A Privacy-Preserving Clustering Approach Toward Secure and Effective Data Analysis for Business Collaboration*

Stanley R. M. Oliveira[1,2]
[1]Embrapa Informática Agropecuária
Av. André Tosello, 209
13083-886 - Campinas, SP, Brasil
oliveira@cs.ualberta.ca

Osmar R. Zaïane[2]
[2]Department of Computing Science
University of Alberta
Edmonton, AB, Canada T6G 2E8
zaiane@cs.ualberta.ca

Abstract

The sharing of data has been proven beneficial in data mining applications. However, privacy regulations and other privacy concerns may prevent data owners from sharing information for data analysis. To resolve this challenging problem, data owners must design a solution that meets privacy requirements and guarantees valid data clustering results. To achieve this dual goal, we introduce a new method for privacy-preserving clustering, called Dimensionality Reduction-Based Transformation (DRBT). This method relies on the intuition behind random projection to protect the underlying attribute values subjected to cluster analysis. The major features of this method are: a) it is independent of distance-based clustering algorithms; b) it has a sound mathematical foundation; and c) it does not require CPU-intensive operations. We show analytically and empirically that transforming a dataset using DRBT, a data owner can achieve privacy preservation and get accurate clustering with a little overhead of communication cost.

*__Note to referees:__ A preliminary version of this work appeared in the Workshop on Privacy and Security Aspects of Data Mining in conjunction with ICDM 2004, Brighton, UK, November 2004. The entire paper has been rewritten with additional detail throughout. We substantially improved the paper both theoretically and empirically to emphasize the practicality and feasibility of our approach. In addition, we introduced a new section with a methodology to evaluate the quality of the clusters generated after applying our method Dimensionality Reduction-Based Transformation (DRBT) to a dataset in which the attributes of objects are either available in a central repository or split across several sites.

Keywords: Privacy-preserving data mining, privacy-preserving clustering, dimensionality reduction, random projection, privacy-preserving clustering over centralized data, and privacy-preserving clustering over vertically partitioned data.

1 Introduction

Cluster analysis plays an outstanding role in data mining applications, such as scientific data explorations, marketing, medical diagnosis, and computational biology [4]. Apart from that, data clustering has been used extensively to find the optimal customer targets, improve profitability, market more effectively, and maximize return on investment supporting business collaboration, etc. [19]. Often, combining different data sources provides better clustering analysis opportunities. For example, it does not suffise to cluster customers based on their purchasing history, but combining purchasing history, vital statistics and other demographic and financial information for clustering purposes can lead to better and more accurate customer behaviour analysis. However, this means sharing data between parties.

Despite its benefits to support both modern business and social goals, clustering can also, in the absence of adequate safeguards, jeopardize individuals' privacy. The problem is not cluster analysis itself, but the way clustering is performed. The concern among privacy advocates is well founded, as bringing data together to support data mining projects makes misuse easier [22].

The fundamental question addressed in this paper is: *how can organizations protect personal data shared for cluster analysis and meet their needs to support decision making or to promote social benefits?* To address this problem, data owners must not only meet privacy requirements but also guarantee valid clustering results.

Clearly, achieving privacy preservation when sharing data for clustering poses new challenges for novel uses of data mining technology. Each application poses a new set of challenges. Let us consider two real-life motivating examples in which the sharing of data poses different constraints:

- Two organizations, 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?

- Suppose that a hospital shares some data for research purposes (e.g., to group patients who have a similar disease). The hospital's security administrator may suppress some identifiers (e.g., name, address, phone number, etc) from patient records to meet privacy requirements. However, the released data may not be fully protected. A patient record may contain other information that can be linked with other datasets to re-identify individuals or entities [27, 28]. How can we identify groups of patients with a similar pathology or characteristics without revealing the values of the attributes associated with them?

The above scenarios describe two different problems of privacy-preserving clustering (PPC). We refer to the former as *PPC over distributed data* and the latter as *PPC over centralized data*. Note that the first scenario is a typical example of PPC to support business collaboration, while the second relies on an application to support a social benefit. To address these scenarios, we introduce a new PPC method called Dimensionality Reduction-Based Transformation (DRBT). This method allows one to find a trade-off between privacy, accuracy, and communication cost.

Communication cost is the cost (typically in size) of the data exchanged between parties in order to achieve secure clustering.

Dimensionality reduction techniques have been studied in the context of pattern recognition [11], information retrieval [5, 9, 14], and data mining [10, 9]. To our best knowledge, dimensionality reduction has not been used in the context of data privacy in any detail. The notable exception is our preliminary work presented in [24].

One of the promising methods designed for dimensionality reduction is random projection. In this work, we use random projection to protect the underlying attribute values subjected to clustering. In tandem with the benefit of privacy preservation, our method DRBT benefits from the fact that random projection preserves the distances (or similarities) between data objects quite nicely, which is desirable in cluster analysis. We show analytically and experimentally that using DRBT, a data owner can meet privacy requirements without losing the benefit of clustering since the similarity between data points is preserved or marginally changed.

The major features of our method DRBT are: a) it is independent of distance-based clustering algorithms; b) it has a sound mathematical foundation; and c) it does not require CPU-intensive operations; and d) it can be applied to address both PPC over centralized data and PPC over vertically partitioned data.

This paper is organized as follows. In Section 2, we provide the basic concepts that are necessary to understand the issues addressed in this paper. In Section 3, we describe the research problem employed in our study. In Section 4, we introduce our method DRBT to address PPC over centralized data and over vertically partitioned data. A taxonomy of the existing PPC solutions is presented in Section 5. The experimental results are presented in Section 6. Finally, Section 7 presents our conclusions.

2 Background

In this section, we briefly review the basics of clustering, notably the concepts of data matrix and dissimilarity matrix. Subsequently, we review the basics of dimensionality reduction. In particular, we focus on the background of random projection.

2.1 Data Matrix

Objects (e.g., individuals, observations, events) are usually represented as points (vectors) in a multi-dimensional space. Each dimension represents a distinct attribute describing the object. Thus, objects are 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 may contain binary, categorical, or numerical attributes. It is referred to as a data matrix, represented as follows:

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

The attributes in a data matrix are sometimes transformed before being used. The main reason is that different attributes may be measured on different scales (e.g., centimeters and kilograms). When the range of values differs widely from attribute to attribute, attributes with large range can influence the results of the cluster analysis. For this reason, it is common to standardize the data so that all attributes are on the same scale.

