Unifying Global and Local Anomaly Detection for Time Series [Scalable Data Science]

Tongkai Lu
SKLSDE Lab, Beihang University
Beijing, China
lutongkai@buaa.edu.cn

Shuai Ma
SKLSDE Lab, Beihang University
Beijing, China
mashuai@buaa.edu.cn

Zhongxi Zhang
SKLSDE Lab, Beihang University
Beijing, China
zhangzx@buaa.edu.cn

ABSTRACT

Anomaly detection for time series data has been proven useful from both academia and industry, and can be categorized into global and local methods based on the amount of data used to determine anomalies. To the best of our knowledge, there are no methods that unify the global and local methods together. In this study, we propose an approach that unifies the global and local anomaly detection to achieve the good performance. To make the unification possible, we discretize time series data and partition the discrete data into global and local anomaly detection parts based on a concept of local factor. We design a rule-based global method using non-redundant association rules with tolerances to handle the difficulty of asynchronous changes of different attributes and the excessive number of rules. We develop a Transformer based on local method with LSH Attention to fit better for discrete data. We finally conduct an extensive experimental study to verify the efficiency and effectiveness of our approach.

1 INTRODUCTION

Time series data captured by various sensors contain patterns such as trends, seasonal fluctuations, irregular cycles and occasional shifts in level or variability in business, economics, medicine, and other scientific fields [47]. Anomaly detection in time series data is useful both for various applications as a stand-alone task and for improving the performance of other data mining tasks as an auxiliary task such as predictions and classifications [5, 9, 11, 13, 26, 40, 41, 48]. Hence, anomaly detection for time series data has drawn significant attentions from both industry and research communities [4, 8, 10, 15, 18, 19, 22, 23, 28, 34–37, 42–44, 49, 51–53, 55, 56, 60].

Anomalies detection for time series data can be essentially classified into two categories: global and local anomaly detection methods, based on the amount of data used to determine anomalies. Global methods detect the anomalies that violate certain patterns applicable to the entire data [4, 15, 16, 18, 19, 22, 23, 25, 28, 31–35, 37, 42, 43, 45, 46, 49, 51–53, 60], while local methods only consider a restricted portion of data (typically the neighbors of an object) by identifying the difference between an object and its neighbors [8, 10, 36, 44, 56]. Due to the complexity of time series and anomalies, a natural idea is to unify the two methods together to detect anomalies more efficiently and effectively in a single framework. However, to the best of our knowledge, there are no methods that unify the global and local methods together.

In this study, we investigate the possibility to unify the global and local anomaly detection methods for time series data. To do this, there are a couple of non-trivial issues to be solved. First, how to decide which portion of anomalies should be detected by global or local methods to take fully advantage of the unification. Second, how to efficiently and effectively detect anomalies of time series data with global methods. Third, the recent method Anomaly Transformer (AT) [56] does have the best accuracy performance for detecting local anomalies, but it is relatively slow. How to improve the efficiency and effectiveness of local anomaly detection when unifying it with global methods.

Contribution & roadmap. To this end, we propose an approach that unifies the global and local anomaly detection for time series data to achieve the good efficiency and accuracy performance.

(1) We propose an approach to unifying a rule-based global anomaly detection method with AT to improve the accuracy, and to alleviate the efficiency issue of AT [56] at the same time (Section 3). Our rule-based global anomaly detection method is fast, and uses discrete data. AT is the SOTA local anomaly detection method in terms of accuracy, and it does not necessarily use discrete data. To make the unification possible, we choose to discretize time series data, and partition the discrete data into global and local anomaly detection parts based on a concept of *local factor* $\in (0, 1]$.

(2) For the global anomaly detection part, we design a rule-based method with a concept of *non-redundant association rules with tolerances* to handle the difficulty of asynchronous changes of different attributes and the excessive number of rules in time series data (Section 4). Our fast rule-based method also helps to alleviate the efficiency issue of the local anomaly detection method.

(3) For the local anomaly detection part, we develop a Transformer based on method by replacing the Self-Attention [54] of AT [56] with LSH Attention [29] to fit for discrete data, which improves the accuracy and reduces the theoretical complexity (Section 5).

(4) Using real-life data SWaT [21] and WADI [3], we conduct an extensive experimental study, which shows that our unified approach significantly improves the efficiency and achieves the best accuracy, compared with the SOTA method AT (Section 6). (a) The running time is decreased from $4,744$s to $2,706$s and from $8,980$s to $4,763$s on SWaT and WADI, respectively. (b) The F1 score is increased on average 4.7% and 11.9% on SWaT and WADI, respectively.

Due to space limitations, the detailed proofs and extra analyses are available at the full version [1].

2 PRELIMINARIES

In this section, we introduce some basic concepts.

Time series [33]. A time series is a sequence of data points in an increasing time order, where data points are associated with a set of attributes (or features) with values, and each attribute is typically associated with a sensor to capture its value.

Itemsets & transactions [2]. Let $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$ be a set of attribute-value pairs, called items. A subset of $\mathcal{I}$ is called an

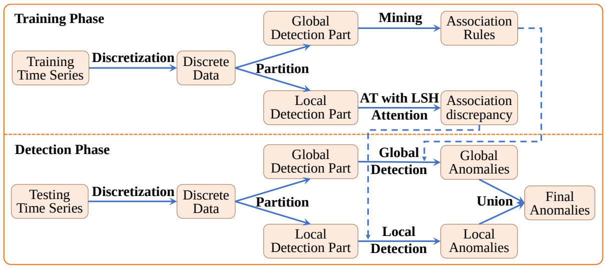


Figure 1: Overview of our unified approach

itemset. Let $\mathcal{D} = \{t_1, \dots, t_n\}$ be a set of transactions, where each transaction t_i ($i \in [1, n]$) is a set of items such that $t_i \subseteq \mathcal{I}$, where a unique identifier tid_i is implicitly associated with a transaction t_i.

Note that a data point of a time series is treated as a transaction, and its set of attribute-value pairs is treated as an itemset. An attribute and timestamp of a data point essentially represent a column and a row of a time series, respectively. The timestamp of a data point can also serve as the unique ID of a transaction.

Operators of itemsets [6]. Operator ϕ associates with transactions $T \subseteq \mathcal{D}$ the items common to all transactions $t \in T$, and operator ψ associates with an itemset $X \subseteq \mathcal{I}$ all the transactions containing X. Closer operator $\gamma(X) = \phi(\psi(X))$, which is the intersection of all the transactions containing X, where $|\gamma(X)| \geq |X|$.

Frequent itemsets [6]. The support of an itemset X, denoted as sup_X, is the percentage of transactions in which it occurs as a subset, *i.e.,* $\text{sup}_X = |\psi(X)|/|\mathcal{D}|$. If the support sup_X of an itemset X is greater than a given threshold δ, it is a frequent itemset.

Association rules [6]. An association rule r is an implication between two frequent itemsets $X, Y \subseteq \mathcal{I}$ of the form $X \rightarrow Y$, where $X \cap Y$ is empty, and X and Y are called the left and right hand sides of the rule, respectively. Its support $\text{sup}_{X \rightarrow Y}$ and confidence $\text{conf}_{X \rightarrow Y}$ is defined as sup_{XY} and $\text{sup}_{XY}/\text{sup}_X$, respectively.

Frequent closed itemsets [6]. An itemset $Y \subset \mathcal{I}$ is a frequent closed itemset if it is frequent and has no itemset $Y' \subset \mathcal{I}$ such that $Y \subset Y'$ and $\text{sup}_{Y'} = \text{sup}_Y$. It has an alternative definition that a frequent itemset $Y \subset \mathcal{I}$ is a frequent closed itemset iff $\gamma(Y) = Y$.

Generators [6]. An itemset $X \subseteq \mathcal{I}$ is a (minimal) generator of a closed itemset Y iff $\gamma(X) = Y$ and there exists no itemset $X' \subseteq \mathcal{I}$ such that $X' \subset X$ and $\gamma(X') = Y$. A generator of cardinality k is called a k-generator.

3 A UNIFIED APPROACH

In this section, we first give an overview of our unified approach to anomaly detection for time series data, shown in Figure 1.

First, we discretize time series data so that each of its attributes has a finite number of values and each data point in a time series is treated as an itemset, which makes the unifying of global and local anomaly detection possible as some methods consider discrete data only. Second, the discrete time series data are partitioned into the global and local anomaly detection parts (referred to as GDP and LDP, respectively) based on a concept of local factor (LF), which is the support of an item (*i.e.,* attribute-value pair). If the local factor of an item is no more than the given threshold, it belongs to GDP, and belongs to LDP, otherwise. That is, we distinguish GDP from LDP in the item level. In this way, the time series data are partitioned into GDP and LDP. Third, we design a rule-based method to detect anomalies in the GDP, by mining non-redundant association rules with a concept of tolerance to the rules to handle the difficulty of asynchronous changes of different attributes and the excessive number of rules in time series data. Fourth, we develop a method based on AT [56] to detect anomalies in the LDP by identifying (local) association discrepancy, where Self-Attention [54] in AT is replaced with LSH Attention [29] to fit for discrete time series data better. Finally, we ensemble the detected global and local anomalies with union, *i.e.,* a transaction (or data point) is a final anomaly no matter whether it is a global or local anomaly.

Timestamp	A
0	0
1	0
2	0
3	0
4	1
5	1

(a) Time Series

Timestamp	A
0	0
1	0
2	0
3	0
4	$-\infty$
5	$-\infty$

(b) GDP

Timestamp	A
0	$-\infty$
1	$-\infty$
2	$-\infty$
3	$-\infty$
4	1
5	1

(c) LDP

Table 1: Example data discretization and partition (LF = 0.5)

The discretization and partition of GDP and LDP of time series data make the unifying of global and local anomaly detection methods possible. The partition of GDP and LDP using the local factor is very flexible. Even better, it naturally brings the benefit of a high recall (considering the union ensemble of anomalies), and the main focus of global and local anomaly detection methods is to improve their precision, where the introducing of data discretization, tolerance and LSH Attention all serves for this purpose. Moreover, data discretization and the fast nature of rule-based anomaly detection also help to alleviate the slow efficiency of AT. In short, our approach achieves a much better performance by unifying the global and local anomaly detection methods, towards to taking advantage of their strengths and avoiding their weaknesses, and to addressing the previously discussed issues in Section 1.

We next introduce the details of data discretization and partition of GDP and LDP of time series, while leaving our global and local anomaly detection methods in Sections 4 and 5, respectively.

