International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6037
***Article Received**: 10 November 2022 **Revised**: 18 December 2022 **Accepted**: 05 January 2023*

Hybrid Cloud-Based Privacy Preserving Clustering as Service for Enterprise Big Data

Amogh Pramod Kulkarni[1], Dr. Manjunath T N[2]
[1]Research scholar, Visvesvaraya Technological University, Belagavi,
Asst. Prof., dept. of computer science & engineering, Presidency University, Bengaluru, kulkarni84@gmail.com .
[2]Professor, Department of Information Science and engineering, B.M.S Institute of Technology and Management, Bengaluru, manju.tn@gmail.com.

ABSTRACT
Clustering as service is being offered by many cloud service providers. It helps enterprises to learn hidden patterns and learn knowledge from large, big data generated by enterprises. Though it brings lot of value to enterprises, it also exposes the data to various security and privacy threats. Privacy preserving clustering is being proposed a solution to address this problem. But the privacy preserving clustering as outsourced service model involves too much overhead on querying user, lacks adaptivity to incremental data and involves frequent interaction between service provider and the querying user. There is also a lack of personalization to clustering by the querying user. This work "Locality Sensitive Hashing for Transformed Dataset (LSHTD)" proposes a hybrid cloud-based clustering as service model for streaming data that address the problems in the existing model such as privacy preserving k-means clustering outsourcing under multiple keys (PPCOM) and secure nearest neighbor clustering (SNNC) models, The solution combines hybrid cloud, LSHTD clustering algorithm as outsourced service model. Through experiments, the proposed solution is able is found to reduce the computation cost by 23% and communication cost by 6% and able to provide better clustering accuracy with ARI greater than 4.59% compared to existing works.

Keywords: - Clustering, cryptographic, cloud, hash, LSHTD

I. INTRODUCTION

Enterprises generate large volumes of data. Enterprise is adopting cloud due to its various features like on demand scalability, pay as you, higher availability, lower CAPEX and OPEX compared to owing the infrastructure. Offloading to cloud storage make it easier to invoke any third-party data mining tools on the data. Mining on these large volumes of data can provide valuable insights about sales, customer service etc. Clustering is a popular data mining operation to group the data records according to their features, attributes and similarities. Though offloading storage to cloud brings lot of values, it also brings lot of security and privacy threats to the data. Sensitive data can be leaked, and privacy can be compromised. Various methods have been adopted for ensuring the security and privacy of the data. The existing methods for privacy preservation fall into categories of: Anonymization, Randomization, Cryptographic techniques, Diversification and Aggregation. Various privacy preserving clustering algorithms have been proposed on transformed data. These algorithms do cluster on privacy transformed data and thus the results provided by them are also privacy preserved.

A detailed survey on the state of art privacy preservation clustering algorithms is presented in related work section. Most of the existing privacy preserving clustering algorithms in outsourced model has some overhead on the querying user. They also involve multiple interactions between the cloud service provider and querying user. These interactions can be tampered, and clustering can be made erroneous. Also, the approaches cannot be scaled for incremental clustering. They also lack personalization in clustering in terms of attributes selection, outlier definition etc. This work addresses these problems. Hybrid clouds are being recently proposed as solution for security and privacy preserving storage. The data is distributed to public and private cloud parts of hybrid cloud based on the sensitivity. The sensitive portions are kept in private cloud managed by the enterprises. The insensitive portions are kept in public cloud. In this work, we propose a hybrid cloud-based privacy preserving clustering as service outsourced model.

The proposed model keeps all sensitive attributes of dataset in the private cloud and distributes the public attributes to the public cloud. For each data item, an adaptive locality hash is generated based on the current distribution of data. The locality hash along with attribute personalization function is distributed to public cloud. A modified clustering algorithm is executed on the locality hash to cluster the data points. On every batch of incremental data, the locality hash is updated based on current distribution of the data.

The proposed solution first transforms the data by normalizing it and mapping to different range of values.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6037
***Article Received**: 10 November 2022 **Revised**: 18 December 2022 **Accepted**: 05 January 2023*

This is done to ensure privacy. Modified locality sensitive hashing is proposed to map the transformed items to buckets. Privacy preserving clustering algorithm is proposed to cluster the data points based on the items mapped to same bucket. Following is some of the important contributions of this work.

This work proposes a low overhead privacy preservation clustering without any cryptographic primitives.

1. Clustering on high dimensional space is transformed to clustering on low dimensional locality hash, thereby improving the effectiveness of clustering.
2. With adaptable locality hashing, the proposed work is able to support incremental data with lower overhead.
3. With attribute personalization method, the querying user can select the attributes to be used for clustering.

The structure of the paper follows: Section 2 describes the related works summary. Proposed method hybrid cloud-based clustering as service model for streaming data is in the section 3. Results and analysis of performance parameters discussed in the section 4. Finally section 5 defines the conclusion of the works.

II. RELATED WORK

Zobaed et al (2020) [1] proposed a privacy preserving clustering scheme for unstructured big datasets. The scheme considers three cases of static, dynamic and semi dynamic datasets. In big data, deciding the cluster center is a costly operation due to repeated iteration. To solve this problem, author proposed a scheme of centroid selection based on diversity of topics. Once the centroids are selected, the dataset items are associated to cluster based on maximum relatedness of token in weighted bipartite graph. In case of semi-dynamic/dynamic data, cluster centroid selection is repeated based on the results of Chi square distribution test. Privacy is ensured by encrypting the plain text and transforming it to tokens. Clustering is done on transformed data. This solution works well when the topic distribution is known priori, when the topic distribution is in large space, the performance overhead is very high due to frequent re-clustering.

