Link Analysis, Privacy Preservation, and Random Perturbations

Hillol Kargupta, Kun Liu, Souptik Datta, and Jessica Ryan
Computer Science and Electrical Engineering Department
University of Maryland Baltimore County
Baltimore, Maryland 21250
{hillol, kunliu1, souptik1, jryan4}@cs.umbc.edu

Krishnamoorthy Sivakumar
School of Electrical Engineering and Computer Science
Washington State University
Pullman, Washington, USA
siva@eecs.wsu.edu

ABSTRACT

Link analysis is playing an increasingly important role in data mining for security and counter-terrorism applications where the data sets/streams are often distributed and privacy-sensitive. This paper explores the data pre-processing problem for link detection and other related problems in a privacy-sensitive environment. It particularly explores random perturbation techniques using additive and multiplicative noise for privacy-preserving data mining. It presents an overview of some of the recent results obtained by the authors regarding the privacy preserving properties of random additive and multiplicative perturbations. At first, it explores the privacy preserving capability of random additive perturbation and questions the efficacy of this approach in hiding sensitive information. Next it explores the result of multiplicative random perturbations for privacy-preserving applications and points out that random projection matrices appear promising for computing statistical aggregates, principal component analysis, and clustering without completely sacrificing the privacy of the data.

1. INTRODUCTION

Many security and counter-terrorism applications need link analysis techniques for identifying dependencies among different features, social networks, and communication profiles. There exists many different algorithms for link detection. Some of them work with relational tabular data; some of them work on more structured data, e.g. graphs. Mining such data sets/streams is a challenging problem. The problem becomes even more challenging when the data is privacy-sensitive. Financial transactions, health-care records, and network communication traffic are a few examples where we often deal with privacy-sensitive data. Figure 1 depicts the data sources of a typical security screening application where the data may be privacy sensitive. Consider the problem of detecting isomorphic sub-graphs, representing certain social group behavior in a large graph G representing a universe of social relationships. In most real-life applications, the graph $\mathcal{G}$ will be constructed based on information from a party (or multiple parties) different from the entity who is trying to analyze the data. Some of the information may be extremely privacy sensitive. As a result, either some portions of the graph or the entire graph can be confidential. Therefore, the party that owns the data (i.e. G) cannot simply hand it over to the data miner without making sure that the data will not be exposed to privacy violation. The challenge is in constructing a representation of the data that preserves (at least in an approximate sense) the underlying links that we want to detect while making sure that the representation does not divulge the original sensitive parts of the data/graph.

There is a growing body of literature on data mining techniques that are sensitive to the privacy issue. This has fostered the development of a class of data mining algorithms [1, 12, 15] that try to protect the data privacy with varying degrees of success [14]. These algorithms try to extract the data patterns without directly accessing the original data and attempt to guarantee that the mining process does not get sufficient information to reconstruct the original data. These efforts are related to the general framework of secured multi-party computation introduced elsewhere [21].

This paper considers the problem of mining multi-party privacy-sensitive data using random perturbation-based techniques. It first presents a negative result. It considers random additive perturbations used by many existing privacy-preserving data mining techniques (e.g. [1, 8]) that try to preserve data privacy by adding random noise while making sure that the underlying distribution is still accurately preserved. It points out that in many cases, the original data can be easily filtered out from the perturbed data using a spectral

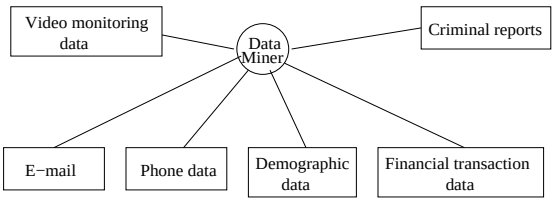

Figure 1: Data sources for a typical security screening application. Many of these sources deal with privacy sensitive data.

decomposition technique. This paper argues that these additive random perturbation-based techniques may not preserve any data privacy at all in many cases. Next, the paper explores multiplicative random projection matrices and points out that they may be useful for computing statistical aggregates, principal components analysis, and clustering from distributed privacy-sensitive data.

Section 2 briefly reviews the common link analysis applications and discusses their relevance to the current work. Section 3 explores additive random perturbation-based techniques for preserving privacy and questions the efficacy of this approach. Section 4 considers multiplicative random projection matrices for privacy-sensitive data mining. Finally, Section 5 concludes this paper.

2. LINK ANALYSIS APPLICATIONS

The techniques described in this paper consider data in the tabular relational format. This section considers different classes of link analysis applications reported in the literature and identifies their relevance to the work presented here.

Link Analysis is finding a growing number of applications in many domains such as social sciences [19], criminal intelligence [17], and large database structuring [10]. Traditional data mining techniques such as association rule mining, clustering, market basket analysis are sometimes used for link analysis. These algorithms usually work from tabular data and the material presented in the coming sections is directly applicable to these application scenarios. However, there also exist a large body of link analysis applications that deal with more complex types of data such as URL sequences, distributed data, and graph structured data to name a few.

The World-Wide-Web is another important domain for link analysis applications. Web search engines are becoming increasingly popular. While the earlier search engines used text analysis techniques to match documents with queries, the use of link analysis techniques has become more common, [4]. Typically the term link analysis, in this context, refers to the study of the algorithms operating over the web's link graph which defines the relationships between pages, based on the hyperlinks from page to page. These applications constitute an important component of this field. However, often these web mining applications are not the types where privacy issues are very critical. If the information is posted at the web then it is not likely to be very privacy sensitive. Therefore, in this paper we shall not directly consider the link analysis problem from web data.

