Adaptive intrusion detection via GA-GOGMM-based pattern learning with fuzzy rough set-based attribute selection

Jinping Liu\textsuperscript{1,a}, Wuxia Zhang\textsuperscript{a}, Zhaohui Tang\textsuperscript{c}, Yongfang Xie\textsuperscript{c}, Tianyu Ma\textsuperscript{b}, Jingjing Zhang\textsuperscript{a}, Guoyong Zhang\textsuperscript{c}, Jean Paul Niyoyita\textsuperscript{d}

\textsuperscript{a} Hunan Provincial Key Laboratory of Intelligent Computing and Language Information Processing, Hunan Normal University, Changsha, Hunan 410081, China
\textsuperscript{b} College of Physics and Electronics, Hunan Normal University, Changsha, Hunan 410081, China
\textsuperscript{c} School of Automation, Central South University, Changsha, Hunan 410083, China
\textsuperscript{d} College of Science and Technology, University of Rwanda, Kigali 3286, Rwanda

**A R T I C L E   I N F O**

Article history:
Received 3 May 2019
Revised 9 July 2019
Accepted 25 July 2019
Available online 26 July 2019

Keywords:
Intrusion detection system
Gaussian mixture model
Greedy algorithm
Fuzzy rough set
Information gain ratio
Pattern learning

**A B S T R A C T**

In this paper, an adaptive network intrusion detection method using fuzzy rough set-based feature selection and GA-GOGMM-based pattern learning is presented. Based on the fuzzy rough set theory, the optimal attribute subset of network connection records is achieved by the information gain ratio criterion in advance. A greedy algorithm-based global optimal Gaussian mixture model (GMM) clustering method, termed GA-GOGMM, is introduced, to extract the intrinsic structure of network instances to achieve highly-discriminable and stable normal and intrusion pattern libraries for the subsequent network intrusion detection (NID). GA-GOGMM-based pattern learning can achieve the optimal GMM of network traffic instances for the pattern clustering while avoiding the negative effect of the empirical initialization of clustering numbers and random initialization of clustering centers with a low computational complexity. An adaptive model updating mechanism is further introduced for the online updating of normal and intrusion pattern libraries to ensure the adaptability of the NID model. Extensive validation and comparative experiments, conducted on a benchmark dataset NSL-KDD and a self-built Nidshench-based network simulation platform, show that the proposed ANID approach leads to a significant improvement in detection accuracies with low false alarms and missing reports on both known and unknown attacks. It can effectively adapt to the dynamic changing network environments with high detection accuracy and low false alarm rate as well as low missing reporting rate, which has significant application prospects.

© 2019 Elsevier Ltd. All rights reserved.

1. Introduction

With the worldwide spread and fast development of the Internet, computer and network have pervaded almost all aspects of our daily life. Especially in the past ten years, dramatically increasing network systems have been established for online services, e.g., online shopping, Internet (mobile) banking, online education, online social interactions and chatting, online stock trading and many other online business transactions, and so on.

However, network is a double-edged sword. Netizen are becoming vulnerable to a wide variety of cyber threats increasingly due to the open property and easy accessibility of free hack tools widespread over the Internet (Wang & Yang, 2018). Recent years, serious network attacking events can be found in the newspaper frequently, e.g., the recently-reported data leak scandal of Facebook, that more than 50 million Facebook users were inappropriately used by a British data analysis company in activities allegedly connected with U.S. President Donald Trump during his presidential campaign; the cyberattack of the US nuclear power plants reported on March 2018, and many others, to mention but a few (Liu et al., 2019).

The extensive growth of cyberthreats in the open Internet has prompted network intrusion detection (NID), a proactive cybersecurity protection technology, to become a critical component of protection infrastructure (Wei, Lee, Jeng, Lamba & Faloutsos, 2019). NID systems attempt to identify malicious activities, e.g., unauthorized uses, misuse and abuse of computer systems by both insider users and external intruders of network systems (Wang & Yang, 2018).