Wei fu et al (2019) [2] proposed a method to estimate the cluster centers for K means algorithm without the need for iterations. The clusters centers are found by running a K fold cross validation on the dataset and estimating the cluster centers using regression. Though the method was able to achieve to find better cluster centers, it needs the right selection of data for K folds. Depending on the K fold selection, clustering accuracy varied. Authors did not propose any heuristics of K fold selection. Mary et al (2012) [3] proposed a density based dynamic data clustering algorithm for incremental dataset. The authors modified the original DBSCAN algorithm in way of treating the outliers. The outliers are treated as unclassified points and considered for clustering on arrival of next batch of data. This allowed for adaptation to incremental dataset. The lag of outlier increases on each incremental round and this increases the computation complexity at later rounds.

Rong et al (2017) [4] considered the problem of joint clustering of datasets encrypted with different keys. Authors used double decryption cryptosystem to transform the encrypted data so that operations like addition, subtraction, multiplication, comparison and equality can be executed. But the computation and communication cost are very high in this approach. Yuan et al (2019) [5] proposed a privacy preserving K-means clustering algorithm that can be outsourced to cloud service providers. The solution is designed to protect the data from privacy leakages by a curious cloud service provider. The solution involves active participation from data owner. Data owner computes the cluster centers and provides the centers to cloud service provides for clustering the encrypted objects. The computation and communication cost are higher in this approach for large dataset.

Rao et al (2015) [6] proposed a privacy preserving outsourced K-means clustering algorithm. User encrypts the data using homomorphic encryption and uploads to the cloud. Cloud service provider executes K-means clustering on the encrypted data. Authors proposed an order preserving encryption to aid the distance computation operations in K means clustering. The solution expects all the users use the same key for data transformation. This can be security lapse even if one of the users is compromised. Zou et al (2020) [7] proposed a privacy preserving outsourced K-means clustering algorithm using multiple keys. The work is based on BCP encryption which has additive homomorphic property and provides double decryption mechanisms. The querying user who needs clustering results offloads entire clustering operation to cloud and does not participate in any operations. The data records are double encrypted to ensure privacy against curious cloud service providers. All the operations needed for clustering like distance measurement, distance comparison etc are secured. The computation and communication complexity are higher in this approach.

Yu et al (2016) [8] proposed a K-means algorithm with differential privacy. Initial cluster centers are selected based on distribution of data points. Privacy preserved is done by adding Laplacian noises to clustering results and sending to user. By this way privacy is preserved to the end user of clustering results. But the mechanism can leak privacy at computation end. Shang et al (2017) [9] solved the problems in differential privacy-based K means algorithm by an optimized canopy algorithm. The number of clusters and the cluster points are decided by the optimized canopy algorithm. Differential privacy-based K means algorithm starts from the results of optimized canopy algorithm. Though privacy preservation is achieved for the cluster centroids, privacy is leaked, when the solution is extended for outsourced clustering model. Zhang et al (2017) [10] proposed a privacy preserving c mean clustering algorithm for big data clustering. The data uploaded to cloud are encrypted using BGV homomorphic encryption technique.

Lin et al (2016) [13] proposed a privacy preserving k-means outsourced model based on randomized kernel matrix. The data contents are encrypted in the randomized kernel matrix and outsourced. At the remote end, the kernel k-means is solved. Service provider is not aware of the data and the cluster centers. The approach is not scalable for large and incremental dataset. Gheid et al (2016) [14] used multiparty additive scheme for privacy preserving K means. The scheme is designed for horizontally partitioned dataset. The solution avoided cryptographic operation to scale to large volume of dataset. Each data owner computes cluster center for their data they own and do a multiparty additive transformation before sharing the cluster center. From the cluster centers, new cluster center is found through a sum method. Though the solution scales well for big data, it not suitable for outsourcing. Hu et al (2018) [15] proposed a privacy preserving K means algorithm based on the concept of negative database. The records are transformed to negative database and distance computation operation of clustering is transformed to estimation from binary string. By this way privacy of the data is preserved for outsourcing.

Zhao et al (2019) [16] proposed a negative database generation algorithm to assist in privacy preserving k means. The distance estimation accuracy can be controlled in a fine-grained manner so that clustering results can also be controlled in granular way. Zhao et al (2021) [17] improvised his earlier work of negative database generation algorithm with consideration for both privacy and accuracy of clustering. Authors introduced new set of parameters to control the selection of different bits in generation of negative database records. Negative database-based methods have two problems – conversion to negative database is not possible for all kinds of data and there is huge overhead on data owner side for negative database construction. Brando et al (2021) [18] proposed a distributed privacy preserving K-mean algorithm. Client compute K-mean for their data locally and send the centroids to a server. Server computes the global centroids based on centroids provided by each client. To ensure the privacy of centroids, each client encrypts the centroid using homomorphic encryption. In addition to centroid, client must also send some statistics about data to server. Due to this, the method cannot work for incremental data. Jiang et al (2020) [19] proposed a two-party collaborative K-means clustering over encrypted data. The protocol is also secure against any party providing getting malicious in centroid computation. The communication complexity is higher when extended for multiple parties.