Data in the form of graph structures shows up in many link analysis applications. Telephone communication networks and intelligence sources usually generate this types of data. These applications usually involve analysis of weighted directed or undirected graphs for detecting different characteristics like social groups, outliers behavior, and instance of target sub-graphs. Although the graphs themselves are not in tabular forms, they can be represented in that form. Adjacency matrix is one possible way to do that. For example, consider the graph shown in Figure 2. Let us assume that links C-E, F-E, and D-E are privacy sensitive. These links may correspond to properties that deal with sensitive features and therefore the exact link weights cannot be disclosed to the third party interested in mining the data.

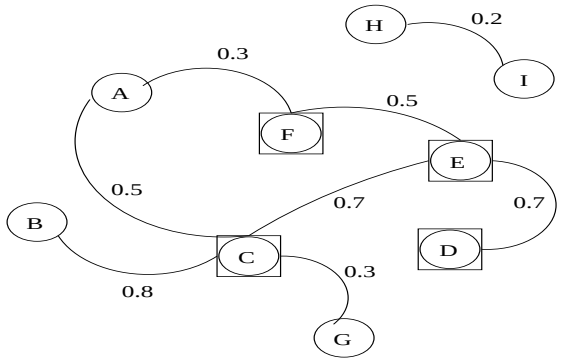

Figure 2: Graph data for link analysis.

	C	D	E	F
C	0	0	0.7	0
D	0	0	0.7	0
E	0.7	0	0	0.5
F	0	0	0.5	0

Table 1: The privacy sensitive links are represented using adjacency matrix-based representation.

One possible solution to this problem is to perturb the sensitive information in a secured fashion so that specific underlying data patterns remain invariant but the data itself appears very different from its original form. This problem can be posed in the following abstract form. Given the sensitive component of the graph shown in Table 1, find a representation of the data that preserves both privacy and the target types of data pattern.

As we see, the data in many link analysis applications can be represented in a tabular form. Therefore the rest of this paper will consider data represented in that form. The following section considers the possibility of using random additive noise to preserve the privacy of the data.

3. DENSITY ESTIMATION AND RANDOM ADDITIVE PERTURBATION

Random additive perturbation [1] is a natural choice to preserve privacy. It works by adding "randomly" generated noise from a given distribution to the values of sensitive attributes. In this section we discuss a spectral filtering technique for reconstructing the original data from the perturbed representation and argue that this apparent masking of data may not necessarily preserve privacy in many cases. First let us briefly review the random value perturbation technique introduced elsewhere [1].

3.1 Perturbing the Data

The random additive perturbation method attempts to preserve privacy of the data by modifying values of the sensitive attributes using a randomized process. The authors of [1] explore two possible approaches — Value-Class Membership and Value Distortion — and emphasize the Value Distortion approach. In this approach, the owner of a dataset returns a value $u_i + v$, where u_i is the original data, and v is a random value drawn from a certain distribution. The n original data values $u_1, u_2, \ldots, u_n$ are viewed as realizations of n independent and identically distributed (i.i.d.) random variables U_i, $i = 1, 2, \ldots, n$, each with the same distribution as that of a random variable U. In order to perturb the data, n independent samples $v_1, v_2, \ldots, v_n$, are drawn from a distribution V. The owner of the data provides the perturbed values $u_1 + v_1, u_2 + v_2, \ldots, u_n + v_n$ and the cumulative distribution function $F_V(r)$ of V. The reconstruction problem is to estimate the distribution $F_U(x)$ of the original data, from the perturbed data.

3.2 Estimation of Density Function from the Perturbed Dataset

Estimating the density function is a common problem in data mining and link analysis is not an exception. The density information can be used for clustering, classification, and other related problems. Perturbed data using additive noise allows estimating the underlying density function reasonably well.

The authors [1] suggest the following method to estimate the distribution $F_U(u)$ of U, given n independent samples $w_i = u_i + v_i$, $i = 1, 2, \ldots, n$ and $F_V(v)$. Using Bayes' rule, the posterior density function $f'_U(u)$ of U, given that $U + V = w$, can be written as

$$f'_U(u) = \frac{f_V(w-u) f_U(u)}{\int_{-\infty}^{\infty} f_V(w-z) f_U(z) dz},$$

where $f_U(.)$, $f_V(.)$ denote the probability density function of U and V respectively. If we have n independent samples $u_i + v_i = w_i$, $i = 1, 2, \ldots, n$, the corresponding posterior density can be obtained by averaging:

$$f'_U(u) = \frac{1}{n} \sum_{i=1}^{n} \frac{f_V(w_i - u) f_U(u)}{\int_{-\infty}^{\infty} f_V(w_i - z) f_U(z) dz}. \qquad (1)$$

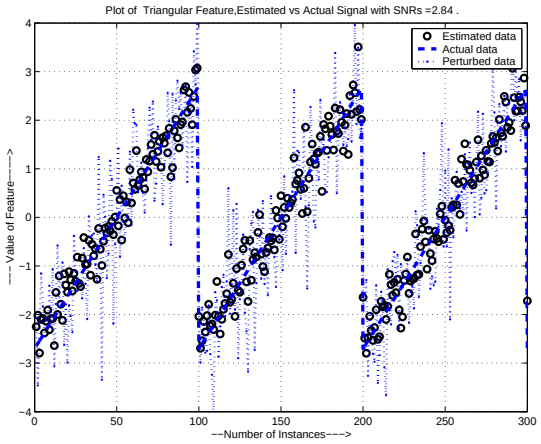

Figure 3: Estimation of triangular data using the spectral filtering technique.

