TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 11/18 pp127–140
DOI: 10.26599/TST.2020.9010031
Volume 27, Number 1, February 2022

IDEA: A Utility-Enhanced Approach to Incomplete Data Stream Anonymization

Lu Yang, Xingshu Chen, Yonggang Luo*, Xiao Lan, and Wei Wang

Abstract: The prevalence of missing values in the data streams collected in real environments makes them impossible to ignore in the privacy preservation of data streams. However, the development of most privacy preservation methods does not consider missing values. A few researches allow them to participate in data anonymization but introduce extra considerable information loss. To balance the utility and privacy preservation of incomplete data streams, we present a utility-enhanced approach for Incomplete Data strEam Anonymization (IDEA). In this approach, a slide-window-based processing framework is introduced to anonymize data streams continuously, in which each tuple can be output with clustering or anonymized clusters. We consider the dimensions of attribute and tuple as the similarity measurement, which enables the clustering between incomplete records and complete records and generates the cluster with minimal information loss. To avoid the missing value pollution, we propose a generalization method that is based on maybe match for generalizing incomplete data. The experiments conducted on real datasets show that the proposed approach can efficiently anonymize incomplete data streams while effectively preserving utility.

Key words: anonymization; generalization; incomplete data streams; privacy preservation; utility

1 Introduction

In recent years, data streams have become an increasingly important data source in big data and Internet-of-Things (IoT) application scenarios, because they allow companies to gain insights into activities through real-time analysis and apply real-time intelligence to new business decisions, user product recommendations, cybersecurity alerts, and so on[1, 2]. However, most data streams contain personally identifiable information of individuals, thereby increasing the risk of privacy disclosure[3]. Any disregard for privacy protection leads to high economic losses and incalculable reputational damage. Therefore the privacy preservation of data streams has attracted increasing attention from industry and academia[4–7]. Anonymization is a commonly used privacy-preserving method that allows the sharing of sensitive data with a guarantee that the individuals, i.e., the subjects of the data, cannot be recognized[8]. Anonymization is mainly achieved through two techniques: generalization and suppression. In generalization, the values of Quasi-IDentifier attributes (QIDs) are replaced with general values. Suppression is the most general replacement method and can be regarded as a special form of generalization[9]. Since the introduction of k-anonymity[10], l-diversity[11], and t-closeness[12] principles, they have been used as the basis of

• Lu Yang is with the College of Computer Science, Sichuan University, Chengdu 610065, China. E-mail: ceciliachn@163.com.
• Xingshu Chen is with the School of Cyber Science and Engineering, Sichuan University, Chengdu 610065, China. E-mail: chenxsh@scu.edu.cn.
• Yonggang Luo, Xiao Lan, and Wei Wang are with the Cyber Science Research Institute, Sichuan University, Chengdu 610065, China. E-mail: iamlyg98@scu.edu.cn; lanxiao@scu.edu.cn; wwzqbx@hotmail.com.
∗ To whom correspondence should be addressed.
Manuscript received: 2020-7-14; revised: 2020-08-26; accepted: 2020-09-01

© The author(s) 2022. The articles published in this open access journal are distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/).

anonymization methods, which have performed well on static datasets[13–16].

Unlike anonymizing static datasets, the aforementioned anonymization methods face new challenges in the handling of data streams. First, a data stream is potentially infinite and flows continuously. Typically, static data are accessed repeatedly to reduce information loss. The multiple scans involved require a large amount of time and space. Such requirement is unacceptable for data streams. Hence, anonymizing data streams should involve a one-time scan and continuous processing. With regard to the timeliness of anonymized data, the long delay of tuples should be avoided during processing. Second, missing values are common in real data streams because of inconsistent acquisition equipment, system failures, or data loss. According to an investigation of the UCI machine learning repository[17], 11 out of 21 health-related datasets have missing values, and the percentage of these missing values even exceeds 15%. The statistics show that missing values occupy a considerable proportion of the real datasets. However, conventional anonymization approaches are developed under the assumption that data do not contain missing values. Hence, the treatment is to remove incomplete records prior to process. However, the direct deletion of incomplete records can involve great information loss. Therefore, we need to establish a method for handling the missing values in anonymization. As the most commonly used method, k-anonymity faces a challenge in anonymizing incomplete data streams. Specifically, k-anonymity is usually transformed into a clustering problem[18], which is aimed at finding a set of clusters, i.e., Equivalence Classes (EC), each of which contains at least k records. In the clustering task, the distance function to measure tuple similarity requires tuples with the same attributes and valid values on each attribute. Such requirement is difficult to satisfy in incomplete data streams. Hence, the lack of similarity measurement for incomplete data is an existing problem that needs to be resolved in incomplete data stream anonymization. Third, a few works have attempted to allow missing values to participate in anonymization, but such an approach introduces missing value pollution[17, 19]. The last step of anonymization is generalization. Given the existence of a missing value, all valid values of other tuples in the same equivalence class are replaced with the most general values (Fig. 1). This scenario creates missing value pollution, which affects the utility of anonymized data and may reduce the accuracy of subsequent data analysis results.

Considering the above issues, we propose a utility-enhanced method for the Incomplete Data strEam Anonymization (IDEA) in this paper. The main contributions of this work are as follows:

• Given the fast and continuous nature of data streams, this study applies a processing framework based on a sliding window to anonymize data streams continuously. The framework adopts a reuse strategy, in which the generalization of anonymized clusters is kept to output tuples. Tuples can choose to be output with a reuse cluster or a new cluster generated by clustering to achieve minimal information loss.

• A two-dimensional similarity measurement method is proposed. In this method, the distance from the dimensions of attribute and tuple is calculated to realize clustering between complete tuples and tuples with missing values. The generated clusters are relatively compact, and the information loss is minimal.

• A generalization method based on maybe match is put forward for incomplete data. Generalizing the clusters using this method can avoid extra missing value pollution.

• The effectiveness of the proposed approach is evaluated on two real datasets. The experimental results show that our approach is efficient and superior to state-of-the-art methods in terms of data utility preservation.

The rest of this paper is organized as follows. In Section 2, the related work in the field of anonymization of data streams and incomplete data is briefly introduced. In Section 3, the basic ideas and important definitions

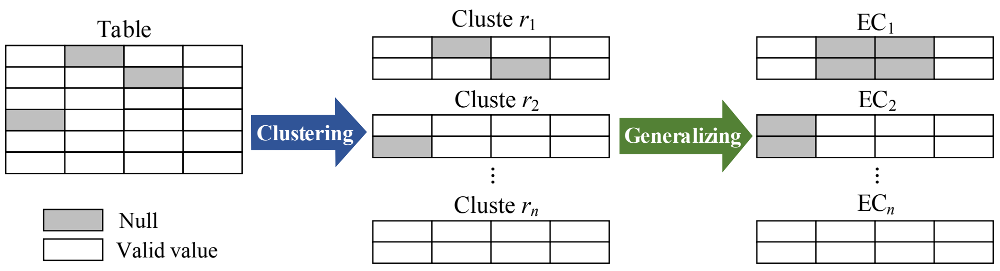


Fig. 1 Missing value pollution in generalization.

of the anonymization algorithm for incomplete data streams are explained in detail. In Section 4, a detailed description of the IDEA algorithm is provided. In Section 5, the results of an extensive experimental evaluation of the proposed algorithm are presented. In Section 6, the conclusions are summarized, and future research directions are identified.

2 Related Work and Problem Analysis

2.1 Data stream anonymization

Generally, anonymization methods for data streams fall into two main categories: specialization tree-based method and clustering-based method.