Almutairi et al (2018) [20] proposed a secure data clustering algorithm with owner participation needed only in encryption stage. An encrypted data matrix is constructed with distance between each records computed and encrypted using homomorphic encryption. This encrypted data matrix is offloaded for privacy preserving clustering. For big data, the cost of computation of encrypted distance matrix is very high and the work is not suitable for incremental dataset.
The problems in implementing privacy preserved clustering as outsourcing model for large data in existing works is summarized in Table 1.

Table 1. Literature summary

Methodology	**Problems**
Zobaed et al (2020)	Performance overhead due to re-clustering is very high
Wei fu et al (2019)	High overhead on querying user in applying repression to detect cluster membership
Mary et al (2012)	Overhead at owner to encrypt all the data and upload to cloud.
Rong et al (2017)	Involves frequent interaction between querying user and cloud service provider
Yuan et al (2019)	
Rao et al (2015)	Use of single key across multiple data owners.
Zou et al (2020)	Higher computation complexity for clustering
Jiang et al (2020)	
Yu et al (2016)	Consider only the privacy of clustering results

Shang et al (2017)	
Zhang et al (2017) [11]	Not scalable and does not support incremental dataset
Lin et al (2016)	
Almutairi et al (2018) [12]	
Gheid et al (2016)	Overhead at owner end to compute initial centroids
Brando et al (2021)	
Hu et al (2018)	Conversion to negative database is not possible for all kinds of data and there is huge overhead on data owner side for negative database construction
Zhao et al (2019)	
Zhao et al (2021)	

III. HYBRID CLOUD BASED PRIVACY PRESERVING CLUSTERING

Hybrid cloud combines the company owned on premise private cloud and third party owned off premise cloud infrastructure. Though there are multiple rationale for adopting the hybrid cloud like guaranteeing a minimum quality of service, satisfy deadline constraints etc, in this work, hybrid cloud is adopted for ensuring security and privacy of the sensitive data. We adopt the framework for privacy aware computing on hybrid cloud proposed by Xu et al (2015) [21]. This framework processes the data set with n attributes$\{y_1, y_2, \ldots y_n\}$. The subset of attributes in tagged as sensitive.

The dataset is vertically portioned into two sets: private part with attributes tagged as sensitive and public part with attributes tagged as not sensitive. The private part of the vertically partitioned data is kept in the private cloud and public part is kept in the public cloud. Locality sensitivity hashing (LSH) [22] is a method for approximate neighbor search in high dimensional space.

It maps the high dimensional data to lower dimensional representation using random hash function such a way that points closer in higher dimensional space maps to same low dimensional space with higher probability. LSH hashes the item repeatedly several times, so that similar items are more likely hashed to same bucket than dissimilar items as shown in Figure 1. Thus, to find the items in a database, which is similar to a query, LSH maps to most relevant bucket and number of buckets are also less. Due to this lookup becomes faster in LSH compared to hashing based lookup.

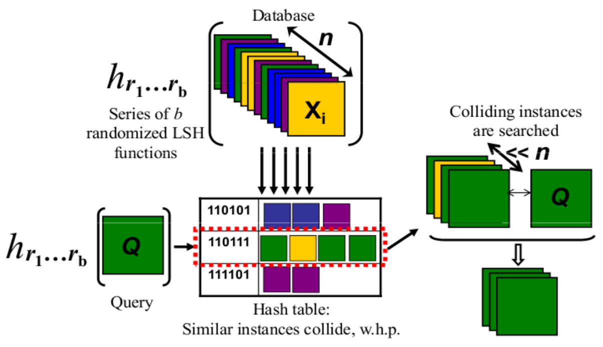
Database
n
X_i
$h_{r_1 \ldots r_b}$
Series of b randomized LSH functions
Colliding instances are searched
<< n
Q
110101
110111
111101
$h_{r_1 \ldots r_b}$
Q
Query
Hash table: Similar instances collide, w.h.p.

Figure 1. LSH lookup

The idea of LSH is to construct hash functions $g: R^d \to U$ such that for any two points p, q

$$if\ \|p-q\| \le r, then\ Pr\ Pr\ [g(p) = g(q)]\ \ is\ high$$

$$if\ \|p-q\| > cr, then\ Pr\ Pr\ [g(p) = g(q)]\ \ is\ small$$

This is achieved with family H of functions

$g(p) =< h_1(p), h_2(p), \ldots . h_k(p) >$

For all data point $p \epsilon P$, p is hashed to buckets $g_1(p), g_2(p), \ldots g_b(p)$

For an input query q, the points are retrieved from the buckets $g_1(q), g_2(q), \ldots$ until all points from b buckets are retrieved.

The effectiveness of LSH comes from use of multiple hash functions instead of single hash. Multiple hash reduces the number of buckets needed for mapping the items in the database. Since the volume of big data is huge, LSH reduces the search space and look up for data becomes faster. Thus, LSH is more suited for big data related lookup operations. For a dataset of size N, in brute force, time taken for similarity matching between each pair for clustering is

$$\frac{N!}{2!(N-2)!} \sim \frac{N^2}{2} = O(N^2) \qquad (1)$$

But, LSH reduces this time to *O(N)* by matching to buckets.

