Privacy-Preserving Clustering by Object Similarity-Based Representation and Dimensionality Reduction Transformation

Stanley R. M. Oliveira[1,2]
[1]Embrapa Informática Agropecuária
André Tosello, 209 - Barão Geraldo
Campinas, SP, Brasil
oliveira@cs.ualberta.ca

Osmar R. Zaïane[2]
[2]Department of Computing Science
University of Alberta
Edmonton, AB, Canada, T6G 1K7
zaiane@cs.ualberta.ca

Abstract

Preserving privacy of individuals when data are shared for clustering is a challenging problem. Data owners must not only meet privacy requirements but also guarantee valid clustering results. In this paper, we show that this dual goal can be achieved by transforming a database using two simple and effective data transformations: Object Similarity-Based Representation (OSBR) and Dimensionality Reduction-Based Transformation (DRBT). The former relies on the idea behind the similarity between objects, and the latter relies on the intuition behind random projection. The major features of our data transformations are: a) they are independent of distance-based clustering algorithms; b) they have a sound mathematical foundation; and c) they do not require CPU-intensive operations.

1. Introduction

In this paper, we focus primarily on Privacy-Preserving Clustering (PPC), notably when personal data are shared before clustering analysis. The challenge is how to protect the underlying data values subjected to clustering without jeopardizing the similarity between objects under analysis. Each application poses a new set of challenges. Let us consider two real-life motivating examples where the sharing of data poses different constraints.

- Small companies have recognized the value in data, especially with the introduction of the knowledge discovery process. However, small companies do not have enough (if any) expertise for doing data analysis, although they have good domain knowledge and understand their data. They have two choices: not mining the data at all, which is not a good option due to competitive reasons, or doing it with help from outside. Outsourcing the data mining process poses a potential security threat to data. A small enterprise could hire the service of a company specialized in data mining, i.e., the data mining would be outsourced. How can this enterprise transform its data before the outsourced data mining without putting in jeopardy the analysis itself?
- An Internet marketing company and an on-line retail company have datasets with different attributes for a common set of individuals. These organizations decide to share their data for clustering to find the optimal customer targets so as to maximize return on investments. How can these organizations learn about their clusters using each other's data without learning anything about the attribute values of each other?

The above scenarios describe two different problems of PPC. We refer to the former as *PPC over centralized data*, and the latter as *PPC over partitioned data*. The existing solutions in the literature address either PPC over centralized data [14, 15], PPC over vertically partitioned data [17], or PPC over horizontally partitioned data [13].

In this paper, we claim that PPC over centralized data and PPC over vertically partitioned data can be addressed by simple and effective transformations on a database. In particular, we show that the difficult goal of achieving full privacy and accuracy can be accomplished by the idea of dissimilarity between objects, but at a high communication cost. We refer to this solution as Object Similarity-Based Representation (OSBR). In order to alleviate the communication cost introduced by OSBR, we show that a trade-off between privacy and accuracy can be accomplished by using the intuition behind random projection. We refer to the latter solution as Dimensionality Reduction-Based Transformation (DRBT).

Dimensionality reduction techniques have been studied in the context of pattern recognition [7], information retrieval [3, 5, 9], and data mining [6, 5]. One of the promising methods designed for dimensionality reduction is random projection. In this work, we use random projection in a different context: *protection of the underlying values subjected to clustering*. In tandem with the benefit of privacy preservation, our solution DRBT benefits from the fact that random projection preserves distances quite nicely, which is desirable in clustering analysis.

The major features of our OSBR and DRBT are: a) they are independent of distance-based clustering algorithms; b) they have a sound mathematical foundation; and c) they do not require CPU-intensive operations.

We show analytically and experimentally that using OSBR and/or DRBT, a data owner can meet privacy requirements without losing the benefits of clustering since the similarity between data points is still preserved.

Our contributions in this paper can be summarized as follows: a) we demonstrate that PPC over centralized data and over vertically partitioned data can be addressed by DRBT, while using OSBR one can address PPC over centralized data. Most importantly, these solutions maintain the usefulness of the data and provide acceptable values in practice to address privacy concerns in clustering; b) we introduce a taxonomy of techniques to address PPC, including OSBR and DRBT in the related work section.

This paper is organized as follows. The background information related to clustering and dimensionality reduction is discussed in Section 2. In Section 3, we describe the research problem employed in our study. The solutions for PPC over centralized and vertically partitioned data are presented in Sections 4 and 5, respectively. The experimental results are presented in Section 6. Related work is reviewed in Section 7. Finally, Section 8 presents our conclusions.

2. Background

2.1. Data Matrix

Objects (e.g. individuals, patterns, events) are usually represented as points (vectors) in a multi-dimensional space. Each dimension represents a distinct attribute describing the object. Thus, an object is represented as an $m \times n$ matrix D, where there are m rows, one for each object, and n columns, one for each attribute. This matrix is referred to as a data matrix, represented as follows:

$$D = \begin{bmatrix} a_{11} & \dots & a_{1k} & \dots & a_{1n} \\ a_{21} & \dots & a_{2k} & \dots & a_{2n} \\ \vdots & & \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mk} & \dots & a_{mn} \end{bmatrix} \quad (1)$$

The attributes in a data matrix are sometimes normalized before being used. The main reason is that different attributes may be measured on different scales (e.g. centimeters and kilograms). For this reason, it is common to normalize the data so that all attributes are on the same scale. There are many methods for data normalization [8]. We review only two of them in this section: *min-max normalization* and *z-score normalization*.