3.1 Data Discretization

We roughly divide the continuous attributes into three categories based on its value distribution and propose three different techniques to discretize them automatically: (1) The value distribution of such attribute is relatively concentrated, and does not have a long-term trend, which can be discretized by K-means directly. (2) The values of such attribute have obvious trends, the changing rates of which can be discretized by K-means directly. (3) The value distribution of such attribute has a multi-peak shape and can be regarded as a superposition of multiple Gaussian distributions, which can be discretized by EM algorithm. These three data discretization methods are enough for our analysis of the time series data. All discretizated attributes are in the form of attribute-value pair as shown in Table 1a. For item P101-2, P101 is the attribute's name, 2 is the discrete value of attribute P101 at timestamp 0 and 1.

Note that it is not necessary to discretize all attributes, and there may exist other alternatives to handle more complicated cases.

Time complexity analysis. All these methods take $O(nKI)$ time for each attribute, where I is the maximum iteration, n is the number of values and K is the cluster number. Hence, it takes $O(mnKI)$ time to discretize a time series, where m is the number of attributes.

Timestamp	Itemset
0	P101-2, MV201-2, ..., P602-1
1	P101-2, MV201-2, ..., P602-1
2	P101-1, MV201-2, ..., P602-1
3	P101-1, MV201-2, ..., P602-1
4	P101-1, MV201-2, ..., P602-1
5	P101-1, MV201-0, ..., P602-1

Table 2: Example time series of SWaT [21]

3.2 Data Partitioning

The partition of GDP and LDP of a time series data is in the item level, and counts on LF $\in (0, 1]$, *i.e.,* the support of an item.

Given a LF threshold η and time series, we obtain the GDP and LDP as follows. (1) For each item (attribute-value pair), it belongs to LDP if its LF $\leq \eta$, and belongs to GDP if its LF $> \eta$. (2) All left vacant places in GDP and LDP, *i.e.,* the places of the LDP values in GDP and the GDP values in LDP, are filled with a special value $-\infty$. Intuitively, an item with a higher LF tends to have more global characteristics, which potentially fits for the use of the global anomaly detection methods, and further the removal of items with lower LF makes the global characteristics more obvious. This makes our choices of relatively simple but effective rule-based global anomaly detection. In contrast, an item with a lower LF tends to have more local characteristics, which potentially fits for the use of the local anomaly detection methods. This makes it possible to unify the global and local anomaly detection methods towards making use of their strengths and avoiding their weaknesses. Considering most anomalies contain both local and global information, our partition method is flexible and the selection of $\leq \eta$ can be in a wide range.

We illustrate this process with an example, shown in Table 1a, where the dicretized time series presented in a table form has a single attribute A and 6 data points, and the timestamp serves as the transaction ID. There are two items A-0 and A-1, where the LF of A-0 is $2/3 > 0.5$ and the LF of A-1 is $1/3 < 0.5$. Therefore, only the values of item A-0 are kept for GDP, which are located at timestamps 1, 2, 3&4, and the values of items A-1 at timestamps 4&5 are replaced with $-\infty$, and the final GDP is shown in Table 1b. Similarly, we have the final LDP shown in Table 1c.

Time complexity analysis. The partitioning process needs to compute the support of all items, which is needed for the mining of association rules in our global anomaly detection method, and, hence, it essentially takes $O(mn)$ time such that m is the number of attributes and n is the number of values.

4 GLOBAL ANOMALY DETECTION

In this section, we first introduce non-redundant association rules with tolerances, and then present the algorithm to mine non-redundant association rules with tolerances on the GDP of training time series data, followed by our rule-based anomaly detection method on the GDP of testing time series data.

4.1 Non-Redundant Rules with Tolerances

For a transaction t and an association rule $X \longrightarrow Y$, they have three types of relationships. We say that t satisfies $X \longrightarrow Y$ if $X \subseteq t$ and $Y \subseteq t$, t violates $X \longrightarrow Y$ if $X \subseteq t$ and $Y \not\subset t$, and t is irrelevant to $X \longrightarrow Y$ if $X \not\subset t$, respectively.

We illustrate their relationships using the example time series shown in Table 2, where each row represents a transaction (or a data point) whose implicit ID is its timestamp. Consider an association rule MV201-2 ⟶ P101-2. Transactions $\{0, 1\}$, $\{2, 3, 4\}$ and $\{5\}$ satisfy, violate and are irrelevant to this rule, respectively.

Association rules with tolerances. However, there is typically a *latency phenomenon* in time series data, as changes among different attributes captured by various sensors may not synchronize, *e.g.,* after the end of an abnormal state, some attributes may return to their normal state faster, but the responses of some attributes may be slower, due to the different mechanisms of sensors. The latency phenomenon raises *false* alarms, and has a negative impact on the precision of anomaly detection.

To handle this phenomenon, we associate an association rule $X \longrightarrow Y$ with *tolerances.* Tolerance $tol^{va}_{X \to Y}$ indicates the maximum number of transactions that continuously violate $X \longrightarrow Y$ right after a transaction satisfies, and tolerance $tol^{vi}_{X \to Y}$ indicates the maximum number of transactions that continuously violate $X \longrightarrow Y$ right after a transaction is irrelevant. The intuitive behind tolerances is that a limited number of transactions continuously violating an association rule should not be treated as violations.

Consider the example time series shown in Table 2, and association rule MV201-2 ⟶ P101-2, which is continuously violated three times right after the satisfaction of transaction 1. This in fact is caused by the latency state value change of attribute P101 from 2 to 1. By taking tolerances into account, such false anomalies can be avoided, and the detection precision can be improved significantly.

Non-redundant association rules with tolerances. To handle the excessive number of rules in time series data, we propose non-redundant association rules with tolerances. Given a set Σ of association rules with tolerances, we say that an association rule $X \longrightarrow Y \in \Sigma$ is *non-redundant* iff there exist no association rules $Z \longrightarrow W \in \Sigma$ such that $Z \subseteq X$, $Y \subseteq W$, $tol^{va}_{Z \to W} \leq tol^{va}_{X \to Y}$ and $tol^{vi}_{Z \to W} \leq tol^{vi}_{X \to Y}$. The rationale behind is if an anomaly can be detected by $X \longrightarrow Y$, it can also be detected by $Z \longrightarrow W$ if it exists.

4.2 Mining Non-Redundant Association Rules with Tolerances

Association rule generation is very expensive, and, hence, it is necessary to generate non-redundant rules directly without generating all rules. Bastide et al. [6] propose a definition of non-redundant rules that are widely adopted [7, 20, 24, 39, 58]. An association rule $X \longrightarrow Y$ is non-redundant iff there exist no rules $Z \longrightarrow W$ such that $Z \subseteq X$, $Y \subseteq W$, $ZY \subset XW$, $\text{sup}_{Z \to W} = \text{sup}_{X \to Y}$ and $\text{conf}_{Z \to W} = \text{conf}_{X \to Y}$ [6]. These non-redundant rules are not for time series data analyses without considering tolerances. But there is a close connection between non-redundant association rules with and without tolerances, shown below. Its proof is in Appendix 9.2

Proposition 4.1. *Non-redundant association rules with tolerances is a subset of non-redundant association rules.*

Algorithm. We now introduce the details of our algorithm, shown in Algorithm 1, for mining non-redundant association rules with tolerances. It takes as input the GDP of the training data, minimum support min_s and confidence min_c, and outputs a minimal set Σ of non-redundant association rules with tolerances.

(1) It first calls algorithm Snow-Touch [50], which utilizes algorithms CHARM [59] and Talky-G [6] to mine closed frequent itemsets CF and generators G, respectively, and builds a connection

Algorithm 1 Mining Non-redundant Rules with Tolerances

```
Input: The GDP of the training data, minimum support $min_s$ and confidence $min_c$.
Output: A minimal set $\Sigma$ of non-redundant association rules with tolerances.
1: Let $\Sigma$ be the set of non-redundant association rules [6];
   /* generated by algorithm Snow-Touch [50]*/
2: Calculate the tolerances of each rule in $\Sigma$;
3: Let $\Sigma_{re}$ be an emptyset;          /*store redundant rules*/
4: for each rule $r_i$ in $\Sigma$ do        /*$i \in [1, |\Sigma|]$*/
5:     for each rule $r_j$ in $\Sigma$ do     /*$j > i$*/
6:         if both $r_i$ and $r_j$ are not in $\Sigma_{re}$ then
7:             if $r_i$ is a redundant rule with tolerances w.r.t. $r_j$ then
8:                 $\Sigma_{re} := \Sigma_{re} \cup \{r_i\}$;
9:             if $r_j$ is a redundant rule with tolerances w.r.t. $r_i$ then
10:                $\Sigma_{re} := \Sigma_{re} \cup \{r_j\}$;
11: return $\Sigma := \Sigma \setminus \Sigma_{re}$.
```

between the *CF* and *G* with a hash table, to generate a set Σ of non-redundant association rules [6] (line 1). (2) It then calculates the tolerances of each rule in Σ (line 2). More specifically, given a rule $X \longrightarrow Y \in \Sigma$, it computes its tolerances as follows. It first computes the transactions T that violate $X \longrightarrow Y$, and then determines all the continuous numbers of transactions in T that violate $X \longrightarrow Y$ right after the satisfaction or irrelevance, and finds the respective maximum numbers as tolerances $tol^{va}_{X \rightarrow Y}$ and $tol^{vi}_{X \rightarrow Y}$. (3) Redundant rules are then repeatedly determined, and stored in Σ_{re} (lines 4-10). Basically, for each rule r_i, it checks its redundance with rule r_j based on the definition of non-redundant association rules with tolerances. Note that if both r_i and r_j are in Σ_{re}, the checking is not necessary. (4) Finally, a minimal set Σ of non-redundant association rules with tolerances is returned (line 11).

Time complexity analysis. The complexity of algorithm Snow-touch is affected by many factors like data sizes, dimensions, supports and data alignments [50], and it is difficult to get an exact complexity. However, its main body is IT-Tree, which is a type of FP-tree. So we can roughly take the time complexity of the latter [30] as the complexity of algorithm Snow-touch, which is $O(m(n-1)+L_{FI})$, where L_{FI} is the sum of all frequent itemset lengths, m is the number of items, and n is the number of transactions in GDP. It is a loose estimate, since algorithm Snow-touch is generally much faster than FP-Growth [57]. It is easy to see that it takes $O(m|\Sigma|)$ time to compute the tolerances, as the time to compute $\psi(X)$ and $\psi(Y)$ can be done while mining association rules. It also takes $O(|\Sigma|^2)$ time to remove redundant association rules with tolerances to derive non-redundant association rules with tolerances. Thus, Algorithm 1 in total takes $O(m(n-1) + L_{FI} + m|\Sigma| + |\Sigma|^2)$ time.

4.3 Rules-Based Global Anomaly Detection

Existing rule-based anomaly detection methods mainly focus on the rule mining part to improve the accuracy and efficiency of detection, while the detection part is largely overlooked [18, 19, 32, 42, 53]. When association rules with tolerances are considered, it should be recognized that the detection part may become costly, especially the computation of the set of association rules violated by a transaction, and needs to record the maximum number of violation times of each rule to cope with the tolerances of rules.