Many variants of LSH have been proposed [23-27]. LSH based clustering is found to scale well for clustering large volume of dataset [28-31]. The existing LSH based methods suffers two important problems

1. Higher error in mapping to bucket for incremental dataset

2. Inefficient to support, attribute selection for clustering

Most LSH method uses all the data points to calculate the hash code for the points. But when the data distribution in incremental data differs from previous, matching to bucket becomes erroneous. Also, most LSH methods constructs bucket based on all the dimensions. When querying user wants to cluster on certain attributes alone, distinct LSH has to be developed with those attributes alone. In high dimensional data space, the number of combinations of attribute is very high making LSH based clustering inefficient. To solve this problem, this work proposes a way of hashing data points in such a way to satisfy all the below requirements

1. Ensure privacy of the data
2. Distance preservation in the hashed data, so that it suitable for clustering
3. Attribute selection for clustering
4. Support incremental dataset

This work assumes that maximum and minimum value is known priori for each attribute in the dataset. The assumption is valid as in most domains the attribute possible values are known in advance. Say in a Forest cover dataset used in our work has attributes such as “Hillshade_9am”, “Wilderness_ Area” having values 0 to 255 and either 0 or 1 respectively. Every time when a batch of data arrives at private cloud, data transformation procedure is invoked. Say the data has N rows with d attributes. In each row, the attributes are first normalized as

$$x_{ti} = \frac{F(x_i) - min}{max_i - min_i}$$
(2)

Where $F(x_i)$, is the numerical value of the attribute i. The function F gives the same value x_i if x_i is a numerical value and returns an index in case x_i is a categorical value. The categorical values are sorted, and index is returned as numerical value. The x_{ti} is rounded to one decimal. The rounded x_{ti} is transformed to 10-digit binary code as below

B $(x_{ti}) = \{1, x_{ti} * 10 \ \ 0, elsewhere$
(3)

Once each attribute value is converted to binary vector, the binary vectors are shuffled in positions according to a random key, which is known only at the private cloud. Each row in the incremental dataset is transformed and the transformed dataset is moved to public cloud. Without the knowledge of shuffle key, the curious public cloud cannot infer any information from the transformed dataset, guaranteeing privacy of the dataset. Say their dimension of all attributes in the transformed dataset is D, and there are N rows, from the high dimensional (N, D) matrix, a signature matrix of size (N, K) is generated with K as the number of hash functions. In this work, for each attribute one hash function is designed. MinHash function is used for as the hash function. Since the transformed dataset is a binary vector, MinHash is the most suitable hashing function. The query to cluster the dataset based on selected list of attributes is given via a secure channel to the private cloud. The private cloud generates a binary mask of size $d * 10$. The mask is filled with 1 in places where the queried attributes are present and filled with 0 over the rest of the position. The mask and number of cluster k is sent to the public cloud to invoke clustering. The algorithm for binary mask construction is given in algorithm 1 At the public cloud, Locality Sensitive Hashing for Transformed Dataset (LSHTD) is invoked as the first step before clustering. The LSH initialization and hash computation function proposed in [32] is modified for computing the LSH hash of transformed data set items. This is done to adapt LSH for attribute selection and to support incremental dataset. Instead of generating G_u^t randomly, transformed data items are first AND with the mask. U items with larger difference in hamming distance to the median are selected for G_u^t. Algorithm 2 shows the LSHTD Locality sensitive hashing for transformed dataset.

```
Initialization of a hash function h∈H
1  For $u = 1$ to $U$, choose a random shift $s_u \in [0,4w]^t$, which specifies the grid $G_u^t = G^t + s_u$ in the $t$-dimensional Euclidean space.
2  Choose a matrix $A \in M_{t,d}$, where each element $A_{ij}$ is distributed according to the normal distribution $N(0,1)$ times a scaling factor, $\frac{1}{\sqrt{t}}$. The matrix $A$ represents a random projection from $R^d$ to $R^t$.
   Computing $h()$ on a point $p \in R^d$
3  Let $p' = Ap$ be the projection of the point $p$ onto the $t$ dimensional subspace given by $A$.
4  For each $u = 1,2, \dots U$
5  Check whether $B(p', w) \cap G_u^t \neq \emptyset$, i.e., whether there exist some $(x_1, x_2, \dots x_t) \in G_u^t$ such that $p \in B\big((x_1, x_2, \dots x_t), w\big)$
6  Once we find such $(x_1, x_2, \dots x_t)$, set $h(p) = (u, x_1, x_2, \dots x_t)$, and stop.
7  Return $0^{t+1}$ if we do not find any such ball.
```

Algorithm 1.LSH proposed in [32]

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6037
***Article Received**: 10 November 2022 **Revised**: 18 December 2022 **Accepted**: 05 January 2023*

The pseudo code for the LSHTD algorithm is given below

```
Initialization of a hash function h $\varepsilon$ Ħ
Input: Binary mask m
1.  For all transformed items
      Transformed items transformed items & m
2.  SDSort the transformed items in descending order of hamming distance to median of the transformed items.
3.  Select U first items from SD as $G_u^t$
Computing h ( ) on a point $p \ \varepsilon \ R^d$
1. $p = p \,\&\, m$
2. For each u =1, 2,..U
    3.Check whether B$(p', w)$ ∩ $G_u^t \neq \emptyset$ , $ie$ whether there exist some $(x_1, x_2, \dots x_t) \ \varepsilon \ G_u^t$ such that $p \ \varepsilon \ B((x_1, x_2, \dots x_t), w)$
    4. Once we find such $(x_1, x_2, \dots x_t), set\ h(p) = (u, x_1, x_2, \dots x_t)$
```

Algorithm 2 .LSHTD Locality sensitive hashing for transformed dataset.

