


ITC 4/48 Information Technology and Control Vol. 48 / No. 4 / 2019 pp. 505-521 DOI 10.5755/j01.itc.48.4.22176	WMFP-Outlier: An Efficient Maximal Frequent-Pattern-Based Outlier Detection Approach for Weighted Data Streams	
	Received 2018/12/02	Accepted after revision 2019/09/30
	 http://dx.doi.org/10.5755/j01.itc.48.4.22176	

WMFP-Outlier: An Efficient Maximal Frequent-Pattern-Based Outlier Detection Approach for Weighted Data Streams

Saihua Cai, Qian Li, Sicong Li, Gang Yuan

College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, P.R. China;
e-mails: caisaih@cau.edu.cn, lacus0327@sina.com, lsc@cau.edu.cn, yuangang2009@gmail.com

Ruizhi Sun

College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, P.R. China;
Scientific Research Base for Integrated Technologies of Precision Agriculture (animal husbandry), the Ministry
of Agriculture, Beijing 100083, P.R. China; e-mail: sunruizhi@cau.edu.cn

Corresponding author: sunruizhi@cau.edu.cn

Since outliers are the major factors that affect accuracy in data science, many outlier detection approaches have been proposed for effectively identifying the implicit outliers from static datasets, thereby improving the reliability of the data. In recent years, data streams have been the main form of data, and the data elements in a data stream are not always of equal importance. However, the existing outlier detection approaches do not consider the weight conditions; hence, these methods are not suitable for processing weighted data streams. In addition, the traditional pattern-based outlier detection approaches incur a high time cost in the outlier detection phase. Aiming at overcoming these problems, this paper proposes a two-phase pattern-based outlier detection approach, namely, WMFP-Outlier, for effectively detecting the implicit outliers from a weighted data stream, in which the maximal frequent patterns are used instead of the frequent patterns to accelerate the process of outlier detection. In the process of maximal frequent-pattern mining, the anti-monotonicity property and MFP-array structure are used to accelerate the mining operation. In the process of outlier detection, three deviation indices are designed for measuring the degree of abnormality of each transaction, and the transactions with the highest degrees of abnormality are judged as outliers. Last, several experimental studies are conducted on a synthetic dataset to evaluate the performance of the proposed WMFP-Outlier approach.

The results demonstrate that the accuracy of the WMFP-Outlier approach is higher compared to the existing pattern-based outlier detection approaches, and the time cost of the outlier detection phase of WMFP-Outlier is lower than those of the other four compared pattern-based outlier detection approaches.

KEYWORDS: outlier detection, weighted maximal frequent-pattern mining, weighted data stream, deviation indices, data mining.

1. Introduction

Because data streams [5] can reflect the variational trends of monitored objects in real time, they have been widely used in practice; for example, agricultural sensors are used to monitor the growth status of greenhouse plants, and weather sensors are used to monitor changes in temperature, humidity, and wind. From data streams, users can accurately assess the current state of the monitored objects to facilitate decision-making; thus, data streams are critical in manufacturing, agriculture and industry. However, outliers [2, 19] are often generated along with data streams and adversely affect the accuracy of the collected data, which further affects the accuracy of data-based decision-making.

In the past ten years, many outlier detection approaches have been proposed for effectively identifying implicit outliers. These approaches can be divided into distance-based outlier detection approaches [1, 10, 12], density-based outlier detection approaches [11, 13, 16] and pattern-based outlier detection approaches [3, 4, 8, 9, 17]. For distance-based and density-based outlier detection approaches, because the distance of each data element from the selected data must be calculated, outlier detection is inefficient for large datasets. In addition, distance-based outlier detection approaches and density-based outlier detection approaches only consider the degree of abnormality of each transaction and not the frequency of occurrence of each pattern; hence, the outliers that are detected via these methods cannot coincide with the outliers that are defined by Hawkins [7]. For pattern-based outlier detection approaches, the frequency of occurrence of each pattern is also considered a major factor that affects the degree of abnormality of the transactions. Because the outliers that are detected via pattern-based outlier detection approaches are closer to the real outliers, pattern-based outlier detection approaches have been studied extensively in recent years. In addition, the frequent-pattern-based outlier detection approaches [8, 17],

maximal frequent-pattern-based outlier detection approaches [3, 4] and infrequent-pattern-based outlier detection approaches [9] gradually form a complete system. Pattern-based outlier detection approaches can be utilized in the following three main steps: (1) mining the frequent patterns or infrequent patterns in the datasets; (2) designing the deviation indices for each transaction; and (3) identifying the implicit outliers in the datasets.

In previous work [1, 6, 9], each data element was considered equally important in pattern-based outlier detection approaches; however, this is not always established in practice. For example, the prices of apples and oranges in a shopping cart are not the same. Therefore, we must consider the price information (also called the *weight*) to make the results of outlier detection conform with the application scenario. However, to the best of our knowledge, no pattern-based outlier detection approach has been proposed until now for detecting the implicit outliers from the data stream by considering the weight conditions. In addition, the scale of the frequent patterns is very large for a small *min_sup* value (minimal *support* threshold), which substantially increases the time cost of the outlier detection phase.

Based on the problems that are discussed above, we propose a weighted maximal frequent-pattern-based outlier detection approach for detecting the implicit outliers in a weighted data stream. The major contributions of this paper can be summarized as follows:

- 1 We design a WFP-Tree structure for storing detailed pattern information and *weight* information and suggest using a two-array structure, namely, a WFP-array, to store the *support* value of each “weighted frequent” 2-pattern.
- 2 We use the weighted maximal frequent patterns to detect the implicit outliers rather than the weighted frequent patterns, to decrease the time cost of the outlier detection phase.

- 3 We design three deviation indices, namely, the weighted maximal frequent pattern deviation index (*WMFPDI*), the weighted infrequent 1-pattern deviation index (*WIPDI*) and the final transaction deviation index (*FTDI*), for measuring the degree of abnormality of the transactions in the sliding window.
- 4 Based on the mined weighted maximal frequent patterns and the defined deviation indices, we propose a **W**eighted **M**aximal **F**requent **P**attern-based **O**utlier detection approach, namely, WMFP-Outlier, for effectively detecting the implicit outliers from a weighted data stream.
- 5 We conduct extensive experiments on synthetic datasets to evaluate the validity and accuracy of the proposed WMFP-Outlier approach.

The remainder of this paper is organized as follows: Section 2 reviews related work on outlier detection approaches. Section 3 introduces preliminaries that are related to this paper and presents the weighted maximal frequent-pattern-based outlier detection framework, the weighted maximal frequent-pattern mining approach and the outlier detection approach. In Section 4, the empirical studies and experimental analysis are discussed. In Section 5, we present the conclusions of the study and discuss directions for future work.

2. Related Work

In the past two decades, the research on outlier detection methods was mainly focused on the following types of approaches: distance-based outlier detection approaches, density-based outlier detection approaches and pattern-based outlier detection approaches.

For distance-based outlier detection, Shaikh and Kitagawa [12] proposed an efficient cell-based outlier detection approach for quickly identifying the implicit outliers based on the Gaussian distribution. Then, they proposed an approximate cell-based outlier detection approach that uses a bounded Gaussian distribution to further improve the efficiency of outlier detection. Aiming at overcoming the problems that are encountered with unsupervised outlier detection methods in high-dimensional

space, Radovanović et al. [10] put forward a unified viewpoint of reverse nearest-neighbor counts and explored the relationship between Hubness and data sparsity. Based on this analysis, they proposed an efficient AntiHub approach for conducting unsupervised outlier detection. Aiming at reducing the high time cost of the traditional distance-based outlier detection approaches, Angiulli et al. [1] proposed families of distance-based parallel and distributed outlier detection methods, namely, BruteForce and SolvingSet, for detecting the implicit outliers using GPU.

For density-based outlier detection, Salehi et al. [11] proposed an efficient memory incremental local outlier detection method, namely, MiLOF, which could identify the outliers from a data stream with limited memory usage. In addition, they proposed an extended version, namely, MiLOF_F, for reducing the scale of the summaries in the memory. Zhang et al. [16] proposed an adaptive kernel density-based outlier detection method, namely, Adaptive-KD, for detecting implicit outliers in nonlinear systems, in which the kernel width parameter could be adaptively set according to the average distance, thereby improving the discrimination ability of the outlierness measure. Tang and He [13] introduced the relative density-based outlier score (RDOS) for measuring the local outlierness of an object and proposed an efficient outlier detection approach that was based on local kernel density estimation (KDE). In local KDE, the k nearest neighbors, reverse nearest neighbors and shared nearest neighbors were used to improve the outlier detection accuracy.

