Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

FAAD: an unsupervised fast and accurate anomaly detection method for a multi-dimensional sequence over data stream*

Bin LI†1, Yi-jie WANG†‡1, Dong-sheng YANG2, Yong-mou LI1, Xing-kong MA1

[1]*Science and Technology on Parallel and Distributed Processing Laboratory, College of Computer, National University of Defense Technology, Changsha 410073, China*

[2]*Block Chain Research Institute of LianLian Pay, Hangzhou 310000, China*

†E-mail: libin16a@nudt.edu.cn; wangyijie@nudt.edu.cn

Received Jan. 15, 2018; Revision accepted May 13, 2018; Crosschecked Mar. 14, 2019

Abstract: Recently, sequence anomaly detection has been widely used in many fields. Sequence data in these fields are usually multi-dimensional over the data stream. It is a challenge to design an anomaly detection method for a multi-dimensional sequence over the data stream to satisfy the requirements of accuracy and high speed. It is because: (1) Redundant dimensions in sequence data and large state space lead to a poor ability for sequence modeling; (2) Anomaly detection cannot adapt to the high-speed nature of the data stream, especially when concept drift occurs, and it will reduce the detection rate. On one hand, most existing methods of sequence anomaly detection focus on the single-dimension sequence. On the other hand, some studies concerning multi-dimensional sequence concentrate mainly on the static database rather than the data stream. To improve the performance of anomaly detection for a multi-dimensional sequence over the data stream, we propose a novel unsupervised fast and accurate anomaly detection (FAAD) method which includes three algorithms. First, a method called "information calculation and minimum spanning tree cluster" is adopted to reduce redundant dimensions. Second, to speed up model construction and ensure the detection rate for the sequence over the data stream, we propose a method called "random sampling and subsequence partitioning based on the index probabilistic suffix tree." Last, the method called "anomaly buffer based on model dynamic adjustment" dramatically reduces the effects of concept drift in the data stream. FAAD is implemented on the streaming platform Storm to detect multi-dimensional log audit data. Compared with the existing anomaly detection methods, FAAD has a good performance in detection rate and speed without being affected by concept drift.

Key words: Data stream; Multi-dimensional sequence; Anomaly detection; Concept drift; Feature selection
https://doi.org/10.1631/FITEE.1800038 **CLC number:** TP391.4

1 Introduction

Anomaly detection refers to the problem of finding patterns in data that do not conform to the expected normal behavior (Chandola et al., 2009). Sequence anomaly detection, which mines sequence anomalies by analyzing the ordering relationship, is widely applied to credit card fraud detection, internal intrusion detection, and aircraft condition monitoring (Wang and Li, 2006; Chandola et al., 2012; Dani et al., 2015; Wang et al., 2018). However, in these fields, sequences are usually multi-dimensional in the data stream, which brings new challenges in anomaly detection compared with the traditional methods for a single-dimensional sequence on static data.

‡ Corresponding author

* Project supported by the National Key R&D Program of China (No. 2016YFB1000101), the National Natural Science Foundation of China (Nos. 61379052 and 61502513), the Natural Science Foundation for Distinguished Young Scholars of Hunan Province, China (No. 14JJ1026), and the Specialized Research Fund for the Doctoral Program of Higher Education, China (No. 20124307110015)

ORCID: Bin LI, http://orcid.org/0000-0003-0876-2694

Table 1 shows an example of the multi-dimensional sequence data. It is a sequence which records a user's operation, access path, call time, and return value. These four fields are regarded as four dimensions. Each row is regarded as a state in a sequence. We call it "a four-dimensional sequence with five states." Our analysis of sequence data does not focus on a single state or dimension but a relation between states or dimensions.

Table 1 Multi-dimensional sequence data

User's operation	Access path	Call time	Return value
Open	\root	598 768 333	True
Read	\home\dataset	598 768 462	True
Cat	\home\name	598 768 987	True
Open	\home\svd	598 769 678	False
Close	\root	598 773 543	True

Challenges of multi-dimensional sequence anomaly detection over the data stream can be summarized as follows: (1) The state space can be explosive in growth as the dimension increases. For an m-dimensional sequence $\mathrm{MS} = \{\mathrm{ms}_0, \mathrm{ms}_1, \ldots, \mathrm{ms}_{n-1}\}$ with length n, every state ms_i $(i = 0, 1, ..., n-1)$ has m dimensions. The state space of MS is $|S| = n \cdot m$, which results in data sparseness, efficiency decrease, and poor anomaly detection. (2) The data stream is continuous and arrives at an unprecedented speed, which requires the anomaly detection method process in a timely manner (Wang et al., 2013; Li et al., 2014; Wang and Ma, 2015). (3) Compared with tha static dataset, concept drift may occur in the data stream, which could affect the performance of anomaly detection. The distribution of newly arrived data may be different from that of historical data, thus leading to a higher false negative rate in detection.

A majority of works of anomaly detection for sequence over the data stream focus on a single-dimensional sequence (Chandola et al., 2008; Budalakoti et al., 2009). Since anomalies need to be detected in the data stream in a timely fashion, these works always have high computation cost and cannot be directly applied to a multi-dimensional sequence which has a large state space. Moreover, most anomaly detections for a multi-dimensional sequence focus on the static data and multivariate time series (Keogh et al., 2001; Lee, 2015; Xianyu et al., 2017) to use frequent item mining technology. However, these methods cannot be applied to the data stream, which always leads to a longer time for modeling and detection. The frequent item mining technology can be applied to only fixed pattern data, and it is difficult to fully mine the sequence relationship. Additionally, some studies provide a supervised learning method to detect anomalies for a multi-dimensional sequence over the data stream (Bao and Wang, 2016). However, since data arrive at a high speed, it is difficult to obtain tag data in time for supervised learning. It is also difficult to obtain anomalous tags from a large amount of normal data because of the serious data imbalance in the data stream, which results in a decrease in the detection rate. Compared with supervised learning, unsupervised learning is more suitable for sequence anomaly detection over the data stream, because it does not rely on tag data and is insensitive to data imbalance. Therefore, very few studies can provide a feasible solution to unsupervised anomaly detection for a multi-dimensional sequence over the data stream.

Motivated by these factors, in this study, we propose a novel unsupervised fast and accurate anomaly detection (FAAD) method for a multi-dimensional sequence over the data stream to achieve higher detection rate and detection speed without being affected by concept drift. FAAD includes three key algorithms and provides the following contributions:

1. To reduce the complexity of the space while fully preserving the information of the multi-dimensional sequence, we propose a feature selection method called "information calculation and minimum spanning tree cluster" (IMC) which can reduce the redundant features in a sequence.

2. A random sampling and subsequence partitioning based on the index probabilistic suffix tree (RSIPST) method is proposed to adapt to the dynamic nature of the data stream. It can accelerate model construction and ensure the anomaly detection rate.

3. To find the concept drift in the data stream and update the existing models in a timely manner, we propose an anomaly buffer based on the model dynamic adjustment (ABMDA) method to reduce the effects of concept drift without adding complexity.

To further validate our method, FAAD is designed and implemented on the streaming platform

Storm to detect multi-dimensional log audit data. Experimental results showed that these three algorithms can perform better and that FAAD can achieve a higher detection rate and a lower false positive rate than the existing methods.

2 Related work

Most works of anomaly detection for sequence focus on single-dimensional sequences. Yang and Wang (2003) and Shu et al. (2015) established a main rule of the one-dimensional sequential relationship to detect anomalies. Yamanishi and Maruyama (2005) adopted hidden Markov models (HMMs) to describe normal sequences and to calculate the anomaly score. Although the HMMs could perform well in a complex environment, its high computation complexity could lead to a poor detection performance. Xiong et al. (2011) proposed an effective Markov model to approximate the conditional probability distribution and to design a novel two-tier Markov model to represent a sequence cluster. Budalakoti et al. (2009) adopted a clustering-based method that uses the longest common subsequence as the similarity measure and combined clustering large applications (CLARA) (Kaufman and Rousseeuw, 2009) to cluster sequence sets. The Bayes method was used to calculate the probability of the testing sequence. The smaller the probability, the more likely these sequences were anomalies. Qian et al. (2012) adopted the edit distance and the shortest distance between two nodes as a new kernel function. They introduced a graph model, increased the flexibility of the model, and achieved better anomaly detection than the traditional editing distance methods. However, because it is complex to find the shortest path and editing distance, it requires a lot of storage space. This method is not suitable for large-scale sequence sets. Li et al. (2012) transformed sequences into numerical feature vectors with a co-occurrence matrix, which could preserve the information of both frequency and order, but the feature space would be enlarged and the vector was very sparse, thus reducing the efficiency.

