Additive Sanitization: A Technique for Pattern-Preserving Anonymization for Time-Series Data

Revathi.S
PG Scholar
Sri Ramakrishna Engineering College
Tamil Nadu, India

Jeyalakshmi.I
Assistant Professor
Sri Ramakrishna Engineering College
Tamil Nadu, India

ABSTRACT
A time series is a set of data normally collected at usual intervals and often contains huge amount of individual privacy. The need to protect privacy and anonymization of time-series while trying to support complex queries such as pattern range and pattern matching queries. The conventional (k, p)-anonymity model cannot effectively address this problem as it may suffer serious pattern loss. In the proposed work a new technique called additive sanitization has been developed which increment the supports of item sets and their subsets in order to reduce pattern loss and prevent linkage attack.

Keywords
Time series, pattern, sanitization, privacy, anonymity

1. INTRODUCTION
Privacy protection in the publication of time series is a difficult one due to the complex nature of the data. Particularly consider an essential problem of anonymizing time series while trying to support the queries. However the time sensitive attribute values and their patterns can be used as strong quasi-identifiers (QI) to launch linkage attack which reidentify some of the records. The desirable solution to prevent linkage attacks is to enforce (k, p)-anonymity on the published database, so that each record has its QI attributes identical to at least k – 1 other records. Even though the conventional (k, p)-anonymity can be used to resist linkage attacks, it cannot efficiently preserve the patterns, which are significant for performing queries on time series. In this paper we develop a novel sanitization strategy, named *additive*, the idea is to increment supports of item sets and their subsets (by virtually adding transactions in the original dataset),minimizing distortion means reducing as much as possible the increments of supports (i.e., number of transactions virtually added).

2. RELATED WORK
In this sector to summarize the existing works of partial information hiding; particularly it was related to time series. The existing partial information hiding approaches can be divided into two types, the perturbation-based approaches and the partition-based approaches. Perturbation-based approaches [1], [2] protect data by adding noises. However, it does not prevent linkage attack. Partition-based approaches first divide tuples of data set into disjoint groups and then release some general information of each group.

K- anonymity [3] and condensation [4] are two approaches in this group. K-anonymity is a necessary approach to privacy preserving data publishing and generalization is the most trendy approach of enforcing k-anonymity. But it has weakness when being applied on time-series data. The limitation of condensation is that it cannot preserve the correlations of attributes for individual data. Microaggregation [5] can be used to prevent linkage attacks on time series. In addition, the data published by microaggregation also suffer pattern loss in an unrestrained manner.

3. PROPOSED SYSTEM
3.1 The (K, P)-anonymity model
Table 1: A Published Data Set T_ Conforming to (k,P)-Anonymity

Group ID	2005	2006	2007	2008	2009	PR
1	[117-176]	[107-181]	[87-188]	[74-197]	[51-213]	Aabbcc
1	[117-176]	[107-181]	[87-188]	[74-197]	[51-213]	Aabbcc
1	[117-176]	[107-181]	[87-188]	[74-197]	[51-213]	Ccbbaa
2	[32-98]	[54-120]	[47-125]	[38-132]	[20-161]	Abbbcc
1	[117-176]	[107-181]	[87-188]	[74-197]	[51-213]	Ccbbaa
2	[32-98]	[54-120]	[47-125]	[38-132]	[20-161]	Abbbcc

In Table 1 the published data set to preserve the pattern information for each time series, there is an additional PR column which implements an alphabetic string representation. Here the QI attributes are generalized based on the formed k-group (k=4). In each k-group, the PR column conforms to P anonymity (p=2).

3.2 The Utility Measures
To generate the utility measures of (k, P)-anonymity model, including the breach probability, which represents the privacy preservation ability, and the information loss, which represents the utility of published data. There are two kinds of information loss, instant value loss (VL), and the pattern loss (PL).

3.3 Instant Value Loss Metric
The instant value loss of Q is given by

$$VL(Q) = \sqrt{\sum (r_i^+ - r_i^-)}^{2/n}$$

For database T, VL (Q) is obtained by summing up the instant value losses of all its members.

3.4 Pattern Loss Metric

The pattern loss can be calculated by the distance between p(Q) and p*(Q),namely

PL(Q)= distance(p(Q),p*(Q)),

where distance(.) is a distance measure defined in the feature vector space of patterns.

4. APPROACHES FOR ENFORCING (K, P)-ANONYMITY

The first approach for enforcing (k, P)-anonymity is to use a **top-down clustering**-like structure as described in the following:

1. Create first-level k-groups from the micro data set.
2. For each k-group, remove PRs from micro data based on the selected PR form. The removed PRs should minimize the pattern loss while respecting the P requirement within its own k-group;
3. For each k-group, make P-subgroups based on the PRs.