For pattern-based outlier detection, the reasons why various subspaces caused abnormalities in the overall spaces are considered; thus, the detected outliers can well satisfy the definition of outliers. He et al. [8] proposed a frequent-pattern-based method, namely, FindFPOF, for identifying the implicit outliers from static datasets, in which the ratio of the *support* value of the contained frequent patterns to the total number of mined frequent patterns was used as the final outlier criterion. However, this simple judging criterion can be changed to further improve the outlier detection performance. In addition, the time cost of the outlier detection phase of the FindFPOF method was relatively high because the scale of the frequent patterns was very large. Aiming at overcom-

ing the sub-optimal efficiency of outlier detection via the FindFPOF method, an improved frequent-pattern-based outlier detection approach, namely, LFP [17], was proposed, which used the ratio of the longest length among the frequent patterns that were contained in the transaction to the length of transaction as the outlier criterion and yielded satisfactory results on multiple datasets. With the objective of further reducing the time cost in the outlier detection phase of the FindFPOF method, the OODFP method [4] and IM_Sunday method [3] were proposed for identifying the implicit outliers, where the maximal frequent patterns were used as the basis for outlier detection to reduce the time cost. To make the identified outliers more consistent with the definition of outliers, Hemalatha et al. [9] proposed a minimal infrequent-pattern-based method, namely, MIFPOF, for detecting the outliers in a data stream. In the outlier detection phase, they provided three abnormality factors for more accurately determining the deviation degrees of the detected transactions: the transaction weighting factor (TWF), the minimal infrequent deviation factor (MIPDF) and the minimal infrequent-pattern-based outlier factor (MIFPOF). However, to the best of our knowledge, no maximal frequent-pattern-based outlier detection approaches have been proposed for identifying the outliers from a weighted data stream until now.

3. The Weighted Maximal Frequent-Pattern-Based Outlier Detection Approach (WMFP-Outlier)

This section presents related preliminaries and definitions and introduces the overall framework of the proposed method. Then, the weighted maximal frequent-pattern mining approach and outlier detection approach for a weighted data stream are introduced in detail using an example.

3.1. Preliminaries and Definitions

Pattern $P=\{p_1, p_2, \dots, p_n\}$ is a set of items. Each transaction $T=\{P_1, P_2, \dots, P_n\}$ consists of a set of patterns and each transaction is identified by a unique id, namely, TID . Data stream $DS=[T_1, T_2, \dots, T_n]$ consists of continuous transactions. For patterns $P_a=\{p_1, p_2, \dots, p_m\}$ and

$P_b=\{p_1, p_2, \dots, p_k\}$, if $m < k$, P_a is called a sub-pattern of P_b and P_b is called a super-pattern of P_a ; P_b is also called a k -pattern. In recent research, the sliding window (SW) model has been used to effectively process the data stream. The SW model can only process the most recent transactions, that is, the sliding window must move back immediately when a new transaction enters the SW . The size of the sliding window is denoted as $|SW|$. To mine the weighted frequent patterns, a user-defined minimal *weighted support* threshold, namely, min_wsup , is used to determine whether the mined patterns are weighted frequent.

Then, an example of a weighted data stream, which is described in Table 1, is used to accurately interpret the definitions, where the value of min_wsup is 2.0 and $|SW|$ is 6.

Table 1

An example of a weighted data stream

<i>TID</i>	<i>Transaction</i>	<i>TID</i>	<i>Transaction</i>
T_1	{B, D, E}	T_2	{A, B, C, D, F}
T_3	{A, B, E, F}	T_4	{B, D, E, F}
T_5	{A, B, C, D}	T_6	{A, B, D, E}
...
<i>Pattern</i>	<i>Weight</i>	<i>Pattern</i>	<i>Weight</i>
A	0.7	B	0.8
C	0.9	D	0.6
E	0.4	F	0.5

Definition 1. Support: The sum of the *count* values of pattern $\{X\}$ in the current sliding window is denoted as $support(X)$, which is defined as

$$support(X) = \sum_{j=1}^{|SW|} count(X, T_j).$$

Example 1. Pattern $\{A\}$ is present in T_2, T_3, T_5 and T_6 ; thus, $support(A)=4$. Similarly, pattern $\{AB\}$ is present in T_2, T_3, T_5 and T_6 ; thus, $support(AB)=4$.

Definition 2. Weight: The *weight* of pattern $\{X\}$ is denoted as $weight(X)$ and defined as

$$weight(X) = \frac{\sum_{p_i \in X} weight(p_i)}{|M|}.$$

$|M|$ denotes the number of 1-patterns $\{p_j\}$ in pattern $\{X\}$.

Example 2. The *weight* of pattern $\{A\}$ is $weight(A)=0.7/1=0.7$ and the *weight* of pattern $\{AB\}$ is $weight(AB)=(0.7+0.8)/2=0.75$.

Definition 3. Weighted support (*wsup*): The *weighted support* of pattern $\{X\}$ in the current sliding window is denoted as $wsup(X)$ and is defined as

$$wsup(X) = weight(X) * support(X).$$

Example 3. The *weighted support* of pattern $\{A\}$ is $wsup(A)=weight(A)*support(A)=0.7*4=2.8$ and the *weighted support* of pattern $\{AB\}$ is $wsup(AB)=weight(AB)*support(AB)=0.75*4=3.0$.

Definition 4. Weighted Frequent Pattern (WFP): If the *weighted support* of pattern $\{X\}$ is not less than the predefined min_wsup , i.e., $wsup(X) \geq min_wsup$, pattern $\{X\}$ is a weighted frequent pattern.

Example 4. The *weighted support* of pattern $\{A\}$ is $wsup(A)=2.8 > 2.0$; thus, pattern $\{A\}$ is a weighted frequent pattern.

Definition 5. Weighted infrequent Pattern (WiFP): If the *weighted support* of pattern $\{X\}$ is less than the predefined min_wsup , i.e., $wsup(X) < min_wsup$, pattern $\{X\}$ is a weighted infrequent pattern.

Example 5. The *weighted support* of pattern $\{C\}$ is $wsup(C)=weight(C)*support(C)=0.9*2=1.8 < 2.0$; thus, pattern $\{C\}$ is a weighted infrequent pattern.

Definition 6. Weighted Maximal Frequent Pattern (WMFP): If the *weighted support* of pattern $\{X\}$ is not less than the predefined min_wsup and no super-pattern of $\{X\}$ is weighted frequent, pattern $\{X\}$ is a weighted maximal frequent pattern.

Example 6. The *weighted support* of pattern $\{ABD\}$ is $wsup(ABD)=2.1 > 2.0$ and the *weighted support* of its super-patterns is less than 2.0; thus, pattern $\{ABD\}$ is a weighted maximal frequent pattern.

The *weighted support* of pattern $\{E\}$ is $wsup(E)=4*0.4=1.6 < 2.0$; thus, pattern $\{E\}$ is a weighted infrequent pattern. However, the *weighted support* of pattern $\{BE\}$ is $wsup(BE)=4*0.6=2.4 > 2.0$; hence, pattern $\{BE\}$ is a weighted frequent pattern. Therefore, the well-known anti-monotonicity property [3] that used to reduce the scale of potential extensible patterns is not applicable in the weighted frequent-pattern mining process.

3.2. The Overall Framework of the WMFP-Outlier Approach

According to the existing pattern-based outlier detection approaches (such as FindFPOF, LFP and OODFP), the complete detection process can be divided into the following two phases: (1) the pattern mining stage and (2) the outlier detection stage. In contrast to the previous approaches, the mining objects in the pattern mining stage of the WMFP-Outlier approach are the weighted maximal frequent patterns; thus, the overall framework of WMFP-Outlier is divided into the weighted maximal frequent-pattern mining phase and the WMFP-based outlier detection phase.

In the weighted maximal frequent-pattern mining phase, detailed valid data information on each transaction is stored into the WFP-Tree structure and the *maximal weight* and pruning strategy are used prior to the extension operation to increase the efficiency of the mining process. In the outlier detection stage, several deviation indices are defined for measuring the degrees of abnormality of the transactions in the sliding window based on the mined WMFPs. Then, the transactions are sorted in ascending order of the calculated deviation degree and the k transactions that have the lowest deviation degrees are identified as outliers. After detecting the transaction in the current sliding window, the sliding window is moved backwards to identify the outliers in the new incoming weighted data stream.

3.3. The Weighted Maximal Frequent-Pattern Mining Approach

In this subsection, we introduce the weighted maximal frequent-pattern mining approach, namely, WMFPM-WDS, which is the basis of outlier detection. The main steps of WMFPM-WDS are to use a WFP-Tree structure to store the detailed information of the new incoming weighted data stream and to use the *maximal weight* and pruning strategy to reduce the potential scale of the extended patterns.

3.3.1. The Main Strategy of WMFPM-WDS

In the data stream environment, the related operations must be processed in a short time. Because the FP-Growth method is more efficient in data mining, it is used as the main strategy in this paper.