Algorithm 2 Global Anomaly Detection with Rules

```
Input: The GDP of testing data and set $\Sigma$ of non-redundant rules with tolerances.
Output: The set $R$ of transaction-rule pairs $(t, r)$ that $t$ violates $r$.
1: Let $R$ be an empty set;    /*store pairs $(t, r)$ with transaction $t$ violating rule $r$*/;
2: for each rule $r \in \Sigma$ do   ArrayS[r].num := 0; ArrayS[r].state := 'sat';
   /*ArrayS is an array that stores the status between rules and transactions*/
3: Let $\Sigma_{pv}$ be an empty set;   /*store rules violated by the previous transaction*/
4: Create prefix trees $PT_l$ and $PT_r$ for all the left and right hand sides of rules in $\Sigma$;
5: for each transaction $t_i$ in the testing GDP do   /*$i \in [1, |\text{GDP}|]$*/
6:     $\Sigma_{cv}$ := the set of rules that $t_i$ violates by using prefix trees $PT_l$ and $PT_r$;
7:     for each rule $r \in \Sigma_{cv}$ do
8:         if $r \notin \Sigma_{pv}$ then                  /*$r$ is violated for the first time*/
9:             ArrayS[r].num := 1;
10:            if transaction $t_{i-1}$ satisfies $r$ then ArrayS[r].state := 'sat';
11:            else ArrayS[r].state := 'na';
12:        else ArrayS[r].num ++;                  /*$r$ is violated previously*/
13:        if (ArrayS[r].num > $tol^{va}_r$ and ArrayS[r].state = 'sat') or
           (ArrayS[r].num > $tol^{vi}_r$ and ArrayS[r].state = 'na') then
14:            $R := R \cup \{(t, r)\}$;
15:    $\Sigma_{pv} := \Sigma_{cv}$;
16: return $R$.
```

Algorithm. We now introduce the details of our method, shown in Algorithm 2, for detecting the global anomalies with non-redundant association rules with tolerances. It takes as input the GDP of testing data and set Σ of non-redundant rules with tolerances, and outputs the set R of transaction-rule pairs (t, r) such that t violates r.

(1) It first initializes the set R to be empty (line 1), and the array *ArrayS* such that for each rule r, ArrayS[r].num = 0, which stores the number of violated transactions of r so far, and ArrayDict[r].state = 'sat', which is the state of the previous transaction *w.r.t.* r when r is firstly violated by a transaction (line 2). ArrayDict[r].state is either 'sat' for satisfiable or 'na' for irrelevant. *ArrayS* serves for the purpose of dealing with tolerances of association rules. It also initializes an empty set Σ_{pv} to store the set of rules violated by the previous transaction (line 3).

(2) It then creates two prefix trees PT_l and PT_r for all the left and right hand sides of the rules in Σ, respectively, to aid the fast search of violated rules for transactions (line 4), whose nodes are items, and the parent nodes appear more frequently than their children in the left or right hand sides of the rules.

(3) It next determines whether a transaction t_i ($i \in [1, |\text{GDP}|]$) in GDP is an anomaly one by one (lines 5-15). (a) The set Σ_{cv} of the rules that t_i violates is identified by using prefix trees PT_l and PT_r as follows (line 6). It first searches the prefix tree PT_r, and adds the rules whose right hand sides are contained in transaction t_i to Σ_{rhs}. During the search, if an item in PT_r does not appear in t_i, its children nodes will be ignored. It then computes the set Σ_{cv} of rules violated by t_i. If $\Sigma_{rhs} = \Sigma$, t_i does not violate any rule, and Σ_{cv} is simply empty. Otherwise, it computes the Σ_{lhs} of the rules whose left hand sides are contained in transaction t_i, similarly, and Σ_{cv} is $\Sigma_{lhs} \setminus \Sigma_{rhs}$. (b) For each rule $r \in \Sigma_{cv}$, it determines whether r is firstly or previously violated. If $r \notin \Sigma_{pv}$, it is violated for the first time, ArrayS[r].num is set to 1, and ArrayS[r].state is set to 'sat' if the previous transaction t_{i-1} satisfies r, and 'na', otherwise (lines 8-11). If $r \in \Sigma_{pv}$, r is violated by the previous transaction t_{i-1}, and ArrayS[r].num is simply increased by 1 (line 12). (c) Whether transaction t violates rule r is determined by the status of ArrayS[r] and the tolerances of r (line 13), and (t, r) is added to R if t violates r (line 14). (d) The last thing to do is to replace Σ_{pv} with Σ_{cv} for handling the next transaction t_{i+1} (line 15).

(4) After all transactions in GDP are handled, the set R of transaction-rule pairs is returned (line 16).

Time complexity analysis. We only need to consider the dominant first step of algorithm 2. It takes $O(|\Sigma| * |\mathcal{I}| * \log |\mathcal{I}|)$ time to create the two prefix trees [14], where $|\mathcal{I}|$ is the number of items in GDP, and the heights of prefix trees are bounded by $\log |\mathcal{I}|$. It takes $O(|\mathcal{I}| * \log |\mathcal{I}|)$ time to search the violated rules for a transaction, and takes $O(|\text{GDP}| * |\mathcal{I}| * \log |\mathcal{I}|)$ time for all transactions. Hence, Algorithm 2 in total takes $O((|\text{GDP}| + |\Sigma|) * |\mathcal{I}| * \log |\mathcal{I}|)$ time.

5 LOCAL ANOMALY DETECTION

In this section, we introduce our local anomaly detection method based on AT [56] on the LDP of time series data by replacing Self-Attention [54] with LSH Attention [29] to fit for the discrete data, which both improves the detection accuracy and reduces the theoretical complexity.

Anomaly Transformer (AT**)**. For ease of understanding, we first briefly introduce AT [56]. AT is the current SOTA anomaly detection method for time series data, which utilizes the adjacent-concentration bias between normal and abnormal data points. Normal data points are easier to build associations with the entire data points, while anomalies are harder due to their rarity. Based on this, AT employs Transformers [54] to model the temporal association of each data point, which is called *series-association* and the learnable Gaussian kernel to represent the adjacent-concentration inductive bias of each data point, which is called *prior-association*. When focusing on a small neighborhood, *i.e.,* 100 data points, the difference between the two associations is used as a criterion to distinguish abnormal from normal data points, which is called *Association Discrepancy* and can be used to detect anomilies.

AT uses a common reconstruction framework that consists of alternating stacked Anomaly Attention blocks and Feed Forward neural network layers. In Anomaly Attention block, it computes the series and prior association and reconstruct the data as the product of the two associations.

AssDis is formulated as the symmetric KL divergence between prior and series associations, where the association differences at different levels are averaged to obtain the combined information of features at different levels. Note that the AssDis of an anomaly is typically larger than normal data points.

The series association can be well trained by data reconstruction. Also, in order to amplify the difference between normal and abnormal data points, AssDis is used as an additional loss. Hence, the loss function of AT is formalized as follows:

$$\mathcal{L}_{\text{Total}}(\hat{\mathcal{X}}, \mathcal{P}, \mathcal{S}, \lambda; \mathcal{X}) = ||\hat{\mathcal{X}} - \mathcal{X}||_F^2 - \lambda \times ||\text{AssDis}(\mathcal{P}, \mathcal{S}; \mathcal{X})||_1. \quad (1)$$

Here $\hat{\mathcal{X}} \in \mathbb{R}^{n \times m}$ is the reconstruction of $\mathcal{X}$, $||\cdot||_F$ and $||\cdot||_1$ are Frobenius and 1-norm, respectively. λ is a regularization coefficient, which is to trade off the loss terms. When $\lambda > 0$, association discrepancy is to be enlarged during optimization. As maximizing the association discrepancy only makes the parameters of Gaussian kernel function converge to zero [38], *i.e.,* invalidating the prior association, a minimax strategy is employed to train the model.

5.1 Anomaly Transformer with LSH Attention

The performance of AT [56] on the LDP of time series data is unstable. This lies in that data discretization is affected by the local factor, especially the resulting vacancies (*i.e.,* $-\infty$) in LDP, and affects the $Q\mathcal{K}^T$ in Equation (5), and may result in an unstable $\mathcal{S}^l$.

To deal with this, we only concentrate on the data point neighborhoods with high similarities based on the long-tailedness [29] of the original distribution of Self-Attention, *i.e.,* the results of the dot product calculation of a few queries and keys dominate the distribution of $\mathcal{S}^l$ after Softmax$(\cdot)$. Our discretization strategy makes the long-tail much more obvious, since it reduces the possible values of the associations. Therefore, for a $q_i \in Q$, we only need to focus on the keys in $\mathcal{K}$ that are closest to q_i, which has almost no effects on the detection accuracy. We adopt LSH Attention [29], instead of Self-Attention [54], to calculate series association $\mathcal{S}$. It adopts a locally sensitive hash function that converts vector q_i into hash $h(q_i)$ and the k_j close to q_i is more likely to have the same hash value, while the one far away is not. This choice also makes LSH Attention much more stable for varied local factors, which shall be verified by the experimental study in Section 6. It can quickly find q_i's nearest neighbors in a high-dimensional space. Besides, to save the number of parameters, LSH Attention lets $Q = \mathcal{K}$ and obtains a shared-QK Attention. Compared with the original Self-Attention of AT, the complexity of LSH Attention decreases to $O(nlogn)$ [29], where n is the number of transactions in LDP.

We refer to AT by replacing the self Attention of the anomaly transformer with LSH Attention as AT-LSH. AT-LSH is trained with a mininmax strategy on the LDP of the training time series data. Its parameter θ is firstly initialized randomly. The training LDP is then input into the model AT-LSH $_\theta(\cdot)$ to get the reconstruction of the training LDP $\hat{\mathcal{X}}$, prior association $\mathcal{P}$ and series association $\mathcal{S}$. Finally, θ is updated with the minimax strategy as follows:

$$\begin{aligned} &\textit{Minimize Phase: } \theta = \theta + \alpha \nabla_\theta \mathcal{L}_{\text{Total}}(\hat{\mathcal{X}}, \mathcal{P}, \mathcal{S}_{\text{detach}}, -\lambda; \mathcal{X}, \theta), \\ &\textit{Maximize Phase: } \theta = \theta + \alpha \nabla_\theta \mathcal{L}_{\text{Total}}(\hat{\mathcal{X}}, \mathcal{P}_{\text{detach}}, \mathcal{S}, \lambda; \mathcal{X}, \theta). \end{aligned} \quad (2)$$

Here $\lambda > 0$, $\alpha \in (0, 1)$ is the learning rate and $*_{\text{detach}}$ means to stop the gradient backpropagation of the association.

5.2 Local Anomaly Detection

We next present the local anomaly detection on the LDP of testing time series data with a trained AT-LSH model.