Wang et al.[20] proposed a k-anonymity method for data streams with a specialization tree, which is defined as a directed tree, in which each node is the generalization of a tuple. The root node is the most general one, that is, the value of each of its attributes is the root value of the corresponding Domain Generalization Hierarchy (DGH). Another data stream k-anonymity algorithm based on sliding windows and the specialization tree KIDS[21] fully considers the data distribution density of a stream and improves the practicality of processing large amounts of data. A tree structure is also adopted in SKY[22]. For δ-constraint k-anonymity, the output stream order deviates from the input stream order. If $\delta = 0$, the output stream is the same as the input stream. In sum, all anonymization algorithms based on specialization trees need to maintain a tree structure in memory and constantly update the trees with newly arrived tuples. The node size of trees directly affects the efficiency of algorithms. If anonymized nodes are periodically deleted to limit tree size, the information loss may increase, because the generalization of deleted nodes is not useful for newly arrived tuples.

The primary task of anonymization is to form an equivalence class, i.e., a set of records that are indistinguishable from one another with respect to the QIDs. If an equivalence class is treated as a cluster[23, 24], anonymization can be taken as a clustering task. Continuously anonymizing streaming data via adaptive clustering (namely CASTLE)[25] is a clustering-based anonymization algorithm for data streams that implements the k-anonymity and l-diversity principles. CASTLE adds incoming tuples to the nearest cluster one by one and checks if the cluster hosting the expired tuples, i.e., the tuples that are set to expire has at least k tuples. If the hosting cluster does not meet the requirement, it is merged with adjacent clusters. In minimizing information loss, a large cluster is split before outputting, and the generalization of the anonymized cluster is kept in buffer to anonymize tuples. However, frequent merging and splitting operations increase the time complexity of algorithms, resulting in reduced efficiency. To overcome the limitation of data distribution in CASTLE, B-CASTLE[26] restricts the number of tuples of buffered anonymized clusters, thereby avoiding the cluster splitting operation and reducing runtime. Although the information loss of existing k-anonymity algorithms is acceptable, the algorithms themselves are time consuming. Assuming that data streams are sensitive to time delay, Zakerzadeh and Osborn[27] proposed a fast anonymization algorithm, called FAANST, for numerical streaming data with window processing. FAANST processes a limited number of tuples each time to simplify the cluster generalization process. The information loss threshold and cluster reuse mechanism in CASTLE are retained to ensure data availability and to reduce algorithm runtime. However, FAANST does not constrain the tuple cache time; thus, tuples may remain in the cache for a long time. The presence of these tuples is not conducive to subsequent data analyses. To solve this problem, Zakerzadeh and Osborn[28] proposed two extensions for FAANST in passive and proactive ways. These extensions ensure tuple output according to a soft deadline, called time delay, and thus guarantee that tuples do not stay in the cache for too long. Guo and Zhang[29] proposed a fast clustering-based anonymization algorithm for data streams, called Fast clustering-based k-Anonymization approach for Data Streams (FADS). This algorithm discards the merging and splitting operations in CASTLE. Here a newly arrived tuple considers whether or not an anonymized cluster covers it. If so, the tuple is output with the anonymized cluster's generalization. Otherwise, an output cluster is generated on the basis of the new tuple and its neighbors. FADS limits the number of anonymized clusters and reduces the time needed to find the covering anonymized cluster. The time and space complexity of algorithms are linear with the size of the data. In recent years, the research of data stream anonymization has focused on optimizing the clustering process or updating the sets of anonymized clusters to improve algorithm performance and to reduce average information loss[30–32]. New methods have been proposed for different scenarios, such as distributed environments, medical big data, and so on[33–35]. However, all of these methods are developed on complete data streams.

2.2 Anonymization for missing data

Most data anonymization technologies focus on complete datasets. Although a few anonymization methods[17, 19] for incomplete data have been proposed and have achieved an acceptable average information loss, they only work on static datasets and cannot be applied to dynamic streaming data.

To deal with attribute loss in the field of IoTs, Otgonbayar et al.[36] introduced a method to anonymize IoT data streams via partitioning. The IoT anonymization algorithm uses partitions to distinguish records with different sets of QIDs, i.e., records with the same missing attributes are divided into the same partition for processing. In conformance to the k-anonymity principle, the partition to be output randomly chooses the most similar and biggest partition to merge with until it becomes eligible to produce a k-anonymous cluster. Considering the data distribution of partitions, K-VARP[37] proposed the R-likeness coefficient, i.e., the number of tuples in the partition whose distance with expired tuple is less than the threshold R. The bigger the coefficient, the tighter the partition. This scenario also favors the partition's merging with the partition hosting the expired tuple. In these partition merging-based methods, the size of the merged partition may be greater than k and thereby causing additional information loss.

3 IDEA Framework

In this section, we present the general overview and several key concepts of the IDEA method. The algorithms are detailed in Section 4.

3.1 Framework overview

In light of the continuity and potentially infinite nature of data streams, we use a count-based sliding window to continuously process incomplete data streams, which are modeled as an infinite sequence of tuples with an incremental order.

Definition 1 (k-anonymization of an incomplete data stream) Let $S(\text{pid}, Q, A)$ be an incomplete data stream; here, pid is the tuple identity, $Q = \{q_1, \ldots, q_m\}$ is the set of QIDs, and $A = \{a_1, \ldots, a_n\}$ is the set of remaining attributes. S^* is the k-anonymized data stream generated from S where pid has been pruned, if S^* satisfies the following conditions:

- for $\forall t \in S, \exists t^* \in S^*$ corresponds to t;
- for $\forall t^* \in S^*$ $|\text{EC}(t^*)| \geqslant k$.

where EC (t^*) is the equivalence class of tuple t^*, and $|\text{EC}(t^*)|$ is the number of distinct values of pid of the tuples in $\text{EC}(t^*)$. The equivalence class is the set of tuples with the same generalized QID values.

Initially, the window is empty. Then, a new tuple comes and is buffered in the window. When the number of tuples in the window accumulates to a predefined threshold δ, the oldest tuple, also called the expired tuple, must be output. The window then slides again to accumulate new tuples. Assuming that each tuple arrives at the same time interval, a tuple stays in the window no longer than $\delta - 1$ time intervals. With this constraint, tuples do not become stuck in the memory for too long, and the freshness of the anonymized data can be preserved.

A partition is used to group and thereby distinguishing tuples with missing values on different attributes in the window. Each newly entered tuple is assigned to a partition according to the set of its valid QIDs, i.e., the QIDs with valid values. Another benefit of grouping is that tuples in the same partition can be regarded as a complete dataset with respect to the partition's attribute set, so that the method for anonymizing complete data can be applied to these tuples.

Definition 2 (Partition) A partition represents a collection of tuples with the same set of valid QIDs. We denote $P(Q_i)$ as the partition based on QID set Q_i; $P(Q_i) = \{t_1(\text{pid}_1, Q_i, A_1), \ldots, t_n(\text{pid}_n, Q_i, A_n)\}$.

Example 1 Consider the incomplete data stream $S(\text{pid}, \{q_1, q_2, q_3, q_4\}, A)$ in Fig. 2 (for simplicity, only QIDs are drawn). Newly arriving tuples are divided into partitions $P_1(\{q_3, q_4\})$, $P_2(\{q_1, q_2, q_4\})$, and $P_3(\{q_1, q_2, q_3, q_4\})$ according to their valid QID sets.

