Secure Third Party Data Clustering Using SecureCL, Φ-Data and Multi-User Order Preserving Encryption

Nawal Almutairi[1], Frans Coenen[2], and Keith Dures[2]

[1] Information Technology Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia
nawalmutairi@ksu.edu.sa

[2] Department of Computer Science, University of Liverpool, Liverpool, UK
{coenen,dures}@liverpool.ac.uk

Abstract. Secure collaborative data clustering using SecureCL is presented. SecureCL is founded on the concept of Φ-data implemented using Super Secure Chain Distance Matrices and encrypted using Multi-User Order Preserving Encryption. The advantage offered, unlike comparable systems, is that SecureCL does not require any user participation once the Φ-data proxy has been encrypted; it does not require recourse to Secure Multi-Party Computation protocols or "secret sharing" mechanisms. The utility of SecureCL is illustrated using Nearest Neighbour Clustering and DBSCAN, although it can be applied to any data clustering algorithm that involves distance comparison. The reported experiments demonstrate that SecureCL can produce securely cluster configurations comparable to those produced using standard, non-encrypted, approaches without entailing any significant computational overhead, thus indicating its suitability in the context of Data Mining as a Service.

Keywords: Cryptography, Privacy, Nearest neighbour, Unsupervised learning, Data mining.

1 Introduction

The increasing demand from data owners for the techniques of data analytics to be applied to their data has led to the emergence of Data Mining as a Service (DMaaS) where a third party conducts the data analysis, using cloud computing facilities, on behalf of the data owner. Examples include Microsoft Azure's Machine Learning Studio and Google Cloud's Machine Learning Engine. The advantages offered are that data owners are liberated from in-house data management and in-house data analysis expertise. DMaaS also provides an opportunity for collaborative data mining which enables analysis over large datasets, supplied by a number of data owners, so as to gain some mutual advantage. However, issues concerning data confidentiality, and associated data privacy preservation concerns, have served to limit the adoption of the collaborative data mining opportunities facilitated by DMaaS platforms, especially in fields where data confidentiality is a legal requirement (Singh & Chatterjee, 2017).

An early solution to the collaborative data mining privacy issues identified above was the research domain of Privacy Preserving Data Mining (PPDM) (Agrawal & Srikant, 2000). The idea was to calculate statistical features, required by data mining algorithms, over data distributed across multiple data owners without sharing any of the source data and without recourse to a third party. The solutions were founded on Secure Multi-Party Computation (SMPC) protocols (Goldreich, 1998). The fundamental idea was for individual data owners to locally process their data to produce local statistical features that could be shared, using a SMPC protocol, to derive global statistical features that could then be used to build a global data model of some kind. However, the significant computation and communication overhead associated with many SMPC protocols required considerable data owner resource, thus rendering PPDM infeasible for large scale collaborative data mining. In addition, involving the data owners as the data mining progressed was seen as providing the potential for non-honest parties to launch "overlapping attacks" (J. Liu, Xiong, Luo, & Huang, 2013).

An alternative solution to the collaborative data mining privacy issue was to maintain data privacy using obfuscation methods and delegate the data analysis to a third party data miner. The data obfuscation could be applied either to selective sensitive data attributes, referred to as "data anonymisation" (Samarati, 2001); or to the entire dataset, referred to as "data perturbation" (L. Liu, Kantarcioglu, & Thuraisingham, 2008). Using data anonymisation, sensitive data attributes are removed and the remaining data irreversibly generalised so that the probability of identifying individuals is minimised. However, the study in (Narayanan & Shmatikov, 2008) shows that data cannot be 100% anonymised. Using data perturbation, values associated with individual data attributes are distorted by adding noise in such a way that the statistical makeup of the dataset is maintained. However, the requirement for adopting the same perturbation method across all parties when conducting collaborative data mining makes the solution vulnerable to security breaches (Anikin & Gazimov, 2017). Obfuscation methods have also been shown to adversely affect the quality of data analysis (L. Liu et al., 2008).

Data encryption can provide a substantial guarantee for data privacy (Goldreich, 2004). Property Preserving Encryption schemes, such as Homomorphic Encryption (HE) and Order-Preserving Encryption (OPE), may therefore provide potential solution to the collaborative data mining privacy issue in that they support primitive operations over cyphertexts without decryption. The precise nature of the operations supported is dependent on the nature of the adopted scheme; however, there is no HE or OPE scheme that provides an entire solution. For example, HE schemes do not support record comparison, whilst OPE schemes do not support any mathematical operations other than the accumulation of data order indicators. Proposed mechanisms used to conduct unsupported operations include: recourse to SMPC protocols (Yao, 1982) as in case of (J. Liu et al., 2013), or recourse to data owner participation as in the case of

(Rahman, Basu, & Kiyomoto, 2017). In both cases an undesirable computation and/or communication overhead is introduced while allowing the potential for non-honest party attacks.

This paper presents a cryptographic solution to secure collaborative data mining that does not feature the disadvantages associated with established Property Preserving Encryption schemes. More specifically this paper presents SecureCL, a mechanism to support secure collaborative data clustering by a third party (a DMaaS provider). The fundamental idea underpinning SecureCL is the Φ-data concept, introduced in (Almutairi, Coenen, & Dures, 2018a), whereby data mining is conducted using an alternative encrypted proxy of the data. The nature of the adopted proxy depends on the nature of the data mining to be applied, and should be such that no recourse to data owner participation or SMPC protocols are required. The data proxy proposed in (Almutairi et al., 2018a) was the Super Secure Chain Distance Matrix (SSCDM) and was also directed at data clustering. However, the work presented in (Almutairi et al., 2018a) only considered data that was either horizontally or vertically partitioned, and focused on DBSCAN only. The work presented here also uses encrypted SSCDM, as the Φ-data proxy, but the associated SecureCL mechanism operates using arbitrary (mixed) data partitioning. The SecureCL mechanism is fully described and extensively evaluated with respect to its scalability and effectiveness. For completeness, the SSCDM proxy is also described together with the bespoke encryption scheme, Multi-User OPE (MUOPE), with which the SSCDM proxy is encrypted. The reported evaluation demonstrates that when using SecureCL: (i) SSCDMs can be readily and securely constructed for all possible data partitioning scenarios - horizontal, vertical and arbitrary; (ii) SecureCL can be used in the context of clustering algorithms that involve distance comparison without modifying the data proxy; (iii) using SecureCL data owner participation is entirely avoided while clustering is in progress, which means that non-honest party attacks are precluded; and (iv) that the proposed solution is scalable.