Other studies of multi-dimensional sequence concentrate on the static database and multivariate time series, which could mine normal or abnormal patterns from the database and express them with an informative data model. Esposito et al. (2008) discussed the frequent patterns of dimensions. They took the possibility into account and expressed the mined complex patterns in a first-order language, in which events may occur along different dimensions. Specifically, multi-dimensional patterns were defined as a set of first-order atomic formulae, which represent events with a variable and the relations between events with a set of dimensional predicates. Jin and Zuo (2007) proposed an incremental multi-dimensional sequence pattern mining method based on a novel data model called "multi-dimensional concept lattice" (MDCL). The MDCL is informative because it consists of both ordered task-relevant dimension and unordered background dimension. Box et al. (2015) used the autoregressive moving average (ARMA) model to process regression analysis to predict the sequence tendency. These methods often use frequent item mining that can be applied to only fixed pattern data but have difficulty in fully mining the sequence relationship. Additionally, anomaly detection on a static dataset is very different from that over the data stream: (1) These methods on the static dataset cannot adapt to the dynamic nature of the data stream because of their high computation costs; (2) Concept drift in the data stream leads to a low detection rate, which brings new challenges to be addressed.

A few studies focus on anomaly detection for a multi-dimensional sequence over the data stream. Cost sensitive support vector machine (C-SVM), proposed by Bao and Wang (2016), first transforms multi-dimensional sequences into feature vectors and detects abnormal sequences over a dynamically imbalanced data stream by testing these vectors based on C-SVM in real time. However, this method focuses on supervised anomaly detection, the labeling data of which will take much effort and time. Compared with supervised learning, unsupervised learning is more suitable for sequence anomaly detection over the data stream because it does not rely on tag data and is insensitive to data imbalance.

3 FAAD for a multi-dimensional sequence over data stream

3.1 Overview

In this subsection, we will introduce the overview of FAAD. Before that, we define the multi-dimensional sequence over a data stream as follows: Let a data stream MS $= \{\mathrm{ms}_0, \mathrm{ms}_1, \ldots, \mathrm{ms}_{n-1}\}$ be an infinite sequence of states, where each state is associated with a time stamp i, i.e., ms_i, and $n-1$ is the identifier of the most recent state ms_{n-1}. A state $\mathrm{ms}_i = \{f_0(i), f_1(i), \ldots, f_{m-1}(i)\}$ is defined as a sequence appearing at the i^{th} time unit, where m is the number of dimensions. $f_j(i)$ is the feature value of ms_i on the j^{th} dimension ($j = 0, 1, \ldots, m-1$). The data model is fixed for the data stream where all states are defined over the same set of dimensions $F = \{f_0, f_1, \ldots, f_{m-1}\}$. For each dimension, we are thus provided with the corresponding list of item sets; for example, in the dimension color, we list {red, blue, yellow} as the optional feature values of this dimension.

Fig. 1 is the overview of the fast and accurate anomaly detection method for a multi-dimensional sequence over the data stream. First, IMC selects the representative feature by computing mutual information and symmetric uncertainty information in the training data to speed up detection and fully preserve the effective information of the multi-dimensional sequence. k representative features $\{f_{l_0}, f_{l_1}, \ldots, f_{l_{k-1}}\}$ ($l_j = 0, 1, \ldots, m-1$) are selected by the minimum spanning tree (MST) to construct a feature subset $S =< S_0, S_1, \ldots, S_{k-1} >$. The j^{th} feature of every state comprises $S_j = \{f_{l_j}(0), f_{l_j}(1), \ldots, f_{l_j}(n-1)\}$. Second, these selected features are processed in RSIPST to construct models. In the sequence modeling phase, since the selected features have a low correlation, each S_i in S is randomly sampled and partitioned to independently construct model M_i in the model set $M = \{M_0, M_1, \ldots, M_{k-1}\}$ by the index probabilistic suffix tree. Third, when the new testing data stream arrives, we select the corresponding features from the testing data stream. The anomaly score is calculated by each model M_i on the data stream, and the weighted sum of all M_i in M determines whether the testing data stream is abnormal or not. Last, ABMDA detects concept drift by a hybrid method and reconstructs new models to reduce the effects of concept drift. The abnormal sequence possibly caused by concept drift in the first detection is added to the anomaly buffer to reconstruct a new model M_i', which will be used to detect these sequences in the buffer again. After the second detection, these sequences are true anomalies if determined as anomalies again; otherwise, they are just normal sequences occurring because of concept drift and wrongly detected. In later subsections, we will describe these three new methods IMC, RSIPST, and ABMDA.

3.2 IMC

A feature selection method IMC is proposed to reduce the spatial complexity of sequence data and improve the modeling performance. Mutual information and symmetric uncertainty information are calculated to measure the correlative and redundant information, which is stored in a complete graph to construct an MST. The tree is divided into k clusters, and a representative feature is selected from each cluster to comprise the feature subset. Fig. 2 is the flow chart of IMC. We will describe the details of IMC in this subsection.

3.2.1 Mutual information and symmetric uncertainty information

The mutual information and symmetric uncertainty information are adopted as the measurement of correlation and redundancy of each feature

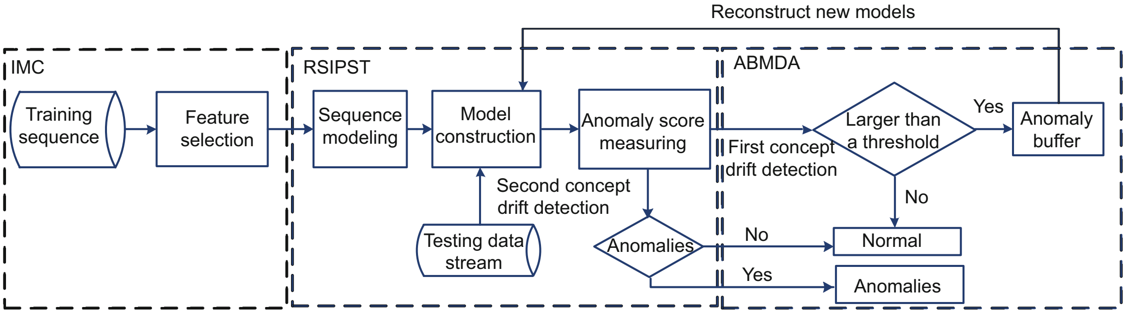


Fig. 1 Overview of an unsupervised fast and accurate anomaly detection method (FAAD)

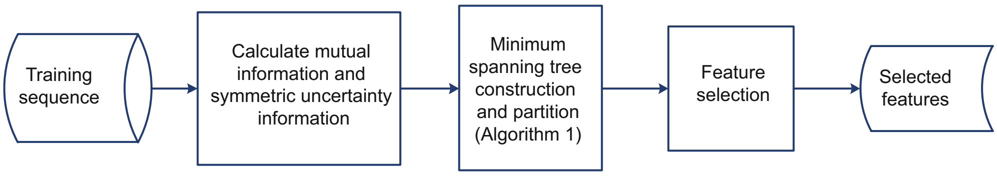

Training sequence
Calculate mutual information and symmetric uncertainty information
Minimum spanning tree construction and partition (Algorithm 1)
Feature selection
Selected features

Fig. 2 Flow chart of the information calculation and minimum spanning tree cluster (IMC)

(Sarhrouni et al., 2012). The mutual information determines how similar the joint distribution $p(X,Y)$ is to the product of factored marginal distribution $p(X)p(Y)$, where X and Y are two random variables. $p(X)$ and $p(Y)$ are the marginal probability distribution functions of X and Y, respectively. The mutual information of these two variables can be defined as

$$I(X,Y)=\sum_{x\in X}\sum_{y\in Y}p(x,y)\log\frac{p(x,y)}{p(x)p(y)}. \tag{1}$$

Compared with the mutual information, the symmetric uncertainty information additionally measures information entropy and reflects the redundant information of the data. In this study, we adopt the symmetric uncertainty information to measure the redundant information between features. The symmetric uncertainty information $\mathrm{SU}(X,Y)$ of two random variables X and Y is defined as

$$\begin{cases}\mathrm{SU}(X,Y)=\dfrac{2\,(H(X)-H(X|Y))}{H(X)+H(Y)},\\ H(X|Y)=-\displaystyle\sum_{y\in Y}p(y)\sum_{x\in X}p(x|y)\log_2 p(x|y),\end{cases} \tag{2}$$

where $H(X)$ is the entropy of random variable X, defined as $H(X)=-\sum_{x\in X}p(x)\log_2 p(x)$, and $H(X|Y)$ is the entropy of X conditioned on Y.

For a training sequence, we construct a complete graph G and calculate the mutual information between features i and j to obtain $I(i,j)$. These vertices and edges in G represent corresponding features and relationships, respectively. When the mutual information $I(i,j)$ is lower than the threshold t, the weight and symmetric uncertainty information of the corresponding edge $e(i,j)$ are set as 0; otherwise, the mutual information is adopted as the weight of this edge and the symmetric uncertainty information $\mathrm{SU}(i,j)$ between these two features is calculated.

3.2.2 MST construction and partition

Fig. 3 shows an example of a complete graph of seven-dimensional feature correlation. The thicker the lines are, the stronger correlations the corresponding features have. In general, if there are n features, graph G contains n vertices and $n(n-1)/2$ edges, bringing the high complexity in multidimensional data. Moreover, the polynomial-time algorithm to decompose the complete graph is unknown. Therefore, we adopt the MST method based on clusters to reduce the number of edges and the complexity of computation.