The goal of anonymization is to output equivalence classes. If a k-anonymous equivalence class is regarded as a cluster, then the implementation of k-anonymity in a data stream can be naturally transformed into a clustering task. Furthermore, because k-anonymity requires each equivalence class to have a size of at least k, the generated cluster accordingly contains at least k records. If the expired partition to which the expired tuple belongs has more than k tuples, then a cluster ready to be output can be formed with the expired tuple and its neighbor in the same partition. We refer to the operation as clusteringInPartition. However, if the size of

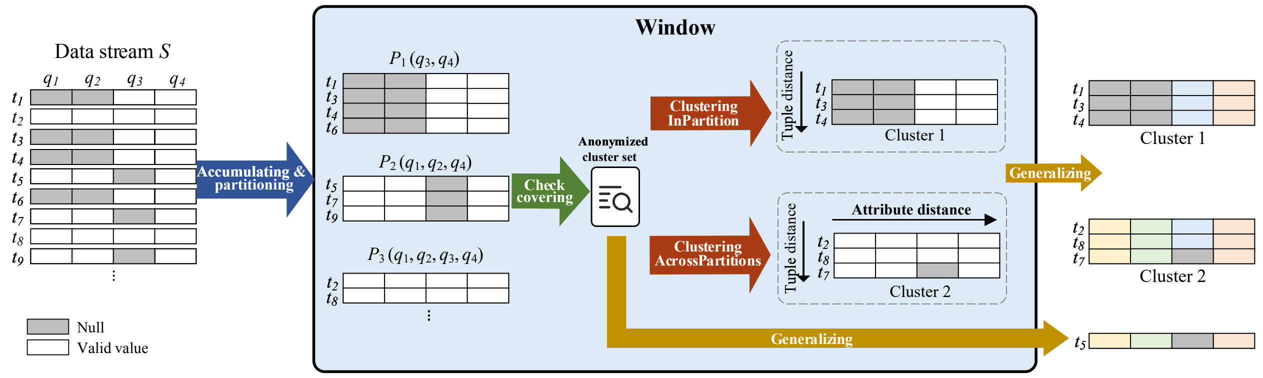


Fig. 2 Framework of IDEA.

the expired partition is less than k, the more partitions are required to generate a cluster. This operation is referred to as clusteringAcrossPartitions. In a special case, all partitions in the memory have less than k tuples, and the expired tuple can only be output with suppression individually.

As the data stream flows continuously and fast, window-based anonymization cannot realize the distribution of the entire data and only processes tuples of the current window. To mitigate these issues, we introduce an anonymized cluster reuse strategy, in which the anonymized cluster whose information loss falls within a given threshold is kept in buffer. The tuple covered by the anonymized cluster is output with the cluster's generalization without any clustering operation. Furthermore, the size of the reuse Cluster Set (CS) is restricted by a constraint β to solve the problem of the time and space complexity of the algorithm increasing infinitely as data continue to arrive.

3.2 Two-dimensional similarity

In a clustering task, the more similar the points, the more compact the generated cluster. In anonymization, the more compact a cluster the smaller the interval, which is used to replace the original value in generalization. Hence, the information loss of the generalized data is low. The value of each attribute is calculated in the distance function to measure the similarity of two records. However, the valid QIDs of tuples in an incomplete data stream are varied because of the existence of missing values. That is, the dissimilarity of these tuples comes not only from the distribution of QID values, but also from the variety of the valid QIDs owned by the records. Therefore, to achieve the clustering of incomplete data, we propose a two-dimensional similarity measurement consisting of attribute distance and tuple distance.

Attribute distance is defined on the basis of the Jaccard coefficient[38] that is used to measure the similarity between finite sets.

Definition 3 (Attribute distance) P_1 and P_2 are two partitions, and Q_1 and Q_2 are the QID sets of P_1 and P_2, respectively. The attribute distance of two partitions is defined as

$$\text{Distance}_{\text{A}}(P_1, P_2) = 1 - \frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} \tag{1}$$

Then, on the basis of the generalized loss metric[39], we define the tuple distance of incomplete data with consideration of numeric and categorical attributes.

Definition 4 (Tuple distance) The distance between two tuples $t_1(\text{pid}_1, Q_1, A_1)$ and $t_2(\text{pid}_2, Q_2, A_2)$ is calculated with the QIDs owned by both tuples at the same time,

$$\text{Distance}_{\text{T}}(t_1, t_2) = \frac{\sum\limits_{q_i \in Q_1 \cap Q_2} d(q_i)}{|Q_1 \cap Q_2|} \tag{2}$$

$$d(q_i) = \begin{cases} \dfrac{|v_{i,1} - v_{i,2}|}{|U_i - L_i|}, & q_i \text{ is numeric;} \\ \dfrac{|\text{leaves}(H_i)| - 1}{|\text{leaves}(\text{DGH}_i)| - 1}, & q_i \text{ is categorical} \end{cases} \tag{3}$$

where q_i is the QID shared between Q_1 and Q_2. If q_i is a numeric attribute, $v_{i,1}$ and $v_{i,2}$ represent the q_i's values of t_1 and t_2, respectively. U_i and L_i are the upper and lower bounds of q_i's value domain, respectively. If q_i is a categorical attribute, H_i represents the lowest common ancestor of the q_i values of t_1 and t_2, and DGH_i is the DGH of q_i. leaves(H_i) and leaves(DGH_i) denote the numbers of leaves of the subtree rooted at H_i and DGH_i, respectively. When the q_i's value of t_1 is that of t_2, $d(q_i) = 0$.

With the two-dimensional similarity measurement, the clustering of incomplete tuples can be divided into two steps. First, the attribute distance is used to select the partitions that are most similar to the expired partition. Second, the tuples most similar to the expired tuple are identified from the selected partitions to form the output cluster according to the tuple distance. These two steps iterate until the cluster size reaches requirement k of k-anonymity. The generated cluster has the smallest size to satisfy k-anonymity requirement and is as tight as possible with the two-dimensional similarity measurement, which immensely preserves the utility of the anonymized data. This operation is called clusteringAcrossPartitions.

Example 2 Consider that t_2 is set for output. P_3 to which t_2 belongs has less than 3 tuples, thus t_2 is output by clusteringAcrossPartitions. $\text{Distance}_{\text{A}}(P_3, P_1) = 1 - 2/4 = 0.5$ and $\text{Distance}_{\text{A}}(P_3, P_2) = 1 - 3/4 = 0.25$. P_2 is relatively close to P_3 and is therefore a candidate partition for P_3. Then the closest tuple t_7 in P_2 is selected to form a 3-anonymity cluster with t_2 and t_8.

As described in Section 3.1, tuples in a partition have the same QID set, and the attribute distance between them is thereby equal to 0. Therefore, in clusteringInPartition, one only needs to calculate the tuple distance to select the neighbors of the expired tuple to form the output cluster. The same is true for complete data anonymization, because tuples in the same partition can be regarded as a complete dataset over the partition's attribute set.

Example 3 Assuming that 3-anonymity is applied to data stream S, the expired tuple t_1 is set to be output. As the size of expired partition P_1 reaches k-anonymity requirement of 3, t_1 selects the two most similar tuples t_3 and t_4 in P_1 to form the output cluster according to the tuple distance. Take the distance between t_1(Null, Null, Male, Mid-school) and t_3(Null, Null, Female, High-school) as an example. With the DGH of gender and education shown in Fig. 3,

$$\text{Distance}_T(t_1, t_3) = \frac{1}{2} \times \left(\frac{\text{leaves(Gender)}}{\text{leaves(Gender)}} + \frac{\text{leaves(Secondary)}}{\text{leaves(Education)}}\right) = \frac{1}{2} \times \left(\frac{2}{2} + \frac{2}{5}\right) = 0.7.$$

3.3 Generalization of incomplete data

The anonymization of a cluster is mainly achieved by generalization (suppression is treated as a special case of generalization). Null is generally used to represent the missing information in database tables[40]. Different matching methods between Null and other values lead to different generalization results. The matching method

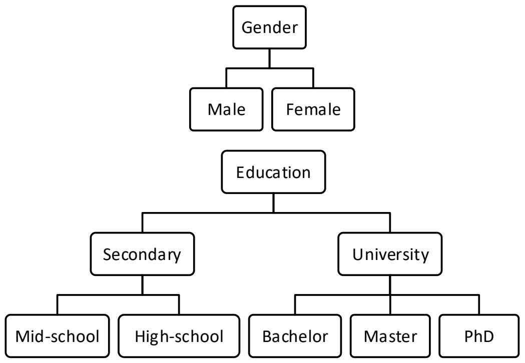


Fig. 3 Domain generalization hierarchy of gender and education.

of Null can be distinguished as three types[41] in the database domain: (1) basic match: Null values match neither Null values nor any other value; (2) extended match: Null values only match Null values and are treated as any other value; (3) maybe match: Null values match any value, including Null values, as wildcards for matching. In the generalization based on basic match, Null has to be removed before processing, because it does not match any value. In the generalization based on extended match, Null is treated as the root node in the DGH, and thus, all the values on the same attribute with Null in the equivalence class are generated to the most general value. Both generalization methods cause high information loss. As the reason beyond missing values is indeterminate, Null can be an arbitrary value. The definition of maybe match is in line with the actual value of Null, and the generalization based on maybe match causes less information loss than the other two matching methods.