2 Related Work

This section presents a review of previous work directed at privacy preserving collaborative data clustering. As noted above, the proposed solutions can broadly be categorised according to whether they require involvement of a "semi-honest" third party or not. If not, security is maintained using an SMPC protocol of some kind; otherwise security is maintained using obfuscation or encryption methods. Both categories are considered in further detail below.

Where a third party is not used, SMPC protocols are adopted whereby security primitives are employed to facilitate privacy preservation (Goldreich, 1998). The protocols are designed to enable two or more parties to compute collaboratively functions or statistics concerning their data without disclosing their respective inputs. The nature of the individual SMPC protocol used is dependent on the nature of the data mining algorithm to be used. There have been a number of SMPC implementations for data clustering using a range of clustering algorithms. Examples include DBSCAN (Jiang, Xue, Ju, Chen, & Ma, 2008; Kumar & Rangan, 2007; J. Liu et al., 2013; Wei-jiang, Liu-sheng, Yong-long, Yi-fei, & Wei-wei, 2007), k-means (Mittal, Kaur, & Aggarwal, 2014) and Nearest Neighbour Clustering (NNC) (Shaneck, Kim, & Kumar, 2006). These implementations considered different numbers of participants (two-party (Kumar & Rangan, 2007; J. Liu et al., 2013; Shaneck et al., 2006; Wei-jiang et al., 2007) and multiple-party (Jiang et al., 2008; Mittal et al., 2014)) and different data partionings (horizontal (Jiang et al., 2008; Kumar & Rangan, 2007; J. Liu et al., 2013; Mittal et al., 2014; Shaneck et al., 2006), vertical (Kumar & Rangan, 2007; J. Liu et al., 2013; Wei-jiang et al., 2007) and arbitrary (J. Liu et al., 2013)). A range of mechanisms have also been employed to determine "distance" between data records, distributed across multiple data owners: (i) "secure scalar product" as in the case of (Jiang et al., 2008; Kumar & Rangan, 2007; Shaneck et al., 2006), (ii) secure accumulation using "secure sum" as in the case of (Wei-jiang et al., 2007) and (iii) secure comparison using "Yao's Millionaires Problem Protocol" (YMPP) as in the case of (Jiang et al., 2008; Kumar & Rangan, 2007; J. Liu et al., 2013; Shaneck et al., 2006; Wei-jiang et al., 2007). In these proposed solutions, data owners are expected to undertake a significant proportion of the work and thus are required to have adequate IT resource. Therefore the use of SMPC protocols, regardless of the precise nature of the adopted protocol, introduces a computation and communication overhead on behalf of data owners, rendering the approach only applicable for small data sets and a limited number of data owners. In terms of security, because of the requirement for significant data owner involvement, a non-honest participant can launch an "overlapping attack" and use the obtained results of distance calculations, and data comparisons with thresholds, to determine how similar the data belonging to other participants is to the attackers own records. This information can then be used to estimate the nature of the data belonging to the other participants as demonstrated in (J. Liu et al., 2013). In the specific context of DBSCAN clustering, a further security risk is that the total number of records within the ϵ-radius is revealed to all participants. These limitations render the SMPC-based solution inadequate for many instances of collaborative data mining.

Where a "semi-honest" third party data miner (a DMaaS provider) is used data privacy is preserved by either obfuscating or encrypting the data prior to outsourcing. The most straightforward form of obfuscation is data anonymisation (Samarati, 2001), which typically operates by removing confidential attributes and then generalising the remaining data until some "syntactic" condition is achieved. Several data clustering algorithms have been implemented using different anonymisation methods such as k-anonymity, l-diversity and t-closeness. However, data anonymisation adversely effects the data utility and hence clustering accuracy (Nergiz & Clifton, 2007). More importantly, anonymisation does not provide sufficient security; as evidenced by experiments reported in (Narayanan & Shmatikov, 2008), which demonstrated the potential for de-anonymising using "linkage attacks"; where the anonymised data is combined with publicly available data so that individual records can be identified when *quasi-identifier* attributes exist in both datasets. An alternative obfuscation method is data perturbation, which broadly operates by introducing "statistical noise". A number of data mining algorithms have been implemented using perturbation; these are surveyed in (Zhu, Li, Zhou, & Philip, 2017). However, most proposed perturbation methods provide the potential for a privacy breach by a non-honest party, as all participants are required to adopt the same method. In addition, data perturbation methods are subject to "external adversarial attacks", where reverse engineering techniques are used to reveal aspects of the original data (Huang, Du, & Chen, 2005).

Data encryption using standard encryption schemes has been proven to provide rigorous security guarantees; but at the cost of data utility (Goldreich, 2004), any form of data manipulation is precluded. A potential solution is to use some form of Property Preserving Encryption scheme that provides simple mathematical or logical operations. Two established Property Preserving Encryption mechanism are Homomorphic Encryption (HE) and Order Preserving Encryption (OPE) (Almutairi, Coenen, & Dures, 2018b; Rahman et al., 2017; Samanthula, Elmehdwi, & Jiang, 2015). In (Rahman et al., 2017), data belonging to multiple data owners was encrypted using a threshold HE scheme, that uses the secret sharing technique, and then outsourced to a third party data miner who utilised the HE mathematical properties of the scheme to conduct DBSCAN-based clustering. Using HE, distances between data records can be calculated; however, the calculated distances cannot be compared hence data owner participation is still required. To obviate the need for data owner participation the usage of two non-colluding third parties has been proposed (Samanthula et al., 2015), one holding encrypted data and another holding the decryption key. Thus unsupported data comparisons can be delegated to the third party key holder who acts on behalf of data owners. For the purpose of data privacy, the data inputs for the required calculations are modified before being passed to the key holder; however, use of a third party key holder raises the potential for a security breach. In (Almutairi et al., 2018b) a combination of OPE and HE was proposed together with the idea of an Encrypted Distance Matrix (EDM); this provided for privacy preserving data clustering without involving data owner participation (or recourse to SMPC protocols). However, the EDM concept was not suited to collaborative data clustering. The SecureCL solution considered in this paper is founded on the use of Super Secure Chain Distance Matrices (SSCDMs) as proposed in (Almutairi et al., 2018a).