Min-max normalization performs a linear transformation on the original data. Each attribute is normalized by scaling its values so that they fall within a small specific range, such as 0.0 and 1.0.

When the actual minimum and maximum of an attribute are unknown, or when there are outliers that dominate the min-max normalization, z-score normalization should be used. In z-score normalization, the values for an attribute A are normalized based on the mean and the standard deviation of A.

2.2. Dissimilarity Matrix

A dissimilarity matrix stores a collection of proximities that are available for all pairs of objects. This matrix is often represented by an $m \times m$ table. In (2), we can see the dissimilarity matrix D_M corresponding to the data matrix D in (1), where each element $d(i, j)$ represents the difference or dissimilarity between objects i and j.

$$D_M = \begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(m,1) & d(m,2) & \dots & \dots & 0 \end{bmatrix} \quad (2)$$

In general, $d(i, j)$ is a nonnegative number that is close to zero when the objects i and j are very similar to each other, and becomes larger the more they differ.

To calculate the dissimilarity between objects i and j one could use the most popular distance measure called Euclidean distance, or others. If i = ($x_{i1}, x_{i2}, ..., x_{in}$) and j = ($x_{j1}, x_{j2}, ..., x_{jn}$) are n-dimensional data objects, the Euclidean distance between i and j is given by:

$$d(i, j) = [\sum_{k=1}^{n} (x_{ik} - x_{jk})^2]^{1/2} \quad (3)$$

In case of binary variables (attributes), one can compute the dissimilarity between objects i and j by using the Jaccard coefficient [8] defined as:

$$d(i, j) = \frac{r + s}{q + r + s} \quad (4)$$

where r is the number of variables that equal 1 for object i but that are 0 for object j; s is the number of variables that equal 0 for object i but equal 1 for object j, and q is the number of variables that equal 1 for both objects i and j. This metric assumes that variables are asymmetric, i.e., the outcomes of the states are not equally important, such as positive and negative outcomes of a disease test.

Nominal variables can be encoded either by asymmetric binary variables or by mapping them to a numerical domain. However, if a dataset contains mixed variables, a more preferable approach is to process all variable types together performing a single cluster analysis. Combining the different variables into a single dissimilarity matrix brings all of the meaningful variables onto a common scale of the interval [0.0, 1.0]. For a dataset containing p variables of mixed types, the dissimilarity $d(i, j)$ between objects i and j is defined as:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}} \tag{5}$$

where the indicator $\delta_{ij}^{(f)} = 0$ if either: (a) x_{if} or x_{jf} is missing; or (b) $x_{if} = x_{jf} = 0$ and variable f is asymmetric binary; otherwise $\delta_{ij}^{(f)} = 1$. The contribution of variable f to the dissimilarity between i and j, $d_{ij}^{(f)}$, is computed dependent on its type:

- If f is binary or nominal: $d_{ij}^{(f)}$ = 0 if $x_{if} = x_{jf}$; otherwise $d_{ij}^{(f)}$ = 1.
- If f is interval-based: $d_{ij}^{(f)} = \frac{|x_{if} - x_{jf}|}{max_h x_{hf} - min_h x_{hf}}$, where h runs over all non-missing objects for variable f.

One advantage of using Equation (5) is that the dissimilarities between objects can be computed even when the variables describing the objects are of different types. Moreover, the dissimilarities are already normalized.

2.3. Dimensionality Reduction

When the data vectors are defined in a high-dimensional space it is computationally intractable to use data analysis or pattern recognition algorithms which repeatedly compute similarities or distances in the original data source. Therefore, it is necessary to reduce the dimensionality before clustering the data [11, 6].

The goal of the methods designed for dimensionality reduction is to map d-dimensional objects into k-dimensional objects, where $k \ll d$ [12]. These methods map each object to a point in a k-dimensional space minimizing the stress function:

$$stress^2 = (\sum_{i,j} (\hat{d}_{ij} - d_{ij})^2)/(\sum_{i,j} {d_{ij}}^2) \tag{6}$$

where d_{ij} is the dissimilarity measure between objects i and j in a d-dimensional space, and $\hat{d}_{ij}$ is the dissimilarity measure between objects i and j in a k-dimensional space. The function *stress* gives the relative error that the distances in k-d space suffer from, on the average.

One of the methods designed for dimensionality reduction is random projection. This method has been shown to have promising theoretical properties since the accuracy obtained after the dimensionality has been reduced using random projection is almost as good as the original accuracy. The key idea of random projection arises from the Johnson-Lindenstrauss lemma [10]: “if points in a vector space are projected onto a randomly selected subspace of suitably high dimension, then the distances between the points are approximately preserved.”

A random projection from d dimensions to k dimensions is a linear transformation represented by a $d \times k$ matrix R, which is generated by first setting each entry of the matrix to a value drawn from an i.i.d. N(0,1) distribution and then normalizing the columns to unit length. Given a d-dimensional dataset represented as an $n \times d$ matrix D, the mapping $D \times R$ results in a reduced-dimension dataset D', i.e.,

$$D'_{n \times k} = D_{n \times d} R_{d \times k} \tag{7}$$

Random projection is computationally very simple. Given the random matrix R and projecting the $n \times d$ matrix D into k dimensions is of order $O(ndk)$, and if the matrix D is sparse with about c nonzero entries per column, the complexity is of order $O(cnk)$ [16].