In contrast to traditional data streams, the super-patterns of an infrequent pattern have a non-zero probability of being frequent in the weighted data stream (pattern $\{BE\}$ is frequent but pattern $\{E\}$ is infrequent in this example). Thus, the well-known anti-monotonicity property [3, 9] that is used to increase the mining efficiency is not suitable for weighted data streams. To overcome this limitation, the primary task of weighted maximal frequent-pattern mining is to design an efficient strategy for enabling the anti-monotonicity property to be used also in weighted data streams, which is highly important for reducing the number of meaningless extension operations. In the process of weighted frequent-pattern mining, the concept of *maximal weight* [14] is used to ensure that the anti-monotonicity property can continue to be applied. However, in previous studies, the *maximal weight* was only used in the process of extending the frequent 1-patterns to 2-patterns. If it is used in each extension process, the efficiency of pattern mining can be further improved. In addition, if the *maximal weight* provides a much tighter bound during the mining process, the efficiency of weighted maximal frequent-pattern mining will be further improved. Based on these two assumptions, we propose an efficient weighted maximal frequent-pattern mining approach, namely, WMFPM-WDS, for effectively mining the WMFPs from a weighted data stream.

Definition 7. Maximal weight (*maxweight*): For a set of *weight* values ($weight = \{weight_1, weight_2, \dots, weight_n\}$) that belong to the weighted data stream, the maximal *weight* value of the valid patterns is called the *maximal weight*.

Example 7. In the example that is presented in Table 1, the *weight* values of the patterns are $\{weight(A)=0.7, weight(B)=0.8, weight(C)=0.9, weight(D)=0.6, weight(E)=0.4, weight(F)=0.5\}$; thus, in this example, the *maximal weight* is $maxweight=0.9$.

Definition 8. Maximal weighted support (*mwsup*): For a pattern $\{X\}$, the *maximal weighted support* of $\{X\}$ is the product of the *support* value and the *maxweight* value; it is defined as

$$mwsup(X) = maxweight(X) * support(X).$$

Example 8. In the example that is presented in Table 1, the *maximal weighted support* of pattern $\{A\}$ is $mwsup(A)=0.9*4=3.6$.

Definition 9. Safe weighted infrequent pattern (SWiFP): For a pattern $\{X\}$, if the *maximal weighted support* of $\{X\}$ is less than the predefined min_wsup , i.e., $mwsup(X) < min_wsup$, pattern $\{X\}$ is a safe weighted infrequent pattern.

Example 9. In the example that is presented in Table 1, the *maximal weighted support* of pattern $\{C\}$ is $mwsup(C)=0.9*2=1.8 < 2.0$; thus, pattern $\{C\}$ is a safe weighted infrequent pattern.

Theorem 1. Anti-monotonicity property: No super-pattern of SWiFP can be weighted frequent.

Proof. Suppose pattern $\{X^k\}$ is a safe weighted infrequent k -pattern and pattern $\{X^{k+1}\}$ is the super-pattern of $\{X^k\}$. Since $\{X^{k+1}\}$ is a super-pattern of $\{X^k\}$, $support(X^{k+1}) \leq support(X^k)$ and $maxweight(X^{k+1}) \leq maxweight(X^k)$. Because $\{X^k\}$ is a safe weighted infrequent pattern, $mwsup(X^k) = maxweight(X^k) * support(X^k) < min_wsup$. It follows that

$$\begin{aligned} wsup(X^{k+1}) &= weight(X^{k+1}) * support(X^{k+1}) \\ &\leq maxweight(X^{k+1}) * support(X^{k+1}) \\ &\leq maxweight(X^k) * support(X^k) \\ &= mwsup(X^k) \\ &< min_wsup. \end{aligned}$$

Therefore, the super-pattern of the safe weighted infrequent pattern is also weighted infrequent and this property is established.

Lemma 1. The use of *maximal weighted support* will not cause any loss of WFPs.

Proof. The *weighted support* value of pattern $\{X^k\}$ is

$$\begin{aligned} wsup(X^k) &= weight(X^k) * support(X^k) \\ &\leq maxweight(X^k) * support(X^k) \\ &= mwsup(X^k). \end{aligned}$$

That is, if the *maximal weighted support* value of pattern $\{X^k\}$ is less than min_wsup , $\{X^k\}$ is not a weighted infrequent pattern. Therefore, pruning the patterns based on *maximal weighted support* will not cause the loss of WFPs.

When the data stream enters the sliding window, the data are scanned for the first time to adjust the insertion order (decreasing *support*) and discard the safe weighted infrequent 1-patterns (they are saved into the **W**eighted **i**n**F**requent **P**attern **L**ibrary

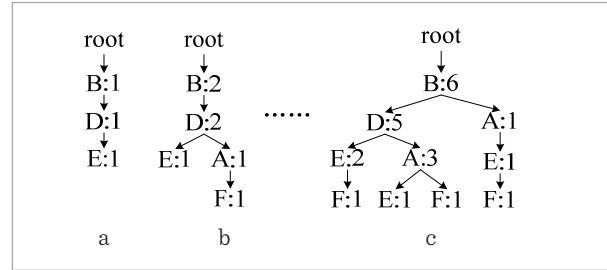
(WiFPL)) to reduce the scale of the constructed WFP-Tree structure. When the data stream is scanned for the second time, the “weighted frequent” 1-patterns (the 1-patterns for which the *maximal weighted support* is not less than *min_wsup*, which are saved into **W**eighted **F**requent **P**attern **L**ibrary (WFPL)) are inserted into the WFP-Tree.

The process is described as follows: The *maximal weight* value of the patterns is 0.9. Since $mwsup(C)=0.9*2=1.8<2.0$, they should be discarded and saved into WiFPL. Then, the *maximal weight* is changed to 0.8. For the maximal weighted frequent patterns, namely, $\{A, B, D, E, F\}$, $support(A)=4$, $support(B)=6$, $support(D)=5$, $support(E)=4$, and $support(F)=3$; thus, the insertion order is adjusted to $B \rightarrow D \rightarrow A \rightarrow E \rightarrow F$. When scanning transaction T_1 , patterns $\{B\}$, $\{D\}$ and $\{E\}$ are inserted into the first branch of the WFP-Tree, as shown in Figure 1(a). When scanning transaction T_2 , patterns $\{B\}$, $\{D\}$, $\{A\}$ and $\{F\}$ are inserted into the second branch of the WFP-Tree, as shown in Figure 1(b). The other four branches of the WFP-Tree are constructed as discussed above and the final WFP-Tree structure is illustrated in Figure 1(c).

After the patterns of each transaction have been stored in the constructed WFP-Tree structure, the pattern mining operation is conducted to effectively mine the *WMFPs* from the weighted data stream. In the process of maximal weighted frequent-pattern mining, the general strategy of the FP-Growth approaches is to construct the conditional FP-Tree for recursively mining the frequent patterns. However, the construction of the conditional FP-Tree is time-consuming because this operation requires the global FP-Tree to be scanned several times to obtain the pattern’s *support* value. Inspired by [14], we can construct the global WFP-array structure for storing the *support* value for each 2-pattern to improve the mining efficiency, where the patterns that do not need to be extended are easy to identify from the WFP-array. The WFP-array structure consists of an $(n-1)*(n-1)$ two-dimensional matrix, where n is the number of “weighted frequent” 1-patterns. The patterns in the WFP-array are stored in decreasing order of *support*, namely, the rows of the WFP-array are the first $(n-1)$ “weighted frequent” 1-patterns and the columns of the WFP-array are the last $(n-1)$ “weighted frequent” 1-patterns. To illustrate the WFP-array

structure, we use the example that is presented in Table 1 to demonstrate the process of constructing the WFP-array structure.

Figure 1
The construction process of the WFP-Tree structure



Example 10. The “weighted frequent” 1-patterns are $\{B\}$, $\{D\}$, $\{A\}$, $\{E\}$ and $\{F\}$; therefore, the patterns in the rows of the WFP-array structure are $\{B\}$, $\{D\}$, $\{A\}$ and $\{E\}$ and the patterns in the columns of the WFP-array structure are $\{D\}$, $\{A\}$, $\{E\}$ and $\{F\}$. In the WFP-array structure, the specified number represents the *support* value of each 2-pattern. For pattern $\{F\}$, according to scans of the WFP-Tree, the number of occurrences of $\{FE\}$ is 2; thus, the corresponding position is specified as 2. The final constructed global WFP-array structure is illustrated in Figure 2.

Figure 2
Global WFP-array structure

D	5			
A	4	3		
E	4	3	2	
F	3	2	2	2
	B	D	A	E
	WFP-array			

Based on the global WFP-array structure, the *support* value of each 2-pattern can be easily determined. However, not all these 2-patterns must be further extended; thus, the next operation is to determine which 2-patterns do not need to be extended. According to Definition 9, if the 2-patterns are SWiFPs, then the “pattern extension” is meaningless. A SWiFP is judged according to the magnitude between the *maximal weighted support* value of the pattern and the

predefined min_wsup ; the *maximal weighted support* value of the pattern is calculated as the product of the *support* value and the *maximal weight* value. Thus, if we obtain the *maximal weight* value and the min_wsup value, we can calculate the critical *support* (which is denoted as cs) value easily. If the real *support* value of the pattern (in the global WFP-array) is less than the cs value, this pattern need not be further extended.