3 Overview of SecureCL

The proposed SecureCL mechanism uses SSCDMs as a Φ-data proxy for the data to be clustered. Figure 1 shows the overall process required to generate the data proxy, and to conduct the secure clustering using the proposed SecureCL mechanism. The numbering used in the figure indicates the individual steps in the process as follows. In step 1 individual data owners create their own CDM using their local dataset D. A CDM, is a $(r-1) \times a$ matrix designed to hold "distance" values, where r is the number of records in a dataset D and a is the number of attributes in the associated attribute set A. In step 2 the Semi-Honest Third Party (STP) and data owners generate the parameters for the bespoken MUOPE scheme that is used in step 3 to encrypt the CDMs. The parameters are generated in a secure collaborative manner using the Paillier HE scheme fully described in (Paillier, 1999). The MUOPE scheme is an OPE scheme designed for use with respect to data distributed across multiple data owners. The scheme preserves the ordering of the encrypted data; this means that the similarity between data records can be determined, an essential requirement for data clustering. The MUOPE scheme is an amalgamation of the schemes proposed in (D. Liu & Wang, 2013) and (Z. Liu, Chen, Yang, Jia, & You, 2016) which were directed at single data owner applications. To generate the MUOPE encryption keys features from the well established Paillier HE scheme (Paillier, 1999) were used. A SSCDM is then generated by the STP with some data owners participation. The SSCDM is a combination of a set of Secure Chain Distance Matrices (SCDMs) which are combined, according to the data partitioning featured in the global dataset, using a *binding process*, step 4, as discussed in Section 5. Once the SSCDM has been generated the STP passes the SSCDM to the third party data miner (step 5); the STP is then ready to repeat the process with new data sets and/or new data owners.

The Third party data miner (a DMaaS provider) is now in a position to provide a secure collaborative data clustering service using the proposed SecureCL process. The data owners can lunch SecureCL by requesting a clustering by specifying the clustering algorithm to be applied and the associated parameters (step 6). Given a collection P of u participating parties, $P = \{p_1, \ldots, p_u\}$ the parties can lunch secure data clustering over the encrypted data proxy (the SSCDM), without further data owner involvement. The entire secure data clustering is conducted by the TPDM (step 7). The data similarities between records, regardless of their data owner, can be determined using the SSCDM and compared using the ordering preserved in the generated MUOPE cyphers. The results of the data clustering are then returned to participating parties, step 8, in such a way that each party will receive the results for their own records.

4 MUOPE Key Generation and Data Encryption

This section describes the process whereby the STP generates the MUOPE encryption keys and, once generated, how these keys are used to encrypt individual CDM distance values. To generate the encryption keys the STP determines the "interval" of the message space $M = [l, h)$ and the associated expanded "interval" of the cypher space $C = [l', h')$; where l and l', and h and h', are the minimum and maximum interval boundaries for the message and cypher spaces respectively (see Figure 2). The intervals should be selected in such a way that $|C| \gg |M|$. The key generation process is as follows:

Message space splitting: The STP randomly splits the message space interval M into t consecutive intervals; $M = \{m_1, \ldots, m_t\}$, where t is a random

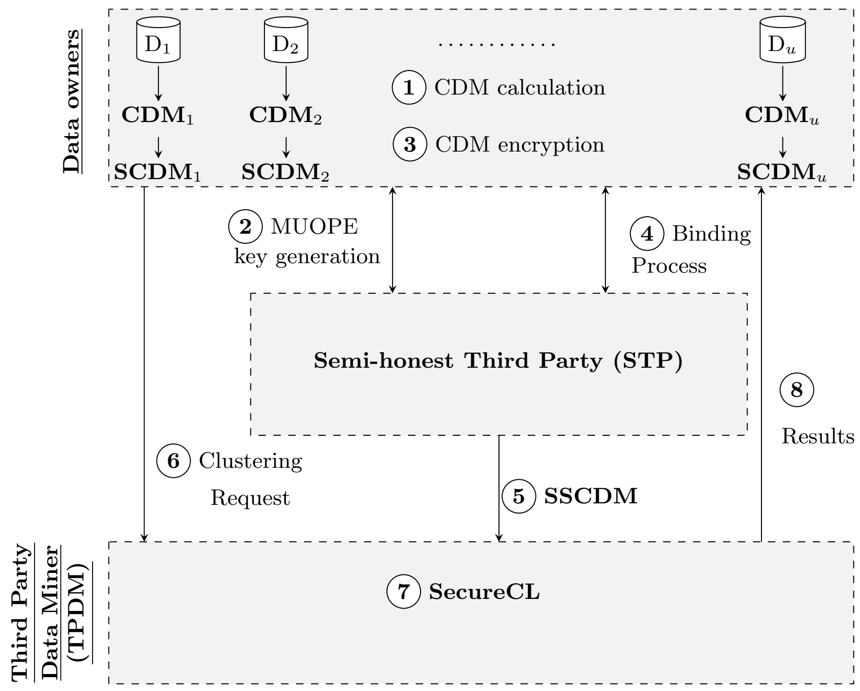


Fig. 1: The overall process to generate a SSCDM and conduct secure data clustering using the SecureCL mechanism

number. The length of interval m_i is decided by randomly selecting the minimum and maximum interval boundaries l_i and h_i (Figure 2).

Non-linear Cypher space expansion: The STP then splits the cypher space C into t intervals; $C = \{c_1, \ldots, c_t\}$. So that the data distribution is hidden, the length of each cypher space interval c_i is determined according to the "data density" in the corresponding message space interval m_i so that message space intervals with high data density will have large corresponding cypher space intervals. The density for each interval in the message space, representing data distributed across multiple data owners, is securely accumulated using a "secure density accumulation" process which utilises the additive property of the established Paillier Encryption scheme (Paillier, 1999). The STP manages the "secure density accumulation", by first generating Paillier key pairs and creating a list V corresponding to M and C ($V = \{v_1, \ldots, v_t\}$). On sartup, V will be initialised with random values which are encrypted using Paillier Encryption to give an encrypted list V'. The encrypted list V', together with the Paillier scheme public key, are then sent to the first data owner (party), p_1, who will calculate the individual density values with reference to their CDM (CDM_1). Data owner p_1 will then encrypt these values (again using Paillier Encryption) and add them to the values already held in V' using the Paillier Encryption additive property. The updated list V' is then sent to the next data owner who will repeat the process, and so on till there are no more data owners to be considered. The last party, p_u, will return V' to the STP who will decrypt it and subtract the original random values used to populate V to give the density value for each interval. These values are then used, by the STP, to determine the length of each cypher space interval. The message space and cypher space interval boundaries are the MUOPE encryption keys required by the MUOPE scheme.