After the random projection, the distance between two d-dimensional vectors i and j is approximated by the scaled Euclidean distance of these vectors in the reduced space as follows:

$$d_{ij} = \sqrt{d/k} \parallel R_i - R_j \parallel \tag{8}$$

where d is the original and k the reduced dimensionality of the dataset. The scaling term $\sqrt{d/k}$ takes into account the decrease in the dimensionality of the data.

The choice of the random matrix R is one of the key points of interest. The elements r_{ij} of R are often Gaussian distributed, but this need not to be the case. Achlioptas [1] showed that the Gaussian distribution can be replaced by a much simpler distribution, as follows:

$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & with\, probability \quad 1/6 \\ 0 & with\, probability \quad 2/3 \\ -1 & with\, probability \quad 1/6 \end{cases} \tag{9}$$

In fact, practically all zero mean, unit variance distributions of r_{ij} would give a mapping that still satisfies the Johnson-Lindenstrauss lemma. Achlioptas’ result means further computational savings in database applications since the computations can be performed using integer arithmetics.

3. Problem Definition

We will approach the problem of PPC by first dividing it into two sub-problems: PPC over centralized data and PPC over vertically partitioned data. We do not address the case of horizontally partitioned data.

3.1. PPC over Centralized Data

In this scenario two parties **A** and **B** are involved, party **A** owning a dataset D and party **B** wanting to mine it for clustering. The dataset is assumed to be a data matrix $D_{m \times n}$, where each of the m rows represents an entity or object, and each entity contains values for each of the n attributes. The matrix $D_{m \times n}$ may contain binary, categorical, or numerical attributes.

Before sharing the dataset D with party **B**, party **A** must transform D to preserve privacy of individual data records. However, the transformation applied to D must not jeopardize the similarity between objects. Our first real-life motivating example is a particular case of this scenario.

The problem of PPC over centralized data can be stated as follows: Let D be a relational database and C a set of clusters generated from D. The goal is to transform D into D' so that the following restrictions hold:

- A transformation $\mathfrak{T}$ when applied to D must preserve the privacy of individual records, so that the released database D' conceals the values of confidential attributes, such as salary, disease diagnosis, credit rating, and others.
- The similarity between objects in D' must be the same as that one in D, or slightly altered by the transformation process. Although the transformed database D' looks very different from D, the clusters in D and D' should be as close as possible since the distances between objects are preserved or marginally changed.

3.2. PPC over Vertically Partitioned Data

Consider a scenario wherein k parties, such that $k \geq 2$, have different attributes for a common set of objects, as mentioned in our second real-life motivating example. In this scenario, the goal is to do a join over the k parties and cluster the common objects. The data matrix for this case is given as follows:

$$\begin{array}{c} \vdash \ \text{Party 1} \ \dashv\vdash \ \text{Party 2} \ \dashv\vdash \ldots \dashv\vdash \ \text{Party } k \ \dashv \\ \left[\begin{array}{cccc} a_{11} \ldots a_{1i} & a_{i+1} \ldots a_{1j} & & a_{1p+1} \ldots a_{1n} \\ \vdots & \vdots & \ldots & \vdots \\ a_{m1} \ldots a_{mi} & a_{mi+1} \ldots a_{mj} & & a_{mp+1} \ldots a_{mn} \end{array}\right] \end{array} \qquad (10)$$

Note that, after doing a join over the k parties, the problem of PPC over vertically partitioned data becomes a problem of PPC over centralized data. For simplicity, we do not consider communication cost here since this issue is addressed later. Therefore, the two restrictions stated in Section 3.1 must hold.

The challenge here is how to move the data of each party to a central party concealing the values of the attributes of each party. This central party could be any of the k parties. However, before moving the data to a central party, each party must disguise its data to protect the privacy of its attribute values. We assume that the existence of an object (ID) in a particular party may be revealed (for the purpose of the join operation), but the values of the associated attributes are private.

4. Object Similarity Representation

4.1. General Assumptions

The solution to the problem of PPC based on the similarity between objects draws the following assumptions:

- The data matrix D subjected to clustering could contain either binary, numerical, or categorical attributes, or even a combination of these attributes.
- The existence of an object (ID) should be replaced by a fictitious identifier.

4.2. PPC over Centralized Data

To address PPC over centralized data, OSBR performs three major steps before sharing the data for clustering as follows:

- *Step 1 - Suppressing identifiers*: Attributes that are not subjected to clustering (e.g., address, phone, etc) are suppressed.
- *Step 2 - Normalizing numerical attributes*: If the attributes subjected to clustering are numerical, they should be normalized. Normalization helps prevent attributes with large ranges (e.g, salary) from outweighing attributes with smaller ranges (e.g., age). If the dataset contains mixed variables, there is no need for normalization. The distances between objects are normalized when computing the dissimilarity matrix using Equation (5), in the next step.
- *Step 3 - Computing the dissimilarity matrix*: In the last step, the pairwise distances between objects are computed. Euclidean distance is widely used for numerical attributes and Jaccard coefficient for binary attributes. If the attributes are mixed types, Equation (5) must be used.