The second approach is a **bottom-up framework** to form P-subgroups from individual records first, and then construct k-groups. The bottom-up approach is described in the following:

1. Remove PRs from the micro data. The extracted PRs should minimize the pattern loss while respecting the P-requirement in the entire data set;
2. Form the second-level P-subgroups based on PRs;

5. ADVANTAGES

- It can prevent linkage attack.
- This model supports customized data publishing.

6. IMPLEMENTATION

The (k, p)-anonymity model consists of 6 modules

1. Computing Pattern Representations
2. Node Splitting
3. Create-tree phase
4. Recycle bad-leaves phase
5. Group formation phase
6. Additive sanitization

6.1 Computing Pattern Representations

When calculating pattern representations, the main aim is to attain minimal pattern loss in the published table. SAX uses a sequence of alphabets, for example, "baabccbc," to represent a time series. Given a set of alphabets a time series can be converted to an alphabet sequence as described in the following:

1. First, to specify an integer parameter level which controls granularity of the resultant SAX representation. .

2. Second, to normalize the time series to have a mean of 0 and standard deviation of 1.

6.2 Node Splitting

The node splitting process is followed by attempts to refine the PRs of the records in a node. For each tree node N, all its members should have identical PR .However, an increment in the level leads to changes in PRs. Therefore, the members of N may have different PRs at N. level + 1. Thus, by increasing the level of N, we can split N into a few partitions of records, so that each partition express records of the same PR at N. level + 1. Beginning from the root, we hold node N in a recursive procedure.

- If N. size<P, then the node is labeled as bad-leaf and the recursion terminates.
- Otherwise if N. level=max-level, then the node is labeled as good leaf and the recursion terminates.
- Else if $P \leq N.\ size < 2 * P$, we try to maximize the level of N as long as all records of N have the identical PR.

Algorithm 1:Node splitting

Data:tree node N,P,max level
Begin
If N.size<P then
N.label=bad-leaf;
If N.level==max-level then
N.label =good-leaf;
If P≤N.size<2 * P then
N.label=good-leaf;
Maximize N.level without node split;
Else
If N can be split then
If total size of all T B-nodes ≥ P then
Generate $child_{merge;}$
$child_{merge.}$.level=N.level;
level of all TG-nodes is N.level +1;
else
level of all child nodes is N.level+1;
else
N.label=good-leaf;
End

6.3 Create-tree phase

In this phase, to produce and organize the PRs in a tree for all time series in T respecting the following P-requirement;

1. The root node for split is the whole data set T;

2. The post processing step in the Naive algorithm is removed**.**

6.4 Recycle bad-leaves phase

To avoid extra suppression, we would recycle most of the bad-leaf nodes by merging them with each other. To achieve the highest PR level for each bad-leaf, we begin the recycling process from the highest PR level between all bad leaves, designated by max-bad-level.

If two bad-leaf nodes BL1 and BL2 have the same level and PR, they can be merged into a new node, represented by nm, which has the same level and PR. If nm contains no fewer than P time series, it is pointed a good-leaf, otherwise a bad-leaf.

Algorithm 2: Recycle bad-leaves

Data: P,leaf-list,current-level,max-bad-level

Result: P-subgroup list

Begin

Current-level=max-bad-level;

While sum of all bad leaves size ≥ P do

If any bad leaves can merge then

Merge them to a new node leaf-merge;

If leaf-merge.size ≥ P then

Leaf-merge.label=good-leaf;

Else

Leaf-merge.label=bad-leaf;

Current-level--;

Suppress all time-series contained in bad leaves;

end

6.5 Group formation phase

The greedy k-group formation phase is described in the following steps:

Step 1: All P-subgroups in PGL containing no fewer than k time series are taken as k-groups and simply moved into GL.

Step 2: In the remaining P-subgroups in PGL, find the P-subgroup s1 with the minimum instant value loss, and then create a new group G = s1.

Step 3: Find another P-subgroup s ∈ PGL - s1, which, if merged with G, produces the minimal value loss VL (G ⋃S).

Step 4: Repeat Step 3 until |G| ≥ k. G is then added into GL and its respective subgroups in PGL are removed.

Step 5: Steps 2-4 are repeated until the total remaining time series in PGL are fewer than k.