The MUOPE keys are used by individual data owners to locally encrypt their CDM. Algorithm 1 gives the pseudo code for encrypting a plaintext value $x \in m_i$ to a MUOPE equivalent value $x' \in c_i$. The algorithm commences by determining the message space interval ID, i, within which x is contained (line 2). The boundaries (keys) of the ith message and cypher space intervals are then retrieved in lines 3 and 4. These values are used to calculate interval $scale_i$ and sample random value δ_i as per lines 5 and 6, where $Sens$ is a data sensitivity value representing the minimum distance between plaintext values in the dataset to be encrypted (calculated as specified in (D. Liu & Wang, 2013)). The value of δ_i is sampled for each interval so that longer intervals with a larger $scale_i$ value will consequently have a larger δ_i value than in the case of shorter intervals. The algorithm will exit (line 8) with cyphertext x' calculated in line 7. Note that the inclusion of the δ_i random value adds an extra level of security, it means that identical attribute values will not have the same encryption. Given two identical records they will be identified as similar, MUOPE does not support equality checking, this has proved advantages in the context of data clustering as will be demonstrated in Section 8.

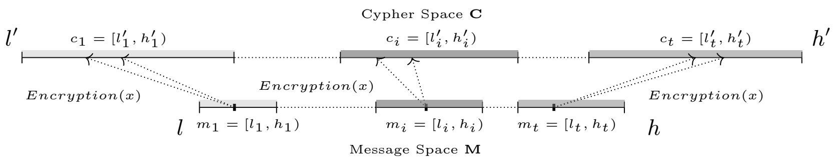


Fig. 2: Message and expanded cypher space splitting

5 Super Secure Chain Distance Matrices (SSCDMs)

A SSCDM is a mechanism for realising the envisioned Φ-data concept in the context of secure collaborative data clustering. As noted above a SSCDM is a combination of a number of SCDMs, generated by individual data owners. The individual data owner SCDM generation process is described in Sub-section 5.1.

Algorithm 1 Order Preserving Encryption algorithm

```
1: procedure ENCRYPTION($x, Sens$)
2:     $i \leftarrow \text{IntervalID}(x)$
3:     $[l_i, h_i] \leftarrow \text{Range}(i)$
4:     $[l'_i, h'_i] \leftarrow \text{Range}'(i)$
5:     $scale_i = \frac{(l'_i - h'_i)}{(l_i - h_i)}$
6:     $\delta_i = \text{Random}(0, Sens \times scale_i)$
7:     $x' = l'_i + scale_i \times (x - l_i) + \delta_i$
8:     Exit with $x'$
```

The process of combining a collection of SCDMs is referred to as “binding”. The nature of the binding depends on the nature of the partitioning. Three different forms of data partitioning can be identified: horizontal (Ceri, Negri, & Pelagatti, 1982), vertical (Navathe, Ceri, Wiederhold, & Dou, 1984) and arbitrary (Sub-sections 5.2 to 5.4). There is also the potential that individual records may be added or deleted; in this case the SSCDM does not need to be regenerated, the relevant content can simply be updated. SSCDM management is considered in Sub-section 6.

5.1 Secure Chain Distance Matrices (SCDMs)

A SCDM is constructed in two steps: (i) CDM calculation and (ii) CDM encryption. The CDM holds distances between attributes in consecutive data records (in whatever ordering they appear in the dataset). The pseudo code for the *CDM Calculation* process is given in Algorithm 2. The algorithm starts by dimensioning the CDM (line 2). The CDM is then populated (lines 3 to 5) in such a way that $CDM_{[i,j]}$ will hold the distance between the jth attribute value in the ith record with the same attribute value in record $i + 1$ (line 5).

Algorithm 2 Chain Distance Matrix calculation

```
1: procedure CDMCALCULATION($D$,$A$)
2:     CDM = $(r - 1) \times a$ 2D array ($r$ = rows in $D$, $a$ columns in $D$)
3:     for $i = 1$ to $i = r - 1$ do
4:         for $j = 1$ to $j = a$ do
5:             $\mathbf{CDM}_{[i,j]} = D_{[i,j]} - D_{[i+1,j]}$
6:     Exit with CDM
```

The next step is to encrypt the CDM using the MUOPE scheme, as described in Section 4, so as to produce a Secure CDM (SCDM). The purpose of this step is to prevent the potential of reverse engineering, given that a CDM is essentially a set of linear equations. Because MUOPE is an OPE scheme, the SCDM can be used to determine the similarity between a record $R_x = \{r_{x_1}, r_{x_2}, \ldots, r_{x_a}\}$ and a record $R_y = \{r_{y_1}, r_{y_2}, \ldots, r_{y_a}\}$ (where $x < y$) using Equation 1. In the case when x = y the distance will clearly be 0.

$$Sim(SCDM, x, y) = \sum_{j=1}^{j=|A|} | \sum_{i=x}^{i=(y-1)} SCDM_{[i,j]} | \quad (1)$$

5.2 SSCDM for Horizontal Data Partition

In this and the following two sub-sections the SSCDM binding process is considered in terms of the three types of partitioning that can be identified, starting with horizontal data partitioning. Horizontal data partitioned is where the participating parties conform to the same set of attributes, A, but each holds different records (Ceri et al., 1982). In other words, the global dataset D is decomposed into "horizontal" segments each belonging to a single party. Multiple SCDMs, representing horizontally partitioned data, can be "bound" by the STP, to form a SSCDM using the *HorizontalBinding* process presented in Algorithm 3. The inputs are: (i) the id, i, of the current party in the sequence, (ii) the SCDM of the next party in the sequence, SCDM$_{i+1}$ and (iii) the global SSCDM accumulated sofar. At the beginning of the process the SSCDM sofar is $SCDM_1$ belonging to the first party. The algorithm starts with the STP generating a random record R_r of length a, $R_r = \{r_{r_1}, r_{r_2}, \ldots, r_{r_a}\}$ which is then encrypted using the MUOPE scheme (line 2). The record R_r is sent to p_i and p_{i+1}. Party p_i will calculate the difference between the MUOPE cypher of the last record in their local dataset D_i and record R_r to give $C_1 = \{c_{1_1}, c_{i_2}, \ldots c_{1_a}\}$ (line 3). At the same time p_{i+1} will calculate the difference between R_r and the MUOPE cypher of the first record in their dataset D_{i+1} to give $C_2 = \{c_{2_1}, c_{2_2}, \ldots c_{2_a}\}$ (line 4). On receiving C_1 and C_2 the STP will next generate a *Pivot* record by adding each attribute in C_1 with the corresponding attribute in C_2 (line 5). The pivot record will then be used to bind $SCDM_i$ (already added in SSCDM) and $SCDM_{i+1}$ (line 6). The process will be repeated using $SCDM_{i+1}$ and $SCDM_{i+2}$ until the entire SSCDM is constructed.