Definition 5 (Maybe match) Let T be a table, and let Q be a set of QIDs. For any $q_i \in Q$, $t_1.q_i$, and $t_2.q_i$ are the QID q's values of t_1 and t_2, respectively. Let $\sim_m$ denote the relationship of maybe match, $t_1, t_2 \in T$ satisfy the maybe match $t_1 \sim_m t_2$, if t_1 and t_2 meet at least one of the following conditions:

- $t_1.q_i = t_2.q_i$;
- $t_1.q_i = \text{Null} \vee t_2.q_i = \text{Null}$.

The generalization of a cluster over an incomplete data stream is given on the basis of maybe match.

Definition 6 (Cluster generalization) Let $C(\{q_1, \ldots, q_n\})$ be a cluster of an incomplete data stream $S(\text{pid}, \{q_1, \ldots, q_n\}, A)$, and let $G(g_1, \ldots, g_n)$ be the generalization of C. For each q_i, $i \in [1, n]$, g_i is computed as follows:

- $g_i = [l_i, u_i]$, where l_i and u_i denote the minimum and maximum values of all the tuples in cluster C on q_i, respectively, if q_i is a numeric attribute;

• $g_i = H_i$, where H_i is the lowest common ancestor of the values of the tuples in cluster C on attribute q_i, if q_i is a categorical attribute;

• g_i = Null, only if the all the values in cluster C on q_i are Null.

Definition 7 (Tuple generalization) Let $t^*(q_1^*, \ldots, q_n^*, A)$ be the generalized tuple with respect to the tuple $t(\text{pid}, q_1, \ldots, q_n, A)$ with the generalization $g(g_1, \ldots, g_n)$. For each q_i^*, $i \in [1, n]$, q_i^* is computed as

$$q_i^* = \begin{cases} g_i, & q_i \neq \text{Null}; \\ q_i, & q_i = \text{Null} \end{cases} \tag{4}$$

In the generalization, Null is not replaced, and the specific value is generalized to a general value.

4 IDEA Algorithm

The main algorithm is IDEA( ), as shown in Algorithm 1, which continuously processes incoming data streams and outputs k-anonymized tuples. The data stream S, requirement k of k-anonymity, window size constraint δ, information loss threshold of reuse cluster τ, and reuse cluster set constraint β are the inputs of IDEA( ). IDEA( ) continuously reads a tuple t from data stream S and assigns it to a corresponding partition p according to its QID. If the matching partition is not found, then a new partition is generated on the basis of t's QID, and t is assigned to the new partition. When the number of tuples in P reaches δ, that is, the window is full, publishTuple( ) is triggered to process the oldest tuple in the buffer. When the data stream has no tuples, the remaining tuples in P are output with publishTuple( ).

Detailed in Algorithm 2, publishTuple( ) anonymizes tuple t that is ready to be output with different methods. As stated previously, the clustering method for expired tuple t varies with the size of expired partition p. If the size of p is not less than k, then t is output with clusteringInPartition( ). If p has less than k tuples and P has more than k tuples, then clusteringAcrossPartitions( ) is applied to output a cluster composed of t and tuples from other partitions. However, P has less than k tuples, then t is published with an eligible reuse cluster in CS covering t and has the smallest generalization information loss. If such cluster does not exist, t is suppressed.

Algorithm 3 describes clusteringInPartition ( ), where t is output either alone using the reuse cluster's generalization or with $k-1$ tuples from the same partition. First, clusteringInPartition (t, p) tries to find an eligible reuse cluster C_r in the reuse cluster set CS. Second, it picks out $k-1$ nearest tuples with t in p using the tuple distance and creates a new cluster C_n on them. If C_r exists and its information loss is less than that of C_n, t is output with C_r's generalization. Otherwise, the

Algorithm 1 IDEA ($S, k, \delta, \tau, \beta$)

```
1:  Let $P$ be a set of buffered partitions, initialized as $\varnothing$;
2:  Let CS be a set of $k$-anonymized clusters for reuse, initialized as $\varnothing$;
3:  while $S \neq \varnothing$ do
4:      Read a tuple $t$ from $S$;
5:      Assign $t$ into a partion of $P$ or create a new partition on $t$;
6:      if $\sum_{p' \in P} |p'| \geqslant \delta$ then
7:          publishTuple( );
8:      end if
9:  end while
10: while $\sum_{p' \in P} |p'| \geqslant 0$ do
11:     publishTuple( );
12: end while
```

Algorithm 2 publishTuple ( )

```
1:  Let $t$ be the expired tuple in $P$ and $p$ be the expired partition containing $t$;
2:  if $|p| \geqslant k$ then
3:      clusteringInPartition ($t$, $p$);
4:  else if $\sum_{p' \in P} |p'| \geqslant k$ then
5:      clusteringAcrossPartitions ($t$, $p$);
6:  else
7:      Find a cluster $C_r$ in CS which covers $t$ with minimum information loss;
8:      if $C_r \neq$ Null then
9:          Publish $t$ with $C_r$'s generalizations;
10:     else
11:         Suppress and publish $t$;
12:     end if
13:     Remove $t$ from $P$;
14: end if
```

Algorithm 3 clusteringInPartition (t, p)

```
1:  Find a cluster $C_r$ in CS which covers $t$ with minimum information loss;
2:  Find $k-1$ neatest neighbors of $t$ in $p$ by tuple distance and create a new cluster $C_n$ on $t$ and its neighbors;
3:  if $C_r \neq$ Null && $C_r$.infoloss < $C_n$.infoloss then
4:      Publish $t$ with $C_r$'s generalizations;
5:      Remove $t$ from $P$;
6:  else
7:      Publish cluster $C_n$;
8:      Remove all tuples of $C_n$ from $P$;
9:      updataReuseClusters ($C_n$);
10: end if
```

whole cluster C_n is generalized and inserted into CS for reuse.

updateReuseClusters ( ), shown in Algorithm 4, tries to insert output cluster C_n, whose generalization information loss is less than the threshold τ, into reuse set CS. If the size of CS exceeds constraint β after insertion, the oldest cluster will be removed from CS. With this limitation, the storage space of the reuse set and the time of needed to search for an eligible reuse cluster are reduced.

In clusteringAcrossPartitions ( ) in Algorithm 5, t is output either alone using the reuse cluster's generalization or with $k-1$ tuples across multiple partitions. Forming output cluster C_n involves two steps. Fisrt, partitions similar to the expired partition p are selected out according to attribute distance. Second, $k-1$ nearest neighbors with t are identified from the selected partitions with tuple distance, and output cluster C_n is formed with t and the identified nearest neighbors. Similar to that in clusteringInPartition ( ), if reuse cluster C_r covering t exists, the generalization of the cluster with smaller information loss between C_r and C_n is used to output t or C_n.