To illustrate how this solution works, let us consider the sample relational database in Table 1. This sample contains real data of the Cardiac Arrhythmia Database available at the UCI Repository of Machine Learning Databases [4]. The attributes for this example are: *age*, *weight*, *h_rate* (number of heart beats per minute), *int_def* (number of intrinsic deflections), *QRS* (average of QRS duration in msec.), and *PR_int* (average duration between onset of P and Q waves in msec.).

ID	age	weight	h_rate	int_def	QRS	PR_int
123	75	80	63	32	91	193
342	56	64	53	24	81	174
254	40	52	70	24	77	129
446	28	58	76	40	83	251
286	44	90	68	44	109	128

Table 1. A cardiac arrhythmia database.

Now suppose this data is made available for research purposes. One may be interested in clustering patients with similar characteristics to give a specific treatment to each group. Our goal here is to protect the underlying attribute values, and at the same time, guarantee accurate clustering results. Following the three steps of OSBR, the dissimilarity matrix D_M corresponding to the data matrix in Table 1, using Equation (3), is given as follows:

$$D_M = \begin{bmatrix} 0 & & & & \\ 2.2436 & 0 & & & \\ 3.3489 & 2.4776 & 0 & & \\ 3.6903 & 3.8844 & 3.1767 & 0 & \\ 3.0203 & 4.0828 & 4.1303 & 3.9955 & 0 \end{bmatrix} \quad (11)$$

The dissimilarity matrix is the dataset shared for clustering. Many clustering algorithms in the literature operate on a dissimilarity matrix [8]. In Section 4.3, we show that a dissimilarity matrix is no longer invertible, as long as the data analysts have no extra knowledge of the original data.

4.3. The Security of the OSBR

Now we move on to showing that sharing a dissimilarity matrix is a secure procedure. Our goal is to show that given the distance between two d-dimensional vectors, one cannot determine the coordinates of these two vectors.

Lemma 1 *Let $DM_{m \times m}$ be a dissimilarity matrix, where m is the number of objects. Given the distance between any two objects, it is impossible to determine the coordinates of the two objects by knowing only the distance between them.*

Proof: Let i and j be any two vectors in a d-dimensional space and let r be the distance between these vectors. For any given distance r, there exist infinitely many pairs of vectors i and j such that $d(i, j) = r$. In fact, for every vector i there exists a vector j such that $d(i, j) = r$. Therefore, the coordinates of i can be chosen completely arbitrarily and there is no way to deduce the coordinates of i from r. □

Even when sufficient care is taken, a solution that adheres to OSBR can still be vulnerable to partial disclosure. For instance, suppose that a user who has access to a dissimilarity matrix, shared by one data owner, knows all the attributes of one particular object o_i. In this case, partial disclosure can occur if o_i is identified in the global matrix. However, since identifiers in centralized data are replaced by fictive identifiers, identifying o_i is almost impossible.

Lemma 2 *Knowing the coordinates of a particular object i and the distance r between i and any other object j, it is possible to estimate the attributes of j.*

Proof: Let i and j be any two vectors in a d-dimensional space and let r be the distance between these vectors. If all the coordinates of i are known, then every coordinate of j cannot differ from the corresponding coordinate of i by more than r since j will lie on the circle of radius magnitude of r. In this case, one can at least have some estimates of the coordinates of j because there are finitely many vectors j. □

4.4. PPC over Vertically Partitioned Data

After illustrating how the problem of PPC over centralized data can be addressed by simply using the concept of dissimilarity matrix, we now evaluate the feasibility of this solution for PPC over vertically partitioned data.

In the context of PPC over vertically partitioned data, OSBR present two limitations, as follows:

- Lemma 2 shows the restriction of OSBR when an adversary has external knowledge of the original data. When two or more parties share data for clustering, if one party knows all the coordinates of a few points, the dissimilarity matrix may disclose the original dataset.
- The significant communication cost of OSBR, as we shall See in the next subsection, indicates that this solution is not attractive for PPC over vertically partitioned data.

The above limitations motivate our next solution based on the intuition of dimensionality reduction.

4.5. The Complexity of the OSBR

We have shown that it is possible to address PPC over centralized data based on the concept of dissimilarity matrix. The main advantages of this solution are: (a) it is independent of distance-based clustering methods; (b) it preserves privacy of values of the attributes subjected to clustering; (c) it is accurate and very simple to implement; and (d) it can handle both numerical and categorical attributes.

On the other hand, the communication cost makes PPC over vertically partitioned data sometimes restrictive. A dissimilarity matrix is a $m \times m$ table, where m is the number of objects under analysis. When m grows, which is not unexpected in data mining applications, this solution becomes too expensive in terms of communication cost. However, only half of the dissimilarity matrix is transmitted from one party to another since distance is a symmetric function. For instance, for a data matrix where m = 1,000,000 objects, the number of pairwise distances (elements of the matrix) is $C_{m,2} = \frac{(m \times (m-1))}{2}$ = 5e+11. Thus, the complexity of OSBR is of order $O(m^2)$.

5. Dimensionality Reduction Transformation

5.1. General Assumptions

The solution to the problem of PPC based on random projections draws the following assumptions:

- The data matrix D subjected to clustering contains only numerical attributes that must be transformed to protect individuals' data values before clustering.
- In PPC over centralized data, the existence of an object identifier (ID) should be replaced by a fictitious identifier. In PPC over vertically partitioned data, the ID of the objects are used for the join purposes between the parties involved in the solution.
- The transformation (random projection) applied to the original data might slightly modify the distance between data points. Such a transformation justifies the trade-off between privacy and accuracy.