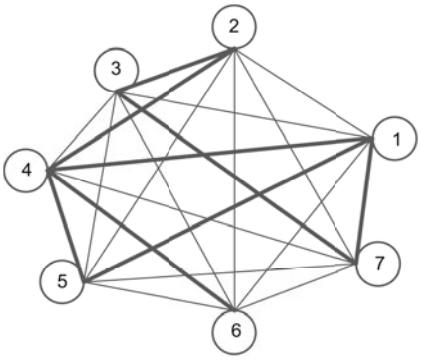


Fig. 3 A complete graph of seven-dimensional feature correlation

The MST cluster method is shown in Algorithm 1. First, we use the Prim algorithm (Kponyo et al., 2013) to build an MST, adopting the negative symmetric uncertainty information as the weight of edges (steps 1 and 2 in Algorithm 1). Second, we partition the MST with the symmetric uncertainty information. Suppose that $\{F_1,F_2,\ldots,F_k\}$ are all features in the same cluster C. If there exists $F_j\in C$ satisfying $\mathrm{SU}(F_i,F_j)<\mathrm{SU}(F_i,C)\bigwedge\mathrm{SU}(F_i,F_j)<\mathrm{SU}(F_j,C)$, we call F_i the redundant feature of F_j. The weight of edge $e(i,j)$ is set as $I(F_i,F_j)^{\mathrm{SU}(F_i,F_j)}$ (steps 3–8 in Algorithm 1). We can reduce the weight of pairs of redundant features with this method. Last, we cut the $(k-1)$ edges with a minimum weight to obtain k weak-correlation clusters with low redundant information (steps 9–12 in Algorithm 1). In particular, $\mathrm{SU}(F_i, C)$ is the correlation between feature F_i and cluster C.

Algorithm 1 MST cluster

```
Input: Complete graph $G$, the number of clusters $k$
Output: $k$ clusters
 1: Use the Prim algorithm to generate the minimum spanning tree MST = Prim($G$)
 2: Forest $F$ = MST
 3: for each edge $E(i,j)$ in forest do
 4:     if $\mathrm{SU}(F_i,F_j) < \mathrm{SU}(F_i,C) \bigwedge \mathrm{SU}(F_i,F_j) < \mathrm{SU}(F_j,C)$ then
 5:         $G.\mathrm{weight}(i,j) = I(F_i,F_j)^{\mathrm{SU}(F_i,F_j)}$
 6:     else $G.\mathrm{weight}(i,j) = I(F_i,F_j)$
 7:     endif
 8: endfor
 9: Sort $G$.weight
10: for the $k$ minimum weight edges do
11:         Delete this edge from $F$ such that $F = F - G.\mathrm{weight}(i,j)$
12: endfor
13: Return Forest $F$
```

3.2.3 Feature selection

A representative feature has the strongest correlation with other features in the same cluster, and knowledge mining on this feature can take the place of mining on other features in the same cluster, which greatly reduces the time on calculation and data spareness. For any feature $F_j \in C$, there must exist a representative feature $F_i \in C$ satisfying

$$\begin{cases} F_i = \arg\max\limits_{F_j \in C} t_j, \\ t_j = \sum\limits_{e \in F_j.\mathrm{edge}} e.\mathrm{weight} \cdot \mathrm{SU}(F_j, C), \end{cases} \tag{3}$$

where t_j is the feature information of F_j. We select F_i in every cluster to obtain k representative features. Fig. 4 shows an example of feature selection. In T_1, the feature information of F_0 is 0.2, F_1 0.6, F_2 0.18, and F_3 0.42. According to Eq. (3), F_1 should be selected as the representative feature in cluster T_1.

3.3 RSIPST

In this subsection, we present an RSIPST method to speed up model construction in Fig. 5. First, the training sequence should be preprocessed by the random sample and subsequence partitioning. Second, we construct models by the index probabilistic suffix tree (PST) based on the preprocessed training sequence. To reduce randomness, we repeat these two steps N times to construct N models for each feature. Third, corresponding features are selected from each sequence in the testing data stream to be detected. Last, we calculate the anomaly score with these models. When the score is larger than a threshold, the sequence is determined as an anomaly.

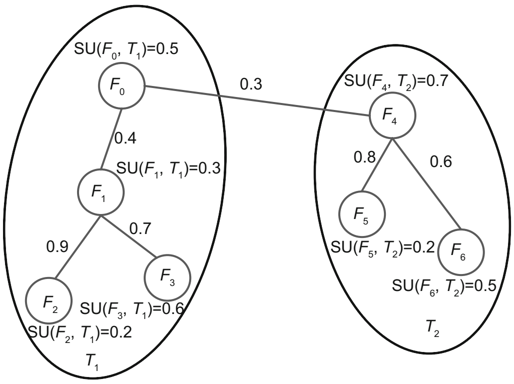


Fig. 4 Feature selection

PST is a compact representation of a variable-order Markov chain, which adopts a suffix tree as its index structure (Ron et al., 1994). Fig. 6 shows an example of PST about a sequence over the state $\{a, b\}$. Each node labeled with a string represents a path from node to root, containing a conditional probability distribution vector. For example, the node labeled ab is (0.606, 0.394), which means that the conditional probability of a after ab is $P(a|ab) = 0.606$ and b after ab is $P(b|ab) = 0.394$. The size of PST is a function of the cardinality of the state space and maximum memory length L. Several pruning mechanisms have been employed to control the size of PST. In Fig. 6, the dashed and solid lines show examples of pruning PST with Pmin = 0.02 and minCount = 25, respectively, where Pmin is the empirical probability threshold and minCount is the least appearing times of the string in the database. Since the pruned nodes take up only a small proportion in the sequence, they do not have an effect on the detection rate. The Bayes rule (Carlin and Louis, 2000) is adopted to generate the probability of sequence $S = \{s_1, s_2, \ldots, s_l\}$ (s_i is the i^{th} state in S):

$$P^{\mathrm{T}}(S) = P^{\mathrm{T}}(s_1)P^{\mathrm{T}}(s_2|s_1)\ldots P^{\mathrm{T}}(s_l|s_1s_2\ldots s_{l-1}). \tag{4}$$

In Fig. 6, for example, we calculate the probability of sequence $< ababb >$ over PST with the pruning lines Pmin = 0.02 and minCount = 25.

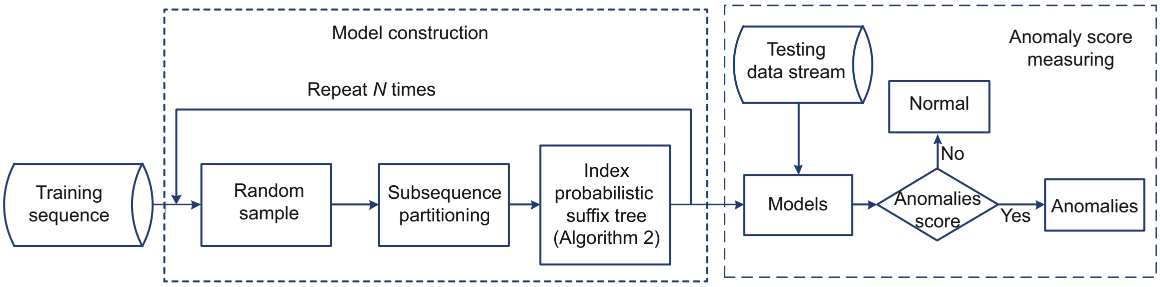


Fig. 5 Flow chart of random sampling and subsequence partitioning based on the index probabilistic suffix tree (RSIPST)

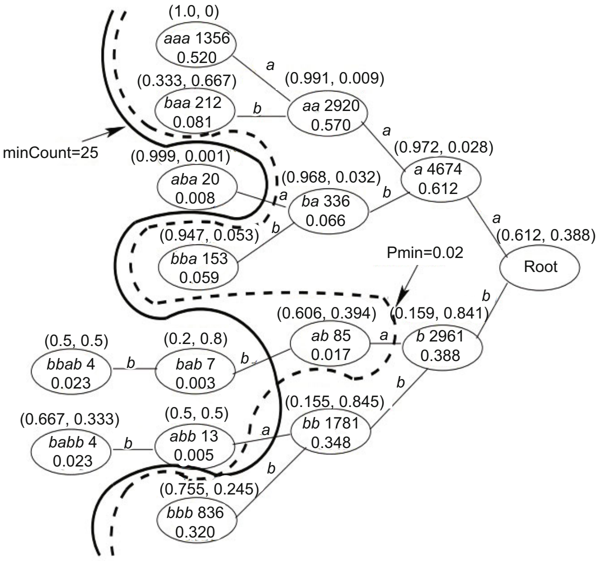


Fig. 6 An example of probabilistic suffix tree (PST)

According to Eq. (4), the longest string matching with *abab* is *bab*. Since the node labeled *bab* is pruned by minCount = 25, $P(b|abab)$ is approximately replaced by $P(b|ab) = 0.394$. Moreover, since *aba* is pruned by these lines, $P(b|aba)$ is replaced by $P(b|ba) = 0.032$. Consequently, the final result of $P(ababb)$ is

$$\begin{aligned} &P^{\mathrm{T}}(ababb) \\ &= P^{\mathrm{T}}(a)P^{\mathrm{T}}(b|a)P^{\mathrm{T}}(a|ab)P^{\mathrm{T}}(b|aba)P^{\mathrm{T}}(b|abab) \\ &= 0.612 \times 0.028 \times 0.606 \times 0.032 \times 0.394 \\ &= 1.309 \times 10^{-4}. \end{aligned}$$

3.3.1 Model construction

With the growth of sequence length and space state, we need to increase the depth of PST to have a better analysis of the sequence relationship. However, adding the depth of PST can result in the growth of size and detection time. We present a method with random sampling and subsequence partitioning based on the index PST (IPST). It can be applied to the situation where the state space is complex.