Example 11. For the patterns that are shown in Figure 2, the two largest valid *weight* values are 0.8 and 0.7; hence, the *maximal weight* value is 0.75 $(=(0.8+0.7)/2)$ and the cs value of the 2-patterns is 2.67 $(=2.0/0.75)$. According to a search of the constructed global WFP-array, the *support* values of patterns $\{DF\}$, $\{AF\}$, $\{EF\}$ and $\{AE\}$ are less than 2.67; thus, they are not “weighted frequent” 2-patterns and should not be included in the subsequent extension process.

Then, the conditional WFP-array structure is constructed for mining the “weighted frequent” 3-patterns. If the number of “weighted frequent” 2-patterns that are prefixed by pattern $\{X\}$ is not less than 2, the conditional WFP-array of pattern $\{X\}$ must be constructed. For the 3-patterns, the three largest valid *weight* values are used to calculate the *maximal weight* value and the cs value is calculated to determine whether the 3-patterns must be further extended. When the number of 3-patterns that are prefixed by pattern $\{XY\}$ is not less than 2, the conditional WFP-array of pattern $\{XY\}$ must be constructed for mining the longer “weighted frequent” patterns. These operations are performed recursively to mine the “weighted frequent” patterns until they cannot be further extended.

When the “weighted frequent” patterns are mined, the real *weighted support* values of the mined longest “weighted frequent” patterns (suppose this pattern is a k -pattern) having different prefixes are calculated to identify the true weighted maximal frequent patterns. If the real *weighted support* value of the k -pattern is less than min_wsup , the real *weighted support* values of the $(k-1)$ -patterns are calculated and compared with min_wsup . These operations are performed recursively if the real *weighted support* value is less than the min_wsup . Otherwise, the k -pattern is the weighted maximal frequent pattern. The mined weighted maximal frequent patterns are stored in the **W**eighted **M**aximal **F**requent **P**attern **L**ibrary (WMFPL). Last, it is checked whether the

mined weighted maximal frequent patterns are global weighted maximal frequent and the global weighted maximal frequent patterns are mined as the final WMFPLs. Detailed pseudo-code for the WMFPM-WDS algorithm is presented as Algorithm 1.

Algorithm 1: WMFPM-WDS

Input: Weighted data stream, min_wsup

Output: WMFPLs

```

01. WFPL= $\Phi$ , WiFPL= $\Phi$ 
02. scan the weighted data stream
03. search for maxweight value
04. foreach 1-pattern  $\{p_i\}$  do
05.   if  $mwsup(p_i) < min\_wsup$  then
06.      $\{p_i\} \rightarrow WiPL$ 
07.   else
08.      $\{p_i\} \rightarrow WFPL$ 
09.   end if
10. end for
11. construct the WFP-Tree structure for the 1-patterns in WFPL
12. construct the global WFP-array structure
13.  $k=2$ 
14. foreach “weighted frequent”  $k$ -pattern  $\{p_1, \dots, p_k\}$  do
15.   if number of “weighted frequent”  $k$ -patterns that are prefixed by  $(p_1, \dots, p_k) \geq 2$  then
16.     construct the conditional WFP-array
17.      $k++$ 
18.     go to 15
19.   else
20.     calculate  $mwsup(p_1, \dots, p_k)$ 
21.     if  $mwsup(p_1, \dots, p_k) \geq min\_wsup$  then
22.        $\{p_1, \dots, p_k\} \rightarrow WFPL$ 
23.     end if
24.   end if
25. end for
26. for  $m$ -pattern  $\{p_1, \dots, p_m\}$  in WFPL do
27.   if  $wsup(p_1, \dots, p_m) < min\_wsup$  then
28.      $m--$ 
29.     go to 27
30.   else
31.      $\{p_1, \dots, p_m\} \rightarrow WMFPL$ 
32.   end if
33. end for
34. for the patterns in WMFPL do
35.   delete the sub-patterns
36. end for
37. return WMFPLs

```

3.3.2. An Example of WMFPM-WDS Approach

In this subsection, we use the example that is presented in Table 1 to explain the WMFPM-WDS approach in detail; the min_wsup value is set to 2.0.

Step 1. Calculate the *maximal weighted support* value for each 1-pattern to identify the safe weighted infrequent 1-patterns. In this example, the *maximal weight* is 0.9 (pattern $\{C\}$) initially, which is used to calculate the *maximal weighted support* value.

For 1-pattern $\{C\}$, $mwsup(C)=0.9*2=1.8<2.0$; thus, it is a safe weighted infrequent 1-pattern and it should be saved into WiFPL and excluded from the subsequent extension process to avoid incurring non-productive time cost. Then, the *maximal weight* is changed to 0.8 (pattern $\{B\}$).

For 1-pattern $\{B\}$, $mwsup(B)=0.9*6=5.4>2.0$; hence, it is a “weighted frequent” 1-pattern.

For 1-pattern $\{A\}$, $mwsup(A)=0.9*4=3.6>2.0$; hence, it is a “weighted frequent” 1-pattern.

For 1-pattern $\{D\}$, $mwsup(D)=0.9*5=4.5>2.0$; hence, it is a “weighted frequent” 1-pattern.

For 1-pattern $\{E\}$, $mwsup(E)=0.9*4=3.6>2.0$; hence, it is a “weighted frequent” 1-pattern.

For 1-pattern $\{F\}$, $mwsup(F)=0.9*3=2.7>2.0$; hence, it is a “weighted frequent” 1-pattern.

Step 2. Insert the “weighted frequent” 1-patterns into the WFP-Tree using decreasing *support* values; the process is illustrated in detail in Figure 1.

Step 3. Construct the global WFP-array structure for storing the *support* value of the “weighted frequent” 2-patterns; the result is presented in detail in Figure 2.

Step 4. Search for the two largest valid *weight* values and calculate the *cs* value for excluding the invalid patterns. The result is presented in detail in Example 11.

Step 5. Construct the conditional WFP-array structure for mining the longer “weighted frequent” patterns. (1) For prefix $\{F\}$, the valid “weighted frequent” 2-pattern is only $\{BF\}$; thus, it is not necessary to construct the conditional WFP-array for pattern $\{F\}$. Therefore, the “weighted frequent” pattern that is prefixed by pattern $\{F\}$ is $\{BF\}$. (2) For prefix $\{E\}$, the valid “weighted frequent” 2-patterns are $\{BE\}$ and $\{DE\}$ and it can be further extended into 3-pattern $\{BDE\}$. For pattern $\{BDE\}$, $support(BDE)=3$ and *cs* is 2.86 ($=2.0/0.7$); hence, it is a valid “weighted frequent” 3-pattern. Therefore, the “weighted frequent” pattern that is prefixed by pattern $\{E\}$ is $\{BDE\}$. (3) For prefix $\{A\}$, the valid “weighted frequent” 2-patterns are $\{BA\}$ and $\{DA\}$ and it can be further extended into 3-pattern $\{BDA\}$. For pattern

$\{BDA\}$, $support(BDA)=3$ and *cs* is 2.86 ($=2.0/0.7$); hence, it is a valid “weighted frequent” 3-pattern. Therefore, the “weighted frequent” pattern that is prefixed by pattern $\{A\}$ is $\{BDA\}$. (4) For prefix $\{D\}$, the valid “weighted frequent” 2-pattern is only $\{BD\}$; thus, it is not necessary to construct the conditional WFP-array for pattern $\{D\}$. Therefore, the “weighted frequent” pattern that is prefixed by pattern $\{D\}$ is $\{BD\}$.

Step 6. Search for the true weighted maximal frequent patterns. (1) For prefix $\{F\}$, the “weighted frequent” pattern is $\{BF\}$ and $wsup(BF)=0.65*3=1.95<2.0$; hence, it is not a weighted frequent pattern. Then, sub-pattern $\{F\}$ is used to determine whether it is weighted frequent. Since $wsup(F)=0.5*3=1.5<2.0$, it is also not a weighted frequent pattern. (2) For prefix $\{E\}$, the “weighted frequent” pattern is $\{BDE\}$ and $wsup(BDE)=0.6*3=1.8<2.0$; hence, it is not a weighted frequent pattern. Then, sub-patterns $\{BE\}$ and $\{DE\}$ are used to determine whether it is weighted frequent. Since $wsup(BE)=0.7*4=2.8>2.0$ and $wsup(DE)=0.5*3=1.5<2.0$, the weighted maximal frequent pattern that is prefixed by pattern $\{E\}$ is $\{BE\}$. (3) For prefix $\{A\}$, the “weighted frequent” pattern is $\{BDA\}$ and $wsup(BDA)=0.7*3=2.1>2.0$; hence, it is a weighted frequent pattern. Therefore, the weighted maximal frequent pattern that is prefixed by pattern $\{A\}$ is $\{BDA\}$. (4) For prefix $\{D\}$, the “weighted frequent” pattern is $\{BD\}$ and $wsup(BD)=0.7*5=3.5>2.0$; hence, it is a weighted frequent pattern. Therefore, the weighted maximal frequent pattern that is prefixed by pattern $\{D\}$ is $\{BD\}$.