One interesting characteristic of the solution based on random projection is that, once the dimensionality of a database is reduced, the attribute names in the released database are irrelevant. In other words, the released database preserves, in general, the similarity between the objects but the underlying data values are completely different from the original ones. We refer to the released database as *disguised database*, which is shared for clustering.

5.2. PPC over Centralized Data

To address PPC over centralized data, DRBT performs three major steps before sharing the data for clustering:

- *Step 1 - Suppressing identifiers*: Attributes that are not subjected to clustering (e.g., address, phone number, etc.) are suppressed.
- *Step 2 - Reducing the dimensions of the original dataset*: After pre-processing the data according to *Step 1*, an original dataset D is then transformed into the disguised dataset D' using random projection.
- *Step 3 - Computing the stress function*: This function is used to verify if the accuracy of the transformed dataset is marginally modified, which guarantees the usefulness of the data for clustering. A data owner can compute the stress function and go back to *Step 2* many times in order to find a good compromise between privacy and accuracy.

To illustrate how this solution works, let us consider the sample relational database in Table 1. We are going to reduce the dimension of that dataset from 6 to 3, one at a time, and compute the error (stress function). To reduce the dimension of this dataset, we apply Equation (7). In this example, the original dataset corresponds to the matrix D. We compute the random matrix R_1 by setting each entry of the matrix to a value drawn from an i.i.d. N(0,1) distribution and then normalizing the columns to unit length. We also compute the random matrix R_2 where each element r_{ij} is computed using Equation (9). We transform D into D' using both R_1 and R_2, one at a time. The random transformation RP_1 refers to the random projection using R_1, and RP_2 refers to the random projection using R_2.

The relative error that the distances in 6-3 space suffer from, on the average, is computed using Equation (6). Table 2 shows the values of the error using R_1 and RP_2. In this Table, K represents the number of dimensions in the disguised database D'.

Transformation	k = 6	k = 5	k = 4	k = 3
RP_1	0.0000	0.0223	0.0490	0.2454
RP_2	0.0000	0.0281	0.0375	0.1120

Table 2. The relative error that the distances in 6-3 space suffer from, on the average.

In this case, we have reduced the dimension of D from 6 to 3, i.e., D' contains 50% less attributes than D. Note that the error is very small for both RP_1 and RP_2. In addition, the attribute values are completely disguised through the random projections to preserve privacy of individuals. One example of the attribute values, after reducing the dimension of D from 6 to 3 attributes, is showed in Table 3. In this Table, we have the attributes labeled *Att1*, *Att2*, and *Att3* since we do not know the labels for the disguised dataset. The dataset D' was disguised using both RP_1 and RP_2.

ID	D' using RP_1			D' using RP_2		
	Att1	Att2	Att3	Att1	Att2	Att3
123	-50.40	17.33	12.31	91.0	-125.0	-97.58
342	-37.08	6.27	12.22	81.0	-98.50	-77.07
254	-55.86	20.69	-0.66	77.0	-93.0	-77.78
446	-37.61	-31.66	-17.58	83.0	-101.0	-73.53
286	-62.72	37.64	18.16	109.0	-123.0	-79.19

Table 3. Disguised dataset D' using RP_1 & RP_2.

As can be seen in Table 3, the attribute values are entirely different from those in Table 1.

5.3. PPC over Vertically Partitioned Data

The solution for PPC over vertically partitioned data is a generalization of the solution for PPC over centralized data. In particular, if we have k parties involved in this case, each party must apply the random projection over its dataset and then send the reduced data matrix to a central party. Note that, any of the k parties can be the central one. We show in Section 5.5 that DRBT greatly alleviates the communication cost when compared with the communication cost in OSBR.

When k parties ($k \geq 2$) share some data for PPC over vertically partitioned data, these parties must satisfy the following constraint:

- *Mutually exclusivity*: To avoid redundancy in the clustering analysis and to alleviate the communication cost, the attributes provided by the k parties should be mutually exclusive. More formally, if $A(D_1), A(D_2)..., A(D_k)$ are a set of attributes of the k parties, $\forall i \neq j \; A(D_i) \cap A(D_j) = \emptyset$.

The solution based on random projection for PPC over vertically partitioned data is performed as follows:

- *Step 1 - Individual transformation*: If k parties, $k \geq 2$, share their data in a collaborative project for clustering, each party K_i must transform its data according to the steps in Section 5.2.
- *Step 2 - Data exchanging or sharing*: Once the data are disguised by using random projection, the k parties are able to exchange the data among them. However, one party could be the central one to aggregate and cluster the data.
- *Step 3 - Sharing clustering results*: After the data have been aggregated and mined in a central party k_i, the results could be shared with the other parties.

5.4. The Security of the DRBT

In the previous sections, we show that transforming a database using random projection is a promising solution for PPC over centralized data and consequently for PPC over vertically partitioned data since the similarities between objects are marginally changed. Now we show that random projection also has promising theoretical properties for privacy preservation. In particular, we show that a random projection from d dimensions to k, where $k \ll d$, is a non-invertible transformation.