Sequences preprocessed by IMC are added to the storage pool R. When the storage pool is full, we decide whether the next arriving sequence is reserved in R depending on a random number r, which is generated when the new sequence arrives. If $r \leq$ ns (ns is the number of sequences in the pool), then the pending sequence replaces the r^{th} sequence in the pool; otherwise, sequences in R do not change. Then we randomly select the starting and ending points to partition the sequence, the length of which is set to j.

After that, we adopt IPST to hasten PST construction. Algorithm 2 shows the IPST algorithm in detail. The main innovations include: (1) We adopt a hash map as an index structure at each level and speed up the retrieval process; (2) We combine the pruning process with the construction process to reduce the traversal time.

In Algorithm 2, first, we create an empty PST pst and create an index structure IM_0 containing only the root (steps 1 and 2 in Algorithm 2). Second, when we construct the i^{th} layer of the tree ($1 \leq i \leq h$), we can obtain all subsequences of length i. For subsequence $s(j, j+i)$, if this subsequence is in IM_i, its prefix and suffix are stored in $\text{HM}_{\text{prefix}}$ and $\text{HM}_{\text{suffix}}$, respectively (steps 5–12 in Algorithm 2). Third, the conditional probability and empirical probability are calculated from $\text{HM}_{\text{prefix}}$ and $\text{HM}_{\text{suffix}}$, respectively. HM_{cp} stores the conditional probability and $\text{HM}_{\text{suffix}}$ is pruned by Pmin (steps 13–15 in Algorithm 2). Last, the index structure IM_i is adopted to find the corresponding node in HM_{cp} and $\text{HM}_{\text{suffix}}$. We update its conditional probability and put the suffix as the child node in PST. The next layer of index IM_{i+1} is constructed by $\text{HM}_{\text{suffix}}$ (steps 16–24 in Algorithm 2).

Algorithm 2 IPST

```
Input: Preprocessed sequence $D_{\text{pre}}$, tree depth $h$, the empirical probability threshold Pmin
Output: PST model pst
 1: Create an empty PST pst
 2: Create a hashmap $\text{IM}_0$ containing the index of root
 3: for each layer $i$ do
 4:     Create three hash maps $\text{HM}_{\text{prefix}}$, $\text{HM}_{\text{suffix}}$, and $\text{HM}_{\text{cp}}$
 5:     for each sequence $D'$ in $D_{\text{pre}}$ do
 6:         for each subsequence $s(j, j+i)$ in $D'$
 7:             if $s(j, j+i)$ in $\text{IM}_i$ then
 8:                 Add $\{s(j, j+i-1) \to s(j+i)\}$ to $\text{HM}_{\text{prefix}}$
 9:                 Add $\{s(j+i-1, j) \to s(j+i)\}$ to $\text{HM}_{\text{suffix}}$
10:             endif
11:         endfor
12:     endfor
13:     Calculate conditional probability with $\text{HM}_{\text{prefix}}$ and calculate empirical probability with $\text{HM}_{\text{suffix}}$
14:     Store conditional probability in hash map $\text{HM}_{\text{cp}}$
15:     Prune $\text{HM}_{\text{suffix}}$ with Pmin
16:     for each prefix $p$ in $\text{HM}_{\text{cp}}$ do
17:         Obtain the node from $\text{IM}_i$ by prefix $p$
18:         Update the node of $p$ by conditional probability in $\text{HM}_{\text{cp}}$
19:     endfor
20:     for each suffix $s$ in $\text{HM}_{\text{suffix}}$ do
21:         Obtain node $n$ from $\text{IM}_i$ by suffix $s$
22:         Put suffix $s$ as the child node of $n$
23:     endfor
24:     Create $\text{IM}_{i+1}$ with $\text{HM}_{\text{suffix}}$
25: endfor
26: Return pst
```

For example, PST is constructed by the sequence $< accactact >$ at level two. The subsequence set is $\{acc, cca, cac, act, cta, tac, act\}$. $\text{HM}_{\text{prefix}}$ stores the prefix information $\{ac{\to}(c,t,t), cc{\to}a, ca{\to}c, ct{\to}a, ta{\to}c\}$ and calculates the conditional probability of the prefix, and it is stored by another hash map HM_{cp}, for example, $P(c|ac) = 1/3$. $\text{HM}_{\text{suffix}}$ stores the suffix information of $\{ca{\to}(c,t,t), cc{\to}a, ac{\to}c, tc{\to}a, at{\to}c\}$. Since level two of index IM_2 has been constructed, we can quickly obtain the node of suffix in $\text{HM}_{\text{suffix}}$. $ca{\to}(c,t,t)$ is taken as an example. The node labeled *ca* is obtained through the index IM_2 and two child nodes *cca* and *tca* are created. The empirical probability can be calculated with $\text{HM}_{\text{suffix}}$. If Pmin is set to 1/4, then the node labeled *cca* is pruned with $P(cca) = 1/7$. Thus, the PST creates level three child nodes and level three index IM_3 at the same time.

Since the random sample may result in randomness, to make it more robust, we repeat the process of sequence modeling N times. For each feature, we obtain N models. Each model is independent since each is trained with random samples and partitionings.

3.3.2 Anomaly score measurement

When testing the data stream, we first select the same representative features that are selected in the training data to detect anomalies. For feature F_i $(i = 1, 2, \ldots, k)$, N corresponding models $M_{iq}, (q = 1, 2, ..., N)$ are generated in model construction. The probability of sequence $\text{TDS}' =< s_1, s_2, \ldots, s_l >$ under model M_{iq} $(q = 1, 2, \ldots, N)$ could be calculated by Eq. (4). We regard the probability as the anomaly score. The larger the $P(\text{TDS}')$ is, the less likely the sequence TDS' is an anomaly. This is because the data distribution of TDS' conforms to that of the training data. However, different sequences have different lengths. To reduce the effects of length and decimal overflow in experiments, we adopt the logarithm in the outcome and the regularization of length in the following formula:

$$P(\text{TDS}') = \frac{1}{l}\left(\log P(s_1) + \sum_{j=2}^{l} \log P(s_j | s_1 s_2 \ldots s_{j-1})\right). \tag{5}$$

$P_{iq}(\text{TDS}')$ is an anomaly score under model M_{iq}, satisfying $M_{iq}(\text{TDS}') = P_{iq}(\text{TDS}')$. We adopt the average of the sum of P_{iq} $(q = 1, 2, ..., N)$ under M_i as Eq. (6) to obtain $M_i(\text{TDS}')$:

$$M_i(\text{TDS}') = \frac{1}{N}\sum_{q=1}^{N} P_{iq}(\text{TDS}'). \tag{6}$$

Since each M_i represents a selected feature in the IMC, we can obtain k models. W_i is the sum of the mutual information of the i^{th} selected feature and other representative features. Thus, the anomaly score on all models can be calculated as

$$A(\text{TDS}') = \frac{\sum_{i=1}^{k} W_i \cdot M_i(\text{TDS}')}{\sum_{i=1}^{k} W_i}. \tag{7}$$

Finally, it can determine whether this sequence

TDS′ is an anomaly compared with the given empirical threshold T, as Eq. (8) shows:

$$f = \begin{cases} \text{normal}, & A(\text{TDS}') > T, \\ \text{anomaly}, & \text{otherwise}. \end{cases} \tag{8}$$

In our later experiments, $T = 1.5$ can have a good result.

3.4 ABMDA

We first give the definition of concept drift over the data stream. Compared with the static dataset, for a data stream $\text{MS} = \{\text{ms}_0, \text{ms}_1, \ldots, \text{ms}_{n-1}\}$, if there exist data $\text{ms}_i \sim \text{ms}_j$ in distribution A in a period and data $\text{ms}_m \sim \text{ms}_n\,(i < j < m < n)$ in distribution $B\,(A \neq B)$ in a later period, then we believe that the concept drift occurs in data $\text{ms}_i \sim \text{ms}_j$ and $\text{ms}_m \sim \text{ms}_n$. Consequently, the model built with historical data cannot characterize the current data well, which makes predictions less accurate as time passes and leads to a higher false negative rate. We propose a method called "ABMDA" to detect concept drift and dynamically adjust models according to the results, ensuring the speed and accuracy of anomaly detection. Fig. 7 is a flow chart of the ABMDA. A hybrid concept drift detection method based on statistics and anomaly detection rate finds anomalies possibly caused by concept drift in the first detection and adds them to the anomaly buffer. When the buffer is full, anomalies in the buffer are adopted to reconstruct new models to detect these anomalies in the buffer again. During this process, the time decay function limits the number of models to ensure the speed of detection. If these anomalies in the buffer are detected as being abnormal again in the second detection, they are determined as true anomalies.