Step 7. Identify the global weighted maximal frequent patterns. Among the weighted maximal frequent patterns, namely, $\{BE\}$, $\{BDA\}$ and $\{BD\}$, pattern $\{BD\}$ is the sub-pattern of $\{BDA\}$ and must be discarded. Finally, the WMFPs in this example are $\{BE\}$ and $\{BDA\}$.

3.4. Outlier Detection Approach

The main objective of the outlier detection phase is to mine the implicit outliers from the weighted data stream according to the deviation degree of each transaction, where the deviation degree is measured by the deviation indices of the mined WMFPs. Therefore, the design of the deviation indices is critical for outlier detection.

3.4.1. Design of the Deviation Indices

According to Hawkins, the main feature of an outlier [7] is that it differs from most other observations. If most patterns that are contained in a transaction occur frequently, the transaction is less likely to be an outlier. Thus, the mined *WMFPs* can be used to measure the degree of abnormality of the detected transactions. The following factors require our attention.

First, longer weighted maximal frequent patterns contain more weighted frequent patterns (the number of the contained weighted frequent patterns of a *k*-*WMFP* is close to 2^k). Therefore, the length of the mined *WMFPs* is a critical factor that affects the outlier judgment. Second, the *weighted support* of each pattern is used to determine whether the pattern is weighted frequent and if a pattern has a large *weighted support* value, this pattern is more likely to be a weighted frequent pattern; thus, the *weighted support* of the *WMFPs* can be used to measure the degree of abnormality of the transactions. Third, if a transaction contains many weighted infrequent 1-patterns, this pattern is more likely to be an outlier because more weighted infrequent patterns can be extended by weighted infrequent 1-patterns; hence, the number of contained weighted infrequent 1-patterns is also a major factor that affects the outlier detection accuracy. Fourth, for transactions T_1 and T_2 , which are of different lengths but containing the same numbers of *WMFPs* and *WiFPs*, the deviation degree between T_1 and T_2 depends on the abnormality density. Therefore, the length of each transaction is also an important factor.

Based on the above analysis, we have roughly identified the factors that affect the deviation degree of each transaction. Thus, we design three deviation indices for measuring the degree of abnormality of each transaction.

Definition 10. Weighted Maximal Frequent Pattern Deviation Index (WMFPDI): For each weighted maximal frequent pattern $\{X\}$, the length of $\{X\}$ is $len(X)$ and the *weighted support* value of $\{X\}$ is $wsup(X)$. Then, *WMFPDI* is defined as

$$WMFPDI(X) = 2^{len(X)} * (wsup(X) - min_wsup). \quad (1)$$

Definition 11. Weighted Infrequent 1-Pattern Deviation Index (WIPDI): For each transaction T_i in the sliding window, the number of contained

weighted infrequent 1-patterns $\{Y\}$ is $M(Y)$. Then, *WIPDI* is defined as

$$WIPDI(T_i) = M(Y) + 1. \quad (2)$$

Definition 12. Final Transaction Deviation Index (FTDI): The length of each transaction T_i in the sliding window is $len(T_i)$ and the number of contained *WMFPs* $\{X\}$ is $N(X)$. Then, *FTDI* is defined as

$$FTDI(T_i) = \sum_{X \in T_i, X \subseteq WMFPL} WMFPDI(X) + \frac{len(T_i) * (N(X) + 1)}{WIPDI(T_i)}. \quad (3)$$

3.4.2. WMFP-Outlier Algorithm

After designing the deviation indices for the weighted maximal frequent patterns, the outliers (abnormal transactions) in current sliding window are detected based on the calculated *FTDI* value. The transactions are sorted in increasing order of *FTDI* value and the transactions that correspond to the lowest *FTDI* values are identified as the outliers, where the value of *k* is specified by the user. The implementation of the proposed WMFP-Outlier method is presented as Algorithm 2.

Algorithm 2: WMFP-Outlier

Input: Weighted data stream, min_wsup , k

Output: Outliers

```

01. call Algorithm 1 // mine WMFPs
02. WMFPDI(X)=0, WIPDI(X)=0, FTDI(Ti)=0
03. foreach {X} in WFPL do
04. WMFPDI(X)=2len(X)*(wsup(X)-min_wsup)
05. end for
06. for i∈[1,|SW|] do
07. foreach weighted infrequent 1-pattern {Y}⊆Ti do
08. WIPDI(Ti)=M(Y)+1
09. end for
10. for {X}⊆Ti do
11. FTDI(Ti) = ∑X∈Ti, X⊆WMFPL WMFPDI(X) +  $\frac{len(T_i) * (N(X) + 1)}{WIPDI(T_i)}$ 
12. end for
13. end for
14. sort the transactions in increasing order of FTDI values
15. Outliers←top k Ti // k is assigned by the user

```

3.4.3. An Example of WMFP-Outlier Approach

To demonstrate the proposed WMFP-Outlier method, we consider the example that is presented in Table 1 and set the min_wsup value to 2.0. The

mined WMFPs are $\{BE\}$ and $\{BDA\}$ and the weighted infrequent 1-patterns (*1-WiFPs*) are $\{C\}$, $\{E\}$ and $\{F\}$. The calculated results for the WMFPDI value, WIPDI value and FTDI value are listed in Table 2.

After the calculation of the FTDI values, the transactions are sorted in increasing order of FTDI value. Thus, the transactions in decreasing order of probability of being an outlier are T_5, T_2, T_3, T_4, T_1 and T_6 .

Table 2

The results of the WMFP-Outlier approach

TID	Contained WMFPs	Contained 1-WiFPs	WMFPDI	WIPDI	FTDI
T_1	$\{BE\}$	$\{E\}$	1.6	2	4.6
T_2	$\{BDA\}$	$\{C\}, \{F\}$	0.8	3	4.13
T_3	$\{BE\}$	$\{E\}, \{F\}$	1.6	3	4.27
T_4	$\{BE\}$	$\{E\}, \{F\}$	1.6	3	4.27
T_5	$\{BDA\}$	$\{C\}$	0.8	2	3.47
T_6	$\{BE\}, \{BDA\}$	$\{E\}$	2.4	2	8.4

4. Experiments and Analysis

To evaluate the effectiveness of the proposed WMFP-Outlier approach, we conduct experiments on a synthetic dataset to evaluate the outlier detection accuracy and the time cost in the outlier detection phase. The size of the synthetic dataset is 600 and each transaction of the synthetic dataset is randomly selected from $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The weight value

of each pattern is randomly generated from (0.0, 1.0) and each probability is maintained to 1 decimal place. For each transaction, the length is randomly selected from $\{5, 6, 7, 8, 9\}$. Then, we randomly implant 3 to 5 errors into every 20 transactions. The elements of the errors are randomly selected from $\{10, 11, 12, 13, 14, 15\}$ and these transactions into which errors have been implanted are labeled as true outliers.

In the experiments, four pattern-based outlier detection methods, namely, the FindFPOF method [8], the LFP method [17], the OODFP method [4] and the MIFPOF method [9], are compared with the proposed method. The experiments are conducted using various sliding window sizes ($|SW|$ is set to 20, 30 and 50) and *min_wsup* is set to 10%, 15% and 20% of $|SW|$, respectively. All algorithms are coded in the Python language and evaluated on a machine with a 2.93 GHz CPU, 4 GB RAM and Windows 10 OS.

4.1. Detection Accuracy of the WMFP-Outlier Approach

In this subsection, we evaluate the detection accuracy of the proposed WMFP-Outlier approach using various sliding window sizes and various *min_wsup* values. The experimental results are presented in Figures 3 to 5. In the figures, the x-axis (“No. of sliding windows”) corresponds to the serial number of the sliding window and the y-axis (“TPR (%)”) corresponds to the outlier detection accuracy of the proposed WMFP-Outlier method. The true-positive rate (TPR) is an accuracy evaluation indicator; it is expressed as $TPR = TP / (TP + FN)$, where the TP denotes the number of true positives (accurate detec-

