcia-Molina. Managing information leakage. In *CIDR*, pages 79–84, 2011.
18. X. Xiao and Y. Tao. Personalized privacy preservation. In *SIGMOD Conference*, pages 229–240, 2006.