There are many methods for data normalization [13]. 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. Min-max normalization maps a value v of an attribute A to v' as follows:

$$v' = \frac{v - min_A}{max_A - min_A} \times (new\_max_A - new\_min_A) + new\_min_A \tag{2}$$

where min_A and max_A represent the minimum and maximum values of an attribute A, respectively, while new_min_A and new_max_A are the new range in which the normalized data will fall.

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 (also called zero-mean 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. A value v is mapped to v' as follows:

$$v' = \frac{v - \overline{A}}{\sigma_A} \tag{3}$$

where $\overline{A}$ and σ_A are the mean and the standard deviation of the attribute A, respectively.

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 (4), 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.

$$D_M = \begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(m,1) & d(m,2) & \dots & \dots & 0 \end{bmatrix} \tag{4}$$

In general, $d(i, j)$ is a non-negative 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.

Several distance measures could be used to calculate the dissimilarity matrix of a set of points in d-dimensional space [13]. The Euclidean distance is the most popular distance measure. 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:

$$d(i, j) = \left[\sum_{k=1}^{n} |x_{ik} - x_{jk}|^2 \right]^{1/2} \tag{5}$$

The Euclidean distance satisfies the following constraints:

- $d(i, j) \geq 0$: distance is a non-negative number.
- $d(i, i) = 0$: the distance of an object to itself.
- $d(i, j) = d(j, i)$: distance is a symmetric function.
- $d(i, j) \leq d(i, k) + d(k, j)$: distance satisfies the triangular inequality.

2.3 Dimensionality Reduction

In many applications of data mining, the high dimensionality of the data restricts the choice of data processing methods. Examples of such applications include market basket data, text

classification, and clustering. In these cases, the dimensionality is large due to either a wealth of alternative products, a large vocabulary, or an expressive number of attributes to be analyzed in Euclidean space, respectively.

When 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 space. It is therefore necessary to reduce the dimensionality before, for instance, clustering the data [16, 10].

The goal of the methods designed for dimensionality reduction is to map d-dimensional objects into k-dimensional objects, where $k \ll d$ [17]. 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.

There exists a number of methods for reducing the dimensionality of data, ranging from different feature extraction methods to multidimensional scaling. The feature extraction methods are often performed according to the nature of the data, and therefore they are not generally applicable in all data mining tasks [16]. The multidimensional scaling (MDS) methods, on the other hand, have been used in several diverse fields (e.g, social sciences, psychology, market research, and physics) to analyze subjective evaluations of pairwise similarities of entities [30].

Another alternative for dimensionality reduction is to project the data onto a lower-dimensional orthogonal subspace that captures as much of the variation of the data as possible. The best and most widely way to do so is Principal Component Analysis [11]. Principal component analysis (PCA) involves a mathematical procedure that transforms a number of (possibly) correlated variables into a smaller number of uncorrelated variables called principal components. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. Unfortunately, PCA is quite expensive to compute for high-dimensional datasets.

Although the above methods have been widely used in data analysis and compression, these methods are computationally costly and if the dimensionality of the original data points is very high it is infeasible to apply these methods to dimensionality reduction.

Random projection has recently emerged as a powerful method for dimensionality reduction. The accuracy obtained after the dimensionality has been reduced, using random projection, is

almost as good as the original accuracy [16, 1, 5]. The key idea of random projection arises from the Johnson-Lindenstrauss lemma [15]: "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."

Lemma 1 *([15]). Given $\epsilon > 0$ and an integer n, let k be a positive integer such that $k \geq k_0 = O(\epsilon^{-2} log\, n)$. For every set P of n points in $\Re^d$ there exists $f : \Re^d \rightarrow \Re^k$ such that for all $u, v \in P$*

$$(1 - \epsilon) \parallel u - v \parallel^2 \leq \parallel f(u) - f(v) \parallel^2 \leq (1 + \epsilon) \parallel u - v \parallel^2.$$

The classic result of Johnson and Lindenstrauss [15] asserts that any set of n points in d-dimensional Euclidean space can be embedded into k-dimensional space, where k is logarithmic in n and independent of d.

In this work, we focus on random projection for privacy-preserving clustering. Our motivation for exploring random projection is based on the following aspects. First, it is a general data reduction technique. In contrast to the other methods, such as PCA, random projection does not use any defined interestingness criterion to optimize the projection. Second, random projection has shown to have promising theoretical properties for high dimensional data clustering [10, 5]. Third, despite its computational simplicity, random projection does not introduce a significant distortion in the data. Finally, the dimensions found by random projection are not a subset of the original dimensions but rather a transformation, which is relevant for privacy preservation. We provide the background of random projection in the next section.

2.4 Random Projection

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. $\sim N(0,1)$ distribution (i.e., zero mean and unit variance) 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 the order $O(ndk)$, and if the matrix D is sparse with about c nonzero entries per column, the complexity is of the order $O(cnk)$ [25].

After applying random projection to a dataset, 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:

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

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.

To satisfy Lemma 1, the random matrix R must hold the follow constraints:

- The columns of the random matrix R are composed of orthonormal vectors, i.e, they have unit length and are orthogonal.
- The elements r_{ij} of R have zero mean and unit variance.

Clearly, 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:

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

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 Privacy-Preserving Clustering: Problem Definition

The goal of privacy-preserving clustering is to protect the underlying attribute values of objects subjected to clustering analysis. In doing so, the privacy of individuals would be protected.

The problem of privacy preservation in clustering 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 just 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.

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. In the centralized data approach, different entities are described with the same schema in a unique centralized data repository, while in a vertical partition, the attributes of the same entities are split across the partitions. 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. In this context, the data are assumed to be a matrix $D_{m\times n}$, where each of the m rows represents an object, and each object contains values for each of the n attributes.

We assume that the matrix $D_{m\times n}$ contains numerical attributes only, and the attribute values associated with an object are private and must be protected. After transformation, the attribute values of an object in D would look very different from the original. Therefore, miners would rely on the transformed data to build valid results, i.e., clusters.