Algorithm 4 updateReuseClusters (C_n)

```
1: if $C_n$.infoloss $< \tau$ then
2:     Insert $C_n$ into CS;
3:     while $|\text{CS}| > \beta$ do
4:         Remove the oldest cluster from CS;
5:     end while
6: end if
```

Algorithm 5 clusteringAcrossPartitions (t, p)

```
1: Find a cluster $C_r$ in CS which covers $t$ with minimum information loss;
2: while $|p| < k$ do
3:     Find partitons similar to $p$ from $P$ by attribute distance and form a set $P_s$;
4:     Insert all tuples of $P_s$ into $p$;
5: end while
6: Find $k-1$ neatest neighbors of $t$ in $p$ by tuple distance and create a new cluster $C_n$ on $t$ and its neighbors;
7: if $C_r \neq$ null && $C_r$.infoloss $<$ $C_n$.infoloss then
8:     Publish $t$ with $C_r$'s generalization;
9:     Remove $t$ from $P$;
10: else
11:     Publish cluster $C_n$;
12:     Remove all tuples of $C_n$ from $P$;
13:     Reassign remaining tuples of $p$ to respective partitions;
14:     updataReuseClusters($C_n$);
15: end if
```

5 Experimental Evaluation

Several experiments have been designed and conducted with two objectives: (1) to verify that the proposed method preserves the utility of anonymized data as much as possible, and (2) to illustrate the efficiency of IDEA, i.e., the delay of our approach does not affect the subsequent real-time processing of data streams. We compare our algorithm with IoT anonymization[36] and K-VARP[37], which are anonymization algorithms for incomplete data streams based on partition merging.

5.1 Experimental setup

To demonstrate the performance of the algorithms on different data distributions, we conduct experiments on two real-world datasets: Adult from the UCI repository[42] and INFORMS datasets[43] from INFORMS. The former is a standard dataset for studying k-anonymity algorithms, and the latter is widely used in related works[19, 44, 45]. To verify the effect of the algorithms on datasets with different data sizes and different degrees of missing values, we construct experimental datasets by selecting a part of the original datasets and randomly inserting missing values to it. Specifically, we denote $|\text{QID}_m|_{\max}$ to represent the maximum number of missing QIDs, which reveals the degree of missing values of dataset. The number of tuples with different numbers of missing QIDs is the same in these datasets. For example, in the dataset with a data size of 30 000 and $|\text{QID}_m|_{\max}$ of 3, the tuples with 0, 1, 2, and 3 missing QIDs are all equal to 7500. The QID attributes of the two datasets used herein are described in Table 1.

All the algorithms are implemented in Java. The experiments are performed on a computer equipped with Intel Core i7 3.0 GHz with 16 GB RAM and Windows 10×64 with JDK 8.0. The algorithm parameters are listed in Table 2, in which the bold ones are default values. According to a previous study[46], the change of τ value has little impact on algorithm performance; thus, to focus on the influence of other parameters on the algorithms, the authors set the middle value of 0.5 as the default value of τ. To buffer more anonymized clusters, we set β, the size of the reuse cluster set, as a reasonably large value. The subsequent experiments also verify that setting β to 200 can save all the added clusters. The default value of R-likeness coefficient R is the same as that in K-VARP.

Table 1 Description of QID attributes of datasets.

Dataset	Attribute	Type	Range (Tree)	
			Min (Height)	Max (Leaves)
Adult	Age	Numeric	0	100
	Edunum	Numeric	0	20
	Hrsperweek	Numeric	0	100
	Workclass	Catgorical	5	8
	Edu	Catgorical	5	16
	Martial	Catgorical	4	7
	Occupation	Catgorical	3	14
	Relationship	Catgorical	3	6
	Race	Catgorical	3	5
	Sex	Catgorical	2	2
INFORMS	Birth-mon	Numeric	1	12
	Birth-year	Numeric	1920	2005
	Income	Numeric	0	210 722
	Sex	Catgorical	2	2
	Race	Catgorical	2	6
	Eduyear	Catgorical	3	18
	Martial	Catgorical	4	6

Note: Range, Min, and Max correspond to type of numeric, and Tree, Height, and Leaves correspond to type of catgorical.

Table 2 Algorithm parameter.

Parameter	Value
$\lvert \text{QID}_m \rvert_{\max}$	**1**, 3, 5, 7, 9
k	5, 10, **50**, 100, 200, 400
$\delta(\times 10^3)$	0.5, 1, **2**, 4, 8, 16
Data size $(\times 10^3)$	5, 10, 15, 20, 25, **30**
τ	**0.5**
β	**200**
R (in K-VARP)	**0.2**

5.2 Evaluation method

5.2.1 Missing pollution rate

The missing value pollution reflects the influence of missing values on generalized data, that the existence of missing values may increase after generalization. One expectation of anonymization is to avoid the missing value pollution of generalization-based algorithms.

Definition 8 (Missing value pollution) Let T^* be the anonymized table generated from T. The generalization method introduces missing value pollution to the original table if $M(T^*) > M(T)$, where $M\ ()$ denotes the number of missing values.

To measure the missing value pollution in the anonymization for incomplete data, we use Missing Pollution Rate (MPR) in calculating the increment of missing values[19].

Definition 9 (MPR) The MPR of T^* is defined as

$$\text{MPR}(T^*) = \frac{M(T^*) - M(T)}{|\text{QID}| \times n} \tag{5}$$

where n is the number of records in the dataset.

5.2.2 Information loss

The utility of anonymized data is mostly measured by the information loss metric in anonymization studies. According to the definition of generalization for incomplete data and the general information loss metric LM[39], the information loss of a record with missing values is given as follows, which can handle both numeric and categorical attributes.

Definition 10 (Information loss of tuple with missing value) A tuple $t(\text{pid}, Q, A)$ is output with the generation $g(g_1, \ldots, g_n)$, where Q is the set of QIDs and A is the set of other attributes. Let Q_v be the set of valid QIDs whose values are not Null. The information loss of the tuple after generalization is

$$\text{IL}(t) = \frac{1}{|Q_v|} \sum_{q_i \in Q_v} \text{IL}(q_i) \tag{6}$$

where $\text{IL}(q_i)$ is the information loss of t on QID q_i caused by the generalization,

$$\text{IL}(q_i) = \begin{cases} \dfrac{|u_i - l_i|}{|U_i - L_i|}, & q_i \text{ is numeric;} \\ \dfrac{|\text{leaves}(H_i)| - 1}{|\text{leaves}(\text{DGH}_i)| - 1}, & q_i \text{ is categorical} \end{cases} \tag{7}$$

where $|U_i - L_i|$ is the value domain of the numeric attribute q_i, and u_i and l_i are the upper and lower bounds of g_i, respectively.

5.3 Data utility

5.3.1 Missing value pollution

For maybe match, a Null value may match with any value. Hence, only other valid values on the attribute are generalized in the generalization based on maybe match. Generalization on valid values only results in more general outputs without introducing new missing values. Hence, in IDEA, which utilizes the generalization based on maybe match for outputting incomplete data, $M(T^*) - M(T) = 0$, that is, the MPR of IDEA is 0. This feature prompts IDEA to preserve data utility while avoiding missing value pollution.