Figure 3

Accuracy of the WMFP-Outlier approach when $|SW|$ is 20

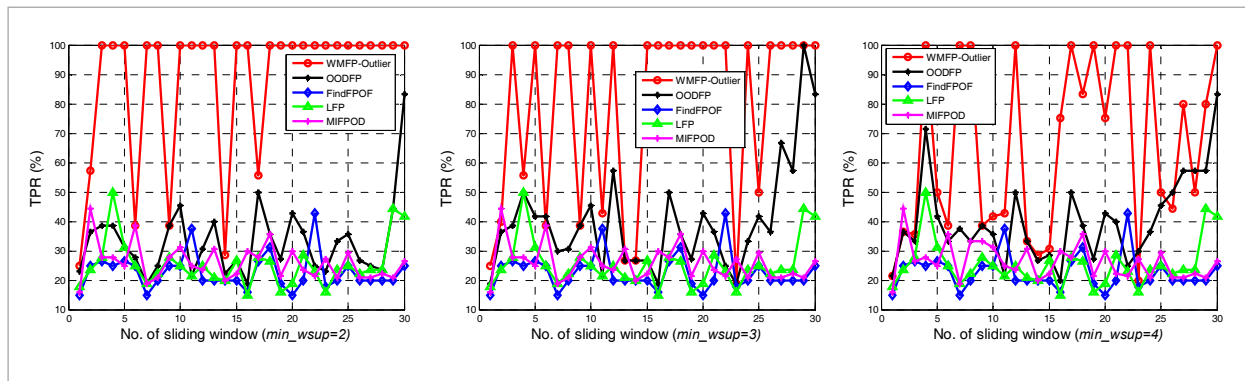


Figure 4

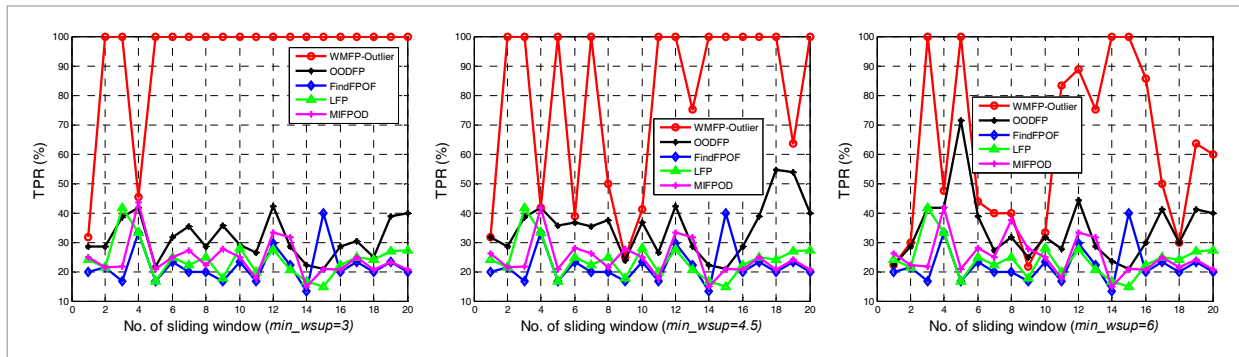
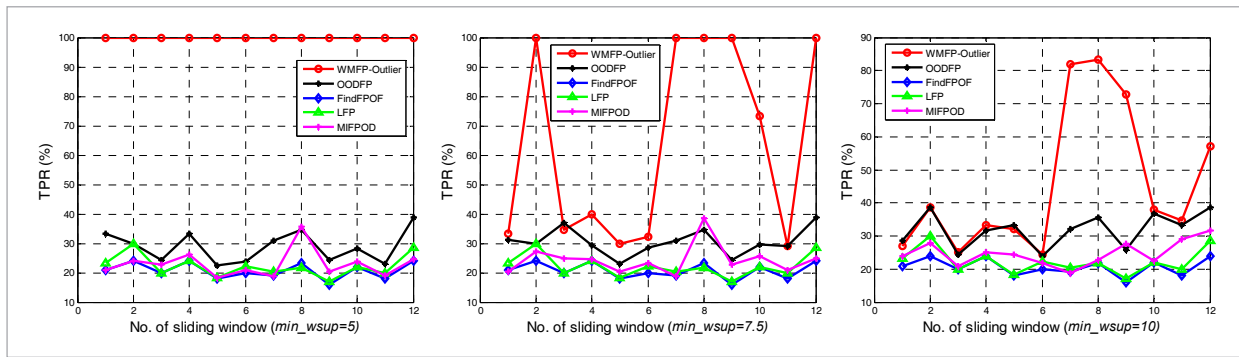
Accuracy of the WMFP-Outlier approach when $|SW|$ is 30

Figure 5

Accuracy of the WMFP-Outlier approach when $|SW|$ is 50

tions) and FN denotes the number of false negative (incorrect detections).

For a sliding window of size 20, the accuracy of the proposed WMFP-Outlier approach is shown in Figure 3. When the min_wsup value is 2, for the WMFP-Outlier approach, six of the thirty sliding windows detect errors and the detection accuracy of the WMFP-Outlier approach is always the highest among the five compared methods. When the min_wsup value is 3, error detection occurs in ten of the thirty sliding windows. In the second, thirteen and twenty-three sliding windows, the outlier detection accuracy of the WMFP-Outlier approach is slightly lower than that of the MIFPOD approach. However, when the min_wsup value is 4, of the thirty sliding windows, only the ten sliding windows have a detection accuracy of 100%. The proposed WMFP-Outlier approach has the highest detection accuracy in most sliding windows. When the size of the sliding window is constant, the outlier detection accuracy of the WMFP-Outlier ap-

proach exhibits a decreasing trend as the min_wsup values increase; hence, the proposed WMFP-Outlier approach is more accurate when the min_wsup value is small.

This is because when the min_wsup value is relatively large, the scale of the mined weighted maximal frequent patterns is very small and the number of patterns that are used during the outlier detection phase is also small; therefore, the outlier detection accuracy is lower. In most cases, the accuracy of the FindFPOF approach is the lowest among the five compared methods, while that of the improved LFP is the second lowest. With the increase of the min_wsup value, the MIFPOD approach gradually demonstrates its advantages; however, its detection accuracy is not stable.

For a sliding window size of 30, the outlier detection accuracies of the five compared methods are shown in Figure 4. When min_wsup is 3, the error detection occurs in two of the twenty sliding windows, the detection accuracy of WMFP-Outlier is the highest among the

five compared methods, and the accuracy of the FindFPOF approach is the lowest among the five compared methods in most sliding windows. When the min_wsup value is 4.5, the error detection of the WMFP-Outlier approach occurs more frequently than that when the min_wsup value is 3; however, the accuracy of the WMFP-Outlier approach is always the highest among the five compared methods. When the min_wsup value is 6, the accuracy of the WMFP-Outlier approach is much lower than that when the min_wsup value is 4.5 and only four sliding windows of the WMFP-Outlier approach have a detection accuracy of 100%. In the first and ninth sliding windows, the accuracy of the WMFP-Outlier approach is not the highest among the five compared methods. Generally, similar to the case when the sliding window size is set to 20, when the min_wsup value is small, the outlier detection accuracy of the WMFP-Outlier approach is very high. The accuracy of the FindFPOF approach is the lowest in most sliding windows.

Figure 5 shows the outlier detection accuracies for the five compared methods when the sliding window size is 50. When the min_wsup value is 5, in all twelve sliding windows, the accuracy of the WMFP-Outlier approach reaches 100%; although the accuracies of the four comparison methods are similar, they do not exceed 40%. When the min_wsup value is 7.5, the accuracy of the WMFP-Outlier approach is substantially lower compared with that when the min_wsup value is set to 5, while 100% detection accuracy is realized only in five sliding windows; however, except for the third and eleventh sliding windows, the accuracy of the WMFP-Outlier approach is higher than those of the compared methods. When the min_wsup value

is 10, the main advantage of the proposed WMFP-Outlier approach in terms of outlier detection accuracy is very small; however, in most sliding windows, the detection accuracy is slightly higher than those of the other four compared methods.

The experimental results demonstrate that the proposed WMFP-Outlier approach is more suitable for detecting the implicit outliers from weighted data streams using small min_wsup values.

4.2. Time Cost of the WMFP-Outlier Approach in the Outlier Detection Phase

For the outlier detection, the detection accuracy and time cost are the two major indices that are used to evaluate the proposed approach. Therefore, this subsection evaluates the time cost of the WMFP-Outlier approach in the outlier detection phase. The experiments are conducted using various sliding window sizes and various min_wsup values and four pattern-based approaches, namely, FindFPOF, OODFP, LFP and MIFPOD, are also compared in the experiments. The experimental results are presented in Figures 6 to 8.

When the sliding window size is 20, the time costs of the five compared approaches for various min_wsup values are shown in Figure 6. For each min_wsup value, the time cost of the FindFPOF approach is the highest among the five compared methods, while the time cost of the LFP approach is the lowest among the five compared methods. For various min_wsup values, the time cost of the proposed WMFP-Outlier approach is less than that of the MIFPOD approach but is very close to that of the OODFP approach.