Before sharing the dataset D with party **B**, party **A** must transform D to preserve the privacy of individual data records. However, the transformation applied to D must not jeopardize the similarity between objects. Our second real-life motivating example, in Section 1, is a particular case of PPC over centralized data.

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 the first real-life example, in Section 1. Here, 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:

$$\vdash \quad \text{Party 1} \quad \dashv\vdash \quad \text{Party 2} \quad \dashv\vdash \quad \ldots \quad \dashv\vdash \quad \text{Party } k \quad \dashv$$

$$\begin{bmatrix} a_{11} \ldots a_{1i} & a_{1i+1} \ldots a_{1j} & & a_{1p+1} \ldots a_{1n} \\ \vdots & \vdots & \ldots & \vdots \\ a_{m1} \ldots a_{mi} & a_{mi+1} \ldots a_{mj} & & a_{mp+1} \ldots a_{mn} \end{bmatrix} \tag{10}$$

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.

In our model for PPC over vertically partitioned data, one of the parties is the central one which is in charge of merging the data and finding the clusters in the merged data. After finding the clusters, the central party would share the clustering results with the other parties. The challenge here is how to move the data from each party to a central party concealing the values of the attributes of each party. However, before moving the data to a central party, each party must transform its data to protect the privacy of the attribute values. We assume that the existence of an object (ID) should be revealed for the purpose of the join operation, but the values of the associated attributes are private.

3.3 The Communication Protocol

To address the problem of PPC over vertically partitioned data, we need to design a communication protocol. This protocol is used between two parties: the first party is the central one and the other represents any of the $k-1$ parties, assuming that we have k parties. We refer to the central party as $party_c$ and any of the other parties as $party_k$. There are two threads on the $party_k$ side, one for selecting the attributes to be shared, as can be seen in Table 1, and the other for selecting the objects before the sharing data, as can be seen in Table 2.

Steps to select the attributes for clustering on the $party_k$ side:
1. Negotiate the attributes for clustering before the sharing of data.
2. Wait for the list of attributes available in $party_c$.
3. Upon receiving the list of attributes from $party_c$:
a) Select the attributes of the objects to be shared.

Table 1: Thread of selecting the attributes on the $party_k$ side.

Steps to select the list of objects on the $party_k$ side:
1. Negotiate the list of m objects before the sharing of data.
2. Wait for the list of m object IDs.
3. Upon receiving the list of m object IDs from $party_c$:
a) Select the m objects to be shared;
b) Transform the attribute values of the m objects;
c) Send the transformed m objects to $party_c$.

Table 2: Thread of selecting the objects on the $party_k$ side.

4 The Dimensionality Reduction-Based Transformation

In this section, we show that the triple-goal of achieving privacy preservation and valid clustering results at a reduced communication cost in PPC can be accomplished by dimensionality reduction. By reducing the dimensionality of a dataset to a sufficiently small value, one can find a trade-off between privacy, accuracy, and communication cost. In particular, random project can fulfill this triple-goal. We refer to this solution as the Dimensionality Reduction-Based Transformation (DRBT).

4.1 General Assumptions

The solution to the problem of PPC based on random projection 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 the data sharing for clustering occurs.
- In PPC over centralized data, the existence of an object (ID) should be replaced by a fictitious identifier. In PPC over vertically partitioned data, the IDs 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, accuracy, and communication cost.

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 a *disguised database*, which is shared for clustering.

4.2 PPC over Centralized Data

To address PPC over centralized data, the 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 dimension 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 determine whether 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 using Equation (6).

To illustrate how this solution works, let us consider the sample relational database in Table 3. This sample contains real data from the Cardiac Arrhythmia Database available at the UCI Repository of Machine Learning Databases [6]. 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 3: A cardiac arrhythmia database.

We are going to reduce the dimension of this 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 a random matrix

R_1 by setting each entry of the matrix to a value drawn from an independent and identically distributed (i.i.d.) $N(0,1)$ distribution and then normalizing the columns to unit length. We also compute a 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. 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 4 shows the values of the error using RP_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	0.2425
RP_2	0.0000	0.0281	0.0375	0.1120

Table 4: 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, the transformed dataset has only 50% of the dimensions in the original dataset. Note that the error is relatively small for both RP_1 and RP_2, especially for RP_2. However, this error is minimized when the random projection is applied to high dimensional datasets, as can be seen in Figure 2, in Section 6.3.

After applying random projection to a dataset, the attribute values of the transformed dataset are completely disguised to preserve the privacy of individuals. Table 5 shows the attribute values of the transformed database with 3 dimensions, using both RP_1 and RP_2. In this table, we have the attributes labeled *Att1*, *Att2*, and *Att3* since we do not know the labels for the disguised dataset. Using random projection, one cannot select the attributes to be reduced beforehand. The attributes are reduced randomly. More formally, $\forall i$ if $Attr_i \in D'$, then $Attr_i \notin D$.

ID	D' using RP_1			D' using RP_2		
	Att1	Att2	Att3	Att1	Att2	Att3
123	-50.40	17.33	12.31	-55.50	-95.26	-107.96
342	-37.08	6.27	12.22	-51.00	-84.29	-83.13
254	-55.86	20.69	-0.66	-65.50	-70.43	-66.97
446	-37.61	-31.66	-17.58	-85.50	-140.87	-72.74
286	-62.72	37.64	18.16	-88.50	-50.22	-102.76

Table 5: Disguised dataset D' using RP_1 and RP_2.

As can be seen in Table 5, the attribute values are entirely different from those in Table 3.

4.3 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.

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

- *Agreement*: The k parties must follow the communication protocol described in Section 3.3.
- *Mutual exclusivity*: We assume that the attribute split across the k parties are 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 only exception is that IDs are shared for the join purpose.

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 4.2.
- *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 themselves. 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.

4.4 How Secure is the DRBT?

In the previous sections, we showed 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 demonstrate that a random projection from d dimensions to k, where $k \ll d$, is a non-invertible transformation.