5.3.2 Information loss on varied parameters

The average information loss of the algorithms compared on the Adult dataset is depicted in Fig. 4. The average information loss of IDEA is less than those of IoT and K-VARP in all the experiments. Hence, our approach is

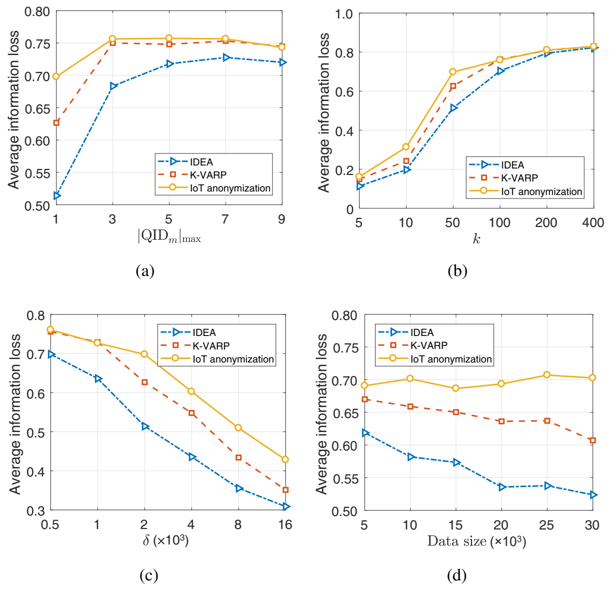


Fig. 4 Average information loss of IoT anonymization, K-VARP, and IDEA on Adult dataset with varying (a) $|\text{QID}_m|_{\max}$, (b) k, (c) δ, and (d) data size.

better than the other two methods in terms of preserving the utility of the anonymized data. The proposed two-dimensional similarity measurement not only enables the clustering across tuples with different missing values, but also makes the generated clusters particularly compact. IoT and K-VARP satisfy the anonymity requirement k by merging similar partitions, and the cluster generated from the merged partitions may have more than k tuples. By contrast, IDEA outputs a cluster with the exact size of k, which is the minimum size to satisfy the k-anonymity requirement; thus, it leads to minimal information loss.

As shown in Fig. 4a, the average information loss increases gradually with augmentation of the degree of missing values. In a fixed-size window, a large number of missing QID attributes result in many relatively small partitions, which facilitate the triggering of clustering across partitions. Given the attribute discrepancy, the information loss of a cluster generated from clustering across partitions is greater than that of a cluster generated from clustering in one partition. Figure 4b shows that the average information loss increases quickly with the increase of k, which denotes the anonymity requirement. The greater the value of k, the better the privacy protection effect. A good privacy protection effect also requires the output cluster to contain more tuples. However, a large cluster usually causes great information loss. Figure 4c reveals the significant impact of window size δ on average information loss. The average information loss decreases as δ increases. As a large window buffers more tuples, a tuple set for output can choose neighbors from more tuples to generate a highly compact cluster and to cause minimal information loss. Figure 4d shows that the average information loss decreases slightly when the data size is large. As more tuples arrive, more clusters are anonymized and added into the reuse set CS. With a large reuse set, tuples can choose from more anonymized clusters to achieve minimal information loss.

The average information loss of the algorithms compared on the INFORMS dataset is depicted in Fig. 5. The experiments varying the parameters on INFORMS reveal similar results, that is, the IDEA approach is superior to the other algorithms in terms of data utility preservation. The results also indicate that IDEA performs well in different data distributions.

5.3.3 Information loss of different generalizations

To verify the effects of different generalization methods, we implement a modified IDEA with generalization based on extended match. We compare the modified IDEA with IDEA, whose the generalization is based on maybe match. As the generalization based on basic match removes records with missing values, the method is not considered in the comparison.

Figure 6a shows that the generalization method based on extended match provokes more information loss than the generalization method based on maybe match used in IDEA. With the increase of the degree of missing

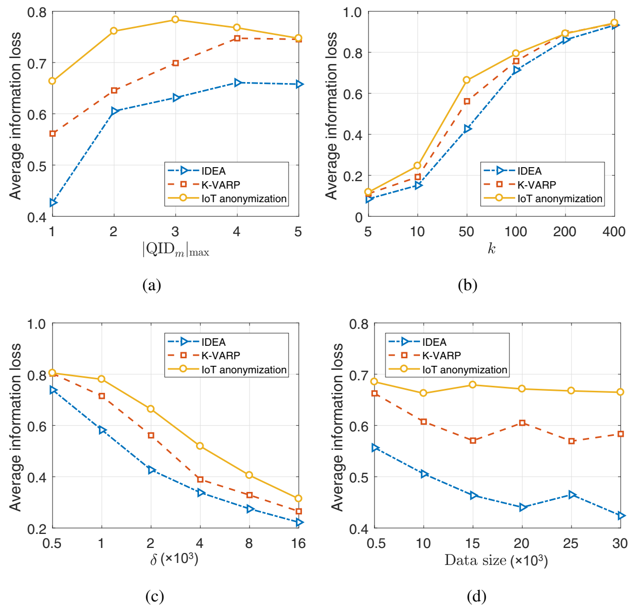


Fig. 5 Average information loss of IoT anonymization, K-VARP, and IDEA on INFORMS dataset with varying (a) $|\text{QID}_m|_{\max}$, (b) k, (c) δ, and (d) data size.

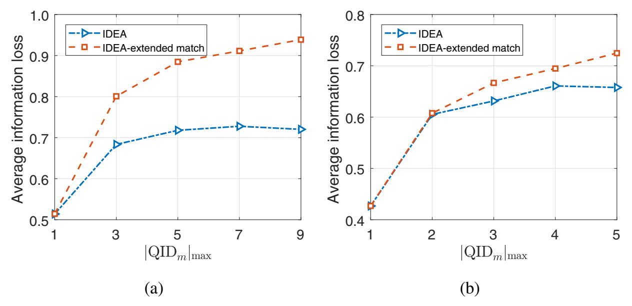


Fig. 6 Average information loss of IDEA and IDEA with generalization based on extended match on (a) Adult and (b) INFORMS datasets with varying $|\mathrm{QID_m}|_{\max}$.

values, the difference between the two methods expands. That is, the greater the number of missing values, the more abvious the advantages of our method in terms of data utility preservation. As mentioned in Section 3.3, the larger the number of missing values is, the more values will be replaced with the most general values in the generalization method based on extended match; the replacement results in great information loss.

5.4 Effectiveness

The total runtime of the algorithms compared on the Adult dataset is depicted in Fig. 7. Compared with IoT and K-VARP, IDEA has greater time consumption. IDEA and K-VARP consume much time in calculating two-dimensional distances and R-likeness, whereas IoT randomly chooses the most similar and largest partitions to merge with expired partitions. As shown in Fig. 8, IDEA has more anonymized clusters for reuse than the other two methods because of its advantage in data utility preservation. Hence, it takes more time to traverse CS to verify whether or not each tuple is covered by the reuse clusters. Nevertheless, IDEA is efficient as its average runtime is less than 2 s.

As shown in Fig. 7a, the runtime of these algorithms grows with the augmentation of the degree of the missing values. When the number of missing values is large, time-consuming clustering across partitions is likely to occur. As revealed in Fig. 7b, the runtime decreases as k increases. A large k implies more output tuples each time that cause few clustering and reduce the total runtime. Figure 7c shows that the time cost increases when the window size δ increases. A large window can buffer more tuples and partitions. When an expired tuple needs to be output, the distance between it and more tuples needs to be calculated. Figure 7d shows the significant impact of data size on the total runtime of the algorithms. A large number of tuples lead to heavy clustering and checking of clusters in CS that can be reused to publish tuples; these processes comsume much time.

The total runtime of the three algorithms compared on the INFORMS dataset is depicted in Fig. 9. The experiments varying k, δ, and data size on INFORMS show similar figures in Fig. 7. Figure 9a presents a different trend from that shown in Fig. 7a. The greater number of missing QID values, the fewer the dimensions involved in the similarity calculation; hence time consumption is decreased. However, this reason and the reason behind the time increment in Fig. 7a affect the runtime of the two datasets differently because of the differences in their data distributions and proportions of attribute types.