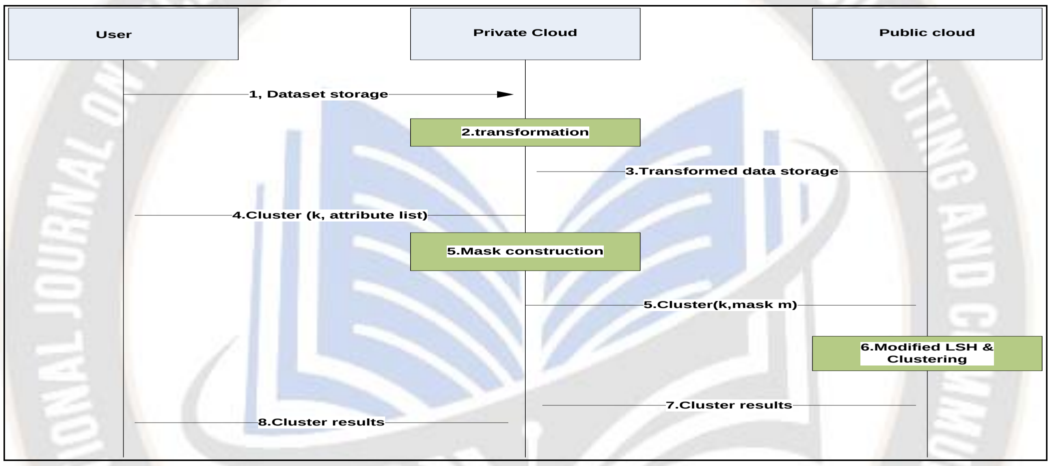

Figure 2. Interaction flow

```
Algorithm: binarymask
Input: attribute list
Output : binary mask M
drandom_shifkey(d)
M[]
For each attribute a in d
    If a in the attribute list
        MM + $\prod_1^{10}$    1
    else
        MM + $\prod_1^{10}$    0
    End
End
Return M
```

Algorithm 3. Binary mask algorithm

With binary mask LSH is adapted for attribute selection-based hash keys. At the end of LSH processing, each transformed item as one of U hash value. Items having the same hash value belong to the same cluster. Figure 2 shows the Interaction flow. The queried number of cluster k is usually less than U. Algorithm 3 shows the Binary mask algorithm. Hamming distance is used for distance calculation between two U points. The U points are merged based on closest distance repeatedly till the U value is reduced to number of cluster k. While merging hash value of any of U points is taken as label for the merged U points. When the number U points are reduced to k, the items with same U hash point are labeled as one cluster. The index of items in each cluster is then returned as output to the private cloud. Private cloud forwards the result to the query user. Algorithm 4 shows the LSHTD clustering algorithm.

```
Algorithm: LSHTD cluster
Input : hash of each data item , number of cluster k, number of hash item U
Ouput: Map of cluster label vs indexes
While U > k
    L<h1,h2>= Calculate distance between each pair of U hash        items
    Sort L<h1,h2>
    Merge L[0] h1, L[0] h2 by replacing h2 with h1
    U=U-1
End
Clusterlabel=1
Result=[]
For h in distinct hash items
    Indexes□Get all index with hash h
    Results.append(Clusterlabel,indexes)
    Clusterlabel=clusterlabel+1
End
Return Result
```

Algorithm 4. LSHTD clustering algorithm

The data owner uploads the data to be stored to private cloud. The data is transformed and the transformed data is stored to public cloud. User can request to cluster data with number of clusters and attribute list. The transformed data on public cloud is clustered using modified LSH based clustering and the clustering results are returned to user. Since the clustering is done on transformed data at public cloud, privacy is ensured even if public data is compromised. The mask needed for clustering is constructed at private cloud and without it, it is not possible to cluster the dataset. Thus, even if the transformed data at public cloud is compromised, without mask, it is not possible to execute any data mining operations like clustering on the data. The clustering is implemented as outsourced service on public cloud with aid of private cloud. Thus, private cloud can charge the users for clustering as service at fine grained level based on the attribute list, number of clusters and volume of data for clustering.

IV. RESULTS

The performance of the proposed-on fly hybrid data stream clustering algorithm is tested for 4 datasets – two synthetic and two real world datasets.

ExclaStar and MRDS are the two synthetic datasets used in this work. Exclastar is a synthetic dataset with 240 points for the star, 391 for the bar and 124 for the pie. MRDS is a synthetic data set of 38.7K records, containing 4 classes with 10% noise [37]. More details of the ExcalStar and MRDS dataset can be found in [38].

Household electricity power consumption, Stock keeping unit, KDD-99 and Forest cover type dataset from UCI machine learning repository [33] are used as real-world datasets. KDD-99 dataset has 4000000 instances with 42 attributes and 67 classes. Due to high complexity, this dataset is selected for testing the clustering efficiency in this work. Forest cover datasets contains tree observations from four areas of the Roosevelt National Forest in Colorado. The dataset has 581012 instances with 54 attributes and 7 different classes belonging to different forest cover. The static dataset is converted to data stream mode using Stream [34]. Stream provides an intuitive interface for experimenting with data streams and data stream algorithms. The performance of proposed solution is compared with Privacy preserving k-means clustering outsourcing under multiple keys (PPCOM) [4] and Secure nearest neighbor clustering (SNNC) [20]. PPCOM used double decryption cryptosystem to transform the encrypted data so that operations like addition, subtraction, multiplication, comparison and equality can be executed every time after decryption of the data and encrypting back the results. In SSNC, the data is transformed using Homomorphic encryption and all the operations like distance computation, centroid computation for clustering are done on Homomorphic encrypted data. In both methods, the cost of computation is very high and it increases linearly for streaming data where new batch of data arrives in some time.