Similar to AT, AT-LSH incorporates the normalized association discrepancy to the reconstruction criterion, which can act as weight to amplify association discrepancy if a data point has a bad reconstruction. Thus, anomalies with bad reconstruction can be easy to detect. The anomaly score $\text{AScore}(\mathcal{X}) \in \mathbb{R}^{n \times 1}$ is a point-wise anomaly criterion of all data points denoted by $\mathcal{X}$:

$$\text{AScore}(\mathcal{X}) = \text{Softmax}\big(-\text{AssDis}(\mathcal{P}, \mathcal{S}; \mathcal{X})\big) \odot \left[||\hat{\mathcal{X}}_{i,:} - \mathcal{X}_{i,:}||_2^2\right]_{i \in [1,n]}. \quad (3)$$

Here $\odot$ is the Hadamard product, denoting the multiplication of the corresponding elements in two vectors, $||\cdot||_2$ is 2-norm and $\mathcal{X}_{i,:}$ denotes taking the i-th row of the two-dimensional matrix $\mathcal{X}$.

First, it takes as input the LDP of the testing time series data into the trained model of AT-LSH to get the reconstruction of the LDP, prior and series association. Second, it calculates the anomaly scores $\text{AScore}(\mathcal{X})$ for all data points. Finally, if the anomaly score of a data point exceeds a given threshold, it is an anomaly, where AT-LSH uses the statistic based method of AT to find a proper threshold for the anomaly scores.

Time complexity analysis. We only need to consider the first step, which is the dominant of our algorithm. By Equation (5), it takes O(|LDP|) time to get prior associations, and $O(|\text{LDP}|^2)$ time to get the reconstruction of LDP. It also takes $O(|\text{LDP}| \log |\text{LDP}|)$ time to get series associations with LSH Attention [29]. Hence, the local detection takes in total $O(|\text{LDP}|^2)$ time.

6 EXPERIMENTAL STUDY

Utilizing two real-life datasets, we conduct an extensive experimental study to validate the advantage of our approach.

6.1 Experimental Settings

Real-life datasets. We use two public real-life time series datasets.

(1) SWaT test bench [21] records the operations of a scaled down sewage treatment plant, collected by sensors for each second. The operation at each timestap is a data point consisting of 25 continuous and 26 discrete attributes with their values. 11 out of 25 continuous attributes are discetized and kept. The training SWaT contains 496, 800 data points with no anomalies, and the testing one contains 449, 919 data points with 12.1% labelled anomalies.

(2) WADI test bench [3], similar to SWaT, is collected by sensors (67 continuous and 60 discrete attributes). Similarly, 8 out of 67 continuous attributes are discetized and kept. The training WADI contains 1, 048, 571 data points with no anomalies, and the testing one contains 172, 801 data points with 5.8% labelled anomalies.

Note that, the training time series contain no anomalies, which is common for time series anomaly detection [16, 25, 31, 33, 56].

Evaluation metrics. We adopt the running time to evaluate the efficiency of anomaly detection and the commonly used Precision, Recall and F1 score (F1) [19] to evaluate the accuracy, respectively.

Algorithms and implementation. Algorithms are all implemented with Python. The tests are conducted on Intel Xeon Gold 6148 CPU @ 2.40GHz and NVIDIA Tesla V100 PCIe 32GB GPU. We only use GPU when training deep learning models, and CPU for the others. For AR method, the threshold of support and confidence are set to 0.7 and 0.9, respectively. For the AT-LSH method, the LSH Attention module uses the same parameter settings in [29], and others are kept the same as AT [56], including the training parameters. For the unified part, local factor is set to 0.5, but it is rather flexible.When quantity measures are evaluated, the tests are repeated over 5 times and the average is reported here. Note that data discretization and partition are taken into account for the efficiency tests.

6.2 Experimental Results

We next present our findings.

Exp-1: Overall comparison with existing methods. In the first set of tests, we compare our unified approach with 5 existing methods, *i.e.,* LSTM [31], OCSVM [25], MAD-GAN [33], GDN [16] and AT [56], together with our global method AR and local method AT-LSH. We use the same settings for the parameters of LSTM, OCSVM, MAD-GAN, GDN and AT as described in their papers. The minimum support and confidence of AR are set to 0.7 and 0.9, respectively, and the local factor threshold is set to 0.5. LSTM, OCSVM, MAD-GAN, GDN and AT use the original data, while our AR, AT-LSH and unified method use the discretized data. Note that our AR and AT-LSH can serve as a method on the entire dataset alone. The results are reported in Table 3.

Dataset	Method	Time (s)	Precision (%)	Recall (%)	F1 Score (%)
SWaT	LSTM [31]	2,074	96.2	62.2	75.6
	OCSVM [25]	301	92.5	69.9	79.6
	MAD-GAN [33]	504	98.9	63.7	77.1
	GDN [16]	486	97.7	68.0	80.2
	AT [56]	4,744	93.7	93.7	93.7
	Our AR	**108**	97.7	81.6	88.9
	Our AT-LSH	3,801	99.1	91.3	95.0
	Ours	2,706	**99.3**	**96.9**	**98.1**
WADI	LSTM [31]	825	92.2	28.6	43.7
	OCSVM [25]	685	91.2	26.3	40.8
	MAD-GAN [33]	276	41.4	33.9	37.1
	GDN [16]	239	97.5	40.2	57.0
	AT [56]	8,980	72.1	**100**	83.8
	Our AR	**84**	93.6	44.9	60.7
	Our AT-LSH	5,891	**97.8**	85.2	91.1
	Ours	4,763	97.1	90.7	**93.8**

Table 3: Overall comparison

AT [56] is the SOTA existing method in terms of the F1 score, but is the slowest method. Our AR alone has a very high precision, and is fastest method. Our AT-LSH is faster than AT, and the running time deceases 20% and 34.4% on SWaT and WADI, respectively, compared with AT. Our unified approach is much faster than AT, and the running time deceases 43.0% and 47.0% (from 4, 744s to 2, 706s and from 8, 980s to 4, 763s) on SWaT and WADI, respectively, compared with AT. Our AT-LSH alone also has a very high precision. Our unified approach achieves the best F1 score and almost the best precision and recall, and its F1 score is on average 4.7% and 11.9% better than AT on SWaT and WADI, respectively. These justify the design of our global method AR and local method AT-LSH, and the advantage of their unification in both efficiency and effectiveness.

Dataset	Discretization	Time (s)	Precision (%)	Recall (%)	F1 Score (%)
SWaT	No	4,474	93.7	**93.7**	93.7
	Yes	**2,627**	**98.9**	92.2	**95.4**
WADI	No	8,980	72.1	**100**	83.8
	Yes	**4,651**	**97.9**	87.6	**92.5**

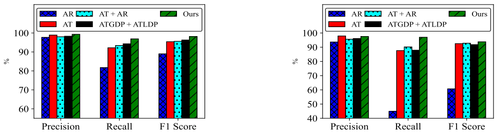
Table 4: With *v.s.* without data discretization

Exp-2: Data discretization. In the second set of tests, to evaluate the effectiveness of data discretization, we train and test AT on the original data and the discretized data, respectively, along the same setting as Exp-1. The results are reported in Table 4.

The running time on the discretized data decreases 44.6% and 48.1% on SWaT and WADI, respectively, partially due to the removal of (mostly useless) attributes that are hard to discretize. AT on the discretized data has a much higher precision and F1 score than AT on the original data on both SWaT and WADI. This indicates that data discretization can improve the precision as our design purpose is to have methods with high precisions. One possible explanation is that data discretization makes the boundary between abnormal and normal data points clearer, which in turn reduces false detections and improves the precision, and the discarded attributes (hard to discretize) may contain noises that have negative impacts on anomaly detection. Data discretization causes a loss of the recall, as it causes certain loss of information. However, this is an acceptable cost as our unified method naturally improves the recall.

Exp-3: Data partition. In the third set of tests, we verify the effectiveness of our data partition based unification along the same setting as Exp-1. We compare our method with two other methods, *i.e.,* AT +AR and ATGDP +ATLDP, on the dicretized data, together with AR and AT. Here AT +AR is the direct union of the detected anomalies of AT and AR on the entire dicretized data, and ATGDP +ATLDP is the union of the detected anomalies of AT on the LDP and GDP of the data. The results are reported in Figure 2.

Among all unifications, our data partition based unification, *i.e.,* the union of AR on GDP and AT-LSH on LDP, achieves the best recall and F1 score and almost the best precision on SWaT and WADI. Our unification also beats AR and AT alone. This justifies our unification of global and local anomaly detection method, which takes advantage of their strengths and avoids their weaknesses.



(a) SWaT **(b) WADI**

Figure 2: Data partition

Exp-4: Our global detection method. In the fourth set of tests, we verify the effectiveness of our global anomaly detection method AR on the GDP of SWaT and WADI along the same setting as Exp-1.

Exp-4.1. To evaluate the impacts of non-redundant rules, we vary the data size from 25% to 100%, the confidence threshold from 0.7 to 1.0, and the support threshold from 0.7 to 0.85 on the GDP of SWaT and WADI, respectively, and the other parameters are along the same setting as Exp-1. The tests are on the GDP of the training data for rule numbers, and on the GDP of the testing data for detection time. The results are reported in Figures 3 & 4 .

The detection time obviously increases with the increase of the testing data size, but the detection time with non-redundant rules increases much slower than with all rules, demonstrating strong scalability for large-scale time series. The detection with non-redundant rules is much faster, as shown in Figure 3a. Indeed, the detection with non-redundant rules is on average 23 and 19 times faster than with all rules on SWaT and WADI, respectively. These confirm to the time complexity analysis of our global detection method.

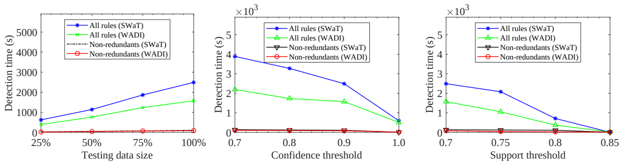
The detection time with all rules drops quickly with the increase of the thresholds of support and confidence, while the detection time with non-redundant rules is much less sensitive to these two parameters, as shown in Figures 3b & 3c. This is because these detections use different number of rules shown in Figures 4b & 4c.

Both the numbers of all rules and non-redundant rules are not very sensitive to the size of training data as shown in Figure 4a, which possible means that 25% of training data may already large enough for generating enough rules, indicating the applicability of our approach to large-scale data. Further, the number of all rules is on average 132 and 462 times of the number of non-redundant rules on SWaT and WADI, respectively. The number of all rules obviously drops quickly with the increase of the thresholds of support and confidence, while the number of non-redundant rules is much less sensitive to these two parameters, as shown in Figures 4b & 4c.