Lemma 3 *A random projection from d dimensions to k dimensions, where $k \ll d$, is a non-invertible linear transformation.*

Proof: A classic result from Linear Algebra asserts that there is no invertible linear transformation between Euclidean spaces of different dimensions [2]. Thus, if there is an invertible linear transformations from $\Re^m$ to $\Re^n$, then the constraint $m = n$ must hold. A random projection is a linear transformation from $\Re^d$ to $\Re^k$, where $k \ll d$. Hence, a random projection from d dimensions to k dimensions is a non-invertible linear transformation. □

5.5. The Complexity of the DRBT

One of the major benefits of a solution that adheres to DRBT is the communication cost to send a disguised dataset from one party to a central one. Unlike OSBT in which a data owner sends a dissimilarity matrix to a central party, in DRBT only a disguised data matrix is subject to communication cost.

In general, a disguised data matrix is of size $m \times k$, where m is the number of objects and k is the number of attributes (dimensions). Considering that $k \ll m$, the complexity of DRBT is of order $O(m)$.

To quantify communication cost of one solution, we consider the number of bits or words required to transmit a dataset from one party to a central or third party. Using DRBT, the bit communication cost to transmit a dataset from one party to another is $O(mlk)$, where l represents the size (in bits) of one element of the $m \times k$ disguised data matrix.

Recall that the communication cost in OSBR is much more expensive, i.e., $O(((m \times (m-1))/2) \times l)$, where l represents the size (in bits) of one element of the $m \times m$ dissimilarity matrix. Clearly, the solution that adheres to DRBT is much more attractive, in terms of communication cost, for addressing the problem of PPC over vertically partitioned data.

6. Experimental Results

We performed a series of experiments to measure the effectiveness of our solution based on dimensionality reduction (DRBT). We do not evaluate the OSBR because this solution requires a simple computation of a dissimilarity matrix and the suppression of identifiers before the release of data for clustering.

All the experiments were conducted on a PC, AMD Athlon 1900/1600 (SPEC CFP2000 588), with 1.2 GB of RAM running a Linux operating system.

To validated DRBT, we used two real datasets available at the UCI Repository of Machine Learning Databases [4]. The datasets are described as follows: a) *mushroom* with 8124 objects and 23 numerical attributes. This dataset contains records drawn from The Audubon Society Field Guide to North American Mushrooms; b) *chess* with 3196 objects and 37 numerical attributes containing information about legal moves of chess.

6.1. DRBT: PPC over Centralized Data

To measure the effectiveness of DRBT for PPC over centralized data, we computed the relative error that the distances in d-k space suffer from, on the average, by using the stress function given in Equation (6).

We computed the random matrix R_1 by setting each entry of the matrix to a value drawn from an i.i.d. N(0,1) distribution and then normalizing the columns to unit length. We also computed the random matrix R_2 where each element r_{ij} is computed using Equation (9). We transformed the datasets using both R_1 and R_2. In the random projection RP_1, we used the random matrix R_1, and in RP_2 we used the random matrix R_2.

Figure 1(A) shows the error produced by RP_1 and RP_2 on the mushroom dataset, while Figure 1(B) shows the error produced by RP_1 and RP_2 on the chess dataset. We reduced the dimensions of mushroom from 23 to 15, taking 3 dimensions at a time. Similarly, we reduced the dimensions of chess from 37 to 17, considering 4 dimensions at a time. The error produced by RP_1 and RP_2 increased slightly. As can be seen, the error produced by RP_1 and RP_2 on the mushroom dataset was less that 0.14, whereas the error on the chess dataset was less than 0.11. In both datasets, the errors produced by RP_2 were slightly lower than those in RP_1, which confirms the same findings in [3].

6.2. DRBT: PPC over Vertically Partitioned Data

To measure the effectiveness of DRBT for PPC over vertically partitioned data, we split the datasets mushroom and chess from 1 up to 4 parties and fixed the number dimensions to be reduced. In particular, we set $k = 12$, i.e., the number of projected dimensions for the dataset mushroom, and $k = 18$ for the dataset chess.

After applying the random projection (RP_1 and RP_2) to the subdatasets of each dataset, we computed the stress error on the subdatasets. Subsequently, we merged the results of each party to compose the aggregate dataset in a central party.

Figure 2(A) shows the error produced by RP_1 and RP_2 on the mushroom dataset when varying the number of parties from 1 up to 4. Likewise, Figure 2(B) shows the error produced by RP_1 and RP_2 on the chess dataset when varying the number of parties from 1 up to 4.

Again, in both cases the errors produced by RP_2 were slightly lower than those in RP_1. In the mushroom dataset, the error produced by RP_1 and RP_2 was less that 0.13, and in the chess dataset the error was less than 0.12.

These results suggest that random projection is a promising method for achieving PPC. Using random projection, a data owner can tune the number of dimensions to be reduced in a dataset trading privacy, accuracy, and communication costs before sharing the dataset for clustering.

7. Related Work

Some effort has been made to address the problem of PPC. We classify the solutions into two major groups: *PPC over centralized data* and *PPC over distributed data*.

A hybrid geometric data transformation method was proposed in [14] to meet privacy requirements as well as to guarantee valid clustering results. This method distorts numerical attributes by translations, scalings, and rotations or even by the combination of these geometric transformations. The key finding of this study was that by transforming a data matrix by rotations only, one would attain both accuracy and a reasonable level of privacy. The investigation also revealed that the Additive Data Perturbation (ADP) method, widely used in statistical databases, offers some level of privacy, but jeopardizes the distances between data points compromising the clustering results.