Lemma 2 *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 [3]. 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. □

When a set of points in a high dimensional space are mapped onto a lower dimensional space by random projection, the coordinates of the points in the low space are completely disguised. In other words, there is no means to reconstruct the coordinates of the points in the original space based on the coordinates of the points in the low dimensional space. Therefore, a data owner would not give away the original points. The only useful information preserved in the lower dimensional space are the distances between the points but with a relatively small error that is acceptable for practical applications. That is the basis of privacy preservation of the DRBT.

4.5 The Accuracy of the DRBT

When using random projection, a perfect reproduction of the Euclidean distances may not be the best possible result. The clusters in the transformed datasets should be equal to those in the original database. However, this is not always the case, and we have some potential problems after dimensionality reduction: a) a noise data point ends up clustered; b) a point from a cluster becomes a noise point; and c) a point from a cluster migrates to a different cluster. In this research, we focus primarily on partitioning methods. In particular, we use K-means [20], one the most used clustering algorithms. Since K-means is sensitive to noise points and clusters all the points in a dataset, we have to deal with the third problem mentioned above (a point from a cluster migrates to a different cluster).

Our evaluation approach focuses on the overall quality of generated clusters after dimensionality reduction. We compare how closely each cluster in the transformed data matches its corresponding cluster in the original dataset. To do so, we first identify the matching of clusters by computing the matrix of frequencies showed in Table 6. We refer to such a matrix as the clustering membership matrix (CMM), where the rows represent the clusters in the original dataset, the columns represent the clusters in the transformed dataset, and $freq_{i,j}$ is the number of points in cluster c_i that falls in cluster c'_j in the transformed dataset.

After computing the frequencies $freq_{i,j}$, we scan the clustering membership matrix calculating precision, recall, and F-measure for each cluster c'_j with respect to c_i in the original dataset [18]. These formulas are given by the following equations:

	c'_1	c'_2	...	c'_k
c_1	$freq_{1,1}$	$freq_{1,2}$	...	$freq_{1,k}$
c_2	$freq_{2,1}$	$freq_{2,2}$	...	$freq_{2,k}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_k	$freq_{k,1}$	$freq_{k,2}$	...	$freq_{k,k}$

Table 6: The number of points in cluster c_i that falls in cluster c'_j in the transformed dataset.

$$Precision\ (P) = \frac{freq_{i,j}}{|c'_i|} \tag{11}$$

$$Recall\ (R) = \frac{freq_{i,j}}{|c_i|} \tag{12}$$

$$F - measure\ (F) = \frac{2 \times P \times R}{(P + R)} \tag{13}$$

where $|X|$ is the number of points in the cluster X.

For each cluster c_i, we first find a cluster c'_j that has the highest F-measure among all the c'_l, $1 \leq l \leq k$. Let $F(c_i)$ be the highest F-measure for cluster c_i, we denote the overall F-measure (OF) as the weighted average of $F(c_i)$, $1 \leq i \leq k$, as follows:

$$OF = \frac{\sum_{i=1}^{k} |c_i| \times F(c_i)}{\sum_{i=1}^{k} |c_i|} \tag{14}$$

In section 6, we present our performance evaluation results for clustering based on Equation (14).

4.6 The Complexity of the DRBT

One of the major benefits of a solution that adheres to the DRBT is the communication cost to send a disguised dataset from one party to a central one. 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).

The complexity of DRBT is of the order $O(m \times k)$, however $k \ll m$.

To quantify the 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.

5 A Taxonomy of PPC Solutions

Some effort has been made to address the problem of PPC. In this section, we present a taxonomy of the existing solutions. We categorize these solutons into two major groups: *PPC over centralized data* and *PPC over distributed data*. In the former approach, different objects are described with the same schema in a unique centralized data repository, while in the latter approach, either the attributes or the records of objects are split across many partitions.

5.1 Solutions for PPC Over Centralized Data

Methods for PPC over centralized data are categorized into two groups: *Attribute Masking* and *Pairwise Object Similarity*.

Attribute Value Masking: This data transformation makes the original attribute values difficult to perceive or understand and preserves all the information for clustering analysis. Our data transformation that falls into this category is called Rotation-Based Transformation (RBT) [23]. The idea behind this technique is that the attributes of a database are split into pairwise attributes selected randomly. One attribute can be selected and rotated more than once, and the angle θ between an attribute pair is also selected randomly. RBT can be seen as a technique bording obfuscation. Obfuscation techniques aim at making information highly illegible without actually changing its inner meaning [7]. In other words, using RBT the original data are masked so that the transformed data capture all the information for clustering analysis while protecting the underlying data values. One interesting application of RBT is privacy preservation of health data [2].

Pairwise Object Similarity: This technique is a data matrix representation in which a data owner shares the distance between data objects instead of the location of the data points. This technique relies on the idea of the similarity between objects, i.e., a data owner shares some data for clustering analysis by simply computing the dissimilarity matrix (matrix of distances) between the objects and then sharing such a matrix with a third party [24]. This solution is simple to implement and addresses PPC over centralized data. One of the

most important advantages of this solution is that it can be applied to either categorical, binary, numerical attributes, or even a combination of these attributes. On the other hand, this solution can sometimes be restrictive since it requires a high communication cost.

5.2 Solutions for PPC Over Distributed Data

Regarding PPC over distributed data, we classify the existing solutions in two groups: *PPC over vertically partitioned data* and *PPC over horizontally partitioned data*.

Vertically Partitioned Data: In a vertical partition approach, the attributes of the same objects are split across the partitions. The idea behind this solution is that two or more parties want to conduct a computation based on their private inputs. The issue here is how to conduct such a computation so that no party knows anything except its own input and the results. This problem is referred to as the secure multi-party computation problem [12, 26, 8]. The existing solution that falls in this category was introduced in [29]. Specifically, a method for k-means was 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.

Horizontally Partitioned Data: In a horizontal partition approach, different objects are described with the same schema in all partitions. A solution for PPC over horizontally partitioned data was proposed in [21]. 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.