Algorithm 3 Horizontal binding process

```
1: procedure HorizontalBinding($i, SCDM_{i+1}, SSCDM$)
2:     $R_r = \{r_1, \ldots, r_a\}$                    ▷ Encrypted using MUOPE
3:     $\mathbf{P}_i$:   $C_1$ = Distances between MUOPE cypher of last record in $D_i$ and $R_r$
4:     $\mathbf{P}_{i+1}$: $C_2$ = Distances between $R_r$ and MUOPE cypher of first record in $D_{i+1}$
5:     $Pivot = C_1 + C_2$
6:     $SSCDM = concatenate(SSCDM, Pivot, SCDM_{i+1})$
7:     Exit with SSCDM
```

Algorithm 4 Vertical binding process

```
1: procedure VerticalBinding($SCDM_i, SSCDM$)
2:     $SSCDM = concatenate(SSCDM, SCDM_i)$
3:     Exit with SSCDM
```

5.3 SSCDM for Vertical Data Partition

Vertical data partition is where the participating parties conform to the same set of records, but each holds different attributes derived from a global set of attributes A (Navathe et al., 1984). In other words, the global dataset D is decomposed into "vertical" segments each belonging to a single party. In this context, multiple SCDMs can be bound following the process given in Algorithm 4. The inputs are $SCDM_i$ belonging to party p_i and the SSCDM accumulated sofar. At the beginning of process, the SSCDM sofar is simply $SCDM_1$, the SCDM belonging to the first data owner. The vertical binding process does not require generation of a *pivot* record as in the case of horizontal binding; it operates by simply appending the SCDMs to one another. This process will be repeated until the entire SSCDM is constructed.

5.4 SSCDM for Arbitrary Data Partition

The arbitrary data partition generalises the vertical and horizontal partitioning cases. As such, arbitrary data partitioning dictates an alternative SCDM binding. The process commences with a *schema agreement* process to derive an ordered global set of attributes A and an ordered global set of records R, this is orchestrated by the STP. The binding process is as shown in Algorithm 5. The inputs are: (i) the number of attributes a, (ii) the number of records r, (iii) the current SCDM ($SCDM_i$) belonging to data owner p_i and (iv) the SSCDM sofar. On start up the SSCDM sofar will simply be a zero valued $(r-1) \times a$ matrix. The process commences with the construction of a temporary $(r-1) \times a$ SCDM, $SCDM'$, initially populated with only zero values (line 2), to which $SCDM_i$ is added (line 3) in such a way that matches the order of the agreed schema. The updated SSCDM is then constructed by adding the content of $SCDM'$ to $SSCDM$ (line 4).

Algorithm 5 Arbitrary binding process

```
1: procedure ArbitraryBinding($a, r, SCDM_i, SSCDM$)
2:     $SCDM' = r - 1 \times a$ matrix populated with 0 values
3:     $SCDM' = SCDM' + SCDM_i$
4:     $SSCDM = SSCDM + SCDM'$
5:     Exit with SSCDM
```

6 SSCDM Management

New records, when added to the global data set D, can simply be appended to the SSCDM sofar in a manner similar to the binding processes described above depending on the nature of the data distribution. Deletion of records is more complex, the appropriate SSCDM elements need to be removed. The pseudo code for achieving this is given in Algorithm 6. The inputs are the index of the "virtual" record to be deleted (virtual because a proxy for the data is being used), *virtualId*, and the size of the attribute set a. The algorithm commences by determining the caller party id, p_{id} (the party that wishes to delete the record), using the *Caller* procedure (line 2). The SSCDM index of the record to be deleted is determined using the *Index* procedure which returns the SSCDM index i associated with virtual record $vr_{virtualId}$ of party p_{id} (line 3). In the case when the index of the record to be deleted is the first or the last element in the SSCDM, the updating process is accomplished by only deleting the $SSCDM_i$ (line 7). Otherwise, the updating process is accomplished by generating a new record that replace two SSCDM rows, $SSCDM_{i-1}$ and $SSCDM_i$. The new generated record will then be used to update $SSCDM_{i-1}$ (lines 5 and 6), while the $SSCDM_i$ will be deleted (line 7).

Algorithm 6 Delete SSCDM element

```
1: procedure DeleteSSCDMElement($virtualId, a$)
2:     $p_{id} \leftarrow$ Caller()
3:     $i \leftarrow$ Index($SSCDM, virtualId, p_{id}$)
4:     if ($i \neq 1$ and $i \neq |SSCDM|$) then
5:         for $j = 1$ to $j = a$ do
6:             $SSCDM_{[i-1,j]} = SSCDM_{[i-1,j]} + SSCDM_{[i,j]}$
7:     Delete $SSCDM_i$
```

7 Third Party Data Clustering Using SecureCL

Using SecureCL the SSCDM Φ data proxy can be applied to a range of clustering algorithms. Since there is no data (only Φ data) a Virtual Records list, $VR = \{vr_1, \ldots, vr_{|SSCDM|+1}\}$ is defined where the indices are used to refer to records held by data owners. For example, the records held by p_1 have indices 1 to $|SCDM_1|+1$. At the end of a data clustering process each party will only receive the cluster labels for the data they own. For the evaluation presented in Section 8 DBSCAN and Nearest Neighbour Clustering were considered. Only Secure Nearest Neighbour Clustering (SNNC) is considered in further detail here. For details concerning the implementation of Secure DBSCAN interested readers are referred to (Almutairi et al., 2018a).

The SNNC is conducted, by the third party data miner, in a similar manner to that of standard NNC (Cover & Hart, 1967) as summarised in Algorithm 7. The inputs are the SSCDM and the desired SNNC threshold σ; agreed by the participating parties. To allow secure data clustering the threshold σ is encrypted using MUOPE to give σ'. The algorithm commences by creating the "virtual" dataset VR (line 2). The first virtual record (vr_1) is added to the first cluster (lines 3 and 4) then iteratively the remaining records are assigned to clusters (lines 5 to 13). As for standard NNC, virtual record vr_i will be assigned to cluster C_m if there exist some virtual record vr_j in cluster m whose distance from vr_i is less than or equals to σ'. Otherwise, vr_i is assigned to a new cluster.