Exp-4.2. To evaluate the impacts of tolerances, we test our global method AR using association rules with and without tolerances, respectively. The results are reported in Table 5.

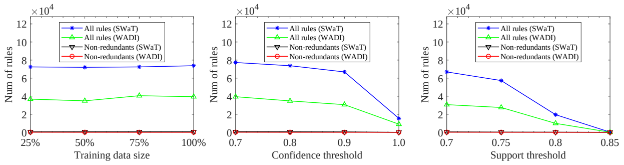
The precision and F1 score of AR are significantly improved when using tolerances, *e.g.,* from 37.4% to 97.7% and from 42.7% to 93.6% for the precision on SWaT and WADI, and from 52.0% to 88.9% and from 49.2% to 60.7% for the F1 score on SWaT and WADI, respectively. The running time of AR with tolerances is slightly increased as an obvious acceptable cost. This justifies the introducing of tolerances for association rules as our main focus is to improve the precision of global anomaly detection.

These together justify the use of non-redundant association rules to avoid the generation of excessive rules, and to improve the efficiency performance of global anomaly detection.



(a) Vary data sizes **(b) Vary confidence** **(c) Vary support**

Figure 3: Detection time with $v.s.$ **without redundance on** GDP



(a) Vary data sizes **(b) Vary confidence** **(c) Vary support**

Figure 4: Rule number with $v.s.$ **without redundance on** GDP

Dataset	Tolerances	Time (s)	Precision (%)	Recall (%)	F1 Score (%)
SWaT	No	**88**	37.4	**85.3**	52.0
	Yes	108	**97.7**	81.6	**88.9**
WADI	No	**68**	42.7	**57.9**	49.2
	Yes	84	**93.6**	44.9	**60.7**

Table 5: With $v.s.$ **without tolerances on** GDP

Exp-5: Our local detection method. In the fifth set of tests, we justify the stableness of our local anomaly detection method AT-LSH on the LDP of SWaT and WADI. We vary the local factor threshold from 0.1 to 0.80 with the other parameters along the same setting as Exp-1, and compare our AT-LSH with AT [56]. The results are reported in Table 6.

The precisions of AT and AT-LSH are relatively stable on both SWaT and WADI when varying the local factor threshold. The recall of AT-LSH is stable on both SWaT and WADI when varying the local factor threshold, while the recall of AT is unstable when varying the local factor threshold. This makes AT unstable for anomaly detection when varying the local factor threshold. In contrast, AT-LSH is much more stable. Further, the F1 score of AT-LSH is better than AT when the local factor falls into a wide range of $[0.25, 0.75]$. These justify the design of our local anomaly detection method using the Anomaly Transformer with LSH Attention.

Summary. From these tests, we find the followings.

(1) The running time of our unified approach is decreased from $4,744$s to $2,706$s and from $8,980$s to $4,763$s on SWaT and WADI, respectively, compared with AT. Our unified approach achieves the best F1 score and almost the best precision and recall, and its F1 score is on average 4.7% and 11.9% better than the existing SOTA method AT on SWaT and WADI, respectively.

(2) The detection time of AT on discrete data is descreased 44.6% and 48.1% on SWaT and WADI, while the precision of AT on discrete data is increased 5.2% and 25.8% on SWaT and WADI. Recall of unification with data partition is increased 3.5% and 6.8% on SWaT and WADI, respectively. Together they make the unifying of global and local method with high efficiency and accuracy possible.

(3) Our global method AR with non-redundant association rules with tolerances has a lower increasing rate compared with all rules

	Local Factor		0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
SWaT	Precision (%)	AT [56]	98.9	99.0	99.0	99.0	99.0	99.1	99.1	99.1	99.1	99.1	99.1	99.1	99.1	99.1	99.1
		AT-LSH	99.0	99.1	99.0	99.0	99.3	98.9	98.9	98.9	99.3	99.3	99.1	99.1	99.0	99.1	99.1
	Recall (%)	AT [56]	88.3	93.9	87.0	87.0	93.8	88.5	88.5	88.5	94.3	94.3	91.8	91.8	87.5	84.9	84.9
		AT-LSH	87.2	93.3	89.6	89.6	93.5	92.6	92.6	92.6	94.1	94.1	93.0	93.0	89.8	84.8	84.8
	F1 Score (%)	AT [56]	93.3	96.4	92.6	92.6	96.3	93.5	93.5	93.5	96.6	96.6	95.3	95.3	92.9	91.4	91.4
		AT-LSH	92.7	96.1	94.1	94.1	96.6	95.6	95.6	95.6	96.1	96.6	95.9	95.9	94.2	91.4	91.4
WADI	Precision (%)	AT [56]	97.6	97.6	97.3	97.0	97.6	97.6	97.6	97.6	97.5	97.5	97.5	97.5	97.5	97.0	95.3
		AT-LSH	97.6	97.6	97.1	97.2	97.6	97.8	97.8	97.8	97.8	97.5	97.5	97.5	97.5	97.1	95.4
	Recall (%)	AT [56]	87.9	87.9	68.1	58.3	81.2	81.2	81.2	81.2	80.3	80.3	80.3	80.3	80.3	68.9	36.8
		AT-LSH	84.6	84.6	67.2	76.1	81.1	81.1	81.1	81.1	81.4	81.4	81.4	81.4	81.4	83.9	52.9
	F1 Score (%)	AT [56]	92.5	92.5	80.1	72.8	88.6	88.6	88.6	88.6	88.1	88.1	88.1	88.1	88.1	80.6	53.1
		AT-LSH	90.6	90.6	79.4	85.4	88.7	88.7	88.7	88.7	88.8	88.8	88.8	88.8	88.8	90.0	68.1

Table 6: AT-LSH *v.s.* AT **on** LDP

when data size increases, showing scalability for large-scale data. AR is also effective, where the use of non-redundant rules improves the detection efficiency with 21 times faster and the use of tolerances improves on average 24.2% for the F1 score.

(4) Our local method AT-LSH is stable, and outperforms AT even when the local factor falls into a wide range of $[0.25, 0.75]$.

7 RELATED WORK

Anomaly detection for time series data can be classified into two categories, *i.e.,* global and local anomaly detection methods, based on the amount of data used to determine anomalies.

Global methods. These methods first mine the global features from the entire time series, and then treat the data points that do not satisfy the features as anomalies. According to the different ways to mine global features, they can be divided into rule-based, cluster-based, reconstruction-based and prediction-based methods.

Rules and their invariants are global features to maintain the conditions during the operations of a system in a given state [53]. Most rule-based methods transform the data points of time series into transactions and attribute-value pairs as items, among which rules are commonly mined with standard frequent patterns [18, 32, 35, 37, 42]. These approaches are usually simple, and ignore the physical properties of rules. A systematic framework for invariant-based anomaly detection is proposed in [19], which further introduces the concept of meaningful rules to reduce the number of rules and to improve the accuracy and efficiency of detection. A supervised classification method LAD is proposed in [15] to extract rules from time series, but its training phase requires the use of the same amount of normal and abnormal data, which is difficult to obtain in practice.

Clustering-based methods learn the global features with classification models, *e.g.,* SVDD [51] and Deep SVDD [43] perform clustering in the mapping feature space of time series, and THOC [46] adopts the idea of hierarchical clustering to fuse multi-scale temporal features and detect anomalies with multi-layer distances.

The reconstruction-based methods learn the global features with a data reconstruction task using deep learning models, and perform anomaly detection with the reconstruction errors, *e.g.,* Variational Auto-Encoder [28, 34], Variational Auto-Encoder based LSTM-VAE [49], OmniAnomaly [49] and VGCRN [12], GAN based MAD-GAN [33], BeatGAN [60], f-AnoGAN [45] and MAD-GAN [33], and transformer based TranAD [52].

Prediction-based methods learn the global features by predicting the data in the next time with deep learning models, and determine the anomalies with the prediction errors, *e.g.,* LSTM based [22, 25], CNN based [31] and GNN based [16, 23].

We adopt rule-based methods as our global detection method due to their high precision and efficiency, and propose non-redundant association rules with tolerances to cope with the physical properties of time series to improve the preformance.

Local methods. These methods consider only the restricted neighborhood of a data point in a time series and distinguish the anomalies from normal data points with the local differences.

Compared with global methods, there are relatively less studies on local anomaly detection. KNN based methods detect anomalies with the neighborhood distances, *e.g.,* [4]. Distance based methods, *e.g.,* LOF [8] and its variants MiLOF [44], DILOF [36] and CELOF [10], identify the anomalies with the distance between a data point and clusters according to the density around the data point. Anomaly Transformer (AT) [56] is a deep learning based method, which utilizes the differences of the associations between normal and abnormal data points in the neighborhood to detect anomalies.

AT [56] is the SOTA method for the anomaly detection of time series data, and we improve AT by replacing its Self-Attention with LSH Attention as our local anomaly detection method to fit better for the local anomaly detection of time series data.

To the best of our knowledge, this study is among the first to unify the global and local anomaly detection for time series data with a better performance. We make the unification possible with data discretization and partition of the time series data into GDP and LDP. The unification also naturally brings the benefit of a high recall due to the union ensemble of anomalies, and our focus goes to improve the precision of both global and local anomaly detection, and to take advantage of their strengths and avoid their weaknesses.

8 CONCLUSIONS

We have proposed an approach to unifying the global and local anomaly detection for times series data with good performance. We have firstly utilized data discretization and partition to make the unification possible, then designed a rule-based global anomaly detection method using non-redundant association rules with tolerances, and developed a Transformer with LSH Attention based local anomaly detection method. We have finally experimentally verified that our unified approach is much better than the SOTA method AT [56] in terms of both accuracy and efficiency.

A couple of issues need further studies. First, although data discetization seems engineering, it seems important for anomaly detection. We are trying other alternatives to handle more complicated cases. Second, we are exploring possibilities to unify global and local methods for other data mining tasks.

REFERENCES

[1] [n.d.]. https://mashuai-ms.github.io/paper-full.pdf.

[2] Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast Algorithms for Mining Association Rules in Large Databases. In *VLDB*.

[3] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P Mathur. 2017. WADI: a water distribution testbed for research in the design of secure cyber physical systems. In *CySWATER@CPSWeek*.

[4] Fabrizio Angiulli and Clara Pizzuti. 2002. Fast Outlier Detection in High Dimensional Spaces. In *PKDD*.

[5] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. 2018. Truth will out: Departure-based process-level detection of stealthy attacks on control systems. In *CCS*.

[6] Yves Bastide, Nicolas Pasquier, Rafik Taouil, Gerd Stumme, and Lotfi Lakhal. 2000. Mining minimal non-redundant association rules using frequent closed itemsets. In *Computational Logic*.

[7] Christian Borgelt. 2012. Frequent item set mining. *WIREs Data Mining Knowl. Discov.* 2, 6 (2012), 437–456.