The performance is compared in terms of

1. Purity
2. Adjusted Rand Index (ARI)
3. Computation time
4. Cloud communication cost

Purity measures the stream clustering accuracy. It is calculated as

$$Purity = \frac{\sum_{i=1}^{K} \frac{|c_i^d|}{|c_i|}}{K} \times 100 \quad (4)$$

Where K is the number of clusters, $|c_i^d|$ is the number of points with dominant class label in cluster i and $|c_i|$ is the number of points in the cluster.

Adjusted Rand Index [35] and Purity are external measures which evaluates the results based on ground truth. Adjusted Rand Index measures the similarity between original class partitioning and clustering. It is calculated as

$$ARI = \frac{RI - expected(RI)}{((RI) - expected(RI)} \quad (5)$$

Where Random index (RI) [36] is calculated as

$$RI = \frac{a+b}{\langle \frac{n}{2} \rangle} \quad (6)$$

Where a is the number of pairs of elements belonging to the same partition in the class set, C, to the same partition in the clustering set, K. b is the number of pairs of elements belonging to different partition in the class set, C, to different partition in the clustering set K and n is the number of data elements. The value of ARI ranges from 0 to 1. The value towards 1 is interpreted as accurate clustering and towards 0 is interpreted as bad clustering.

Each data instance in the dataset is assigned with a cluster label and this is used as ground truth for clustering evolution. The results for purity across the solutions for different datasets are given in table2.

Table 2. Different datasets solutions

Dataset	**Purity**		
	PPCOM	**SNNC**	**Proposed**
Exclastar	0.81	0.80	0.88
MRDS	0.84	0.82	0.89
KDD-99	0.87	0.84	0.91
Forest cover	0.86	0.86	0.90

The purity in the proposed solution is on average 5.58% higher compared to PPCOM and 7.2% higher compared to SNNC. The faster adaptivity to incremental data with use of LSHTD has increased the clustering accuracy in terms of purity in the proposed solution. Figure 3 shows purity comparison.

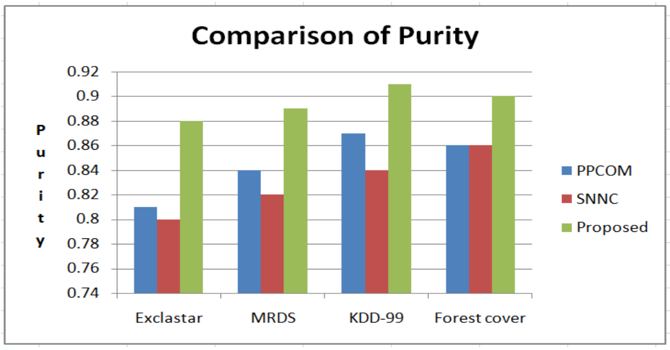

Figure 3. Purity comparison

The results for ARI across the solutions for different dataset are given in table3.

Table 3. Different datasets solutions

Dataset	**ARI**		
	PPCOM	**SNNC**	**Proposed**
Exclastar	0.80	0.79	0.86
MRDS	0.83	0.81	0.87
KDD-99	0.84	0.81	0.87
Forest cover	0.85	0.82	0.88

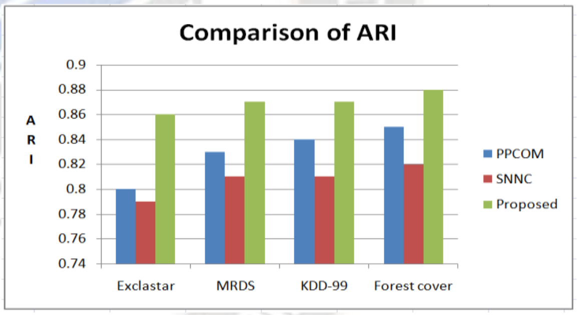

Figure 4. ARI comparison

The ARI in the proposed solution is on average 4.59% higher compared to PPCOM and 8.04% higher compared to SNNC. The ARI is improved in the proposed solution due to selection of U seed points based on diversity in hamming distance and reduction from U to K using distance-based merging. Figure 4 shows ARI comparison. The results for computation time across the solutions for different datasets is given in table4.

Table 4. Different datasets solutions

Dataset	**Computation time (min)**		
	PPCOM	**SNNC**	**Proposed**
Exclastar	67	52	42
MRDS	66	46	38
KDD-99	35	31	24
Forest cover	39	33	27

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6037
***Article Received**: 10 November 2022 **Revised**: 18 December 2022 **Accepted**: 05 January 2023*

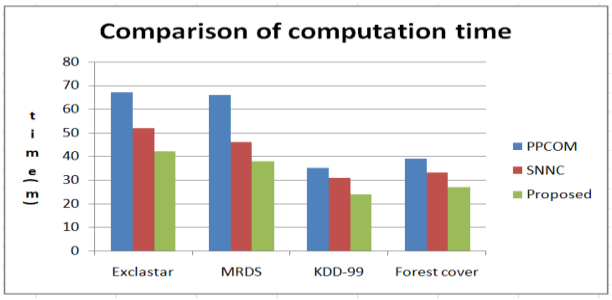

Figure 5. Computation time comparison

The computation time in the proposed solution is on average 58.01 % lower compared to PPCOM and 23.66% lower compared to SNNC. The computation time has reduced in proposed solution due to avoidance of cryptographic primitives for data transformation and iterative procedures in clustering. Figure 5 shows computation time comparison. The results for communication cost across the solutions for different datasets is given in table5.