Algorithm 7 Secure Nearest Neighbour Clustering

```
1: procedure SecureNearestNeighbourClustering($SSCDM, \sigma'$)
2:     $C = \emptyset$, $VR$ =list of record IDs, $k = 1$
3:     $C_k = \{vr_1\}$
4:     $C = C \cup C_k$
5:     for $i = 2$ to $i = |VR|$ do
6:         $[m, smallDist] \leftarrow$ findNearestRecordCluster($i, C, SSCDM$)
7:         if $smallDist \leq \sigma'$ then
8:             $C_m = C_m \cup vr_i$
9:             update $C_m$ in Cluster list $C$
10:        else
11:            $k = k + 1$
12:            $C_k = \{vr_i\}$
13:            $C = C \cup C_k$
14:    Exit with C
15: procedure findNearestRecordCluster($index$, $C$,$SSCDM$)
16:    $smallDist$ =MaxNumber
17:    for $clusterID = 1$ to $clusterID = |C|$ do
18:        for $j = 1$ to $j = |C_{clusterID}|$ do
19:            $Distance = Sim(SSCDM, index, j)$
20:            if $Distance < smallDist$ then
21:                $smallDist = Distance$
22:                $m = clusterID$
23:    Exit with $m$ and $smallDist$
```

Therefore, in line 6, the *findNearestRecordCluster* procedure is called to return the cluster ID of the nearest record and their corresponding distance value. The *findNearestRecordCluster* procedure uses the SSCDM to determine the similarity between records. The record vr_i will be assigned to cluster m when the distance is less than the threshold (lines 7 to 9), otherwise a new cluster is generated (lines 11 to 13).

8 Experimental Evaluation

The evaluation of the *Φ*-data concept, implemented using SSCDMs and the MUOPE scheme, in the context of SecureCL, is presented in this section. For the evaluation the operation of secure encrypted clustering, using Secure NNC (SNNC) and Secure DBSCAN (SDBSCAN), was compared with the operation of the corresponding "standard", un-encrypted variations of NNC and DBSCAN. The criteria considered were: (i) data owner participation, (ii) clustering efficiency, (iii) clustering accuracy, (iv) security, (v) scalability and (vi) memory resource requirement. For the evaluation both synthetic data and fifteen datasets from the UCI data repository (Lichman, 2013) were used, the later listed in Table 1.

Table 1: Cluster configuration for standard and secure DBSCAN and NNC (differing results highlighted in bold font)

UCI Data set	$MPts$	ϵ	σ	DBSCAN		SDBSCAN		NNC		SNNC	
				Num. Clus.	Sil. Coef.	Num. Clus.	Sil. Coef.	Num. Clus.	Sil. Coef.	Num. Clus.	Sil. Coef.
1. Arrhythmia	2	600	1	6	0.472	6	0.472	452	1.000	452	1.000
2. Banknote Auth.	2	3	5	7	0.922	7	0.922	21	0.895	21	0.895
3. Blood Trans.	2	10	68	**27**	**0.971**	**33**	**0.976**	**34**	0.999	**35**	0.999
4. Breast Cancer	2	5	10	**4**	**0.678**	**1**	**0.485**	**108**	**0.903**	**135**	**0.926**
5. Breast Tissue	2	100	1	3	0.628	3	0.628	105	1.000	105	1.000
6. Chronic kidney	2	70	100	19	0.970	19	0.970	243	0.999	243	0.999
7. Dermatology	2	10	18	**16**	**0.853**	**15**	**0.881**	**32**	**0.919**	**37**	**0.915**
8. Ecoli	2	60	1	1	-1.000	1	-1.000	2	0.353	2	0.353
9. Ind. Liver Pat.	3	40	99	7	0.789	7	0.789	100	0.997	100	0.997
10. Iris	5	2	1	2	0.722	2	0.722	**15**	**0.922**	**16**	**0.927**
11. Libras Move.	5	5	4	11	0.715	11	0.715	224	0.969	224	0.969
12. Lung Cancer	2	20	1	1	0.053	1	0.053	32	1.000	32	1.000
13. Parkinsons	3	10	73	5	0.829	5	0.829	11	0.953	11	0.953
14. Pima Disease	5	20	100	4	0.691	4	0.691	22	0.956	22	0.956
15. Seeds	5	1	1	7	0.852	7	0.852	103	0.979	103	0.979

8.1 Data Owner Participation

Data owner participation was evaluated in terms of runtimes required to prepare data for SecureCL and the amount of data owner involvement when clustering was in progress. For preparing the data, data owners are required to: (i) calculate CDMs (`CDM Cal`), (ii) encrypt CDMs (`CDM Enc`) and (iii) provide their data density using “Secure density accumulation” as discussed in Section 4 (`Dens Cal`). The number of elements in CDMs, to be calculated and encrypted, is $(r - 1) \times a$, thus the required time for `CDM Cal`, `CDM Enc` and `Dens Cal` is a function of the data set size. To evaluate the complexity of the data preparation, ten synthetic datasets, increasing in size from $1,000$ to $10,000$ in steps of $1,000$, were used with the number of attributes kept constant at $a = 125$. Figure 3 shows the reported results. As expected, runtime linearly increased with the size of data. For example, in the case when $r = 1K$ the runtimes for `CDM Cal`, `CDM Enc`, and `Dens Cal` are $162.69ms$, $494.31ms$ and $162.68ms$ respectively, while for $r = 10K$ the runtimes are $444.5ms$, $2356.94ms$ and $492.88ms$ respectively. Whatever the case, the reported results indicate that, regardless of the number of records considered data owner participation for preparing data did not introduce any significant overhead. A dataset measuring $10K \times 125$ can be prepared in a matter of seconds. Using SecureCL, unlike in the case of alternative solutions to secure collaborative clustering found in the literature, there is no requirement for any data owner participation while clustering is in progress. This was achieved without recourse to key delegation, as in the case of “secret sharing”, and without resorting to SMPC protocols.

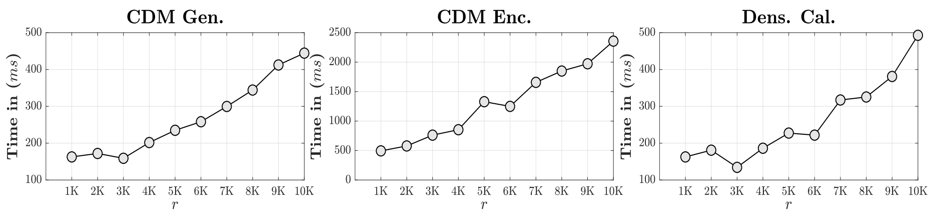


Fig. 3: Time required (ms) for data owner participation in terms of number of records

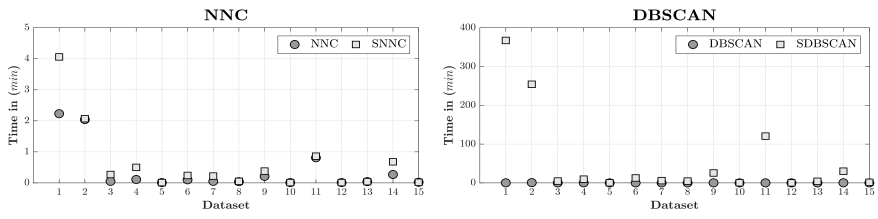


Fig. 4: Comparison of run times using standard and secure clustering algorithms

8.2 Clustering Efficiency (Runtime)

The utilisation of the SSCDM proxy, and order preserving comparison, will clearly introduce a computational overhead compared to standard algorithms (using plaintext data). The runtimes required to cluster the experimental datasets using standard algorithms were compared with the runtimes required using SecureCL. The results are shown in Figure 4. The algorithm thresholds and parameters ($MPts$, ϵ and σ) are as given in Columns 2 to 4 of Table 1; selected because, from the literature, these were shown to give good results; in practical these are prescribed by data owners. The results indicate that the runtime required by SNNC and SDBSCAN, as expected, is greater than that require by standard NNC and DBSCAN. The difference is caused by the utilisation of SSCDMs to determine similarity; the bigger the dataset the larger the SSCDM the greater the time required to use the SSCDM for determining similarity. However, it is argued here that these runtimes were still within acceptable boundaries.