For sufficiently large number of samples n, we expect the above density function to be close to the real density function $f_U(u)$. In practice, since the true density $f_U(u)$ is unknown, we need to modify the right-hand side of Equation 1. The authors suggest an iterative procedure where at each step $j = 1, 2, \ldots$, the posterior density $f_U^{j-1}(u)$ estimated at step $j-1$ is used in the right-hand side of Equation 1. Detailed description of this approach can be found elsewhere [1]. A related approach to estimate the density function and a discussion on quantifying privacy can be found in [2].

The following section presents a spectral filtering algorithm for filtering the noise out of the perturbed data. We use this filter to show that the original data can be accurately reconstructed from the randomized data and therefore this approach may not be suitable for preserving privacy.

3.3 Separating the Data from the Noise

This section points out that although the data may look apparently different after the random additive perturbation, it is possible to extract the original data by using spectral filtering techniques. Detailed description of the material discussed in the following can be found elsewhere [14].

Consider an $m \times n$ data matrix U and a noise matrix V with same dimensions. The random value perturbation technique generates a modified (or perturbed) data matrix $U_p = U + V$. Our objective is to extract U from U_p. Although the noise matrix V may introduce seemingly significant difference between U and U_p, it may not be successful in hiding the data. Random noise has well defined probabilistic properties that may be used to identify the noise component of the perturbed data matrix U_p in an appropriate representation. The rest of this section argues that the spectral representation of the data allows us to do exactly that.

Consider the covariance matrix of U_p:

$$\begin{aligned} U_p^T U_p &= (U+V)^T (U+V) \\ &= U^T U + V^T U + U^T V + V^T V. \end{aligned} \qquad (2)$$

Note that when the signal vector (columns of U) and ran-

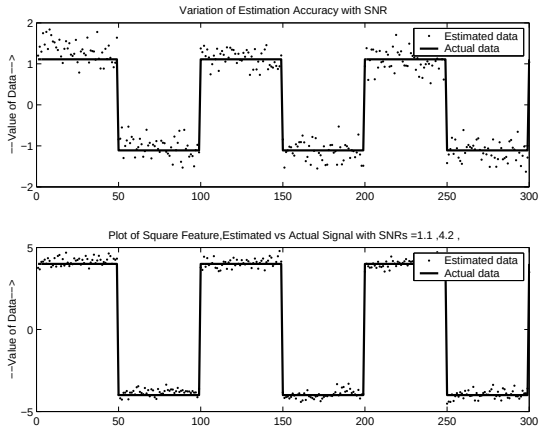

Figure 4: A higher noise content (low SNR) leads to less accurate estimation. SNR in upper figure is 1.1, while that for lower figure is 4.2.

dom noise vector (columns of V) are uncorrelated, we have $E[U^T V] = E[V^T U] = 0$. This assumption is valid in practice since the noise V that is added to the data U is generated by a statistically independent process. If the number of observations is sufficiently large, we have that $U^T V \approx 0$. Equation 2 can now be simplified as follows:

$$U_p^T U_p = U^T U + V^T V \tag{3}$$

Since the correlation matrices $U^T U$, $U_p^T U_p$, and $V^T V$ are symmetric and positive semi-definite, let

$$U^T U = Q_u \Lambda_u Q_u^T, \; U_p^T U_p = Q_p \Lambda_p Q_p^T, \text{ and} \tag{4}$$

$$V^T V = Q_v \Lambda_v Q_v^T, \tag{5}$$

where Q_u, Q_p, Q_v are orthogonal matrices whose column vectors are eigenvectors of $U^T U$, $U_p^T U_p$, $V^T V$, respectively, and Λ_u, Λ_p, Λ_v are diagonal matrices with the corresponding eigenvalues on their diagonals.

It has been shown elsewhere [14] that for "reasonable" signal-to-noise ratio,

$$\Lambda_p \approx \Lambda_u + \Lambda_v. \tag{6}$$

Suppose the signal covariance matrix has only a few dominant eigenvalues, say $\lambda_{1,(u)} \geq \cdots \geq \lambda_{k,(u)}$, with $\lambda_{i,(u)} \leq \epsilon$ for some small value ϵ and $i = k+1, \ldots, n$. This condition is true for many real-world signals. Suppose $\lambda_{k,(u)} > \lambda_{1,(v)}$, the largest eigenvalue of the noise covariance matrix. It is then clear that we can separate the signal and noise eigenvalues Λ_u, Λ_v from the eigenvalues Λ_p of the observed data by a simple thresholding at $\lambda_{1,(v)}$. Note that equation 6 is only an approximation. However, in practice, one can design a filter based on this approximation to filter out [14] the perturbation from the data. This filtering approach first separates the signal eigenstates from those belonging to the noisy eigenstates and then use the signal eigenstates to construct an approximation of the original data by projecting the perturbed data on to the subspace spanned by the signal eigenvectors. In other words, $\hat{U} = U_p A_u A_u{}^T$, where A_u is the matrix whose columns are the eigenvectors corresponding to the signal eigenvalues.