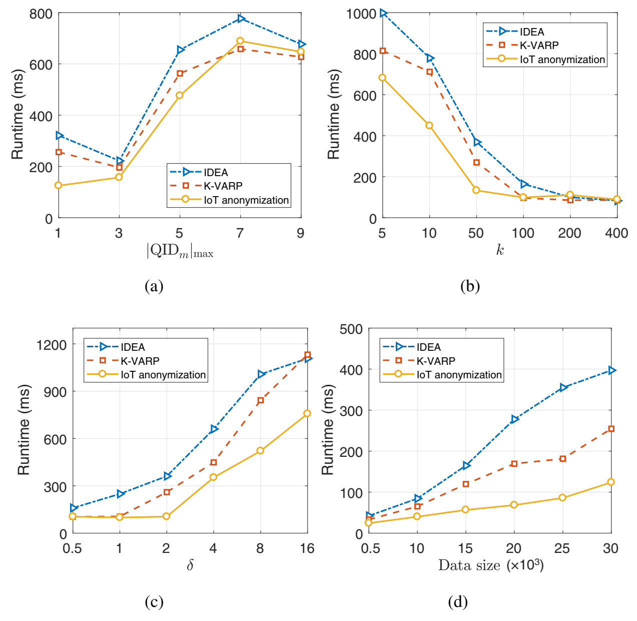


Fig. 7 Runtime of IoT anonymization, K-VARP, and IDEA on Adult dataset with varying (a) $|\mathrm{QID}_m|_{\max}$, (b) k, (c) δ, and (d) data size.

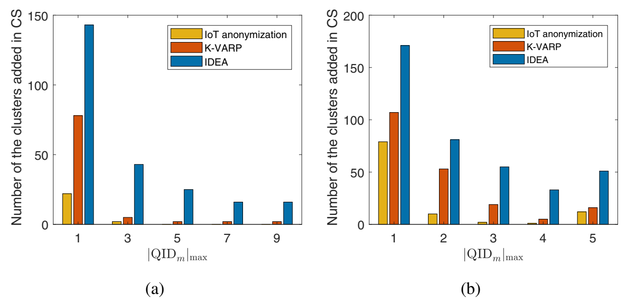


Fig. 8 Number of clusters added in reuse cluster set CS on (a) Adult and (b) INFORMS datasets with varying $|\mathrm{QID}_m|_{\max}$.

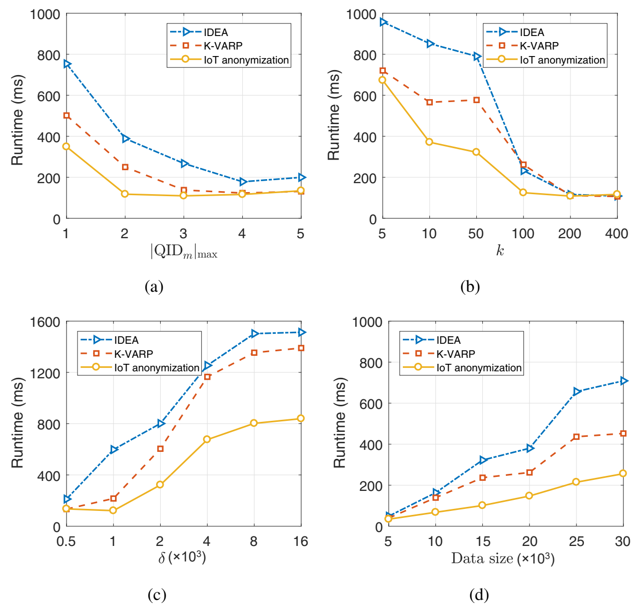


Fig. 9 Runtime of IoT anonymization, K-VARP, and IDEA on INFORMS dataset with varying (a) $|\mathrm{QID}_m|_{\max}$, (b) k, (c) δ, and (d) data size.

6 Conclusion

In this work, we propose a utility-enhanced approach for incomplete data stream anonymizaiton. This approach is capable of anonymizing incomplete data streams and preserving the more utility of anonymized data within an acceptable time. We implement a sliding window-based processing framework for anonymizing continuous and potentially infinite data streams. IDEA considers the dimensions of attributes and tuples as the similarity measurement to accomplish clustering between complete and incomplete tuples and to obtain more compact clusters with minimal information loss. In addition, a generalization method based on maybe match strategy is proposed to generalize incomplete data without causing extra missing value pollution. The experiments conducted on two real datasets validate the applicability and performance of the proposed approach.

For our future work, we aim to implement the proposed approach with the data stream processing framework in a distributed environment.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Nos. U19A2081 and 61802270), and the Fundamental Research Funds for the Central Universities (No. 2020SCUNG129).

References

[1] J. Gama, *Knowledge Discovery From Data Streams*. Boca Raton, FL, USA: Chapman & Hall/CRC Press, 2010.

[2] X. Zeng, X. Chen, G. Shao, T. He, and L. Wang, DTA-HOC: Online https traffic service identification using DNS in large-scale networks, *Tsinghua Science and Technology*, vol. 25, no. 2, pp. 239–254, 2020.

[3] S. Yu, Big privacy: Challenges and opportunities of privacy study in the age of big data, *IEEE Access*, vol. 4, pp. 2751–2763, 2016.

[4] K. Al-Hussaeni, B. C. M. Fung, and W. K. Cheung, Privacy-preserving trajectory stream publishing, *Data and Knowledge Engineering*, vol. 94, pp. 89–109, 2014.

[5] Z. Pervaiz, A. Ghafoor, and W. G. Aref, Precision-bounded access control using sliding-window query views for privacy-preserving data streams, *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 7, pp. 1992–2004, 2015.

[6] S. Liu, Q. Qu, L. Chen, and L. M. Ni, SMC: A practical schema for privacy-preserved data sharing over distributed data streams, *IEEE Transactions on Big Data*, vol. 1, no. 2, pp. 68–81, 2015.

[7] X. Chen, L. Yang, and Y. Luo, Big data security technology, *Advanced Engineering Sciences*, vol. 49, no. 5, pp. 1–12, 2017.

[8] S. A. Abdelhameed, S. M. Moussa, and M. E. Khalifa, Privacy-preserving tabular data publishing: A comprehensive evaluation from web to cloud, *Computers & Security*, vol. 72, pp. 74–95, 2018.

[9] L. Xu, C. Jiang, J. Wang, J. Yuan, and Y. Ren, Information security in big data: Privacy and data mining, *IEEE Access*, vol. 2, pp. 1149–1176, 2014.

[10] L. Sweeney, K-anonymity: A model for protecting privacy, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[11] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, L-diversity: Privacy beyond *k*-anonymity, *TKDD*, vol. 1, no. 1, p. 3, 2007.

[12] N. Li, T. Li, and S. Venkatasubramanian, T-closeness: Privacy beyond k-anonymity and l-diversity, in *Proceedings of the 23rd International Conference on Data Engineering* Istanbul, Turkey, 2007, pp. 106–115.

[13] S. Yaseen, S. M. A. Abbas, A. Anjum, T. Saba, A. Khan, S. U. R. Malik, N. Ahmad, B. Shahzad, and A. K. Bashir, Improved generalization for secure data publishing, *IEEE Access*, vol. 6, pp. 27156–27165, 2018.

[14] X. Huang, J. Liu, Z. Han, and J. Yang, A new anonymity model for privacy-preserving data publishing, *China Communications*, vol. 11, no. 9, pp. 47–59, 2014.

[15] X. He, Y. Xiao, Y. Li, Q. Wang, W. Wang, and B. Shi, Permutation anonymization: Improving anatomy for privacy preservation in data publication, in *Proc. of New Frontiers in Applied Data Mining-PAKDD 2011 International Workshops*, Shenzhen, China, 2011, pp. 111–123.

[16] Q. Wei, Y. Lu, and Q. Lou, Privacy-preserving data publishing based on de-clustering, in *Proc. of 7th IEEE/ACIS International Conference on Computer and Information Science*, Portland, OR, USA, 2008, pp. 152–157.