Algorithm 3:Group formation

Data:PGL**,**k,P

Result: Group list GL

Begin

For each P-subgroup that size ≥ 2*P do

Split it by top-down clustering;

If any P-subgroup that size ≥ k then

Add it into GL and remove it from PGL;

While |PGL| ≥ k do

Find s_1 and G = s_1;

While |G| < k do

Find s_{min} and add s_{min} into G;

Remove all P-subgroups in G from PGL and put G in GL;

For each remaining P-subgroup s' do

Find corresponding G' and add s' into G';

end

6.6 Additive sanitization

In the additive sanitization, not to actually adding transactions: this is just to emphasize that the transformed set of frequent item sets maintains database-compatibility. Besides, it will contain exactly the same item sets, but with some supports increased. In the sanitization approach the idea is to increment supports of item sets and their subsets (by virtually adding transactions in the original dataset), minimizing distortion means reducing as much as possible the increments of supports (i.e., number of transactions virtually added).

This algorithm is composed by **two phases**: during the first phase all maximal channels are merged as much as possible, according to Maximal Channels Merging; then the resulting set of merged channels is used in the second phase to select the item sets whose support must be increased. Therefore, according to the additive sanitization, we increase the support of the item set by k. It takes in input closed frequent item sets Cl(D, σ) and maximal inference channels M Ch(k, Cl(D, σ)), and returns O^k, which in this case is the sanitized version of Cl(D, σ).

Algorithm 4: Additive Sanitization

Input:$Cl(D,\sigma), \mu Ch(k, Cl(D,\sigma))$

Output : O^k

$S \leftarrow \emptyset$;

For all $(C_I^J, f_I^J) \in \mu Ch(k, Cl(D,\sigma))$ do

If $\exists C_{I'}^{J'} \in S \; s.t \; C_I^J$ and $C_{I'}^{J'}$ can be merged

Then

S← $S \setminus \{C_{I'}^{J'}\}$;

S← $S \cup (C_I^J \bowtie C_{I'}^{J'})$;

Else

S← $S \cup \{C_I^J\}$;

For all (I, sup_D(I)) ∈ $Cl(D,\sigma)$ *do*

For all $C_{I'}^{J'} \in S$ s.t I ⊆ I' d*o*

$sup_D(I) \leftarrow sup_D$(I)+k;

$O^k \leftarrow Cl(D,\sigma)$

7. CONCLUSION

Conclusion of this work provides a new technique called **additive sanitization** may greatly reduce the information loss. In the additive sanitization, not to really adding transactions: this is just to show up that the transformed set of frequent item sets maintains database-compatibility. Besides it will contain exactly the same item sets, but with some supports increased. The idea of this approach is to increment supports of item sets and their subsets (by virtually adding transactions in the original dataset), minimizing distortion means reducing as much as possible the increments of supports (i.e., number of transactions virtually added). Finally, this method concluded that by increasing the support count of item sets, it will reduce the pattern and information loss.

8. REFERENCES

[1] S. Papadimitriou, F. Li, G. Kollios, and P.S. Yu, "Time Series Compressibility and Privacy," Proc. 33rd Int'l Conf. Very Large Data Bases (VLDB), pp. 459-470, 2007.

[2] L. Singh and M. Sayal, "Privacy Preserving Burst Detection of Distributed Time Series Data Using Linear

Transforms," Proc. IEEE Symp. Computational Intelligence and Data Mining (CIDM), pp. 646-653, 2007.

[3] L. Sweeney, "k-Anonymity: Privacy Protection Using Generalization and Suppression," Int'l J. Uncertainty Fuzziness and Knowledge-Based Systems, vol. 10, no. 5, pp. 571-588, 2002.

[4] C.C. Aggarwal and P.S. Yu, "A Condensation Approach to Privacy Preserving Data Mining," Proc. Ninth Int'l Conf. Extending Database Technology (EDBT), pp. 183-199, 2004.

[5] J. Nin and V. Torra, "Towards the Evaluation of Time Series Protection Methods," Information Sciences, vol. 179, no. 11, pp. 1663-1677, 2009.

[6] R. Agrawal and R. Srikant. Privacy-preserving data mining. In Proceedings of the 2000 ACM SIGMOD on Management of Data.

[7] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases.In Proceedings of the 1993 ACM SIGMOD.

[8] E. Dasseni, V. S. Verykios, A. K. Elmagarmid, and E. Bertino. Hiding association rules by using confidence and support. In Proceedings of the 4th International Workshop on Information Hiding, 2001.

[9] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke.Privacy preserving mining of association rules. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002.

[10] S. Rizvi and J. R. Haritsa. Maintaining data privacy in association rule mining. In Proceedings of the 28th VLDB Conference, 2002.