5.3 Attribute Reduction

This is the approach presented in this paper: the attributes of a database are reduced to a smaller number. The small number of attributes is not a subset of the original attributes since the transformation disguises the original attribute values by projecting them onto a random space. Our data transformation that lies in this category is called Dimensionality Reduction-Based Transformation (DRBT) [24]. This data transformation can be applied to both PPC

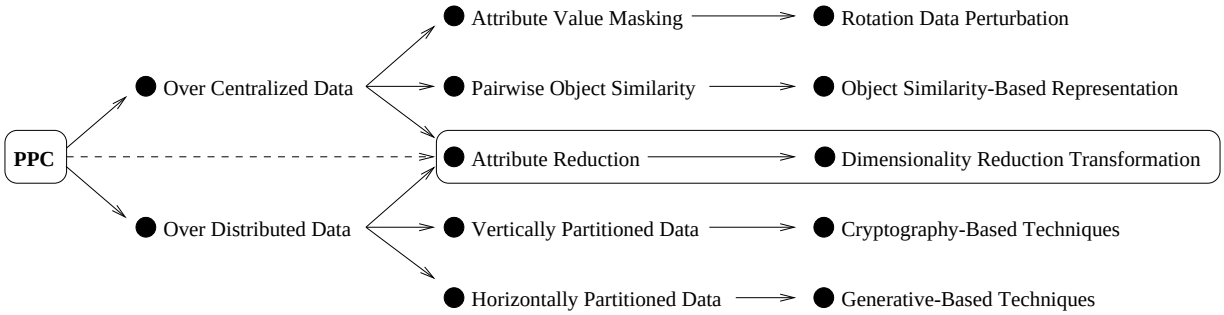


Figure 1: A taxonomy of PPC solutions.

over centralized data and PPC over vertically partitioned data. The idea behind this data transformation is that by reducing the dimensionality of a database to a sufficiently small value, one can find a trade-off between privacy and accuracy. Once the dimensionality of a database is reduced, the released database preserves (or slightly modifies) the distances between data points. In tandem with the benefit of preserving the similarity between points, this solution protects individuals' privacy since the underlying data values of the objects subjected to clustering are completely different from the original ones.

As can be seen in Figure 1, the only solution to address both PPC over centralized data and PPC over distributed data is the DRBT.

6 Experimental Results

In this section, we empirically validate our method DRBT. We start by describing the real datasets used in our experiments. We then describe the methodology used to validate our method. Subsequently, we study the effectiveness of our method to address PPC over centralized data and PPC over vertically partitioned data. We conclude this section discussing the main lessons learned from our experiments.

6.1 Datasets

We validated our method DRBT for privacy-preserving clustering using five real datasets. These datasets are described as follows:

1. **Accidents:** This dataset concerning traffic accidents was obtained from the National Institute of Statistics (NIS) for the region of Flanders in Belgium. The transactions are traffic accident forms filled out by police officers for each traffic accident that occurred involving injuries or deaths on a public road in Belgium. There are 340,183 traffic accident records included in the dataset. We used 18 columns of this dataset after removing missing values.

2. **Mushroom:** This dataset is available at the UCI Repository of Machine Learning Databases [6]. Mushroom contains records drawn from The Audubon Society Field Guide to North American Mushrooms. There are 8,124 records and 23 numerical attributes.

3. **Chess:** The format for instances in this database is a sequence of 37 attribute values. Each instance is a board-descriptions of a chess endgame. The first 36 attributes describe the board. The last (37th) attribute is the classification: "win" or "nowin". Chess is available at the UCI Repository of Machine Learning Databases [6] and contains 3,196 records. There is no missing value in this dataset.

4. **Connect:** This database contains all legal 8-ply positions in the game of connect-4 in which neither player has won yet, and in which the next move is not forced. Connect is composed of 67,557 records and 43 attributes without missing values. This dataset is also available at the UCI Repository of Machine Learning Databases [6].

5. **Pumsb:** The Pumsb dataset contains census data for population and housing. This dataset is available at http://www.almaden.ibm.com/software/quest. There are 49,046 records and 74 attribute values without missing values.

Table 7 shows the summary of the datasets used in our experiments. The columns represent, respectively, the database name, the total number of records, and the number of attributes in each dataset.

Dataset	#records	# attributes
Accidents	340,183	18
Mushroom	8,124	23
Chess	3,196	37
Connect	67,557	43
Pumsb	49,046	74

Table 7: A summary of the datasets used in our experiments

6.2 Methodology

We performed two series of experiments to evaluate the effectiveness of DRBT when addressing PPC over centralized data and PPC over vertically partitioned data. Our evaluation approach focused on the overall quality of generated clusters after dimensionality reduction. One question that we wanted to answer was:

What is the quality of the clustering results mined from the transformed data when the data are both sparse and dense?

Our performance evaluation was carried out through the following steps:

- *Step 1*: we normalized the attribute values of the five real datasets used in our experiments. To do so, we used the z-score normalization given in Equation (3). The results presented in the next sections were obtained after normalization.

- *Step 2*: we considered random projection based on two different approaches. First, the traditional way to compute random projection, by setting each entry of the random matrix R_1 to a value drawn from an i.i.d. $N(0,1)$ distribution and then normalizing the columns to unit length. Second, we used the random matrix R_2 where each element r_{ij} is computed using Equation (9). We refer to the former random projection as RP_1 and the latter as RP_2. We repeated each experiment (for random projection) 5 times. In the next section, we present results by showing only the average value.

- *Step 3*: 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). The stress function was computed for each dataset.

- *Step 4*: we selected K-means to find the clusters in our performance evaluation. Our selection was influenced by the following aspects: (a) K-means is one of the best known clustering algorithm and is scalable; (b) When using random projection, a perfect reproduction of the Euclidean distances may not be the best possible result. However, the rank order of the distances between the vectors is meaningful. Thus, when running K-means over the transformed data, one can find the clusters that would be mined from the original datasets with a reasonable accuracy.

- *Step 5*: we compared how closely each cluster in the transformed dataset matches its corresponding cluster in the original dataset. We expressed the quality of the generated clusters by computing the F-measure given in Equation (14). Considering that K-means is not deterministic (due to its use of random seed selection), we repeated each experiment 10 times. We then computed the minimum, average, maximum, and standard deviation for each measured value of the F-measure. We present the results by showing only the average value.