[8] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: Identifying Density-Based Local Outliers. In *SIGMOD*.

[9] Defense Use Case. 2016. Analysis of the cyber attack on the Ukrainian power grid. *E-ISAC* 388 (2016), 1–29.

[10] Liang Chen, Wei Wang, and Yun Yang. 2021. CELOF: Effective and fast memory efficient local outlier detection in high-dimensional data streams. *Applied Soft Computing* 102 (2021), 107079.

[11] Thomas M Chen and Saeed Abu-Nimeh. 2011. Lessons from stuxnet. *Computer* 44, 4 (2011), 91–93.

[12] Wenchao Chen, Long Tian, Bo Chen, Liang Dai, Zhibin Duan, and Mingyuan Zhou. 2022. Deep Variational Graph Convolutional Recurrent Network for Multivariate Time Series Anomaly Detection. In *ICML*.

[13] Vickram Chundhoo, Gopinath Chattopadhyay, Gour Karmakar, and Gayan Kahandawa Appuhamillage. 2021. Cybersecurity Risks in Meat Processing Plant and Impacts on Total Productive Maintenance. In *ICMIAM*.

[14] Richard H. Connelly and F. Lockwood Morris. 1995. A Generalization of the Trie Data Structure. *Math. Struct. Comput. Sci.* 5, 3 (1995), 381–418.

[15] Tanmoy Kanti Das, Sridhar Adepu, and Jianying Zhou. 2020. Anomaly detection in industrial control systems using logical analysis of data. *Comput. Secur.* 96 (2020), 101935.

[16] Ailin Deng and Bryan Hooi. 2021. Graph neural network-based anomaly detection in multivariate time series. In *AAAI*.

[17] Paweł D. Domański. 2020. *Frequency Based Methods.* Springer International Publishing, 91–94.

[18] Entisar E Eljadi and Zulaiha Ali Othman. 2011. Anomaly detection for PTM's network traffic using association rule. In *DMO*.

[19] Cheng Feng, Venkata Reddy Palleti, Aditya Mathur, and Deeph Chana. 2019. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In *NDSS*.

[20] Liqiang Geng and Howard J. Hamilton. 2006. Interestingness Measures for Data Mining: A Survey. *CSUR* 38, 3 (2006), 9–es.

[21] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. 2016. A dataset to support research in the design of secure water treatment systems. In *CRITIS*.

[22] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. 2017. Anomaly detection in cyber physical systems using recurrent neural networks. In *HASE*.

[23] Siho Han and Simon S. Woo. 2022. Learning Sparse Latent Graph Representations for Anomaly Detection in Multivariate Time Series. In *KDD*.

[24] Nan-Chen Hsieh. 2004. An integrated data mining and behavioral scoring model for analyzing bank customers. *Expert Systems with Applications* 27, 4 (2004), 623–633.

[25] Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M Poskitt, and Jun Sun. 2017. Anomaly detection for a water treatment system using unsupervised machine learning. In *ICDM Workshops*.

[26] Blake Johnson, Dan Caban, Marina Krotofil, Dan Scali, Nathan Brubaker, and Christopher Glyer. 2017. Attackers deploy new ICS attack framework "TRITON" and cause operational disruption to critical infrastructure. *Threat Research Blog* 14 (2017).

[27] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. 2001. An online algorithm for segmenting time series. In *ICDM*.

[28] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

[29] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The Efficient Transformer. In *ICLR*.

[30] Walter A. Kosters, Wim Pijls, and Viara Popova. 2003. Complexity Analysis of Depth First and FP-Growth Implementations of APRIORI. In *MLDM*.

[31] Moshe Kravchik and Asaf Shabtai. 2018. Detecting cyber attacks in industrial control systems using convolutional neural networks. In *CPS-SPC@CCS*.

[32] Jennifer Leopold, Bruce McMillin, Rachel Stiffler, and Nathan Lutes. 2020. Comparison of Design-Centric and Data-Centric Methods for Distributed Attack Detection in Cyber-Physical Systems. In *Critical Infrastructure Protection*.

[33] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. 2019. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In *ICANN*.

[34] Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. 2021. Multivariate Time Series Anomaly Detection and Interpretation Using Hierarchical Inter-Metric and Temporal Embedding. In *KDD*.

[35] Matthew V Mahoney and Philip K Chan. 2003. Learning rules for anomaly detection of hostile network traffic. In *ICDM*.

[36] Gyoung S. Na, Donghyun Kim, and Hwanjo Yu. 2018. DILOF: Effective and Memory Efficient Local Outlier Detection in Data Streams. In *KDD*.

[37] Kazuyo Narita and Hiroyuki Kitagawa. 2008. Outlier detection for transaction databases using association rules. In *WAIM*.

[38] Radford M Neal. 2007. Pattern Recognition and Machine Learning. *Technometrics* 49, 3 (2007), 366–366.

[39] Shrivastava Neeraj and Lodhi Singh Swati. 2012. Overview of non-redundant association rule mining. *Research Journal of Recent Sciences* 1, 2 (2012), 108–112.

[40] Nell Nelson. 2016. The impact of dragonfly malware on industrial control systems. *SANS Institute* (2016).

[41] C Osborne. 2021. Colonial Pipeline attack: Everything you need to know. *ZD Net* (2021).

[42] Koyena Pal, Sridhar Adepu, and Jonathan Goh. 2017. Effectiveness of association rules mining for invariants generation in cyber-physical systems. In *HASE*.

[43] Lukas Ruff, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Robert A. Vandermeulen, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep One-Class Classification. In *ICML*.

[44] Mahsa Salehi, Christopher Leckie, James C. Bezdek, Tharshan Vaithianathan, and Xuyun Zhang. 2016. Fast Memory Efficient Local Outlier Detection in Data Streams. *TKDE* 28, 12 (2016), 3246–3260.

[45] Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. 2019. f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis* 54 (2019), 30–44.

[46] Lifeng Shen, Zhuocong Li, and James Kwok. 2020. Timeseries anomaly detection using temporal hierarchical one-class network. In *NeurIPS*.

[47] Neil J Smelser and Paul B Baltes. 2001. *International encyclopedia of the social & behavioral sciences.* Elsevier Amsterdam.

[48] Ralf Spenneberg, Maik Brüggemann, and Hendrik Schwartke. 2016. Plc-blaster: A worm living solely in the plc. *Black Hat Asia* 16 (2016), 1–16.

[49] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network. In *KDD*.

[50] Laszlo Szathmary, Petko Valtchev, Amedeo Napoli, Robert Godin, Alix Boc, and Vladimir Makarenkov. 2014. A fast compound algorithm for mining generators, closed itemsets, and computing links between equivalence classes. *Ann. Math. Artif. Intell.* 70, 1 (2014), 81–105.

[51] Robert Tibshirani, Guenther Walther, and Trevor Hastie. 2001. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63, 2 (2001), 411–423.

[52] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. 2022. TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. *PVLDB* 15, 6 (2022), 1201–1214.

[53] Muhammad Azmi Umer, Aditya Mathur, Khurum Nazir Junejo, and Sridhar Adepu. 2017. Integrating design and data centric approaches to generate invariants for distributed attack detection. In *CPS-SPC@CCS*.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*.

[55] Chao Wang, Bailing Wang, Hongri Liu, and Haikuo Qu. 2020. Anomaly detection for industrial control system based on autoencoder neural network. *Wirel. Commun. Mob. Comput.* 2020 (2020), 1–10.

[56] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2022. Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy. In *ICLR*.

[57] M.J. Zaki and C.-J. Hsiao. 2005. Efficient algorithms for mining closed itemsets and their lattice structure. *TKDE* 17, 4 (2005), 462–478.

[58] Mohammed Javeed Zaki. 2004. Mining Non-Redundant Association Rules. *DMKD* 9, 3 (2004), 223–248.

[59] Mohammed J Zaki and Ching-Jui Hsiao. 2002. CHARM: An efficient algorithm for closed itemset mining. In *SIDMA*.

[60] Bin Zhou, Shenghua Liu, Bryan Hooi, Xueqi Cheng, and Jing Ye. 2019. BeatGAN: Anomalous Rhythm Detection using Adversarially Generated Time Series. In *IJCAI*.

9 APPENDIX

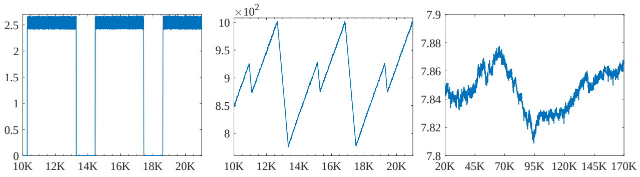
9.1 Detail of data discretization

The values of different continuous attributes may have different properties, which makes it barely impossible to discretize the time series data with a single method. Hence, we propose three different techniques to discretize time series data.

Discretization by clusters. The value distribution of certain continuous attributes is relatively concentrated, and does not have a long-term trend, such as Water Flow shown in Figure 5a. For such attributes, we use K-means to cluster their values directly, where we use a heuristic method, elbow criterion, to decide its parameter K. Parameter K is set to 2 initially, and is gradually increased by 1 until there is a sharp drop of the sum of squared errors. For a continuous attribute, we use the frequency statistics-based method [17] to determine whether the values of the attribute are concentrated. If so, we discretize the attribute by clusters.

Discretization by trend. The values of certain continuous attributes have obvious trends, and their distribution is not concentrated, such as Water Level shown in Figure 5b. For this type of attributes, we cluster their values with the trends. To alleviate the impact of noise, we smooth the values with moving average, after which we use a sliding window-based algorithm [27] to segment the values by calculating the changing rates of segments. The segments are then clustered with K-means to reduce their number, where its parameter K is determined by the changing rate distributions. For a continuous attribute, if it can't be discretized by clusters, we then segment the values by calculating the changing rates of segments and and use the frequency statistics-based method [17] to determine whether the changing rates of segments are concentrated. If so, we discretize the attribute by trend.

Discretization by the EM algorithm. The value distribution of certain continuous attributes has a multi-peak shape, such as Hydrochloric Acid (HA) Level shown in Figure 5c. For such attributes, we treat their values as a superposition of multiple Gaussian distributions to build a Gaussian mixture model, and use the EM algorithm to cluster the values so that we obtain to which Gaussian distribution the value at a certain moment belongs, and the discrete value of the attribute at this moment. For a continuous attribute, if it can't be discretized by clusters of trend, we use wavelet transform to query the number of peaks. If the number of peaks is less than a given threshold, we discretize the attribute by EM algorithm. For attributes that do not satisfy the above three conditions, we do not to discretize the attribute.