The filtered data based on this approximation turns out to be an accurate estimation of the original data. Figure 3 shows one such estimation of data when the actual data has a triangular trend. Extensive experimental results presented elsewhere [14] also support the observation.

The accuracy of the suggested method depends upon different factors. One is the relative amount of noise added to the actual data. The method works well as long as the relative noise content remain within a specific limit. In fact if that is not the case then the data mining algorithm will also have trouble extracting accurate patterns from the data. We define the term "Signal-to-Noise Ratio" (SNR) to quantify the relative amount of noise added to actual data to perturb it.

$$\text{SNR} = \frac{\text{Value of Actual Data}}{\text{Value of Noise Added to the Data}}$$

As the noise added to the actual value increases, the SNR decreases. Our experiments show that this method predicts the actual data reasonably well up to a SNR value of 1.0 (i.e. 100% noise). Figure 4 shows the difference in estimation accuracy as the SNR increases from 1. The dataset used has square trend in its values. The upper figure shows the estimation corresponding to 24% noise(mean SNR = 4.2), and the lower figure shows estimation corresponding to 90% noise (mean SNR = 1.1).

The second important factor is the inherent noise in the original dataset before we add noise explicitly for preserving privacy. The spectral filtering technique will remove the random noise regardless of its source. Therefore, if the data set contains some noisy eigenstates it will be removed since we do not have to identify whether this noise component originated from the original data set or from the privacy-preserving data transformation. As a result, sometimes the filtered data may look quite different from the original data set.

We have performed experiments with artificial dataset having specific trend in its value as well as real world dataset containing random component. The results show that for dataset with specific trend like the one shown in Figure 3, due to absence of any random component in actual data, Equation 6 holds closely, giving a close estimation of the actual data. However, for some real life datasets with inherent noise, the eigenvalues of signal and noise may not always be clearly non-overlapping and separable. In that case Equation 6 may not be applicable. Figure 5 shows that our method gives a close estimation of actual data when the dataset has specific trend of a sine curve, and SNR of the perturbed data is 1.1. It also shows that the performance is significantly better than that of a moving average filter. We also applied our method to 'Ionosphere data' available from [24] which has random component in its values. We perturbed the original data with random noise such that mean SNR is same as the artificial dataset, i.e. 1.1. Figure 6 shows that the recovery quality is poor compared to datasets with definite trend since the actual dataset has some random

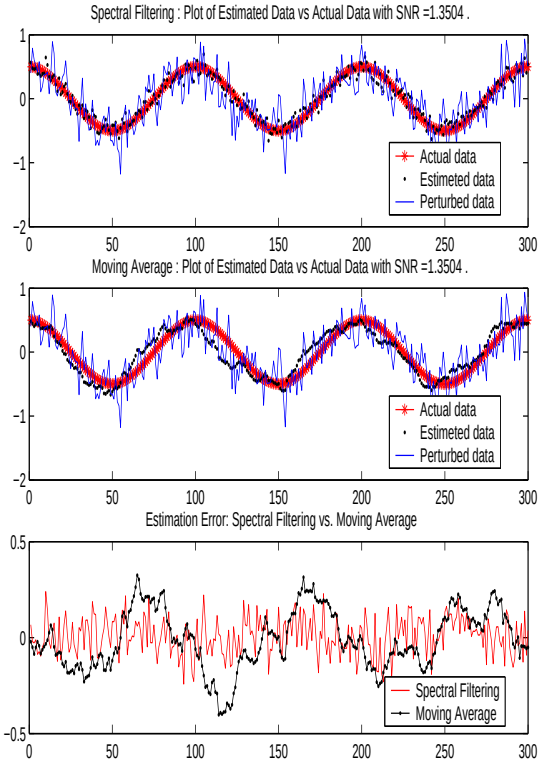

Figure 5: Spectral filtering recovers actual data with specific trend in its value closely. It works better than a moving average filter. This seems to be consistent over many experiments that we have performed using different data sets.

component.

This section clearly points out that for data sets where the underlying "signal" is non-random, spectral filtering techniques may be used to break the wall of privacy offered by some of the random additive noise-based privacy-preserving data mining algorithms, at least for the data sets used in our experiments. We need a solution to this problem. The following section explores the possibility of using a combination of multiplicative random projection matrices for privacy-preserving applications.

4. MULTIPLICATIVE PERTURBATIONS FOR PRESERVING PRIVACY AND RANDOM PROJECTIONS

The previous section pointed out some of the problems of privacy preserving data mining techniques that use additive random matrices. This section considers multiplicative random matrices for preserving privacy. If U be the data matrix and V be an appropriately sized random noise matrix then we are interested in the properties of the perturbed data $U_p = UV$. In general, we explore the properties of the data transformation induced by a collection of such multiplicative random noise matrices.