Table 5. Different datasets solutions

Dataset	**Communication cost (MB)**		
	PPCOM	**SNNC**	**Proposed**
Exclastar	2000	1920	1817
MRDS	2500	2100	2010
KDD-99	300	252	201
Forest cover	320	292	241

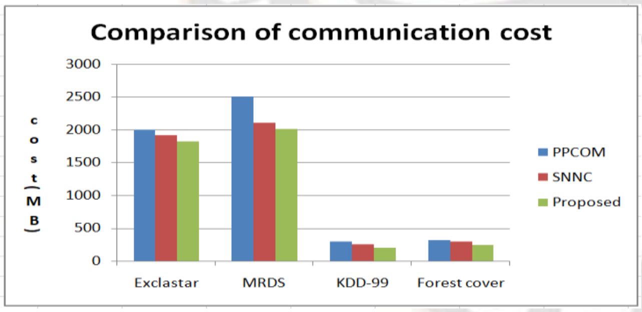

Figure 6. Communication cost comparison

The communication cost in the proposed solution has on average reduced by 19.9% compared to PPCOM and 6.93% compared to SNNC. Figure 6 shows Communication cost comparison. The only communication factor in the proposed solution is transmission of transformed data, query and query results. The size of transformed data in the proposed solution is low compared to transformed data in PPCOM and SNNC.

V. CONCLUSION

This work proposed a hybrid cloud-based privacy preserving clustering as outsourcing model. As part of the solution, LSHTD, a lightweight data transformation, LSH based clustering is proposed. The proposed solution LSHTD is 40.83% computationally efficient relative to mean efficiency of PPCOM and SNNC, because the LSHTD avoids cryptographic primitives and iterative clustering procedures. Also, the proposed LSHTD reduces communication cost by 13.41% compared with mean of communication cost of PPCOM and SNNC. The communication cost has also reduced due to lower data consuming, data transformation. In addition, the proposed solution is able to adapt to incremental data as reveled by purity and ARI results. Extending the solution to a hierarchical clustering model is in the scope of future work.

Declaration:

Ethics Approval and Consent to Participate: No participation of humans takes place in this implementation process
Human and Animal Rights: No violation of Human and Animal Rights is involved.
Funding: No funding is involved in this work. Conflict of Interest: Conflict of Interest is not applicable in this work.
Authorship contributions: There is no authorship contribution
Acknowledgment: No acknowledge

REFERENCES

[1] Zobaed, S., Gottumukkala, R.N., & Salehi, M. (2020). Privacy-Preserving Clustering of Unstructured Big Data for Cloud-Based Enterprise Search Solutions. *ArXiv, abs/2005.11317*.
[2] Wei Fu and Patrick O Perry. Estimating the number of clusters using cross-validation. Journal of Computational and Graphical Statistics, pages 1–12, January 2019.
[3] Angel Latha Mary and KR Shankar Kumar. A density based dynamic data clustering algorithm based on incremental dataset. Journal of Computer Science, 8(5):656–664, February 2012
[4] Hong Rong, Huimei Wang, Jian Liu, Jialu Hao, Ming Xian, "Privacy-Preserving -Means Clustering under Multiowner Setting in Distributed Cloud Environments", Security and Communication Networks, vol. 2017, Article ID 3910126, 19 pages, 2017
[5] J. Yuan and Y. Tian, "Practical Privacy-Preserving MapReduce Based K-Means Clustering Over Large-Scale Dataset" in IEEE Transactions on Cloud Computing, vol. 7, no. 02, pp. 568-579, 2019
[6] F.-Y. Rao, B. K. Samanthula, E. Bertino, X. Yi, and D. Liu, "Privacy-preserving and outsourced multi-user k-means clustering," in Proceedings of the 1st IEEE International Conference on Collaboration and Internet

Computing, CIC 2015, pp. 80–89, October 2015.

[7] Ying Zou, Zhen Zhao, Sha Shi, Lei Wang, Yunfeng Peng, Yuan Ping, Baocang Wang, "Highly Secure Privacy-Preserving Outsourced k-Means Clustering under Multiple Keys in Cloud Computing", Security and Communication Networks, vol. 2020

[8] Q. Yu, Y. Luo, C. Chen, and X. Ding, “Outlier-eliminated k-means clustering algorithm based on differential privacy preservation,” Applied Intelligence, vol. 45, no. 4, pp. 1179–1191, 2016.

[9] T. Shang, Z. Zhao, Z. Guan, and J. Liu, “A DP canopy k-means algorithm for privacy preservation of hadoop platform,” in Proceedings of the CSS 2017, Lecture Notes in Computer Science, vol. 10581, pp. 189–198, Springer, Xi’an, China, October 2017

[10] H. Rong, H. Wang, J. Liu, J. Hao, and M. Xian, “Outsourced k-means clustering over encrypted data under multiple keys in spark framework,” in Proceedings of the SecureComm 2017, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 238, pp. 67–87, Springer, Niagara Falls, ON, Canada, October 2017.

[11] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, “Pphopcm: privacy-preserving high-order possibilistic c-means algorithm for big data clustering with cloud computing,” IEEE Transactions on Big Data, 2017.