3.4.1 Score based on statistics and anomaly rate

Appendixes A and B show the hybrid concept drift detection in detail based on anomaly detection rate and statistics, respectively. First, the proportion of current anomalies P_t, the average m, and the variance s of the proportion of historical anomalies P are calculated. Then we obtain the difference value pd between the proportion of current anomalies P_t and historical anomalies P. The larger the pd is, the more likely the current anomalies are caused by concept drift, and the more likely these anomalies are false anomalies. Second, word frequency statistics in the current data are adopted to construct the frequency matrix **DM**. The difference sd between matrices **DM** and **HM** is calculated, where **HM** is the frequency matrix of historical data. The larger the sd is, the more likely concept drift occurs. Last, a and b provided by users are adopted to compute the probability of concept drift as follows:

$$\text{cpd}(D) = a \cdot \text{pd} + b \cdot \text{sd}. \tag{9}$$

If cpd is larger than a threshold j, these anomalies are added to the anomaly buffer AB to construct new models; otherwise, they are determined as true anomalies. P_t and **DM** are adopted to update P and **HM**, respectively.

3.4.2 Model dynamic adjustment

Because of the timeliness of data stream, a model dynamic adjustment based on the time decay function is proposed to ensure the speed of detection. When the anomaly buffer is full, a new model $M_i{}'$ is constructed by these anomalies and is added to the model set M. Meanwhile, according to the latest detection result, the time decay function dynamically adjusts the weight of models in M to keep the number of models in a stable range. This method addresses that the problem of efficiency decreases in detection when too many models exist in the model set.

Algorithm 3 shows the details of the model dynamic adjustment algorithm. First, ABMDA constructs a new model $M_i{}'$ from the anomaly buffer and adds $M_i{}'$ to model set M, the weight of which is set to 1. The time decay function $W(t) = \exp(-t)$ updates weights of all models in model set M as t changes (steps 1–6 in Algorithm 3). Second, these models with new weights are adopted to detect an anomalous sequence in the anomaly buffer again. If these sequences are determined as anomalies again, then they are added to the anomaly dataset OD (steps 7–12 in Algorithm 3). Third, if the detection result of model $M_i{}'$ is the same as that of the model set M, the weight of this model $M_i{}'$ is reset to 1; otherwise, the weight of this model is updated by the time decay function. When $M_i{}'$.weight is smaller than a threshold t, the model is deleted from the model set M (steps 13–21 in Algorithm 3).

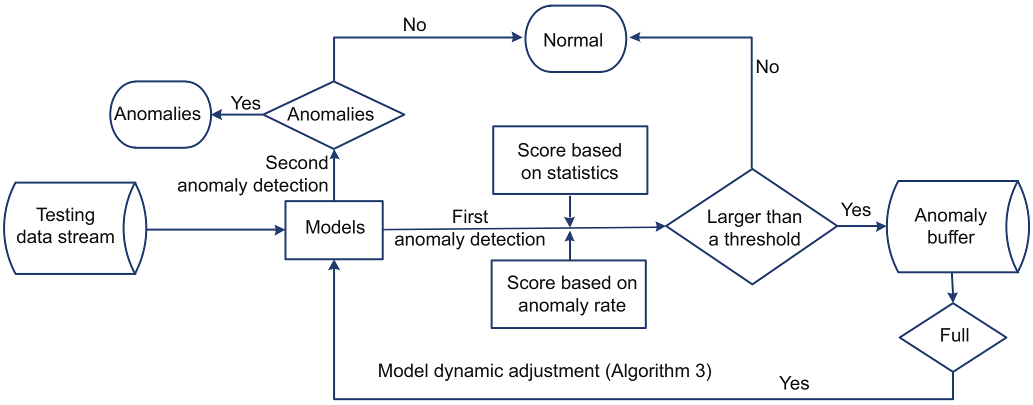


Fig. 7 Flow chart of ABMDA

Algorithm 3 Model dynamic adjustment based on time decay function

```
Input: Anomalous sequences in anomaly buffer OB, threshold $t$, model set $M$
Output: Anomaly dataset OD
Use OB to construct new model $M_i{}'$
Let the weight of $M_i{}'$ be 1 (that is, $M_i{}'$.weight = 1)
Add model $M_i{}'$ to the model set $M$
for each model $M_i{}'$ in the model set $M$ do
    Use time decay function to update the weight of each model
endfor
for each sequence OB′ in OB do
    Use the model set $M$ to detect sequence OB′
    if OB′ is an anomaly then
        Add OB′ in anomaly dataset OD
    endif
endfor
for each model $M_i$ in model set $M$ now do
    if $M_i$.result == $M$.result then
        Reset $M_i$.weight = 1
    else if $M_i$.weight < $t$ then
        Delete model $M_i$ from $M$
    else
        Update the weight of $M_i$ by time decay function
    endif
endfor
Return anomaly dataset OD
```

4 Experiments and results

We designed and implemented the prototype of FAAD on the platform Storm to detect anomalies for a multi-dimensional sequence over the data stream. We first adopted the Arcene and Dorothea datasets to test the performance of feature selection, and compared it with the traditional methods, i.e., fast correlation based filter (FCBF) (Yu and Liu, 2003) and correlation based feature selection (CFS) (Hall, 2000). Second, we used a single-dimensional synthetic data of Unix user behavior from Purdue University to evaluate the performance of model construction in complexity, time, and detection rate. Compared with the traditional method PST with different parameters, our construction method performs better. Third, we adopted the Darpa 99 dataset to verify that our FAAD method can effectively detect anomalies for multi-dimensional sequences compared with several existing methods. Last, we adopted some users' data in the Darpa 99 dataset to simulate concept drift and measured the performance of FAAD when concept drift occurs with different parameters.

4.1 Dataset description

4.1.1 Arcene and Dorothea datasets

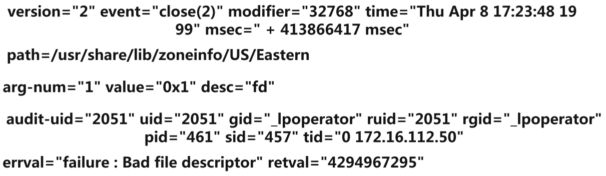
We adopted the Arcene and Dorothea datasets that are publicly available from the University of California Irvine. Each of these two datasets includes 10 000 features. Thus, they are regarded as the benchmark dataset to select representative features. The Arcene dataset consists of 900 samples, including 398 positive samples and 502 negative samples. The Dorothea dataset consists of 1950 samples, including 190 positive samples and 1760 negative samples.

4.1.2 Unix user behavior dataset

Lane (1998) from Purdue University used single-dimensional shell commands on the Unix platform as audit data to detect anomaly user behavior. We adopted the synthetic data of the Unix user behavior as the experimental data to test the complexity of detection models. RSIPST uses the hidden Markov model (HMM) to generate the synthetic data with 100 000 sequences whose lengths are between 150 and 200 (Chandola et al., 2008).

4.1.3 Darpa 99 dataset

The Darpa 99 dataset consists of daily system logs (basic security mode) containing all system calls performed by all processes over a seven-week period. Each of them consists of tokens that represent system calls using the syntax exemplified in Fig. 8.



```
version="2" event="close(2)" modifier="32768" time="Thu Apr 8 17:23:48 19
                    99" msec=" + 413866417 msec"
path=/usr/share/lib/zoneinfo/US/Eastern
arg-num="1" value="0x1" desc="fd"
audit-uid="2051" uid="2051" gid="_lpoperator" ruid="2051" rgid="_lpoperator"
                    pid="461" sid="457" tid="0 172.16.112.50"
errval="failure : Bad file descriptor" retval="4294967295"
```

Fig. 8 A sample system call record from the Darpa 99 dataset

The system call record consists of several features. Seven arguments were extracted as features (Table 2). To simplify the expression, numbers 0–6 were adopted to represent the above seven features. The dataset consists of 200 000 logs and the anomaly logs take up 10%.

Table 2 A sample record of the extracted features

Call	Desc	Return value	Path	Call time	Run time (ms)	Arg
Open	fd	Success	/root/	8:01:11	41 392 013	2

Arg: argument number

4.2 Performance of feature selection

In this subsection, to verify that IMC can effectively select features, we compared it with two traditional feature selection methods, FCBF (Yu and Liu, 2003) and CFS (Hall, 2000), and then tested the representativeness of these selected features with classification algorithms naive Bayes (NB), decision tree C4.5, and SVM. FCBF is a fast filter method which can identify relevant features and redundancy among relevant features without pairwise correlation analysis. CFS mines the features that have a high correlation with the class as representative features. We adopted the Arcene and Dorothea datasets to carry out our experiments. Since the number of clusters of IMC needs to be assigned by the user, while FCBF and CFS automatically determine the number of clusters according to the information between features, we set the proportion of selected features among all features from 1% to 25%.

Fig. 9 shows the correction classification rate (CCR) on the NB, decision tree C4.5, and SVM classifiers. We can see from Fig. 9a, when the proportion is about 1%, the CCR of CFS is higher than those of FCBF and IMC. Since CFS is more likely to select features that have a high correlation with the cluster, it can obtain more representative features than the other two methods when the proportion is 1%. However, with the growth of the proportion of selected features, the CCR of selected features from CFS increases slowly. This is because redundant information is not considered by CFS and the proportion of redundant features increases; thus, these features slightly improve only the CCR.