This paper specifically considers random projection matrices and points out [15] that multiplicative noise may be used

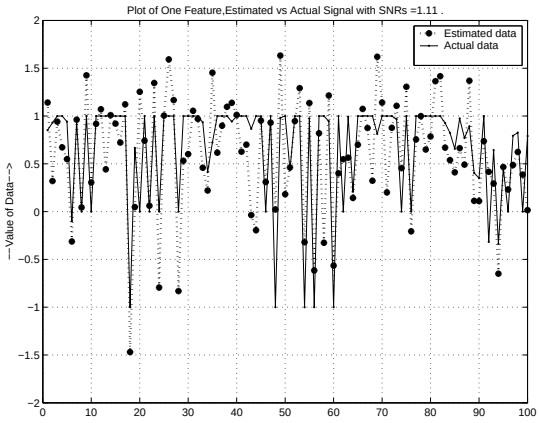

Figure 6: Spectral filtering performs poorly on a dataset where the original data has some inherent random noise components. It removes both the inherent noise data and also the noise explicitly added to preserve privacy.

to compute basic statistical aggregates, principal component analysis, and clustering without apparently completely sacrificing the data privacy.

In this section, we restrict ourselves to the problem of computing the correlation matrix from multi-party data set(s) where the owner of the data does not trust the third party who developed the data mining program. Although, correlation computation is a relatively simple kind of statistical operation, its frequent use in data mining (including link analysis) applications calls for the development of its privacy sensitive counter-part. Moreover, computing the correlation matrix allows us to solve several other related problems. We shall discuss this in more details soon. Let us first define the problem we plan to pursue.

Consider a distributed environment with two parties—$\mathcal{P}_u$ and $\mathcal{P}_c$. Party $\mathcal{P}_u$ owns the data set U and $\mathcal{P}_c$ owns the data mining program. Now $\mathcal{P}_c$ wants to mine the data set U; however $\mathcal{P}_u$ does not trust $\mathcal{P}_c$. Although $\mathcal{P}_u$ wants $\mathcal{P}_c$ to be able to extract the underlying patterns, it would like to make sure that $\mathcal{P}_c$ cannot reconstruct the data set.

It is natural to wonder why we cannot provide the correlation matrix to a third party directly since normally the raw data set cannot be reconstructed only from the correlation matrix? It is certainly possible if the client party is just interested in the correlation matrix, the data is owned by a single party, and the owner of the data has the resources to compute it. However, the objective of the privacy-sensitive data mining technology is somewhat different, in our opinion. In this domain, it is normal to assume that the owner of the data is not necessarily the data miner itself and the owner may not have any resource for mining the data. Moreover, the final objective is to allow the data miner only limited access to some representation of the data in such a way that a class of mining objectives (not just a single operation) are fulfilled. Providing just the correlation matrix to the client party does not meet the overall objective. This approach also does not work for heterogeneous distributed

data sets [15].

The correlation computation problem is directly related to several other common data mining tasks. A solution to this problem will help addressing a number of related problems. Some of them are discussed below. Correlation computation is very similar to the problem of computing the inner product from data [9, 20]. Zhu and Shasha exploited [22] an interpretation of the correlation coefficient as a measure of Euclidean distance between two data vectors. Note that many clustering applications make use of Euclidean distance; thus if we can compute the correlation matrix securely, we may also be able to implement a privacy preserving clustering algorithm. Vaidya and Clifton [18] proposed a privacy preserving association rule mining algorithm on vertically partitioned distributed data. The key insight is that if the entire transaction database is a boolean matrix where 1 represents the presence of the item (column feature) in a transaction, while 0 represents an absence correspondingly, the *support* of an itemset is nothing but the inner product of the vectors representing the sub-itemsets with both parties. Du and Zhan [7] presented a technique for building decision tree classifier from distributed privacy-sensitive data. The secure inner product computation acts as the building blocks for node-splitting evaluation and a secured correlation computation algorithm can be directly used for that.

Although, the purpose of this section is to investigate the properties of multiplicative noise, the following analysis is easier to understand in the context of deterministic orthogonal matrices. The following section presents that perspective.

4.1 Secure Correlation Computation and Orthogonal Matrices

The Pearson Product-Moment Correlation Coefficient, or correlation coefficient for short, is a measure of the degree of linear relationship between two random variables, X and Y. It is usually estimated from the given data set, comprised of m tuples (x_i, y_i), using the following expression:

$$Corr(X, Y) = \sum_{i=1}^{m} x_i y_i \tag{7}$$

We assume that the data columns are normalized so that they have 0 mean and unit length (ℓ_2 norm).

Computing correlations using the above expression in a straight forward fashion requires direct access to the data. We cannot compute the correlation coefficient using Equation 7 unless we know the values of the tuples (x_i, y_i). However, in a privacy sensitive application we cannot allow that. In this scenario, the data matrix U belongs to someone else. We can get the meta-data information that tells us about the underlying schema, the number of observed attributes, and the number of observed data points. Our goal is to compute the correlation matrix by observing some representation of U that does not allow reconstruction of the original data matrix U.

Let U be an $m \times n$ matrix, R_1 be an $n \times n$ random orthogonal matrix, and R_2 be an $m \times m$ randomly chosen orthogonal matrix. Now consider the following sequence of linear transformations of the data matrix U.

$$\begin{aligned} U_1 &= UR_1; \quad U_2 = U_1^T R_2 = R_1^T U^T R_2 \qquad (8)\\ U_2 U_2^T &= (R_1^T U^T R_2)(R_1^T U^T R_2)^T = R_1^T U^T R_2 R_2^T U R_1 \end{aligned}$$

Since both R_1 and R_2 are orthogonal matrices we can write,

$$R_1 U_2 U_2^T R_1^T = R_1 R_1^T U^T R_2 R_2^T U R_1 R_1^T = U^T U \tag{9}$$

Now recall that $U^T U$ is nothing but the correlation matrix of U. So if the owner of the data set U computes U_2 and hands over that and the matrix R_1 to a third party, the correlation matrix can still be computed by that party. However, since the matrix R_2 is hidden there is no way to exactly reconstruct the matrix U from U_2 and R_1. The following lemma formally states this claim.