(a) Water Flow **(b) Water Level** **(c) HA Level**

Figure 5: Example continuous attributes in SWaT [21]

Note that although these three data discretization methods seem enough for our analysis of the time series data, there potentially exist other alternatives to handle more complicated cases. Also, we don not discretize all continuous attributes. In fact, we only use 11 out of 25 continuous attributes and 8 out of 67 continuous attributes in SWaT and WADI, respectively, which is enough for our detection.

9.2 Proof of proposition 4.1

Proof. This is done by showing that any redundant association rule that belongs to the complement of non-redundant association rules [6], it belongs to the complement of non-redundant association rules with tolerances. Note that the complete set of association rules [6] is the same as that of association rules with tolerances.

Suppose that there is a redundant association rule $X \longrightarrow Y$ in the complement of non-redundant association rules [6]. Then there is a non-redundant association rule $Z \longrightarrow W$ such that $Z \subseteq X$, $Y \subseteq W$, $ZY \subset XW$, $\mathrm{sup}_{Z \to W} = \mathrm{sup}_{X \to Y}$ and $\mathrm{conf}_{Z \to W} = \mathrm{conf}_{X \to Y}$ by the definition. We next show that $X \longrightarrow Y$ is in the complement of non-redundant association rules with tolerances.

By the definitions of associations rules and non-redundant association rules [6], we have $\mathrm{sup}_{ZW} = \mathrm{sup}_{Z \to W} = \mathrm{sup}_{X \to Y} = \mathrm{sup}_{XY}$ and $\mathrm{sup}_{ZW}/\mathrm{sup}_{Z} = \mathrm{conf}_{Z \to W} = \mathrm{conf}_{X \to Y} = \mathrm{sup}_{XY}/\mathrm{sup}_{X}$. From these, we have $\mathrm{sup}_{Z} = \mathrm{sup}_{X}$. By the definition of operators in Section 2, $\mathrm{sup}_{Z} = |\psi(Z)|/|\mathcal{D}|$ and $\mathrm{sup}_{X} = |\psi(X)|/|\mathcal{D}|$, which leads to $|\psi(X)| = |\psi(Z)|$, and $\psi(Z) \supseteq \psi(X)$ as $Z \subseteq X$. From these, we have $\psi(Z) = \psi(X)$.

(1) When $X \neq Z$, we construct an association rule $Z \longrightarrow XY \setminus Z$.

(a) The transactions that satisfy $Z \longrightarrow XY \setminus Z$ and $X \longrightarrow Y$ are obviously the same, which are $\psi(XY)$. (b) The transactions that violate $Z \longrightarrow XY \setminus Z$ and $X \longrightarrow Y$ are $\psi(Z) \setminus \psi(XY)$ and $\psi(X) \setminus \psi(XY)$, respectively. As $\psi(Z) = \psi(X)$, we have $\psi(Z) \setminus \psi(XY) = \psi(X) \setminus \psi(XY)$, *i.e.,* the transactions that violate $Z \longrightarrow XY \setminus Z$ and $X \longrightarrow Y$ are the same. (c) The above leads to that the transactions irrelevant to $Z \longrightarrow XY \setminus Z$ and $X \longrightarrow Y$ are the same. From the above, we have $tol^{va}_{X \to Y} = tol^{va}_{Z \to XY \setminus Z}$ and $tol^{vi}_{X \to Y} = tol^{vi}_{Z \to XY \setminus Z}$. These show that $X \longrightarrow Y$ is a redundant association rule with tolerances.

(2) When $X = Z$, we construct an association rule $X \longrightarrow W$.

(a) The transactions that satisfy $X \longrightarrow W$ and $X \longrightarrow Y$ are $\psi(XW)$ and $\psi(XY)$, respectively. As $XY \subseteq XW$, we have $\psi(XY) \supseteq \psi(XW)$. As $\mathrm{sup}_{XW} = \mathrm{sup}_{X \to W} = \mathrm{sup}_{X \to Y} = \mathrm{sup}_{XY}$, we have $|\psi(XW)| = |\psi(XY)|$. From these, we have $\psi(XW) = \psi(XY)$. (b) The transactions that violate $X \longrightarrow W$ and $X \longrightarrow Y$ are $\psi(X) \setminus \psi(XW)$ and $\psi(X) \setminus \psi(XY)$, respectively, which are obviously the same as $\psi(XW) = \psi(XY)$. (c) We now have that the transactions irrelevant to $Z \longrightarrow XY \setminus Z$ and $X \longrightarrow Y$ are the same. From the above, $tol^{va}_{X \to Y} = tol^{va}_{X \to W}$ and $tol^{vi}_{X \to Y} = tol^{vi}_{X \to W}$. These together show that $X \longrightarrow Y$ is a redundant association rule with tolerances.

Putting (1) and (2) together, we have the conclusion. □

Remarks. (1) Proposition 4.1 indicates that we can mine non-redundant association rules with tolerances based on non-redundant association rules [6]. (2) It is mostly likely that non-redundant association rules with tolerances is a proper subset of non-redundant association rules [6], *e.g.,* for a set of itemsets {{ABC}, {AD}, {ABC}, {ACE}} with the minimum support and confidence are set to 0.5.

9.3 Introduction of Anomaly Transformer

Anomaly Transformer (AT) is the current SOTA anomaly detection method for time series data, which utilizes the adjacent-concentration bias between normal and abnormal data points. Normal data points are easier to build associations with the entire data points, while anomalies are harder due to their rarity. Based on this, AT employs Transformers [54] to model the temporal association of each data point, which is called *series-association* and the learnable Gaussian kernel to represent the adjacent-concentration inductive bias of each data point, which is called *prior-association*. When focusing on a small neighborhood, *i.e.,* 100 data points, the difference between the two associations is used as a criterion to distinguish abnormal from normal data points, which is called *Association Discrepancy*.

AT uses a common reconstruction framework that consists of alternating stacked Anomaly Attention blocks and Feed Forward neural network layers. For a model containing L layers with m attributes and n input data points $\mathcal{X} \in \mathbb{R}^{n \times m}$, its overall equations of the l-th layer are formulated as follows:

$$\begin{aligned} \mathcal{Z}^l &= \mathrm{LayerNorm}\big(\mathrm{AnomalyAttention}(\mathcal{X}^{l-1}) + \mathcal{X}^{l-1}\big), \\ \mathcal{X}^l &= \mathrm{LayerNorm}\big(\mathrm{FeedForward}(\mathcal{Z}^l) + \mathcal{Z}^l\big). \end{aligned} \tag{4}$$

Here $\mathcal{X}^l, \mathcal{Z}^l \in \mathbb{R}^{n \times d_{\text{model}}}$ denote the output and hidden representation of the l-th layer, respectively. The initial input $\mathcal{X}^0 = \mathrm{Embedding}(\mathcal{X})$ represents the embedded LDP of time series data.

Its Anomaly Attention layer adopts Self-Attention [54] to model both prior and series associations. For the prior association, a learnable Gaussian kernel that concentrates on adjacent data points is used to compute the relative distance between data points in the neighborhood. The series association is obtained from the Self-Attention map. The Anomaly Attention in the l-th layer is:

$$\begin{aligned} &\mathit{Initialization}\colon \mathcal{Q}, \mathcal{K}, \mathcal{V}, \sigma = \mathcal{X}^{l-1} W_{\mathcal{Q}}^l, \mathcal{X}^{l-1} W_{\mathcal{K}}^l, \mathcal{X}^{l-1} W_{\mathcal{V}}^l, \mathcal{X}^{l-1} W_{\sigma}^l \\ &\mathit{Prior\ Association}\colon \mathcal{P}^l = \mathrm{Rescale}\left(\left[\frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(\frac{|j-i|^2}{2\sigma_i^2}\right) \right]_{i,j \in [1,n]} \right) \\ &\mathit{Series\ Association}\colon \mathcal{S}^l = \mathrm{Softmax}\left(\frac{\mathcal{Q}\mathcal{K}^T}{\sqrt{d_{\text{model}}}} \right) \\ &\mathit{Reconstruction}\colon \hat{\mathcal{Z}}^l = \mathcal{S}^l \mathcal{V}. \end{aligned} \tag{5}$$

Here $\mathcal{Q}, \mathcal{K}, \mathcal{V} \in \mathbb{R}^{n \times d_{\text{model}}}$ represent the query, key, value of Self-Attention [54] and the learned scale, respectively. $W_{\mathcal{Q}}^l, W_{\mathcal{K}}^l, W_{\mathcal{V}}^l \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ are learnable parameters. $\sigma \in \mathbb{R}^{n \times 1}$ is the learned scale to get $\mathcal{P}^l$, the i-th element of which σ_i corresponds to the i-th data point. Further, Rescale$(\cdot)$ is used to transform the association weights to discrete distributions $\mathcal{P}^l$ by dividing the row sum. Softmax$(\cdot)$ is used to normalize the Attention map along the last dimension and let each row of $\mathcal{S}^l$ be a discrete distribution.

Its association discrepancy is formulated as follows:

$$\mathrm{AssDis}(\mathcal{P}, \mathcal{S}; \mathcal{X}) = \left[\tfrac{1}{L} \textstyle\sum_{l=1}^{L} \left(\mathrm{KL}\big(\mathcal{P}_{i,:}^l || \mathcal{S}_{i,:}^l\big) + \mathrm{KL}\big(\mathcal{S}_{i,:}^l || \mathcal{P}_{i,:}^l\big) \right) \right]_{i \in [1,n]}. \tag{6}$$

Here AssDis $\in \mathbb{R}^{n \times 1}$ and $\mathrm{KL}(\cdot||\cdot)$ is the KL divergence between two discrete distributions corresponding to every row of $\mathcal{P}^l$ and $\mathcal{S}^l$. The i-th element of AssDis corresponds to the i-th data point of LDP. As shown in Equation (6), AssDis is formulated as the symmetric KL divergence between prior and series associations, where the association differences at different levels are averaged to obtain the combined information of features at different levels. Note that the AssDis of an anomaly is typically larger than normal data points.

The series association can be well trained by data reconstruction. Also, in order to amplify the difference between normal and abnormal data points, AssDis is used as an additional loss. Hence, the loss function of AT is formalized as follows:

$$\mathcal{L}_{\text{Total}}(\hat{\mathcal{X}}, \mathcal{P}, \mathcal{S}, \lambda; \mathcal{X}) = ||\hat{\mathcal{X}} - \mathcal{X}||_F^2 - \lambda \times ||\mathrm{AssDis}(\mathcal{P}, \mathcal{S}; \mathcal{X})||_1. \tag{7}$$

Here $\hat{\mathcal{X}} \in \mathbb{R}^{n \times m}$ is the reconstruction of $\mathcal{X}$, $||\cdot||_F$ and $||\cdot||_1$ are Frobenius and 1-norm, respectively. λ is a regularization coefficient, which is to trade off the loss terms. When $\lambda > 0$, association discrepancy is to be enlarged during optimization. As maximizing the association discrepancy only makes the parameters of Gaussian kernel function converge to zero [38], *i.e.,* invalidating the prior association, a minimax strategy is employed to train the model.