In Figs. 9a–9c, when the proportion is 10%–15%, the CCR of IMC is significantly higher than that when the proportion is 5%. This is because when IMC selects features, it will delete the redundant information. When the proportion is 10%–15%, it can select the best features to represent the most information. When the proportion continues to grow, the CCR does not dramatically grow further. Moreover, the CCR of FCBF gradually increases with the growth of proportion. It means that FCBF enlarges the coverage of features to improve the CCR. However, more selected features will lead to a longer time on model construction and detection.

Consequently, compared with FCBF and CFS, most IMC achieves the highest CCR on NB, decision tree C4.5, and SVM classifiers without selecting too many features.

4.3 Performance of model construction

Since we used every selected feature in the training data to construct every detection model M_i, we adopted the single-dimensional Unix user behavior dataset to evaluate the performance of RSIPST in model construction. Ten thousand sequences were selected as training data and others as test data. Anomalies took up 5% of the whole data. First, we compared RSIPST with the PST method on complexity, running time, and detection rate. Second, since RSIPST can choose sample rate and

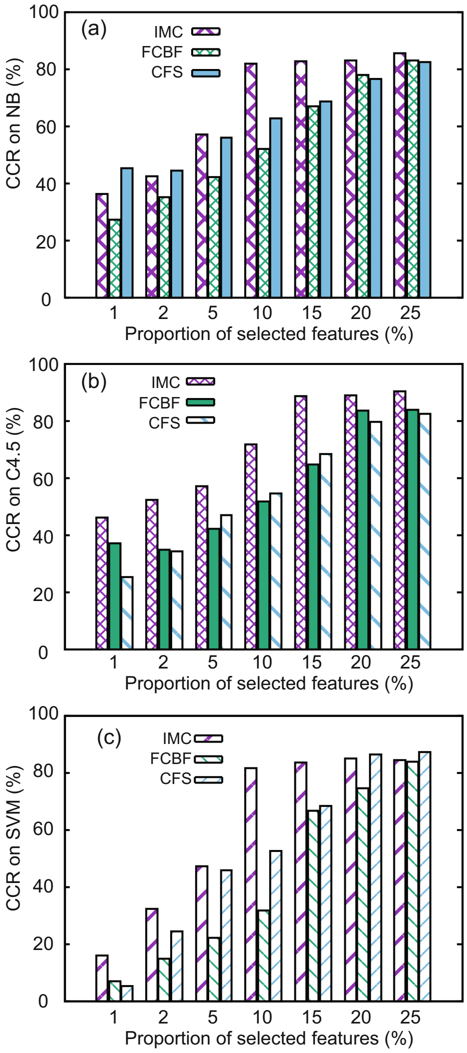


Fig. 9 Correction classification rate (CCR) on different classification algorithms based on the Arcene and Dorothea datasets: (a) naive Bayes; (b) C4.5; (c) SVM

IMC: information calculation and minimum spanning tree cluster; FCBF: fast correlation based filter; CFS: correlation based feature selection

subsequence length, to analyze the relationship between them, we constructed the RSIPST on the Unix user behavior data with different arguments.

We estimated the complexity of our model by the number of nodes in RSIPST. The forest scale marked as TreeNum is the number of repeated times in sequence modeling. The tree depth is the layer of PST. As Fig. 10a shows, when TreeNum is 4, 6, and 8, the number of nodes in RSIPST is smaller than that of the traditional PST. Only when TreeNum is 10, is RSIPST more complex than PST, as the index structure reduces the complexity of model construction when RSIPST repeats fewer times.

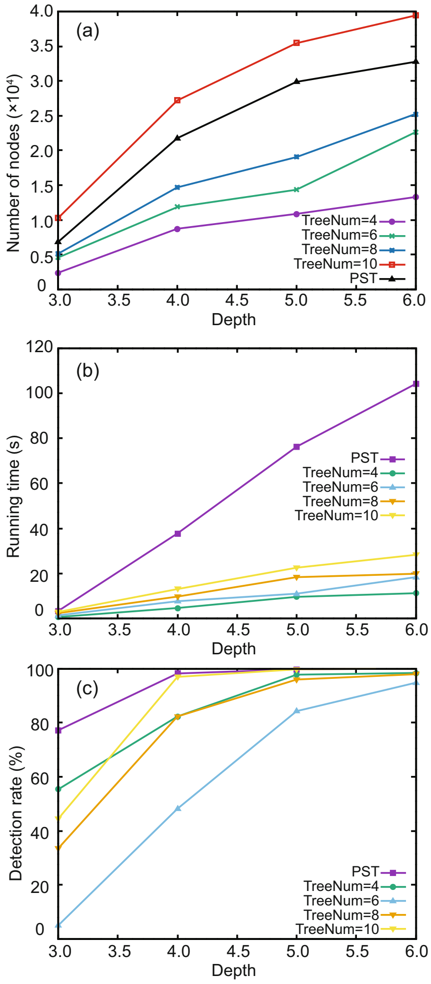


Fig. 10 Effects of forest scales and traditional probabilistic suffix tree (PST) on the number of nodes (a), running time (b), and detection rate (c) based on the Unix user behavior dataset

As we can see from Fig. 10b, since the index structure was adopted in RSIPST, even if TreeNum=10 has a more complex model, the construction time of RSIPST is far less than that of PST. Moreover, as the model is deeper, the gap between the traditional PST and RSIPST is clearly larger. RSIPST has a stable trend, while the time of PST dramatically increases. Fig. 10c shows the detection rate on different depths. The detection rate increases with the growth of the number and the depth of models. When the scale is 10 and the depth is four, the result of RSIPST can reach the top value and RSIPST behaves better than the traditional method PST at depth four. This is because the subsequence partitioning in RSIPST can reduce the complexity of the sequence, and then RSIPST can mine the pattern of long sequences in models by a smaller tree depth. However, the traditional PST method without subsequence partitioning must improve the anomaly detection rate by deepening models. In our experiments, we set eight as the tree number because of its good detection rate, lower complexity, and suitable construction time.

Table 3 shows the effects of the sample rate, subsequence length, and state number on running time, detection rate, and false positive rate. In these experiments, when the sample rate r and subsequence length l satisfy $r \cdot l = 4$, RSIPST always exhibits a good detection performance and a low false positive rate. Furthermore, with the growth of the sample rate, the anomaly detection rate can be improved by increasing the length of subsequence, which leads to more time on detection.

4.4 Anomaly detection

In this subsection, we used the the Darpa 99 dataset to test the efficiency of FAAD on anomaly detection for a multi-dimensional sequence over data stream without concept drift (We will discuss the experiments on concept drift in the next subsection). The user 2103 was randomly selected, and regarded as a normal user to construct a behavior model as training data. Then we used other users' data as the testing data stream. Six users were selected in the testing data as abnormal users.

First, we carried out experiments to select the representative features from the Darpa 99 dataset. In the step of feature selection, FAAD selects feature subsets {0}, {0, 5}, and {0, 1, 6} when the number of clusters k is set to 1, 2, and 3, respectively. Fig. 11 shows the changes of the detection rate under these selected feature subsets with different users. As we can see from the figure, when $k = 2$, the feature subset {0, 5} has a higher detection rate in all users than the other feature subsets. In addition, Fig. 12 shows the comparison of the detection rate and time on feature subset {0, 5} with {0} and {0, 3, 5}. As Fig. 12 shows, the more features are selected to construct models, the higher detection rate and the longer detection time they bring. This is because more features can completely represent the whole dataset. This improves the detection rate and leads to the increase of detection time. In Fig. 12, feature 3 is added to the feature subset {0, 5}. It brings a little improvement on detection rate, but a huge increase in running time. Consequently, we took $k = 2$ and feature subset {0, 5} as a trade-off between detection time and detection rate in our later experiments.

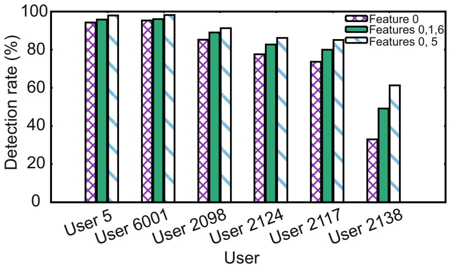


Fig. 11 Effects of different selected feature subsets on detection rate based on the Darpa 99 dataset

Second, since we have tested the efficiency of model construction, we compared our anomaly detection method FAAD with the SVM method (Parveen et al., 2013) and the pivotal method (Tandon and Chan, 2003). The SVM method is an effective classification method which is widely applied in anomaly detection. The pivotal method is an effective multi-dimensional anomaly detection method for a sequence which has a high detection rate in the Darpa 99 dataset. We extracted feature subset {0, 5} from the testing data stream to detect anomalies

Table 3 Effects of different arguments on running time, detection rate, and false positive rate based on the Unix user behavior dataset

Sample rate	Subsequence length	Detection rate (%)	False positive rate (%)	Running time (s)	State number
0.1	10	85.88	1.227	7.429	20
0.2	10	85.74	0.48	9.128	20
0.2	20	99.15	0.74	18.119	20
0.4	10	97.86	1.30	13.172	20
0.4	20	99.23	4.48	27.557	20
0.2	20	98.30	1.38	24.00	40
0.4	10	91.84	1.64	16.248	40
0.2	20	96.46	1.98	34.724	60
0.4	10	90.513	2.40	24.68	60