Figure 6

Time cost of the outlier detection phase when $|SW|$ is 20

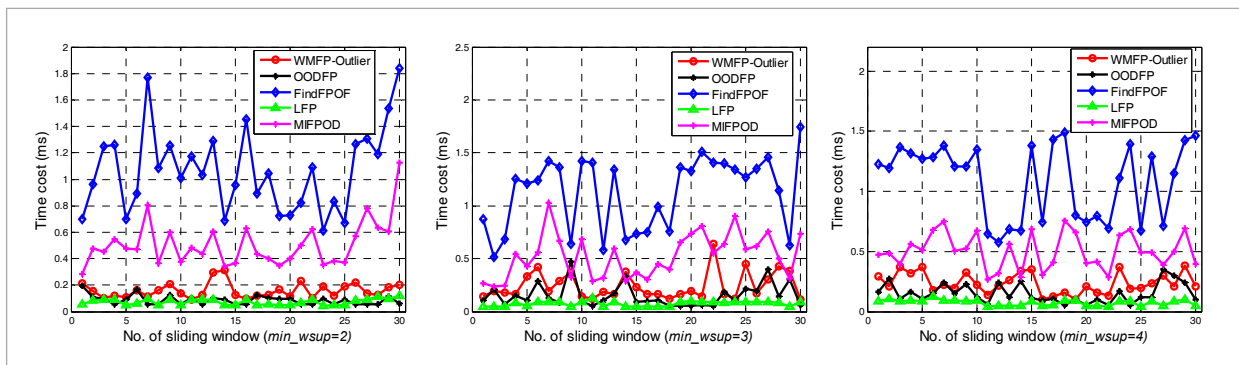


Figure 7

Time cost of the outlier detection phase when $|SW|$ is 30

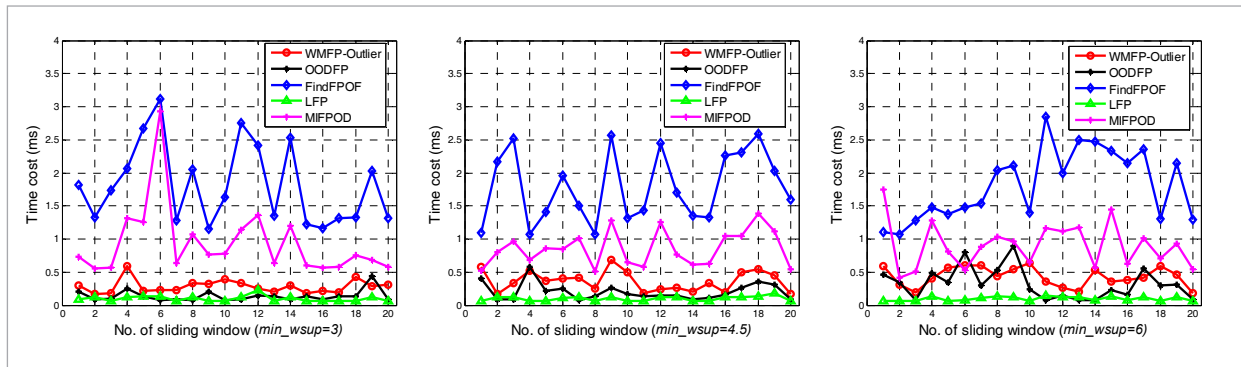
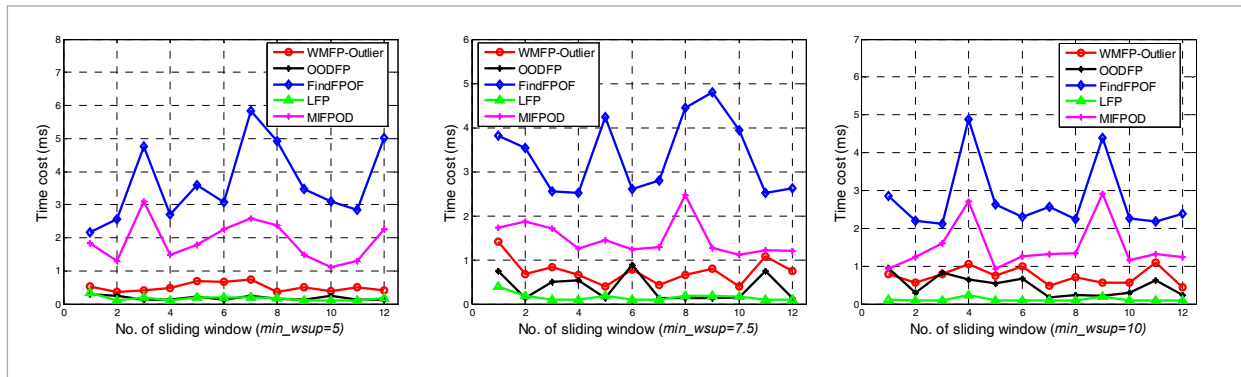


Figure 8

Time cost of the outlier detection phase when $|SW|$ is 50



In addition, the time cost of the WMFP-Outlier approach is highly stable and the min_wsup value has little effect on the time cost of the WMFP-Outlier approach. This experimental result demonstrates that the use of weighted maximal frequent patterns can reduce the time cost of the outlier detection phase.

For a sliding window size of 30, the time costs of the compared five approaches using various min_wsup values are shown in Figure 7. Among the five compared approaches, the time cost of the LFP approach is the lowest and the time cost of the FindFPOF approach is the highest in most sliding windows; the time cost of the MIFPOD approach the second highest. The time cost trend of the WMFP-Outlier approach is relatively stable under various min_wsup values. When the min_wsup value is small, the time cost of the WMFP-Outlier approach is slightly higher than that of the OODFP approach and with the

increase of the min_wsup value, the time cost of the WMFP-Outlier approach is only slightly lower than that of the OODFP approach in a few sliding windows, however, overall, the time costs of the WMFP-Outlier approach and the OODFP approach are comparable. For various min_wsup values, the time cost of the proposed WMFP-Outlier approach varies little; hence, the min_wsup value has little effect on the time cost of the WMFP-Outlier approach.

For a sliding window size of 50, the time costs of the five compared methods for various min_wsup values are shown in Figure 8. When the min_wsup value is set to 5, the time costs of the OODFP approach and the LFP approach are almost the same. When the min_wsup value is set to 7.5 or 10, the time cost of the LFP approach is slightly lower than that of the OODFP approach. Among the five compared methods, the time cost of the FindFPOF approach is the highest and the time cost of the MIFPOD approach is the

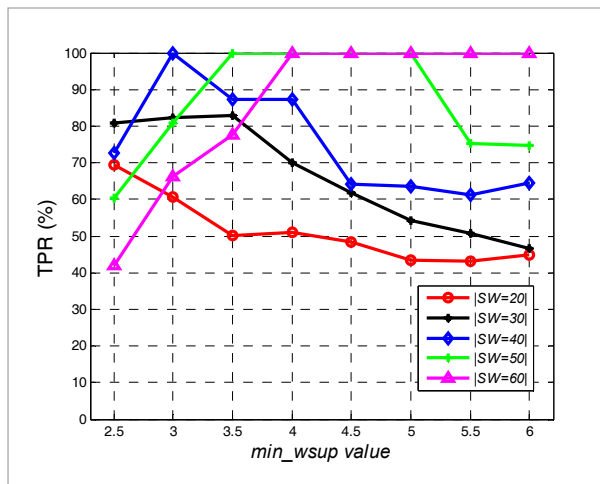
second highest. As the min_wsup value increases, the time cost of the WMFP-Outlier approach becomes slightly lower than that of the OODFP approach in a few sliding windows. Overall, the change in the time consumption of the WMFP-Outlier approach is very small and the time cost of the proposed WMFP-Outlier approach is lower than those of the FindFPOF and MIFPOD approaches.

4.3. Impact of $|SW|$ on the Accuracy and Time Cost of the Outlier Detection Phase

In subsections 4.1 and 4.2, we present the detection accuracy and time cost results of the WMFP-Outlier approach for a fixed sliding window size ($|SW|$) and discuss the effect of the min_wsup value on the detection accuracy and the time cost. However, the size of the sliding window can also affect the detection accuracy and the time cost. Therefore, in this subsection, the impacts of $|SW|$ on the accuracy and time cost of outlier detection are evaluated. In the experiment, the value of min_wsup is selected from $\{2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6\}$ and the sliding window size is selected from $\{20, 30, 40, 50, 60\}$. The experimental results are shown in Figures 9 and 10.

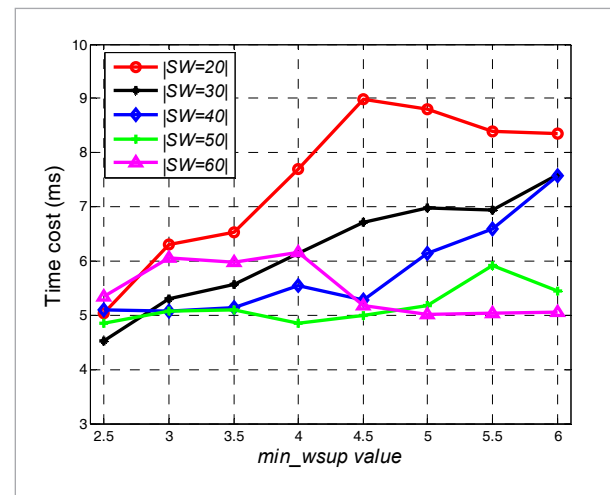
According to Figure 9, when min_wsup is set to 2.5 and the sliding window size is set to 60, the detection accuracy of the WMFP-Outlier approach is the lowest, whereas the detection accuracy is the highest when the sliding window size is set to 30. With the increase of the min_wsup value, when the $|SW|$ is

Figure 9
Impact of $|SW|$ on the outlier detection accuracy



60, the outlier detection accuracy of the WMFP-Outlier approach exhibits an increasing trend; when $|SW|$ is set to 20 or 30, the detection accuracy of the WMFP-Outlier approach exhibits a decreasing trend. The main reason is that in large sliding windows, the number of mined WMFPs is larger; hence, the accuracy will be much higher. When the min_wsup value is not less than 3.5, the outlier detection accuracy is also higher in large sliding windows because the number of mined WMFPs is very large in a large sliding window. Therefore, the sliding window size affects the outlier detection accuracy and to improve the outlier detection accuracy, the ratio of min_wsup to $|SW|$ should not be set to a large value for the proposed WMFP-Outlier approach.