LEMMA 1. *[15]* Given an $m \times n$ real-valued data matrix U, two random orthogonal matrices R_1, and R_2 such that $R_1 \neq I$ and $R_2 \neq I$, and U_2, as defined above by Equation 8. The matrix U is not uniquely defined by U_2 and R_1.

An intuitive proof sketch is given here.

PROOF. By exploiting the orthogonality property, we can rewrite Equation 8 as,

$$U_2^T R_1^T = R_2^T U \tag{10}$$

Thus given U_2 and R_1, one cannot determine U from Equation 10, this is based on the premise that the possible solutions is infinite when the number of equations is less than the number of unknown variables.

A further analysis of Equation 10 shows the above model holds the following ambiguities:

- If R_2^T has m columns and $rank(R_2^T) = m$, there are infinitely many solutions for U. The reason is that for any R_2^T, there exists a pseudoinverse matrix R_2^{T+} such that $R_2^{T+} R_2^T = I$, and $U = R_2^{T+} U_2^T R_1^T$.
- If R_2^T has m columns and $rank(R_2^T) < m$ (note this condition is false for orthogonal matrix), thus for any R_2^T, given one solution U such that $U_2^T R_1^T = R_2^T U$, we can construct the set of all solutions as $U + \{Y | R_2^T Y = 0\}$, where 0 here denotes the zero matrix of the same dimensions as $U_2^T R_1^T$. The columns of such matrices Y are arbitrary vectors z such that $R_2^T z = 0$, i.e., z is an arbitrary vector belonging to the null space of R_2^T.
- We cannot determine the scale of U. The reason is that, with both U and R_2^T being unknown, any scalar multiplier in one of the observations of U could always be cancelled by dividing the corresponding column vector of R_2^T.
- We cannot determine the order of the observations of U. The reason is that, with both U and R_2^T being unknown, any permutation matrix P and its inverse can be substituted in Equation 10 to give $U_2^T R_1^T = R_2^T P^{-1} P U$. The matrix PU can be viewed as a new

representation of U and $R_2^T P^{-1}$ can be viewed as a new representation of R_2^T. Thus we cannot distinguish between U and any other pertumations of it.

Thus for any two arbitrarily chosen matrices R_1 and R_2, the matrix U can be computed from U_2 and R_1 if and only if R_2 is an identity matrix. This contradicts the premise. □

Since the matrix U_2 is generated by two linear transformations, in the rest of this paper we shall call it a "double-sided" transformation of the matrix U. The following section explores a randomized approach to this problem. It shows that randomly generated projection matrices can be used to compress the data while preserving the correlation information without exposing the raw data.

4.2 Secure Correlation Computation and Random Projection Matrices

In this section we consider random noise matrices whose elements are random variables with given probability laws. We shall use these matrices to perturb the data without destroying some of the underlying patterns. However, the perturbations will be multiplicative, unlike the additive type that we have seen earlier in this paper. We will also be particularly interested in projection matrix-based random perturbations.

From Equation 9 we note that the matrices R_1 and R_2 should satisfy the following constraint: $R_1 R_1^T = R_2 R_2^T = I$. So far we treated R_1 and R_2 as square and orthogonal matrices. Now let us change the gear and redefine them.

Let R_1 be an $n \times k_1$ dimensional random noise matrix whose entries are independent, identically distributed (i.i.d.) according to some unknown distribution with zero mean and unit variance. Similarly let R_2 be an $m \times k_2$ dimensional random matrix with i.i.d. entries with zero mean and unit variance. The randomized approach presented in this section exploits the fact that $R^T R$ approximates an identity matrix on average.

Intuitively, this result echoes the observation made elsewhere [11] that in a high-dimensional space vectors with random directions are almost orthogonal. A similar result was proved elsewhere [3].

LEMMA 2. *[15]* Let R be a $p \times q$ dimensional random matrix such that each entry $r_{i,j}$ of R is independently chosen according to some unknown distribution with mean 0 and variance 1. Then, $E[RR^T] = qI$

Lemma 2 can be used to prove [15] the following result.

LEMMA 3. *[15]* Given an $n \times k_1$ dimensional random matrix R_1 and an $m \times k_2$ dimensional random matrix R_2 with i.i.d. entries chosen from N(0,1), and the "doubly-projected"

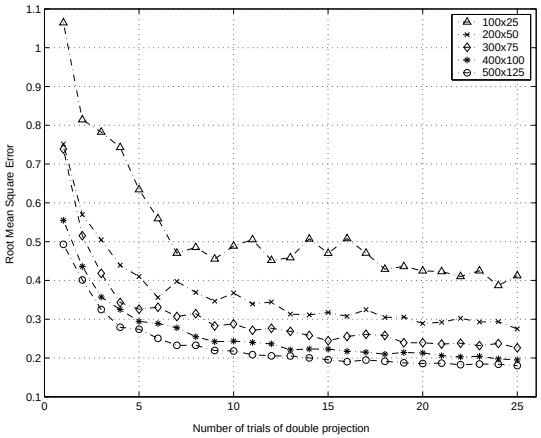

Figure 7: Performance of random projection-based algorithm with respect to different dimensional data sets. $k_1 = 60\%n$, $k_2 = 60\%m$.

matrix U_2 defined by Equation 8. Then,

$$\begin{aligned} diag[E(R_1 U_2 U_2^T R_1^T)] &= \\ (k_1^2 + 2k_1 + (n-1)k_1) k_2 diag(U^T U) & \\ E(R_1 U_2 U_2^T R_1^T) - diag[E(R_1 U_2 U_2^T R_1^T)] &= \\ (k_1 + {k_1}^2) k_2 (U^T U - diag(U^T U)) & \end{aligned}$$

where the columns of U have been z-score normalized and $diag(A)$ is the diagonal matrix of matrix A.