8.3 Clustering Accuracy

The "correctness" of clustering configurations produced using SDBSCAN and SNNC were measured by comparing the results obtained with those obtained using the equivalent standard (un-encrypted) approaches. Silhouette coefficient (Sil. Coef.) was used as the evaluation metric; a value between -1 to 1, the closer the value is to 1 the better the clustering. To demonstrate that the proposed solution operates correctly, SecureCL should produces comparable Sil. Coef. values to those produced using the standard approaches (using the same parameters and/or thresholds). The number of produced clusters was also compared. The results are presented in Columns 5 to 12 of Table 1. From the table, in most cases, SecureCL produced identical configuration to those produced using standard approaches; only in 7 cases out of 30 were different results obtained. This was because of the random element included in MUOPE as a consequence of which equality testing is not supported. Interestingly, with respect to the differing results, SecureCL produced slightly better configurations in four out of the seven cases (Iris and Breast cancer in context of NNC, and Dermatology and Blood Trans in context of DBSCAN). The dataset Blood Trans. produced a different number of classes using NNC but with the same Sil. Coef. In the remaining two cases (Breast Cancer in context of DBSCAN and Dermatology in context of NNC) the standard approach was slightly better. The results demonstrate that SecureCL provides a suitable solution to secure collaborative data clustering in that accuracy is not adversely affected.

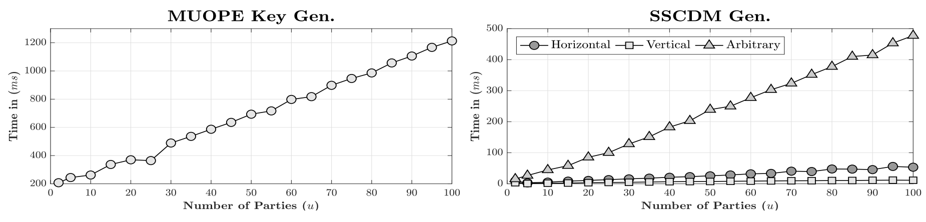
8.4 Security Analysis

In this section, the security of the proposed SecureCL mechanism is evaluated by identifying potential attacks that can be directed to breach the outsourced data privacy. For the purpose of this evaluation, the third party data miner was treated as "passive adversary" in the context of what is usually referred to as the "semi-honest model". In this model, the third party data miner is expected to "honestly" execute the SecureCL facilitated clustering; but during the process uses the results, and any intermediate knowledge, to extract additional information about the parties' data, as defined in (Goldreich, 2004). This was considered a reasonable assumption since the main objective of the third party (the DMaaS provider) is to deliver high quality and effective services to clients (data owners).

Using SecureCL attacks directed over the actual data are entirely precluded because of the adopted Φ-data concept which prevents data from being confided, in any form, to a third party data miner or shared with any other participant. The data proxy received by the third party data miner is the SSCDM (encrypted using MUOPE), hence the only possible attacks are Cyphertext Only Attacks (COAs) that can be launched when the attacker somehow has access to the SSCDM. COAs are more likely to succeed when the attacker has background knowledge about the original data (data frequency and/or distribution) that can be associated with the order preserving features of MUOPE to reveal cyphertexts of highly frequent data items. However, as a countermeasure to COAs, the MUOPE scheme reduces the information leakage in the generated cyphertexts by obscuring the data frequency and distribution. The data distribution is obscured using the concept of *message space splitting* and *non-linear cypher space expansion*, which is derived from the shared data density, in such a way that message space intervals with high data density will have larger (expanded) cypher space intervals. This feature associated with the MUOPE encryption function will guarantee that plaintexts in high density intervals will equate to long cypher space intervals. The data frequency is hidden using the encryption function; this generates different cyphertexts for the same plaintext value even when the same encryption keys are used. This feature is essential to prevent the utilisation of frequency analysis that could allow an attacker to relate cyphertexts with the same frequency to known plaintext values (if such plaintext values where available) and then identify the cyphertexts that match this frequency. In the proposed solution no decryption take place on the third party side; hence additional security. The third party data miner, who compares data records in the encrypted SSCDM and assigns records to appropriate clusters, cannot initiate overlapping attacks that rely on the results of comparisons and the real data, because the real data will not be available. The entire clustering process is delegated to the third party data miner and each party will receive the class labels for their own dataset, hence a non-honest party cannot launch any form of attack.

8.5 Scalability

The scalability of the proposed solution was evaluated by analysing the runtime as the number of participating parties (u) increases. Using `SecureCL`, increasing the number of participants would have an effect on the efficiency of: (i) the MUOPE key generation process (`MUOPE Key Gen`) and (ii) the binding process for generating SSCDMs (`SSCDM Gen`). Experiments were conducted using a range of values for u from 10 to 100 increasing in step of 5; $u = 2$ and $u = 4$ were also considered. For the experiments a $r = 7000$ and $a = 125$ synthetic data set, distributed equally across the participants, was used, and the runtimes recorded. The results are shown in Figure 5. As expected, the time complexity for both `MUOPE key Gen` and `SSCDM Gen` increased linearly with number of parties. However, the increased run time was not significant. When $u = 10$ the `MUOPE key Gen` was undertaken in $262ms$; when $u = 100$, $1,213ms$ was required. The binding process runtime for horizontal and vertical partitioning was negligible, whilst for arbitrary partitioning this was higher due to the requirement for schema agreement.