Figure 10
Impact of $|SW|$ on the time cost of outlier detection



According to Figure 10, when min_wsup is set to 2.5, the time costs of the proposed WMFP-Outlier approach are similar among sliding window sizes. When min_wsup is set to 6, the time cost of the WMFP-Outlier approach is very large for small sliding windows. The time cost of the proposed WMFP-Outlier approach is positively correlated with the min_wsup value, except when min_wsup is set to 60. When the min_wsup value is set relatively large, the time cost of the WMFP-Outlier approach is more competitive for large sliding windows. The experimental results demonstrate that the size of the sliding window affects the time cost of the WMFP-Outlier approach in the outlier detection phase.

5. Conclusions

In this paper, a weighted maximal frequent-pattern-based outlier detection approach, namely, WMFP-Outlier, is proposed for detecting the implicit outliers in a weighted data stream. The outlier detection process can be divided into two phases: (1) the weighted maximal frequent-pattern mining phase and (2) the pattern-based outlier detection phase. In the weighted maximal frequent-pattern mining process, the WFP-Tree structure is used to store the detailed pattern information and weight information and the WFP-array structure and anti-monotonicity property are used to accelerate WMFP mining. In the pattern-based outlier detection process, we define three deviation indices, namely, the weighted maximal frequent pattern deviation index (WMFPDI), the weighted infrequent 1-pattern deviation index (WIPDI) and the final transaction deviation index (FTDI), for measuring the degree of abnormality of each transaction. The transactions that have lower FTDI values are more likely to be implicit outliers.

The performance of the WMFP-Outlier approach is evaluated on a synthetic dataset, and the detection accuracy and the time cost of the outlier detection phase are evaluated under various sliding window sizes and various *min_wsup* values. The experimental results demonstrate that the detection accuracy of

the proposed WMFP-Outlier approach is higher than those of the FindFPOF, LFP, OODFP and MIFPOD approaches in most scenarios. In all sliding windows, the time cost of the WMFP-Outlier approach is lower than that of the FindFPOF approach and the time cost of the WMFP-Outlier approach is also lower than that of the MIFPOD approach in most sliding windows. In addition, when the *min_wsup* value is relatively small, the detection accuracy of the proposed WMFP-Outlier approach is very high. The proposed WMFP-Outlier approach takes full account of the influence of weight information on outlier detection; therefore, it is more suitable for detecting implicit outliers from a realistic weighted data stream.

In the future, we will consider applying our proposed mining strategy to the mining process of weighted closed frequent-pattern mining and investigate the application of the weighted closed frequent-pattern-based outlier detection approach to a weighted data stream. In addition, we are prepared to design an automatic threshold optimization selection method for improving the outlier detection accuracy.

Acknowledgement

This work was supported by the Fundamental Research Funds for the Central Universities under grant number 2018XD004.

References

1. Angiulli, F., Basta, S., Lodi, S., Sartori, C. GPU Strategies for Distance-Based Outlier Detection. *IEEE Transactions on Parallel Distributed Systems*, 2016, 27(11), 3256-3268. <https://doi.org/10.1109/TPDS.2016.2528984>
2. Ayadi, A., Ghorbel, O., Obeid, A., Abid, M. Outlier Detection Approaches for Wireless Sensor Networks: A Survey. *Computer Networks*, 2017, 129, 319-333. <https://doi.org/10.1016/j.comnet.2017.10.007>
3. Cai, S., Sun, R., Cheng, C., Wu, G. Exception Detection of Data Stream Based on Improved Maximal Frequent Itemsets Mining. *Proceedings of 11th Springer Chinese Conference on Trusted Computing and Information Security (CTCIS 2017)*, Changsha, China, September 14-17, 112-125. https://doi.org/10.1007/978-981-10-7080-8_10
4. Feng, L., Wang, L., Jin, Bo. Research on Maximal Frequent Pattern Outlier Factor for Online High Dimensional Time-Series Outlier Detection. *Journal of Convergence Information Technology*, 2010, 5(10), 66-71. <https://doi.org/10.4156/jcit.vol5.issue10.9>
5. Hahsler, M., Bolanos, M., Forrest, J. An Extensible Framework for Data Stream Clustering Research with R. *Journal of Statistical Software*, 2017, 76(14), 1-50. <https://doi.org/10.18637/jss.v076.i14>
6. Hao, S., Cai, S., Sun, R., Li, S. An Efficient Outlier Detection Approach Over Uncertain Data Stream Based on Frequent Itemset Mining. *Information Technology and Control*, 2019, 48(1), 34-46. <https://doi.org/10.5755/j01.itc.48.1.21162>
7. Hawkins, D. M. *Identification of Outliers*. London: Chapman and Hall, 1980. <https://doi.org/10.1007/978-94-015-3994-4>
8. He, Z., Xu, X., Huang, Z. J., Deng, S. FP-Outlier: Frequent Pattern Based Outlier Detection. *Computer Science*

- and Information Systems, 2005, 2(1), 103-118. <https://doi.org/10.2298/CSIS0501103H>
9. Hemalatha, C.S., Vaidehi, V., Lakshmi, R. Minimal Infrequent Pattern Based Approach for Mining Outliers in Data Streams. *Expert Systems with Applications*, 2015, 42(4), 1998-2012. <https://doi.org/10.1016/j.eswa.2014.09.053>
 10. Radovanović, M., Nanopoulos, A., Ivanović, M. Reverse Nearest Neighbors in Unsupervised Distance-Based Outlier Detection. *IEEE Transactions on Knowledge and Data Engineering*, 2015, 27(5), 1369-1382. <https://doi.org/10.1109/TKDE.2014.2365790>
 11. Salehi, M., Leckie, C., Bezdek, J. C., Vaithianathan, T., Zhang, X. Fast Memory Efficient Local Outlier Detection in Data Streams. *IEEE Transactions on Knowledge and Data Engineering*, 2016, 28(12), 3246-3260. <https://doi.org/10.1109/TKDE.2016.2597833>
 12. Shaikh, S. A., Kitagawa, H. Efficient Distance-Based Outlier Detection on Uncertain Datasets of Gaussian Distribution. *World Wide Web*, 2014, 17(4), 511-538. <https://doi.org/10.1007/s11280-013-0211-y>
 13. Tang, B., He, H. A Local Density-Based Approach for Outlier Detection. *Neurocomputing*, 2017, 241, 171-180. <https://doi.org/10.1016/j.neucom.2017.02.039>
 14. Yun, U., Lee, G., Ryu, K. H. Mining Maximal Frequent Patterns by Considering Weight Conditions over Data Streams. *Knowledge-Based Systems*, 2014, 55, 49-65. <https://doi.org/10.1016/j.knosys.2013.10.011>
 15. Yun, U., Lee, G., Lee, K. M. Efficient Representative Pattern Mining Based on Weight and Maximality Conditions. *Expert Systems*, 2016, 33(5), 439-462. <https://doi.org/10.1111/exsy.12158>
 16. Zhang, L., Lin, J., Karim, R. Adaptive Kernel Density-Based Anomaly Detection for Nonlinear Systems. *Knowledge-Based Systems*, 2018, 139, 50-63. <https://doi.org/10.1016/j.knosys.2017.10.009>
 17. Zhang, W., Wu, J., Yu, J. An Improved Method of Outlier Detection Based on Frequent Pattern. *Proceedings of WASE International Conference on Information Engineering (ICIE)*, Beidaihe, China, August 14-15, 2010, 3-6. <https://doi.org/10.1109/ICIE.2010.97>
 18. Zhao, X., Zhang, X., Wang, P., Chen, S., Sun, Z. A Weighted Frequent Itemset Mining Algorithm for Intelligent Decision in Smart Systems. *IEEE Access*, 2018, 6, 29271-29282. <https://doi.org/10.1109/ACCESS.2018.2839751>
 19. Zhu, X., Zhang, J., Li, H., Fournier-Viger, P., Lin, J. C. W., Chang, L. FRIOD: A Deeply Integrated Feature-Rich Interactive System for Effective and Efficient Outlier Detection. *IEEE Access*, 2017, 5, 25682-25695. <https://doi.org/10.1109/ACCESS.2017.2771237>