This result points out that one can estimate the correlation matrix $U^T U$ by computing the average of $R_1 U_2 U_2^T R_1^T$, which requires several transmissions of R_1 and U_2. This leads to the question that how much privacy is lost with each trial, if at all any? Lemma 1 proves there is no way to reconstruct U because U is not uniquely defined by U_2 and R_1 when R_1 and R_2 are random orthogonal matrices. This claim can be extended to any random matrices defined by Lemma 2. Moreover, if V is an orthogonal matrix and R_2 is a random matrix whose entries are i.i.d. with N(0,1) distribution, the entries of VR_2 are again i.i.d. with N(0,1) distribution. So the observations of U_2 which is equal to $R_1^T U^T R_2$ have the same distribution as would observations of $R_1^T U^T V R_2$, and thus we cannot use them to distinguish between U^T and any other matrices.

The total communication cost of this algorithm is $O(T(nk_1 + k_1 k_2))$ where T is the number of trials of double projection. Note the fact that the higher the dimensionality of the data set, the more likely the random vectors are orthogonal. So when the dimension of the data increases, the number of transmissions required to get a fixed error bound (which is normally in $O(2/k_2)$) decreases, which in turn decreases the communication cost. Figure 7 validates this claim. Figure 8 demonstrates the overall performance of the random projection algorithm with respect to k_1 and k_2 on Nasdaq100 data. [1]

[1] Available from http://quote.yahoo.com

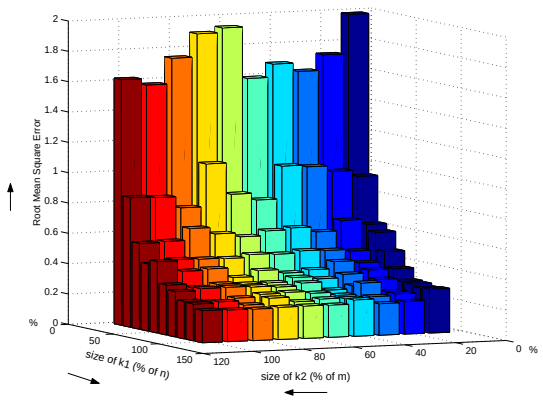

Figure 8: Performance of random projection-based algorithm with respect to k_1 and k_2 on Nasdaq data set. The estimated correlation matrix is an average of 25 independent trials.

4.3 Computing Correlation from Distributed Data

This section points out that the proposed technique can also be directly applied to compute correlation matrices from multiple distributed data sites. Let us first consider homogeneous sites [13] where each site observes the same set of attributes, but the observations are different. This scenario is sometimes called the horizontally partitioned distributed data mining scenario.

Let U and V be the two data sets owned by two different parties. Both the data sets observe the same set of attributes. Each column vector of the data sets has been normalized to have zero mean and unit length. Let x and y be two such attributes. Also let $Corr_U(x, y)$, $Corr_V(x, y)$, and $Corr_{U \cup V}(x, y)$ be the correlation matrices estimated from data sets U, V, and $U \cup V$ respectively. Then we can write,

$$Corr_{U \cup V}(x, y) = \frac{Corr_U(x, y) + Corr_V(x, y)}{2} \qquad (11)$$

The proposed approach exploits this useful decomposability property. After obtaining the estimated correlation matrix from both sites, we can combine them by adding them together to produce the overall estimated correlation matrix for the entire data set without ever seeing the raw data from either site.

An extension of this approach for handling distributed heterogeneous data sites is introduced elsewhere [15]. Here we only describe a naive approach that shares the same philosophy. Let U and V be the two data sets owned by two different parties $\mathcal{P}_u$ and $\mathcal{P}_v$ where each party observes a different set of features of the same observation. Let one of the parties generate a random matrix R as an encryption key and send that key to the other party through secure communication channel. Then each party encrypts their data by projecting them onto this random matrix and sends the new representations of original data sets, i.e., $R^T U$ and $R^T V$ to the third party respectively. The overall correlation can be computed at the third party through $(R^T U)^T (R^T V) = U^T V$ without directly accessing the raw data from each site. This simple technique assumes that the two participating data sites are collaborative. Further analysis and experimental results documenting the performance of these algorithms to mine distributed multi-party privacy-sensitive data can be found elsewhere [15].