A more accurate investigation on PPC using geometric transformation is presented in [15]. In particular, it is shown that distorting attribute pairs in a database by using only rotations is a promising approach. In this work, a spatial data transformation method is introduced, called Rotation-Based Transformation (RBT). The method is designed to protect the underlying attribute values subjected to clustering without jeopardizing the similarity between data objects under analysis.

Regarding PPC over distributed data, we classify the existing solutions in two groups: *PPC over vertically partitioned data* and *PPC over horizontally partitioned data*. In a horizontal partition, different entities are described with the same schema in all partitions, while in a vertical parti-

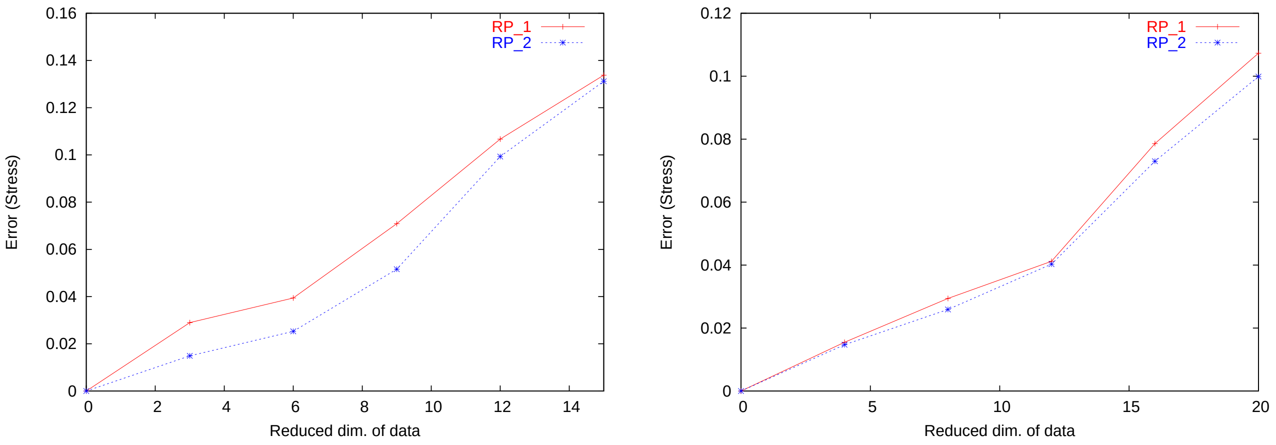

Figure 1. (A): The error produced on *mushroom* dataset. (B): The error produced on *chess* dataset.

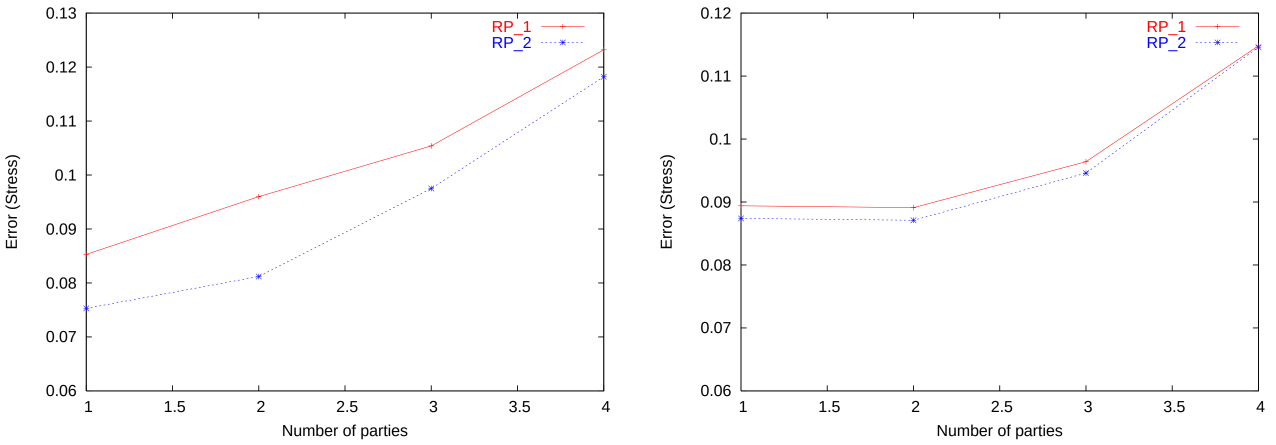

Figure 2. (A): The error produced on *mushroom* dataset. (B): The error produced on *chess* dataset.

tion the attributes of the same entities are split across the partitions.

The work presented in [17] addresses PPC over vertically partitioned data. The solution is based on secure multipart computation. Specifically, a method for k-means is proposed when different sites contain different attributes for a common set of entities. In this solution, each site learns the global clusters, but learns nothing about the attributes at other sites. This work ensures reasonable privacy while limiting communication cost.

A solution for PPC over horizontally partitioned data was proposed in [13]. This solution is based on generative models. In this approach, rather than sharing parts of the original data or perturbed data, the parameters of suitable generative models are built at each local site. Then such parameters are transmitted to a central location. The best representative of all data is a certain “mean” model. It was empirically shown that such a model can be approximated by generating artificial samples from the underlying distributions using Markov Chain Monte Carlo techniques. This approach achieves high quality distributed clustering with acceptable privacy loss and low communication cost.