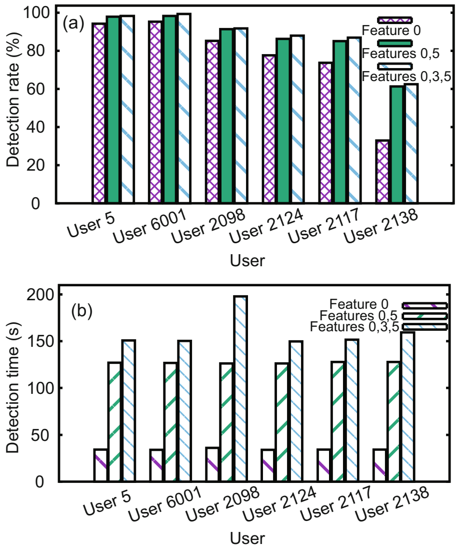


Fig. 12 Effects of feature subsets {0}, {0, 5}, and {0, 3, 5} on detection rate (a) and detection time (b) based on the Darpa 99 dataset

on these two features. We chose the system call (feature 0) as the pivotal feature and the other features as supplementary features. Then we analyzed the sequenced relationship of pivotal feature and used point anomaly detection methods to analyze the supplementary features. Fig. 13 shows a comparison of our method with the pivotal method and the SVM-based method. As the results show, our method has the best detection rate. Furthermore, the false positive rates of our method, pivotal method, and SVM method are about 3%–5%, 6%, and 12%, respectively.

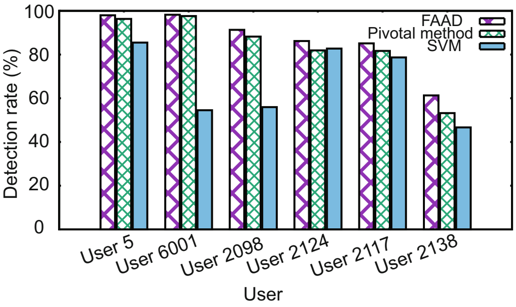


Fig. 13 Effects of different anomaly detection methods on detection rate based on the Darpa 99 dataset

4.5 Concept drift

Currently, most anomaly detection methods for a multi-dimensional sequence over the data stream do not focus on concept drift detection, except that Bao and Wang (2016) provided a concept drift detection method based on its outcome of SVM. However, SVM is a supervised learning method for labeled data, and it is not suitable for the unlabeled data in our study. Additionally, Bao and Wang (2016)'s idea on concept drift detection is not universal. It is suitable for only the SVM detection method and cannot be combined with other methods. Consequently, in our experiments on concept drift detection, we just discussed different settings of the proposed approach.

In this subsection, to verify the performance of FAAD on anomaly detection when concept drift occurs, we randomly selected three users' data to simulate the occurrence of concept drift. The first user's data were taken as normal and training data, the second were taken as the concept drifted data, and the third were taken as the abnormal data. The last two users' data were the testing data in these sections.

First, we discussed changes in the detection rate with different sizes of anomaly buffer. Second, we measured effects of proportions of weights on the oncept drift detection rate. Third, contrast experiments between FAAD and FAAD-without decay function (FAAD-WDF) have been carried out to show effects of the time decay function on the complexity of detection. Last, we measured the changes of the false negative rate with the arriving data stream. We concluded that FAAD can produce a high anomaly detection rate and speed when concept drift occurs.

First, we showed changes of the detection rate with different sizes of anomaly buffer in FAAD. Fig. 14 shows that the anomaly detection rate increases with the growth of buffer size. When the size of the buffer was set to 0, the concept drift detection rate was very low since it does not have new models constructed by anomalies in the buffer. As the buffer size increased, FAAD can find more anomalies caused by concept drift. However, when the buffer size was larger than 700, the detection rate remained at the level about 81%.

Second, we evaluated effects of different weights on the anomaly detection rate when the buffer size was 700. Proportion P is a/b (a is the weight of the method based on statistics, and b is the weight of the method based on the detection rate). As Table 4 shows, the anomaly detection rate gradually increases with the decrement of proportion

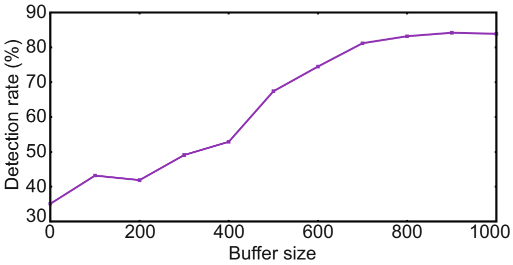


Fig. 14 Effects of buffer size on detection rate based on the Darpa 99 dataset

P. When the proportion $P = 10$, the detection rate reaches the highest value at 81.256%. After that, the detection rate slightly decreases and remains at about 74%. Consequently, a hybrid method based on statistics and the anomaly detection rate can detect anomalies with a higher detection rate than each method which detects anomalies individually.

Table 4 Effects of different proportions of weights on detection rate based on the Darpa 99 dataset

Proportion P	Detection rate (%)
100	45.214
50	59.486
10	81.256
1	76.529
0.1	74.983
0.02	73.438
0.01	75.216

Third, we compared FAAD with FAAD-WDF to verify that the time decay function can limit the number of models. The size of buffer and proportion P in this experiment were set to 700 and 10, respectively. Fig. 15 shows changes in the number of models in FAAD and FAAD-WDF. As we can see from the figure, since data distribution changes with the arrival of new sequences, FAAD can keep the number of models in a constant range of about 20, while the number of models in FAAD-WDF gradually increases. Increasing the number of models makes the detection more complex and leads to a longer detection time.

Last, Fig. 16 shows the changes of the false negative rate. With new sequences arriving, the false negative rate caused by concept drift dramatically increases since the history model cannot effectively detect anomalies with changes of data distribution. When the 6^{th} batch of the sequence arrives, the anomaly buffer is full, and new models are constructed to detect true anomalies. The false negative rate drops from 94% to 4.4%, which shows that FAAD can ensure a low false negative rate.

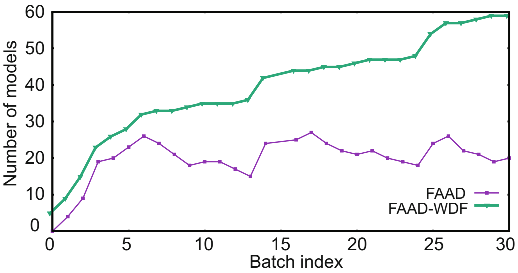


Fig. 15 Numbers of models in FAAD and FAAD-WDF based on the Darpa 99 dataset

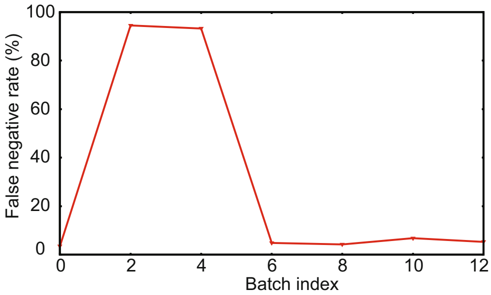


Fig. 16 Effects of ABMDA on false negative rate based on the Darpa 99 dataset

5 Conclusions and future work

In this study, we have provided an unsupervised fast and accurate anomaly detection (FAAD) method for a multi-dimensional sequence over the data stream. FAAD focuses on the multi-dimensional sequence over the data stream and addresses new challenges. It uses IMC, RSIPST, and ABMDA to reduce redundant dimensionality, speed up model construction, and reduce the effects of concept drift in the data stream. Compared with existing methods, our analytical and experimental results demonstrated that FAAD can adapt to a multi-dimensional sequence over the data stream and perform effectively in anomaly detection. Moreover, FAAD can reduce the false negative rate caused by concept drift without adding complexity of our models.

In future work, we are planning to apply this method of mining anomalies to other domains, such as remote sensing and program analysis.

References

Bao H, Wang YJ, 2016. A C-SVM based anomaly detection method for multi-dimensional sequence over data

stream. Proc IEEE 22nd Int Conf on Parallel and Distributed Systems, p.948-955.
https://doi.org/10.1109/ICPADS.2016.0127

Box GE, Jenkins GM, Reinsel GC, et al., 2015. Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken, USA.

Budalakoti S, Srivastava AN, Akella R, et al., 2006. Anomaly Detection in Large Sets of High-Dimensional Symbol Sequences. TM-2006-214553, NASA Ames Research Center, USA.

Budalakoti S, Srivastava AN, Otey ME, 2009. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *IEEE Trans Syst Man Cybern C*, 39(1):101-113.
https://doi.org/10.1109/TSMCC.2008.2007248

Carlin BP, Louis TA, 2000. Bayes and Empirical Bayes Methods for Data Analysis (2nd Ed.). Chapman & Hall/CRC Press, Boca Raton, FL, USA.