[12] N. Almutairi, F. Coenen, and K. Dures, “K-means clustering using homomorphic encryption and an updatable distance matrix: secure third party data clustering with limited data owner interaction,” in Proceedings of the DaWaK 2017, Lecture Notes in Computer Science, vol. 10440, pp. 274–285, Springer, Lyon, France, August 2017.

[13] K.-P. Lin, “Privacy-preserving kernel k-means clustering outsourcing with random transformation,” Knowledge and Information Systems, vol. 49, no. 3, pp. 885–908, 2016

[14] Z. Gheid and Y. Challal, “Efficient and privacy-preserving k-means clustering for big data mining,” in Proceedings of the 2016 IEEE Trustcom/BigDataSE/ISPA, pp. 791–798, IEEE, Tianjin, China, August 2016.

[15] Hu Xiaoyi,Lu Liping,"Privacy-Preserving K-Means Clustering Upon Negative Databases",Springer International Publishing,2018

[16] D. Zhao *et al.*, "A Fine-grained Privacy-preserving k-means Clustering Algorithm Upon Negative Databases," *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 1945-1951

[17] Dongdong Zhao, Xiaoyi Hu, Shengwu Xiong, Jing Tian,"k-means clustering and kNN classification based on negative databases",Applied Soft Computing,2021

[18] Brandão, André & Mendes, Ricardo & Vilela, Joao. (2021). Efficient Privacy Preserving Distributed K-Means for Non-IID Data. 10.1007/978-3-030-74251-5_35.

[19] Jiang, Z.L., Guo, N., Jin, Y., Lv, J., Wu, Y., Liu, Z., Fang, J., Yiu, S.M., Wang, X.: Efficient two-party privacy-preserving collaborative k-means clustering protocol supporting both storage and computation outsourcing. Information Sciences 518, 168–180 (2020)

[20] Almutairi, N., Coenen, F., & Dures, K. (2018). Third Party Data Clustering Over Encrypted Data Without Data Owner Participation: Introducing the Encrypted Distance Matrix. *DaWaK.*

[21] X. Xu and X. Zhao, "A Framework for Privacy-Aware Computing on Hybrid Clouds with Mixed-Sensitivity Data," 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems, 2015

[22] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the twentieth annual symposium on Computational geometry. 253–262.

[23] Mayank Bawa, Tyson Condie, and Prasanna Ganesan. 2005. LSH forest: self-tuning indexes for similarity search. In Proceedings of the 14th international conference on World Wide Web. 651–660

[24] Junhao Gan, Jianlin Feng, Qiong Fang, and Wilfred Ng. 2012. Locality-sensitive hashing scheme based on dynamic collision counting. In Proceedings of the 2012 ACM SIGMOD international conference on management of data. 541–552.

[25] Qiang Huang, Jianlin Feng, Yikai Zhang, Qiong Fang, and Wilfred Ng. 2015. Query-aware locality-sensitive hashing for approximate nearest neighbor search. Proceedings of the VLDB Endowment 9, 1 (2015), 1–12

[26] Wanqi Liu, Hanchen Wang, Ying Zhang, Wei Wang, and Lu Qin. 2019. I-LSH: I/O efficient c-approximate nearest neighbor search in highdimensional space. In 2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE, 1670–167

[27] Yifang Sun, Wei Wang, Jianbin Qin, Ying Zhang, and Xuemin Lin. 2014. SRS: solving c-approximate nearest neighbor queries in high dimensional euclidean space with a tiny index. Proceedings of the VLDB Endowment (2014).

[28] McConville, R., Cao, X., Liu, W., & Miller, P. (2016). Accelerating Large Scale Centroid-based Clustering with Locality Sensitive Hashing. In Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE) (pp. 649-660). Institute of Electrical and Electronics Engineers (IEEE).

[29] C. Oprişa, M. Checicheş and A. Năndrean, "Locality-sensitive hashing optimizations for fast malware clustering," *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2014, pp. 97-104

[30] Jafari, Omid & Maurya, Preeti & Nagarkar, Parth & Islam, Khandker & Crushev, Chidambaram. (2021). A Survey on Locality Sensitive Hashing Algorithms and

their Applications.
[31] Khader, Mariam & Al-Naymat, Ghazi. (2020). Density-based Algorithms for Big Data Clustering Using MapReduce Framework: A Comprehensive Study. ACM Computing Surveys. 53. 1-38. 10.1145/3403951.
[32] Veeraiah N, Krishna BT. Intrusion detection based on piecewise fuzzy C-means clustering and fuzzy Naïve Bayes rule. Multimedia Research. 2018 Oct;1(1):27-3.
[33] https://archive.ics.uci.edu/ml/datasets/
[34] M. Hahsler, M. Bolanos, and J. Forrest, stream: Infrastructure for Data Stream Mining, 2015, R package version 1.2-2.
[35] N.X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clustering's comparison: is a correction for chance necessary? in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 1073–1080.
[36] Rand, W. M. (1971) Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association, 66, 846–850
[37] L. Wan, W.K. Ng, X.H. Dang, P.S. Yu, K. Zhang, Density-based clustering of data streams at multiple resolutions, ACM Trans. Knowl. Discov. Data 3 (3) (2009) 49–50.
[38] Xu, Ji & Wang, Guoyin & Li, Tianrui & Deng, Weihui & Guanglei, Gou. (2017). Fat Node Leading Tree for Data Stream Clustering with Density Peaks. Knowledge-Based Systems. 120. 99-117. 10.1016/j.knosys.2016.12.025.