The work presented here differs from the related work in some aspects: First, we address PPC over centralized data by using both OSBR and DRBT, but the later solution (DRBT) can be used to address PPC over vertically partitioned data at a reasonable communication cost, which emphasizes the generality of this solution. Second, our solutions are independent of distance-based clustering algorithms.

8. Conclusions

In this paper, we have showed analytically and experimentally that PPC by simple transformations is to some extent possible. Our study revealed that PPC can be achieved by simple and effective solutions. In particular, we showed that the challenging goal of achieving full privacy and accuracy can be accomplished by the idea of dissimilarity between objects, but at a high communication cost. We referred to this solution as Object Similarity-Based Representation (OSBR). In particular, we showed that OSBR is inefficient for PPC over vertically partitioned data when an adversary has external knowledge of some attributes subjected to clustering. As a result, OSBR is more atractive to address PPC over centralized data.

In order to alleviate the communication cost introduced by OSBR, we showed that a trade-off between privacy, accuracy, and communication costs can be accomplished by using the intuition behind random projection. We referred to the latter solution as Dimensionality Reduction-Based Transformation (DRBT). This solution is promising to either PPC over centralized or vertically partitioned data since it greatly alleviates communication costs while preserving the accuracy of reduced data as good as the accuracy of the original data.

The highlights of our approaches are as follows: a) they are independent of distance-based clustering algorithms; b) they have a sound mathematical foundation; and c) they do not require CPU-intensive operations.

The contributions in this paper can be summarized as follows: a) we demonstrated that PPC over centralized data and over vertically partitioned data can be addressed by OSBR and DRBT, respectively. Our solutions maintain the usefulness of the data and provide acceptable values in practice to address privacy concerns in clustering; b) we introduced a taxonomy of solutions to address PPC, including OSBR and DRBT.

9. Acknowledgments

Stanley R. M. Oliveira was partially supported by CNPq/Brazil, under grant No. 200077/00-7. Osmar Zaïane was partially supported by a Research Grant from NSERC, Canada. A conversation with Dr. Jörg Sander and Reza Sherkat provided initial impetus for this work. Dr. Vladimir G. Troitsky diligently checked the soundness of our mathematical proofs.

References

[1] D. Achlioptas. Database-Friendly Random Projections. In *Proc. of the 20th ACM Symposium on Principles of Database Systems*, pages 274–281, Santa Barbara, CA, USA, May 2001.

[2] J. W. Auer. *Linear Algebra With Applications*. Prentice-Hall Canada Inc., Scarborough, Ontario, Canada, 1991.

[3] E. Bingham and H. Mannila. Random Projection in Dimensionality Reduction: Applications to Image and Text Data. In *Proc. of the 7th ACM SIGKDD Cnternational Conference on Knowledge Discovery and Data Mining*, pages 245–250, San Francisco, CA, USA, 2001.

[4] C.L. Blake and C.J. Merz. UCI Repository of Machine Learning Databases, University of California, Irvine, Dept. of Information and Computer Sciences, 1998.

[5] C. Faloutsos and K.-I. Lin. FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. In *Proc. of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, CA, USA, June 1995.

[6] X. Z. Fern and C. E. Brodley. Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach. In *Proc. of the 20th International Conference on Machine Learning (ICML 2003)*, Washington DC, USA, August 2003.

[7] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. 2nd. Edition. Academic Press, 1990.

[8] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[9] H. V. Jagadish. A Retrieval Technique For Similar Shapes. In *Proc. of the 1991 ACM SIGMOD International Conference on Management of Data*, pages 208–217, Denver, Colorado, USA, May 1991.

[10] W. B. Johnson and J. Lindenstrauss. Extensions of Lipshitz Mapping Into Hilbert Space. In *Proc. of the Conference in Modern Analysis and Probability*, pages 189–206, volume 26 of Contemporary Mathematics, 1984.

[11] S. Kaski. Dimensionality Reduction by Random Mapping. In *Proc. of the International Joint Conference on Neural Networks*, pages 413–418, Anchorage, Alaska, May 1999.

[12] J. B. Kruskal and M. Wish. *Multidimensional Scaling*. Sage Publications, Beverly Hills, CA, USA, 1978.

[13] S. Meregu and J. Ghosh. Privacy-Preserving Distributed Clustering Using Generative Models. In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, pages 211–218, Melbourne, Florida, USA, November 2003.

[14] S. R. M. Oliveira and O. R. Zaïane. Privacy Preserving Clustering By Data Transformation. In *Proc. of the 18th Brazilian Symposium on Databases*, pages 304–318, Manaus, Brazil, October 2003.

[15] S. R. M. Oliveira and O. R. Zaïane. Achieving Privacy Preservation When Sharing Data For Clustering. In *Proc. of the Workshop on Secure Data Management in a Connected World (SDM'04) in conjunction with VLDB'2004*, pages 67–82, Toronto, Ontario, Canada, August 2004.

[16] C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent Semantic Indexing: A Probabilistic Analysis. In *Proc. of the 17th ACM Symposium on Principles of Database Systems*, pages 159–168, Seattle, WA, USA, June 1998.

[17] J. Vaidya and C. Clifton. Privacy-Preserving K-Means Clustering Over Vertically Partitioned Data. In *Proc. of the 9th ACM SIGKDD Intl. Conf. on Knowlegde Discovery and Data Mining*, pages 206–215, Washington, DC, USA, August 2003.