We should point out that the steps described above were performed to evaluate the effectiveness of the DRBT when addressing PPC over centralized and vertically partitioned data.

6.3 Measuring the Effectiveness of the DRBT over Centralized Data

To measure the effectiveness of DRBT in PPC over centralized data, we started by computing the relative error that the distances in d-k space suffer from, on the average. To do so, we used the two random projection approaches (RP_1 and RP_2) mentioned in Step 3 of Section 6.2.

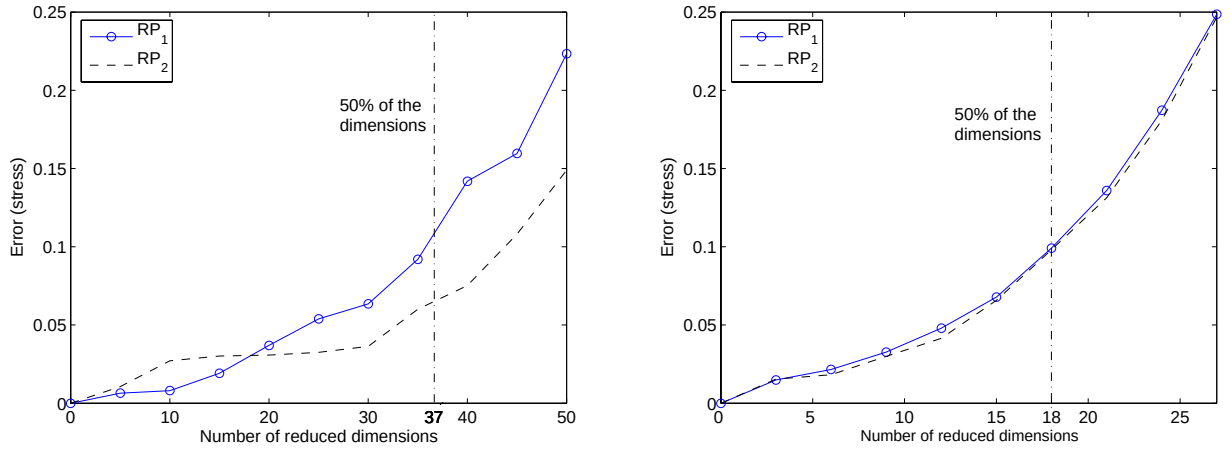
A word of notation: hereafter we denote the original dimension of a dataset as d_o and reduced dimension of the transformed dataset as d_r. This notation is to avoid confusion between the reduced dimension of a dataset (k) and the number of clusters used as input of the algorithm K-means.

An important feature of the DRBT is its versatility to trade privacy, accuracy, and communication cost. The privacy preservation is assured because random projection is a non-invertible transformation, as discussed in Section 4.4. We here study the trade-off between accuracy and communication cost. The accuracy is represented by the error that the distances in d_o-d_r space suffer from, while the communication cost is represented by the number of dimensions that we reduce in the datasets. We selected two datasets: Pumsb and Chess with 74 and 37 dimensions, respectively. We reduced the dimensions of these datasets and computed the error. Figure 2(a) shows the error produced by RP_1 and RP_2 on the dataset Pumsb and Figure 2(b) shows the error produced by RP_1 and RP_2 on the dataset Chess. These results represent the average value of five trials. The error produced by RP_1 and RP_2 on the other datasets are available at Appendix A.

We observed that, in general, RP_2 yielded the best results in terms of the error produced on the datasets (the lower the better). In the dataset Chess the difference between RP_2 and RP_1 was not significant. These results confirm the same findings in [5] and backup the theory of random projection (the choice of the random matrix) proposed in [1]. We noticed from the figures that the DRBT trades well accuracy (error) and communication cost (number of reduced dimensions) when the data are reduced up to 50% of the dimensions. In this case, the trade-off between the error and the communication cost is linear. However, reducing more than 50% of the dimensions, the communication cost is improved but the accuracy is compromised since the error produced on the datasets grows faster. Therefore, a data owner should consider carefully this trade-off before releasing some data for clustering.

After evaluating the error produced on the datasets, we used the algorithm K-means to find the clusters in the original and transformed datasets. We varied the number of clusters from 2 to 5 in the five datasets. Subsequently, we compared how closely each cluster in the transformed dataset matches its corresponding cluster in the original dataset by computing the F-measure given in Equation (14).

Table 8 shows the results of the F-measure for the Accidents dataset. We reduced the original 18 dimensions to 12. We repeated each experiment 10 times and computed the minimum,



(a) (b)

Figure 2: (a) The error produced on the dataset *Pumsb* ($d_o = 74$); (b) The error produced on the dataset *Chess* ($d_o = 37$).

average, maximum, and standard deviation for each measured value of the F-measure. We simplify the results by showing only one dataset (Accidents). The values of the F-measure for the other datasets can be found in Appendix B. Note that we computed the values of the F-measure only for the random projection RP_2 since its results were slightly better than those yielded by RP_1.

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.931	0.952	0.941	0.014	0.903	0.921	0.912	0.009

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.870	0.891	0.881	0.010	0.878	0.898	0.885	0.006

Table 8: Average of the F-measure (10 trials) for the Accidents dataset ($d_o = 18, d_r = 12$).

We noticed that the values of the F-measure for the Chess and Connect datasets (see Appendix B) were relatively low when compared with the results of the F-measure for the other datasets. The main reason is that the data points in these datasets are densely distributed. Thus, applying a partitioning clustering algorithm (e.g., K-means) to datasets of this nature increases the number of misclassified data points. On the other hand, when the attribute values of the objects are sparsely distributed, the clustering results are much better. Consider, for example, the Iris dataset available at the UCI Repository of Machine Learning Databases [6]. Iris is perhaps the best known database to be found in the pattern recognition literature. This

dataset has two clusters well defined and the data are sparsely distributed. We reduced the original 5 dimensions to 3. Then we applied random projection RP_2 to the Iris dataset and computed the minimum, average, maximum, and standard deviation for each measured value of the F-measure. We repeated each experiment 10 times. Table 9 shows that the standard deviation for two clusters ($k = 2$) was zero and the average of the F-measure was one.