Dataset	Testing data size (%)	Tolerances	Time (s)	Precision (%)	Recall (%)	F1 Score (%)
SWaT	25	No	**19**	6.2	**51.1**	11.1
		Yes	27	**84.5**	29.3	**43.5**
	50	No	**24**	16.4	**60.8**	25.8
		Yes	52	**89.4**	40	**55.3**
	75	No	**42**	23.5	**83.7**	36.7
		Yes	77	**98**	80	**88.1**
	100	No	**88**	37.4	**85.3**	52
		Yes	108	**97.7**	81.6	**88.9**
WADI	25	No	**12**	5.7	**22.3**	9.1
		Yes	16	**77.4**	16.7	**27.5**
	50	No	**35**	21.9	**36.5**	27.4
		Yes	38	**86.7**	31.3	**46**
	75	No	**52**	39.6	**52.9**	45.3
		Yes	69	**90.1**	42.1	**57.4**
	100	No	**68**	42.7	**57.9**	49.2
		Yes	84	**93.6**	44.9	**60.7**

Table 7: With *v.s.* without tolerances with varying data size

Dataset	Support Threshold	Tolerances	Time (s)	Precision (%)	Recall (%)	F1 Score (%)
SWaT	0.7	No	**88**	37.4	**85.3**	52.0
		Yes	108	**97.7**	81.6	**88.9**
	0.75	No	**57**	37.9	**81.5**	51.7
		Yes	61	**99.4**	66.3	**79.6**
	0.8	No	**22**	44.7	**71.7**	55.1
		Yes	49	**99.7**	63.7	**77.7**
	0.85	No	**6**	61.1	**66.0**	63.4
		Yes	**6**	**99.8**	62.4	**76.7**
WADI	0.7	No	**68**	42.7	**57.9**	49.2
		Yes	84	**93.6**	44.9	**60.7**
	0.75	No	**29**	45.1	**46.7**	45.9
		Yes	39	**96.3**	38.2	54.7
	0.8	No	**21**	54.4	**38.6**	45.2
		Yes	27	**97.4**	32.4	**48.6**
	0.85	No	**15**	60.8	**35.6**	44.9
		Yes	16	**97.5**	30.5	**46.5**

Table 8: With *v.s.* without tolerances with varying support

Dataset	Confidence Threshold	Tolerances	Time (s)	Precision (%)	Recall (%)	F1 Score (%)
SWaT	0.7	No	**112**	19.5	**90.4**	32.0
		Yes	136	**74.7**	83.4	**78.8**
	0.8	No	**102**	25.4	**89.9**	39.6
		Yes	121	**93.4**	82	**87.4**
	0.9	No	**88**	37.4	**85.3**	52.0
		Yes	108	**97.7**	81.6	**88.9**
	1.0	No	**5**	**98.9**	**67.0**	**79.8**
		Yes	8	**98.9**	**67.0**	**79.8**
WADI	0.7	No	**86**	16.3	**80.1**	27.1
		Yes	106	**64.6**	50.2	**56.5**
	0.8	No	**74**	23.5	**63.4**	34.3
		Yes	93	**75.7**	47.8	58.6
	0.9	No	**68**	42.7	**57.9**	49.2
		Yes	84	**93.6**	44.9	**60.7**
	1.0	No	**6**	**95.7**	**39.6**	**56**
		Yes	12	**95.7**	**39.6**	**56**

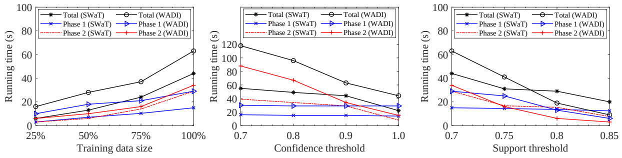
Table 9: With *v.s.* without tolerances with varying confidence

9.4 Extra Tests on Our Global Detection Method

In this set of tests, we further verify the effectiveness of our global anomaly detection method AR on the GDP of SWaT and WADI, which is complementary to Exp-4.

Exp-Ex1. To further analyze rule generation procedure of AR, we evaluate the time of different mining phases and the total time. While mining, we vary the training data size from 25% to 100%, the confidence threshold from 0.7 to 1.0, and the support threshold from 0.7 to 0.85 on the GDP of SWaT and WADI, respectively, and the other parameters are along the same setting as Exp-1. The results are reported in Figures 6, where phase 1 represents generating association rules [6] with algorithm Snow-Touch [50] and phase 2 represents the rest dealing with tolerances in Algorithm 1.

Both the total time for generating non-redundant rules with tolerances and the running time of phase 1 & 2 increase with the increase of training data sizes, where the total time and the running time of phase 2 are more sensitive as shown in Figure 6a. These results are consistent with our time complexity analysis. The total time for generating non-redundant rules with tolerances and the running time of phase 2 drop more quickly with the increase of the thresholds of support and confidence, while the running time of phase 1 is much less sensitive to these two parameters, as shown in Figures 6b & 6c. Also, in most occasions, the running time of phase 2 dominates the total running time for generating non-redundant rules with tolerances. However, the running time for all cases is much faster compared with AT [56].



(a) Vary data sizes **(b) Vary confidence** **(c) Vary support**

Figure 6: Running time of different phases on GDP

Exp-Ex2. To further evaluate the impacts of tolerances, we test our global method AR using association rules with and without tolerances. Different from Exp-4.1, we vary the testing data size from 25% to 100%, the confidence threshold from 0.7 to 1.0, and the support threshold from 0.7 to 0.85 on the GDP of SWaT and WADI, respectively, and the other parameters are along the same setting as Exp-1. The tests are on the GDP of the testing data for detection accuracy and time. The results are reported in Table 7, 8 & 9.

The precision and F1 score of AR are consistently improved significantly with the use of tolerances for all support and confidence thresholds. In Table 7, the average precision and F1 score of AR have been improved 342.6% and 119.6% on SWaT, and 216.4% and 46.0% on WADI, respectively. In Table 8, the average precision and F1 score of AR have been improved 119.0% and 45.3% on SWaT, and 90.0% and 13.7% on WADI, respectively. In Table 9, the average precision and F1 score of AR have been improved 101.2% and 64.7% on SWaT, and 85.0% and 39.1% on WADI, respectively.

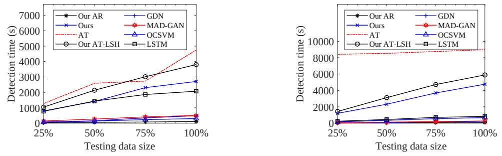
The running time of AR with tolerances is slightly increased as an obvious acceptable cost, with an average of 31.9% and 24.8% increase for all support and confidence thresholds on SWaT and WADI, respectively. Considering that AR is much faster than other methods, and the extra cost of the detection time is acceptable compared with its improvement of the precision and F1 scores.

These further justify the introducing of tolerances for association rules to improve the precision of global anomaly detection.

9.5 Extra Tests on Scalability

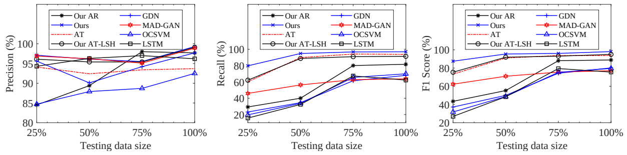
In this set of tests, we further analyze the performance of our unified detection method, AR and AT-LSH along with other baseline methods in Exp-1 when the testing date size varies, which is complementary to Exp-4.1.

Exp-Ex4. To further verify the effectiveness and efficiency of our unified method as well as our global method AR and local method AT-LSH, we compare with the existing methods in Exp-1, by varying the testing data size from 25% to 100%, while the other setting is the same as Exp-1. The results are reported in Figures 7, 8 & 9.



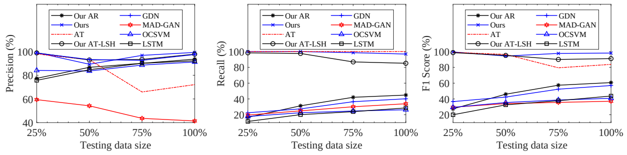
(a) SWaT **(b) WADI**

Figure 7: Detection time with varying data size



(a) Precision **(b) Recall** **(c) F1 Score**

Figure 8: Detection accuracy with varying data size on SWaT



(a) Precision **(b) Recall** **(c) F1 Score**

Figure 9: Detection accuracy with varying data size on WADI

The detection time of all methods increases linearly with the increase of testing data sizes as shown in Figure 7. This is because these methods perform anomaly detection once a data point or a segment of continuous data points. However, our unified approach and AT-LSH increases slower than AT, which demenstrates the scalability of our method for large-scale data. And our unified approach is much faster than the SOTA method AT.

The detection accuracy (especially the F1 score) of our unified approach, AT-LSH and AT is not sensitive to the data sizes as shown in Figures 8 & 9, which justifies the scalability of our unified approach and AT-LSH as our local anomaly detection method. AR performs very well in terms of detection time and accuracy, which justifies our choice of AR as our global detection method.

These further indicate the scalability of our unified approach.

	Local Factor	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
SWaT	Precision (%)	98.3	98.5	99.3	99.3	99.3	99.3	99.3	99.3	99.3	99.3	99.4	99.4	99.3	99.1	99.1
	Recall (%)	89.9	96.0	92.3	92.3	95.1	95.2	95.2	95.2	96.9	96.9	95.6	95.6	95.4	85.8	85.8
	F1 Score (%)	93.9	97.2	95.7	95.7	97.2	97.2	97.2	97.2	98.1	98.1	97.5	97.5	97.3	92.0	92.0
WADI	Precision (%)	97.6	97.6	97.2	97.4	97.4	97.4	97.4	97.4	97.1	97.1	97.1	97.1	97.1	97.4	97.0
	Recall (%)	96.9	96.9	90.1	96.7	90.2	90.2	90.2	90.2	90.7	90.7	90.7	90.7	90.7	96.9	81.9
	F1 Score (%)	97.2	97.2	93.5	97.1	93.7	93.7	93.7	93.7	93.8	93.8	93.8	93.8	93.8	97.1	88.8

Table 10: Our unified approach with varying LF

9.6 Extra Tests on Local factor

Exp-Ex3. To further verify the flexibility of the local factor in our unified approach on SWaT and WADI, we vary the local factor threshold from 0.1 to 0.8 with the other parameters along the same setting as Exp-1. The results are reported in Table 10.

The precision, recall and F1 score of the SOTA method AT are 93.7%, 93.7% and 93.7%, respectively, on SWaT, and are 72.1%, 100% and 83.8%, respectively, on WADI. This means that our unified approach outperforms AT on both datasets even when the local factor falls into a wide range of $[0.15, 0.70]$ as shown in Table 10.