[17] Q. Gong, M. Yang, and J. Luo, Data anonymization approach for incomplete microdata, *Journal of Software*, vol. 24, no. 12, pp. 2883–2896, 2013.

[18] J. Tekli, B. al Bouna, Y. B. Issa, M. Kamradt, and R. A. Haraty, (k, l)-clustering for transactional data streams anonymization, in *Proc. of Information Security Practice and Experience-14th International Conference*, Tokyo, Japan, 2018, pp. 544–556.

[19] Q. Gong, M. Yang, Z. Chen, W. Wu, and J. Luo, A framework for utility enhanced incomplete microdata anonymization, *Cluster Computing*, vol. 20, no. 2, pp. 1749–1764, 2017.

[20] W. Wang, J. Li, C. Ai, and Y. Li, Privacy protection on sliding window of data streams, in *Proc. of 2007 International Conference on Collaborative Computing: Networking, Applications and Worksharing* (*CollaborateCom 2007*), New York, NY, USA, 2007, pp. 213–221.

[21] J. Zhang, J. Yang, J. Zhang, and Y. Yuan, Kids: k-anonymization data stream base on sliding window, in *Proc. of 2010 2nd International Conference on Future Computer and Communication*, Shanghai, China, 2010, pp. V2-311–V2-316.

[22] J. Li, B. C. Ooi, and W. Wang, Anonymizing streaming data for privacy protection, in *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering Workshop*, Cancún, Mexico, 2008, pp. 1367–1369.

[23] J. J. V. Nayahi and V. Kavitha, Privacy and utility preserving data clustering for data anonymization and distribution on Hadoop, *Future Generation Computer Systems*, vol. 74, pp. 393–408, 2017.

[24] H. Chhinkaniwala and S. Garg, Tuple value based multiplicative data perturbation approach to preserve privacy in data stream mining, doi:10.5121/ijdkp.2013.3305..

[25] C. Jianneng, C. Barbara, F. Elena, and T. Kian-Lee, CASTLE: A delay-constrained scheme for ks-anonymizing data streams, in *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering Workshop*, Cancún, Mexico, 2008, pp. 1376–1378.

[26] P. Wang, J. Lu, L. Zhao, and J. Yang, B-CASTLE: An efficient publishing algorithm for k-anonymizing data streams, in *Proc. of 2010 2nd WRI Global Congress on Intelligent Systems, GCIS 2010*, Wuhan, China, 2010, pp. 132–136.

[27] H. Zakerzadeh and S. L. Osborn, FAANST: Fast anonymizing algorithm for numerical streaming data, in *Proceedings of the 5th International Workshop on Data Privacy Management, and 3rd International Conference on Autonomous Spontaneous Security*, Athens, Greece, 2011, pp. 36–50.

[28] H. Zakerzadeh and S. L. Osborn, Delay-sensitive approaches for anonymizing numerical streaming data, *International Journal of Information Security*, vol. 12, no. 5, pp. 423–437, 2013.

[29] K. Guo and Q. Zhang, Fast clustering-based anonymization approaches with time constraints for data streams, *Knowledge-Based Systems*, vol. 46, pp. 95–108, 2013.

[30] G. Yang, J. Yang, J. Zhang, and Y. Chu, Research on data streams publishing of privacy preserving, in *Proc. of 2010 IEEE International Conference on Information Theory and Information Security*, Beijing, China, 2010, pp. 199–202.

[31] J. Xie, J. Zhang, J. Yang, and B. Zhang, Anonymization algorithm based on time density for data stream, *Journal on Communications*, vol. 35, no. 11, pp. 191–198, 2014.

[32] A. B. Sakpere and A. V. D. M. Kayem, Adaptive buffer resizing for efficient anonymization of streaming data with minimal information loss, in *Proc. of 2015 International Conference on Information Systems Security and Privacy* (*ICISSP*), Loire Valley, France, 2015, pp. 1–11.

[33] S. A. Abdelhameed, S. M. Moussa, and M. E. Khalifa, Restricted sensitive attributes-based sequential anonymization (RSA-SA) approach for privacy-preserving data stream publishing, *Knowl.-Based Syst.*, vol. 164, pp. 1–20, 2019.

[34] J. Wang, C. Deng, and X. Li, Two privacy-preserving approaches for publishing transactional data streams, *IEEE Access*, vol. 6, pp. 23648–23658, 2018.

[35] J. Zhang, H. Li, X. Liu, Y. Luo, F. Chen, H. Wang, and L. Chang, On efficient and robust anonymization for privacy protection on massive streaming categorical information, *IEEE Trans. Dependable Sec. Comput.*, vol. 14, no. 5, pp. 507–520, 2017.

[36] A. Otgonbayar, Z. Pervez, and K. Dahal, Toward anonymizing IoT data streams via partitioning, in *Proc. of 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems*, Brasilia, Brazil, 2016, pp. 331–336.

[37] A. Otgonbayar, Z. Pervez, K. P. Dahal, and S. Eager, *K*-VARP: *k*-anonymity for varied data streams via partitioning, *Inf. Sci.*, vol. 467, pp. 238–255, 2018.

[38] P. Jaccard, The distribution of the flora in the alpine zone, *New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912.

[39] V. S. Iyengar, Transforming data to satisfy privacy constraints, in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, 2002, pp. 279–288.

[40] R. van der Meyden, *Logical Approaches to Incomplete Information: A Survey*. Boston, MA, USA: Springer, 1998.

[41] M. Ciglic, J. Eder, and C. Koncilia, K-anonymity of microdata with NULL values, in *Proc. of International Conference on Database and Expert Systems Applications*, Cham, Switzerland, pp. 328–342, 2014.

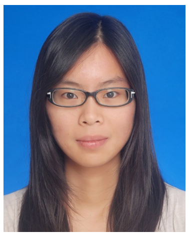
[42] U. M. L. Repository, Adult data set, https://archive.ics.uci.edu/ml/datasets/Adult, 2020.

[43] INFORMS data set, https://sites.google.com/site/informsdataminingcontest/, 2020.

[44] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu, Utility-based anonymization using local recoding, in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia, PA, USA, 2006, pp. 785–790.

[45] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis, Fast data anonymization with low information loss, in *Proceedings of the 33rd International Conference on Very Large Data Bases*, Vienna, Austria, 2007, pp. 758–769.

[46] K. Guo and Q. Zhang, Fast clustering-based anonymization algorithm for data treams, *Journal of Software*, vol. 24, no. 8, pp. 1852–1867, 2013.



Lu Yang received the BS degree from Sichuan University in 2014. She is currently a PhD candidate at the College of Computer Science, Sichuan University. Her research interests include big data security and data privacy.

Xingshu Chen received the MS and PhD degrees from Sichuan University in 1999 and 2004, respectively. She is currently a full professor at the School of Cyber Science and Engineering, Sichuan University. Her research interests include cloud computing, cloud security, distributed file system, big data processing, network protocol analysis, and new media supervision. She is a member of China Information Security Standardization Technical Committee.

Yonggang Luo received the PhD degree from Sichuan University in 2016. He is currently an assistant researcher at the Cyber Science Research Institute, Sichuan University. His research interests include big data platform security and network security analysis using big data technology.

Xiao Lan received the PhD degree from the Institute of Information Engineering, Chinese Academy of Sciences in 2018. She is currently an assistant researcher at the Cyber Science Research Institute, Sichuan University. Her research interests include applied cryptography, authenticated key exchange protocol, and blockchain.

Wei Wang received the PhD degree from the University of Electronic Science and Technology of China, Chengdu, China in 2017. He is currently an associate professor at Sichuan University, Chengdu, China. His recent studies focus on investigating the spreading mechanisms of information, epidemic, rumor, and associated critical phenomena in complex networks. He has published more than 60 papers in the field of network science and spreading dynamics.