The technique may be vulnerable to other filtering techniques like Independent Component Analysis (ICA) [16] if the multiplicative randomization matrix R is a square (not a projection) matrix. We have performed several preliminary experiments and noted that ICA does not perform well in estimating signal from data protected by random projections. Actually, it is generally not possible to design linear filters to simultaneously recover all the original data when the they are projected into a lower dimensional space by the projection matrices [5]. However, this calls for further study. Note that ICA is not directly comparable with our proposed spectral filtering technique described in section 3 of this paper. The reason is ICA works only if the independent components are non-Gaussian [23], while the spectral filtering technique reported here deals with Gaussian random noise.

This multiplicative perturbation method is fundamentally different from the additive perturbation technique described in section 3. The dimensions of the perturbed matrices U_1 and U_2 are different from the original data matrix U in most of the cases when the multiplicative randomization matrices R_1 and R_2 are non-square matrices. Additive perturbation does not project data in different dimension as done by multiplicative perturbation technique.

5. CONCLUSIONS

Preserving the privacy of the data is increasingly becoming an important issue in many link analysis applications. Therefore, the next generation of link analysis algorithms must be equipped to deal with techniques that prohibit compromising data privacy. This paper reviewed some of the recent results obtained by the authors in this area in the context of some of the existing work reported in the literature. It first pointed out that additive random perturbation based techniques may have questionable privacy-preserving properties for many data sets. Next it considered multiplicative random projection matrices and explored them for privacy-sensitive computation of statistical aggregates from distributed sources. While the approach seems promising we need more study to evaluate its potential for privacy-preserving data mining applications.

Acknowledgments

The authors acknowledge supports from the NASA (NRA) NAS2-37143 and the United States National Science Foundation CAREER award IIS-0093353. The authors would like to thank Haimonti Dutta for helping to prepare part of this manuscript.

6. REFERENCES

[1] R. Agrawal and S. Ramakrishnan. Privacy-preserving data mining. In *Proceedings of SIGMOD Conference*, pages 439–450, 2000.

[2] D. Agrawal and C. Aggarwal. On the Design and

Quantification of Privacy Preserving Data Mining Algorithms. In *Proceedings of Symposium on Principles of Database Systems*, pages 247–255, 2001.

[3] R. Arriaga and S. Vempala. An algorithmic theory of learning: Robust concepts and random projection. In *Proc. of the 40th Foundations of Computer Science*, New York, New York, 1999.

[4] S. Brin and L. Page The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Networks 30(1-7), pages 107-117, 1998.

[5] X.-R. Cao and R. wen Liu. A general approach to blind source separation. *IEEE Trans. Signal Processing*, 44:562–571, 1996.

[6] D. J. Cook and L. B. Holder. Graph Based Data mining. IEEE Intelligent Systems, 15(2), pages 32-41, 2000.

[7] W. Du and Z. Zhan. Building decision tree classifier on private data. In *Workshop on Privacy, Security, and Data Mining at The 2002 IEEE International Conference on Data Mining (ICDM'02)*, Maebashi City, Japan, December 9 2002.

[8] A. Evfimievski. Randomization in Privacy-Preserving Data Mining. ACM SIGKDD Explorations, Volume 4, issue 2, pages 43–48, 2003.

[9] R. Falk and A. Well. Many faces of the correlation coefficient. *Journal of Statistics Education*, 5(3), 1997.

[10] H. Goldberg and T. Senator. Restructuring databases for knowledge discovery by consolidation and link formation. First International Conference on Knowledge Discovery and Data Mining, AAAI Press, 1995.

[11] R. Hecht-Nielsen. Context vectors: general purpose approximate meaning representations self-organized from raw data. *Computational Intelligence: Imitating Life*, pages 43–56, 1994.

[12] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data, 2002.

[13] H. Kargupta, B. Park, D. Hershberger, and E. Johnson. Collective data mining: A new perspective towards distributed data mining. In *Advances in Distributed and Parallel Knowledge Discovery, Eds: Kargupta, Hillol and Chan, Philip*. AAAI/MIT Press, 2000.

[14] H. Kargupta, S. Datta, and K. Sivakumar. (2003). Random value perturbation: Does it really preserve privacy? Technical Report TR-CS-03-25, Computer Science and Electrical Engineering Department, University of Maryland, Baltimore County.

[15] K. Liu, H. Kargupta, and J. Ryan. (2003). Random projection and privacy preserving correlation computation from distributed data. Technical Report TR-CS-03-24, Computer Science and Electrical Engineering Department, University of Maryland, Baltimore County.

[16] L. C. Parra. An Introduction to Independent Component Analysis and Blind Source Separation. Sarnoff Corporation, CN-5300, Princeton, NJ 08543, 1999.

[17] M. K. Sparrow. The application of network analysis to criminal intelligence: An assessment of the prospects. *Social Networks*, volume 13, pp. 251-274.

[18] J. S. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *In The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, July 2002.

[19] S. Wasserman and K. Faust. Social Network Analysis: Methods and Applications. Cambridge University Press, 1994.

[20] K. L. Weldon. A simplified introduction to correlation and regression. *Journal of Statistics Education*, 8(3), 2000.

[21] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Symposium on Foundations of Computer Science (FOCS)*, pages 160–164. IEEE Computer Society Press, 1982.

[22] Y. Zue and D. Shasha. Statstream: Statistical monitoring of thousands of data streams in real time. In *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.

[23] A. Hyvarinen and E. Oja. Independent Component Analysis: A Tutorial. *Helsinki University of Technology, Finland*, April 1999.

[24] UCI Machine Learning Repository. http://www.ics.uci.edu/ mlearn/MLSummary.html.