(a) **(b)**

Fig. 5: Runtime to generate OPE keys and construct SSCDMs as the number of participants (data owners) increases

8.6 Memory Requirement Compared to Related Work

The memory resources required by SSCDMs was compared to the EDM-based solution presented in (Almutairi et al., 2018b); which also required no data owner participation in the context of data clustering, but did not support collaborative data clustering as in the case of SSCDMs. An EDM is a 2D matrix where the first and second dimensions are correlated to the number of records r in the dataset; thus the EDM size grows exponentially with the number of records. More formally, the number of EDM elements is equal to $\frac{r(r+1)}{2}$, while the number of elements in a SCDM is equal to $(r-1)\times a$. Therefore, a SSCDM is more compact and more appropriate for big data where the number of records are much bigger than the number of attribute features in the data. For example a dataset with $10K$ records and 100 attributes would result in $9,99900$ element SSCDM, as compared to a $50,005K$ element EDM, a significant difference.

9 Conclusion and Future Work

This paper has presented SecureCL, a secure collaborative data clustering mechanisms, suited to the provision of DMaaS. The mechanism is founded on the usage of the concept of Φ-data realised using SSCDMs encrypted using a bespoke encryption scheme, MUOPE. The Φ-data concept liberates data owners from sending their data (in any form) to a third party data miner or sharing it with other participants. SecureCL provides the following advantages:

1. Data clustering is entirely delegated to a third party without the requirement of resorting to computationally expensive SMPC protocols or secret key sharing.
2. The quality of the data clustering is comparable to that produced using standard, non-encrypted methods.
3. No data owner participation is required once data proxy encryption has been completed.
4. Non-honest data owner attacks, and overlapping attacks, are precluded since the third party does not have access to data to conduct overlapping attacks and data owners are not involve in the process of data clustering.
5. The solution can be scaled to a large number of participants.

It should also be noted that the data proxy, the SSCDM, can be employed with respect to a range of data clustering algorithms, not just NNC and DBSCAN as used for illustrative purposes in this paper. For future work, the authors intend to evaluate the SSCDM in context of data classification and consider the application of MUOPE in the context of database security.

References

Agrawal, R., & Srikant, R. (2000). Privacy-preserving data mining. In *Proceedings of the 2000 sigmod international conference on management of data* (p. 439-450). ACM.

Almutairi, N., Coenen, F., & Dures, K. (2018a). Secure third party data clustering using ϕ data: Multi-user order preserving encryption and super secure chain distance matrices. In M. Bramer & M. Petridis (Eds.), *Artificial intelligence xxxv* (p. 3-17). Cham: Springer.

Almutairi, N., Coenen, F., & Dures, K. (2018b). Third party data clustering over encrypted data without data owner participation: Introducing the encrypted distance matrix. In *International conference on big data analytics and knowledge discovery* (p. 163-173).

Anikin, I. V., & Gazimov, R. M. (2017). Privacy preserving DBSCAN clustering algorithm for vertically partitioned data in distributed systems. *International Siberian Conference on Control and Communications*, 1-4.

Ceri, S., Negri, M., & Pelagatti, G. (1982). Horizontal data partitioning in database design. In *Proceedings of the 1982 acm sigmod international conference on management of data* (p. 128-136). ACM.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, *13*(1), 21–27.

Goldreich, O. (1998). Secure multi-party computation. *Manuscript. Preliminary version*, *78*.

Goldreich, O. (2004). *Foundations of cryptography.* Cambridge University Press.

Huang, Z., Du, W., & Chen, B. (2005). Deriving private information from randomized data. In *Proceedings of the 2005 sigmod international conference on management of data* (p. 37-48).

Jiang, D., Xue, A., Ju, S., Chen, W., & Ma, H. (2008). Privacy-preserving DBSCAN on horizontally partitioned data. In *International symposium on it in medicine and education* (p. 1067-1072).

Kumar, K. A., & Rangan, C. P. (2007). Privacy preserving DBSCAN algorithm for clustering. In *International conference on advanced data mining and applications* (p. 57-68).

Lichman, M. (2013). *UCI machine learning repository.* Retrieved from http://archive.ics.uci.edu/ml

Liu, D., & Wang, S. (2013). Nonlinear order preserving index for encrypted database query in service cloud environments. *Concurrency Computation: Practice and Experience*, *25*(13), 1967-1984.

Liu, J., Xiong, L., Luo, J., & Huang, J. Z. (2013). Privacy preserving distributed DBSCAN clustering. *Transactions on Data Privacy*, *6*(1), 69-85.

Liu, L., Kantarcioglu, M., & Thuraisingham, B. (2008). The applicability of the perturbation based privacy preserving data mining for real-world data. *Data and Knowledge Engineering*, *65*(1), 5–21.

Liu, Z., Chen, X., Yang, J., Jia, C., & You, I. (2016). New order preserving encryption model for outsourced databases in cloud environments. *Journal of Network and Computer Applications*, *59*, 198-207.

Mittal, D., Kaur, D., & Aggarwal, A. (2014). Secure data mining in cloud using homomorphic encryption. *IEEE International Conference on Cloud Computing in Emerging Markets*, 1-7.

Narayanan, A., & Shmatikov, V. (2008). Robust De-anonymization of large sparse datasets. In *Proceedings of the 2008 symposium on security and privacy* (p. 111-125).

Navathe, S., Ceri, S., Wiederhold, G., & Dou, J. (1984). Vertical partitioning algorithms for database design. *ACM Transactions on Database Systems*, *9*(4), 680-710.

Nergiz, M. E., & Clifton, C. (2007). Thoughts on k-anonymization. *Data and Knowledge Engineering*, *63*(3), 622 - 645.

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of eurocrypt* (p. 223-238).

Rahman, M. S., Basu, A., & Kiyomoto, S. (2017). Towards outsourced privacy-preserving multiparty DBSCAN. In *22nd pacific rim international symposium on dependable computing* (p. 225-226).

Samanthula, B. K., Elmehdwi, Y., & Jiang, W. (2015). k-Nearest Neighbor classification over semantically secure encrypted relational data. *IEEE Transactions on Knowledge and Data Engineering*(5), 1261-1273.

Samarati, P. (2001). Protecting respondents identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, *13*(6), 1010-1027.

Shaneck, M., Kim, Y., & Kumar, V. (2006). Privacy preserving nearest neighbor search. *Sixth IEEE International Conference on Data Mining*, 541.

Singh, A., & Chatterjee, K. (2017). Cloud security issues and challenges: A survey. *Journal of Network and Computer Applications*, *79*, 88 -115.

Wei-jiang, X., Liu-sheng, H., Yong-long, L., Yi-fei, Y., & Wei-wei, J. (2007). Privacy-preserving DBSCAN clustering over vertically partitioned data. In *the 2007 international conference on multimedia and ubiquitous engineering* (p. 850-856).

Yao, A. C. (1982). Protocols for secure computations. In *23rd annual symposium on foundations of computer science* (pp. 160–164).

Zhu, T., Li, G., Zhou, W., & Philip, S. Y. (2017). Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering*(8), 1619-1638.