Chandola V, Mithal V, Kumar V, 2008. Comparative evaluation of anomaly detection techniques for sequence data. Proc 8th IEEE Int Conf on Data Mining, p.743-748.
https://doi.org/10.1109/ICDM.2008.151

Chandola V, Banerjee A, Kumar V, 2009. Anomaly detection: a survey. *ACM Comput Surv*, 41(3), Article 15.
https://doi.org/10.1145/1541880.1541882

Chandola V, Banerjee A, Kumar V, 2012. Anomaly detection for discrete sequences: a survey. *IEEE Trans Knowl Data Eng*, 24(5):823-839.
https://doi.org/10.1109/TKDE.2010.235

Dani MC, Freixo C, Jollois FX, et al., 2015. Unsupervised anomaly detection for aircraft condition monitoring system. Proc IEEE Aerospace Conf, p.1-7.
https://doi.org/10.1109/AERO.2015.7119138

Esposito F, di Mauro N, Basile TMA, et al., 2008. Multi-dimensional relational sequence mining. *Fundam Inform*, 89(1):23-43.

Hall MA, 2000. Correlation-based feature selection for discrete and numeric class machine learning. Proc 17th Int Conf on Machine Learning, p.359-366.

Jin Y, Zuo WL, 2007. Multi-dimensional concept lattice and incremental discovery of multi-dimensional sequential patterns. *J Comput Res Dev*, 44(11):1816-1824 (in Chinese).

Kaufman L, Rousseeuw PJ, 2009. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, New York, USA.

Keogh E, Chakrabarti K, Pazzani M, et al., 2001. Dimensionality reduction for fast similarity search in large time series databases. *Knowl Inform Syst*, 3(3):263-286.
https://doi.org/10.1007/PL00011669

Kponyo JJ, Kuang YJ, Zhang EZ, et al., 2013. VANET cluster-on-demand minimum spanning tree (MST) prim clustering algorithm. Proc Int Conf on Computational Problem-Solving, p.101-104.
https://doi.org/10.1109/ICCPS.2013.6893585

Lane T, 1998. Machine Learning Techniques for the Domain of Anomaly Detection for Computer Security. Purdue University, Indiana, USA.

Lee CH, 2015. A multi-phase approach for classifying multi-dimensional sequence data. *Intell Data Anal*, 19(3):547-561. https://doi.org/10.3233/IDA-150731

Li C, Tian XG, Xiao X, et al., 2012. Anomaly detection of user behavior based on shell commands and co-occurrence matrix. *J Comput Res Dev*, 49(9):1982-1990 (in Chinese).

Li XY, Wang YJ, Li XL, et al., 2014. Parallelizing skyline queries over uncertain data streams with sliding window partitioning and grid index. *Knowl Inform Syst*, 41(2):277-309.
https://doi.org/10.1007/s10115-013-0725-8

Parveen P, Mcdaniel N, Weger Z, et al., 2013. Evolving insider threat detection stream mining perspective. *Int J Artif Intell Tools*, 22(5):1360013.
https://doi.org/10.1142/S0218213013600130

Qian Q, Wu JL, Zhu W, et al., 2012. Improved edit distance method for system call anomaly detection. Proc IEEE 12th Int Conf on Computer and Information Technology, p.1097-1102. https://doi.org/10.1109/CIT.2012.223

Ron DN, Singer Y, Tishby N, 1994. Learning probabilistic automata with variable memory length. Proc 7th Annual Conf on Computational Learning Theory, p.35-46.
https://doi.org/10.1145/180139.181006

Sarhrouni E, Hammouch A, Aboutajdine D, 2012. Application of symmetric uncertainty and mutual information to dimensionality reduction and classification of hyperspectral images. *Int J Eng Technol*, 4(5):268-276.
https://doi.org/10.1145/180139.181006

Shu XK, Yao DF, Ryder BG, 2015. A formal framework for program anomaly detection. Proc 18th Int Symp Research in Attacks, Intrusions, and Defenses, p.270-292. https://doi.org/10.1007/978-3-319-26362-5_13

Tandon G, Chan P, 2003. Learning rules from system call arguments and sequences for anomaly detection. Proc ICDM Workshop on Data Mining for Computer Security, p.20-29.

Wang Y, Ma X, 2015. A general scalable and elastic content-based publish/subscribe service. *IEEE Trans Parall Distr Syst*, 26(8):2100-2113.
https://doi.org/10.1109/TPDS.2014.2346759

Wang YJ, Li S, 2006. Research and performance evaluation of data replication technology in distributed storage systems. *Comput Math Appl*, 51(11):1625-1632.
https://doi.org/10.1016/j.camwa.2006.05.002

Wang YJ, Li XY, Li XL, et al., 2013. A survey of queries over uncertain data. *Knowl Inform Syst*, 37(3):485-530.
https://doi.org/10.1007/s10115-013-0638-6

Wang YJ, Pei X, Ma X, et al., 2018. TA-update: an adaptive update scheme with tree-structured transmission in erasure-coded storage systems. *IEEE Trans Parall Distr Syst*, 29(8):1893-1906.
https://doi.org/10.1109/TPDS.2017.2717981

Xianyu JC, Rasouli S, Timmermans H, 2017. Analysis of variability in multi-day GPS imputed activity-travel diaries using multi-dimensional sequence alignment and panel effects regression models. *Transportation*, 44(3):533-553.
https://doi.org/10.1007/s11116-015-9666-2

Xiong TK, Wang SR, Jiang QS, et al., 2011. A new Markov model for clustering categorical sequences. Proc IEEE 11th Int Conf on Data Mining, p.854-863.
https://doi.org/10.1109/ICDM.2011.13

Yamanishi K, Maruyama Y, 2005. Dynamic syslog mining for network failure monitoring. Proc 11th ACM SIGKDD

Int Conf on Knowledge Discovery in Data Mining, p.499-508. https://doi.org/10.1145/1081870.1081927

Yang J, Wang W, 2003. CLUSEQ: efficient and effective sequence clustering. Proc 19th Int Conf on Data Engineering, p.101-112. https://doi.org/10.1109/ICDE.2003.1260785

Yu L, Liu H, 2003. Feature selection for high-dimensional data: a fast correlation-based filter solution. Proc 20th Int Conf on Machine Learning, p.856-863.

Appendix A: Concept drift detection based on detection rate

By the definition of anomalies, the data distribution of anomalies is different from that of the most data and the data size is far smaller than normal. Consequently, for a data stream D in a stable distribution ϕ, the proportion of anomalies is always low, and the average and variance of detection rate are stable.

The data are generated in the ϕ distribution and the detection result is the random variable Z. $Z =$ false shows that the result of detection is false; otherwise, $Z =$ true. After n random experiments, $T_D(Zf)$ is supposed to represent the proportion of false detection results. The random variable $T_D(Zf)$ obeys the binomial distribution, where parameters are n and p. Consequently, based on the central-limit theorem (the number of samples is large enough $(n \geq 30)$), the binomial distribution is satisfied with the normal distribution, where average $\mu = p$ and variance $\sigma = \sqrt{p(1-p)/n}$.

Based on the reasons mentioned above, we can compute the difference of the proportion of historical anomalies and current anomalies. Let the proportion of anomalies in each time period be P_i and the proportion of current anomalies be P_t. Thus, the proportion of historical anomalies obeys the normal distribution $P_i \sim N(\mu, \sigma^2)$, where $\mu = \frac{1}{t}\sum\limits_{i=1}^{t} P_i$ and $\sigma^2 = \frac{1}{t}\sum\limits_{i=1}^{t}(\mu - P_i)^2$. The cumulative distribution function of the proportion of historical anomalies is $\phi(x) = [1 + \text{erf}(x/\sqrt{2})]/2$, where erf() is the error function (also called the "Gauss error function"). Eq. (A1) is adopted as the standard of the proportion of current anomalies and proportion of historical anomalies. The larger P_t deviates from the historical value, the larger $F(P_t)$ is, which shows that the data distribution has changed and the concept drift occurs.

$$F(x) = \frac{1}{1-\phi(x)}. \tag{A1}$$

Appendix B: Concept drift detection based on statistics

Concept drift occurs when the data distribution changes. Statistics are used to express the data distribution and detect whether concept drift happens by comparing the statistical information of historical data with current data. In this step, we face two challenges in detecting the data distribution for a sequential data stream: (1) Since data are infinite in the data stream, the historical data cannot be completely stored; (2) The statistical information cannot be directly extracted from the data stream due to its pattern and features. To solve these problems, we calculate the difference between traditional data and current data. Matrix $\boldsymbol{M}$ is the word frequency statistics of sequential data. $M_{i,j}$ represents the frequency of the j^{th} value of the i^{th} attribute. The average of frequency is adopted as the approximate attribute value since the complete data cannot be stored. The difference of **HM** and **DM** evaluates whether the concept drift occurs. Let $A_{i,j} = \text{HM}_{i,j} - \text{DM}_{i,j}$, and the 1-norm of matrix $\boldsymbol{A}$ is $\|\boldsymbol{A}\|_1 = \max\limits_{j}\sum\limits_{i=1}^{m}|A_{i,j}|$. Consequently, the difference of **HM** and **DM** is shown as Eq. (B1). $G(\mathbf{HM}, \mathbf{DM})$ measures the gap between the historical frequency matrix and the current frequency matrix. The larger G is, the more likely the concept drift occurs.

$$G(\mathbf{HM}, \mathbf{DM}) = \max_{j}\sum_{i=1}^{m}\left|\text{HM}_{i,j} - \text{DM}_{i,j}\right|. \tag{B1}$$