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	1.000	1.000	1.000	0.000	0.094	0.096	0.948	0.010

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.773	0.973	0.858	0.089	0.711	0.960	0.833	0.072

Table 9: Average of the F-measure (10 trials) for the Iris dataset ($d_o = 5, d_r = 3$).

6.4 Measuring the Effectiveness of the DRBT over Vertically Partitioned Data

Now we move on to measure the effectiveness of DRBT to address PPC over vertically partitioned data. To do so, we split the Pumsb dataset (74 dimensions) from 1 up to 4 parties (partitions) and fixed the number of dimensions to be reduced (38 dimensions). Table 10 shows the number of parties, the number of attributes per party, and the number of attributes in the merged dataset which is subjected to clustering. Recall that in a vertically partitioned data approach, one of the parties will centralize the data before mining.

No. of parties	No. of attributes per party	No. of attributes in the merged dataset
1	1 partition with 74 attributes	38
2	2 partitions with 37 attributes	38
3	2 partitions with 25 and 1 with 24 attributes	38
4	2 partitions with 18 and 2 with 19 attributes	38

Table 10: An example of partitioning for the Pumsb dataset.

In this example, each partition with 37, 25, 24, 19, and 18 attributes was reduced to 19, 13, 12, 10, and 9 attributes, respectively. We applied the random projections RP_1 and RP_2 to each partition and then merged the partitions in one central repository. Subsequently, we

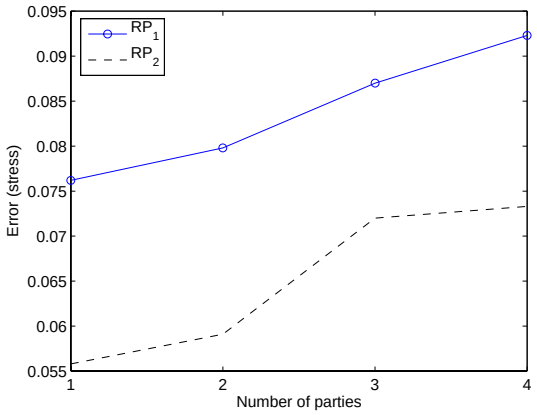


Figure 3: The error produced on the dataset *Pumsb* over vertically partitioned data.

computed the stress error on the merged dataset and compared the error with that one produced on the original dataset (without partitioning). Figure 3 shows the error produced on the Pumsb dataset in the vertically partitioned data approach. As we can see, the results yielded by RP_2 were again slightly better than those yielded by RP_1.

Note that we reduced approximately 50% of the dimensions of the dataset Pumsb and the trade-off between accuracy and communication cost is still efficient for PPC over vertically partitioned data.

We also evaluated the quality of clusters generated by mining the merged dataset and comparing the clustering results with those mined from the original dataset. To do so, we computed the F-measure for the merged dataset in each scenario, i.e., from 1 up to 4 parties. We varied the number of clusters from 2 to 5. Table 11 shows values of the F-measure (average and standard deviation) for the Pumsb dataset over vertically partitioned data. These values represent the average of 10 trials considering the random projection RP_2.

No. of parties	k = 2		k = 3		k = 4		k = 5	
	Avg	Std	Avg	Std	Avg	Std	Avg	Std
1	0.909	0.140	0.965	0.081	0.891	0.028	0.838	0.041
2	0.904	0.117	0.931	0.101	0.894	0.059	0.840	0.047
3	0.874	0.168	0.887	0.095	0.873	0.081	0.801	0.073
4	0.802	0.155	0.812	0.117	0.866	0.088	0.831	0.078

Table 11: Average of the F-measure (10 trials) for the Pumsb dataset over vertically partitioned data.

We notice from Table 11 that the results of the F-measure slightly decrease when we increase the number of parties in the scenario of PPC over vertically partitioned data. Despite this fact,

the DRBT is still effective to address PPC over vertically partitioned data in preserving the quality of the clustering results as measured by F-measure.

6.5 Discussion on the DRBT When Addressing PPC

The evaluation of the DRBT involves three important issues: security, communication cost, and quality of the clustering results. We discussed the issues of security in Section 4.4 based on Lemma 2, and the issues of communication cost and space requirements in Section 4.6. In this Section, we have focused on the quality of the clustering results.

We have evaluated our proposed data transformation method (DRBT) to address PPC. We have learned some lessons from this evaluation, as follows:

- *The application domain of the DRBT*: we observed that the DRBT does not present acceptable clustering results in terms of accuracy when the data subjected to clustering are dense. Slightly changing the distances between data points by random projection results in misclassification, i.e., points will migrate from one cluster to another in the transformed dataset. This problem is somehow understandable since partitioning clustering methods are not effective to find clusters in dense data. The Connect dataset is one example which confirms this finding. On the other hand, our experiments demonstrated that the quality of the clustering results obtained from sparse data is promising.

- *The versatility of the DRBT*: using the DRBT, 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. Most importantly, the DRBT can be used to address PPC over centralized and vertically partitioned data.

- *The choice of the random matrix*: from the performance evaluation of the DRBT we noticed that the random projection RP_2 yielded the best results for the error produced on the datasets and the values of F-measure, in general. The random projection RP_2 is based on the random matrix proposed in Equation (9).

7 Conclusions

In this paper, we have showed analytically and experimentally that Privacy-Preserving Clustering (PPC) is to some extent possible. To support our claim, we introduced a new method to address PPC over centralized data and over vertically partitioned data, called the Dimensionality Reduction-Based Transformation (DRBT). Our method was designed to support business collaboration considering privacy regulations, without losing the benefit of data analysis. The

DRBT relies on the idea behind random projection to protect the underlying attribute values subjected to clustering. Random projection has recently emerged as a powerful method for dimensionality reduction. It preserves distances between data objects quite nicely, which is desirable in cluster analysis.

We evaluated the DRBT taking into account three important issues: security, communication cost, and accuracy (quality of the clustering results). Our experiments revealed that using DRBT, a data owner can meet privacy requirements without losing the benefit of clustering since the similarity between data points is preserved or marginally changed. From the performance evaluation, we suggested guidance on which scenario a data owner can achieve the best quality of the clustering when using the DRBT. In addition, we suggested guidance on the choice of the random matrix to obtain the best results in terms of the error produced on the datasets and the values of F-measure.

The highlights of the DRBT are as follows: a) it is independent of distance-based clustering algorithms; b) it has a sound mathematical foundation; c) it does not require CPU-intensive operations; and d) it can be applied to address PPC over centralized data and PPC over vertically partitioned data.

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] M. P. Armstrong, G. Rushton, and D. L. Zimmerman. Geographically Masking Health Data to Preserve Confidentiality. *Statistics in Medicine*, 18:497–525, 1999.

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

[4] M. Berry and G. Linoff. *Data Mining Techniques - for Marketing, Sales, and Customer Support*. John Wiley and Sons, New York, USA, 1997.

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

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

[7] C. Collberg, C. Thomborson, and D. Low. A Taxonomy of Obfuscating Transformations. Technical report, TR–148, Department of Computer Science, University of Auckland, New Zealand, July 1997.

[8] W. Du and M. J. Atallah. Secure Multi-Party Computation Problems and their Applications: A Review and Open Problems. In *Proc. of 10th ACM/SIGSAC 2001 New Security Paradigms Workshop*, pages 13–22, Cloudcroft, New Mexico, September 2001.

[9] 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.

[10] 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.

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

[12] O. Goldreich, S. Micali, and A. Wigderson. How to Play Any Mental Game - A Completeness Theorem for Protocols with Honest Majority. In *Proc. of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, USA, May 1987.

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

[14] 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.

[15] 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.

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

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

[18] B. Larsen and C. Aone. Fast and Effective Text Mining Using Linear-Time Document Clustering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 16–22, San Diego, CA, USA, August 1999.

[19] V. S. Y. Lo. The True Lift Model - A Novel Data Mining Approach to Response Modeling in Database Marketing. *SIGKDD Explorations*, 4(2):78–86, December 2002.

[20] J. Macqueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley: University of California Press, Vol. 1, 1967.

[21] 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.

[22] S. R. M. Oliveira. *Data Transformation For Privacy-Preserving Data Mining*. PhD thesis, Department of Computing Science, University of Alberta, Edmonton, AB, Canada, December 2004.

[23] 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.

[24] S. R. M. Oliveira and O. R. Zaïane. Privacy-Preserving Clustering by Object Similarity-Based Representation and Dimensionality Reduction Transformation. In *Proc. of the Workshop on Privacy and Security Aspects of Data Mining (PSADM'04) in conjunction with the Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 21–30, Brighton, UK, November 2004.

[25] 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.

[26] B. Pinkas. Cryptographic Techniques For Privacy-Preserving Data Mining. *SIGKDD Explorations*, 4(2):12–19, December 2002.

[27] P. Samarati. Protecting Respondents' Identities in Microdata Release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[28] L. Sweeney. k-Anonymity: A Model for Protecting Privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[29] 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.

[30] F. W. Young. *Multidimensional Scaling*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1987.

A Results of the Stress Function Applied to the Datasets

Chess	$d_r = 37$	$d_r = 34$	$d_r = 31$	$d_r = 28$	$d_r = 25$	$d_r = 22$	$d_r = 16$
RP_1	0.000	0.015	0.024	0.033	0.045	0.072	0.141
RP_2	0.000	0.014	0.019	0.032	0.041	0.067	0.131

Table 12: The error produced on the *Chess* dataset ($d_o = 37$).

Mushroom	$d_r = 23$	$d_r = 21$	$d_r = 19$	$d_r = 17$	$d_r = 15$	$d_r = 13$	$d_r = 9$
RP_1	0.000	0.020	0.031	0.035	0.048	0.078	0.155
RP_2	0.000	0.017	0.028	0.029	0.040	0.079	0.137

Table 13: The error produced on the *Mushroom* dataset ($d_o = 23$).

Pumsb	$d_r = 74$	$d_r = 69$	$d_r = 64$	$d_r = 59$	$d_r = 49$	$d_r = 39$	$d_r = 29$
RP_1	0.000	0.006	0.022	0.029	0.049	0.078	0.157
RP_2	0.000	0.007	0.030	0.030	0.032	0.060	0.108

Table 14: The error produced on the *Pumsb* dataset ($d_o = 74$).

Connect	$d_r = 43$	$d_r = 37$	$d_r = 31$	$d_r = 25$	$d_r = 19$	$d_r = 16$	$d_r = 13$
RP_1	0.000	0.016	0.037	0.063	0.141	0.159	0.219
RP_2	0.000	0.016	0.028	0.062	0.122	0.149	0.212

Table 15: The error produced on the *Connect* dataset ($d_o = 43$).

Accidents	$d_r = 18$	$d_r = 16$	$d_r = 14$	$d_r = 12$	$d_r = 10$	$d_r = 8$	$d_r = 6$
RP_1	0.000	0.033	0.034	0.044	0.094	0.144	0.273
RP_2	0.000	0.018	0.023	0.036	0.057	0.108	0.209

Table 16: The error produced on the *Accidents* dataset ($d_o = 18$).

B Results of F-measure for the Clusters Mined from the Transformed Datasets

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.529	0.873	0.805	0.143	0.592	0.752	0.735	0.050
Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.597	0.770	0.695	0.063	0.569	0.761	0.665	0.060

Table 17: Average of F-measure (10 trials) for the Chess dataset ($d_o = 37, d_r = 25$).

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.972	0.975	0.974	0.001	0.689	0.960	0.781	0.105
Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.727	0.864	0.811	0.058	0.747	0.884	0.824	0.051

Table 18: Average of F-measure (10 trials) for the Mushroom dataset ($d_o = 23, d_r = 15$).

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.611	0.994	0.909	0.140	0.735	0.991	0.965	0.081
Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.846	0.925	0.891	0.028	0.765	0.992	0.838	0.041

Table 19: Average of F-measure (10 trials) for the Pumsb dataset ($d_o = 74, d_r = 38$).

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.596	0.863	0.734	0.066	0.486	0.863	0.623	0.103
Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.618	0.819	0.687	0.069	0.572	0.763	0.669	0.069

Table 20: Average of F-measure (10 trials) for the Connect dataset ($d_o = 43, d_r = 28$).