Traditional NID technologies can be categorized into signature-based and anomaly-based technologies. Signature-based NID, or
mise detection, compares the patterns of network activities with the signatures of known attacks to perform attack alarms if a match is found. Misuse detection usually has a high detection rate and low false alarm rate for the known attacks, but it is incapable of detecting unforeseen attacks. Anomaly-based technologies detect deviations as malicious from the learned normal patterns based on representative normal instances (Karami, 2018), which has the merit of high detection rates of novel intrusions but is prone to generate high false alarm rate, especially under the recent rapidly changing and heterogeneous network environment.

As the proliferation of the heterogeneous network systems, people can access the network by various devices, e.g., personal computer, smartphone, iPad, PDAs, with myriad network protocols, which prompts the network environment to become increasingly complex with dynamic changing property. Under dynamic and heterogeneous network environments, attacking modes and attackers’ manners are evolving and retrofitting constantly (Aparicio-Navarro, Kyriakopoulos, Yu, Parish & Chambers, 2017; Roshan, Miche, Akusok & Lendasse, 2018). Due to the unceasing emergence of unforeseen intrusions and the dynamically changing network environment, developing a NID system with high accuracy and low false alarm, is still a challenging task (Khammassi & Krichen, 2017).

It is well accepted that there is a certain similarity within the normal network connection instances or within the intrusion instances of a specific attacking manner, but there is a great difference between the intrusion instances and the normal network connection instances (Atli et al., 2018). Hence, intrusions can be determined by the method of pattern matching, i.e., measuring similarities between a network instance and the normal network patterns as well as attack patterns, learned by clustering algorithms. Based on this concept, an adaptive NID (ANID) method is introduced in this work based on the clustering-based pattern mining of normal and attack network instances.

Traditional clustering algorithms can be categorized into about five categories: partitioning-based, hierarchy-based, density-based, grid-based and model-based methods (Xu & Tian, 2015). K-means (Nguyen & Baets, 2019) is a classic partitioning-based clustering method. K-means algorithm with its variants, i.e., K-mediods (Xu & Tian, 2015), taking the sum of squared errors as the objective criterion to assess the clustering quality, is easy to implement, but it needs to determine the cluster number manually in advance and its clustering results are seriously influenced by the initial clustering centers.

Hierarchical clustering (Chen, Martínez, Chin & Tsui, 2018; Liu, He, Zhang, Xu & Tang, 2018) have the merit of easy discovering the hierarchical structure of different clusters. However, the selection of clustering levels depends on subjective knowledge and experience. In addition, hierarchical clustering has high computational complexity and it is sensitive to the singular value in the learning data.

Density-based clustering method, i.e., the well-known DBSCAN (Jahirabadkar & Kulkarni, 2014), does not require the number of categories and can find clusters of arbitrary shapes. However, this method is sensitive to the internal parameters, e.g., the density threshold and the adaptive density value used to rapidly converge to high-density regions.

Grid-based clustering method (Wu & Wilamowski, 2016) is highly efficient because its computational complexity is independent to the instance number of training data but dependent on instances in unit neighboring grid space. However, this method is also sensitive to the internal model parameter and has defects in dealing with high dimensional data or data with irregular distributions.

Model-based clustering method provides a model for each cluster, e.g., Gaussian mixture model (GMM), and then searches data sets that well fit each component model to perform data clustering. This method usually does not strictly categorize a data instance into a crisp class, but often expresses them in a probabilistic form. This method can usually achieve a better clustering effect (Liu et al., 2018). However, it also needs to provide the number of clusters in advance.

In summary, these off-the-shelf clustering approaches need to determine the number of clusters manually or should set an appropriate stop criterion based on the understanding of the internal structure of the learning data. Hence, the final clustering results depend heavily on the initialized parameters, i.e., the clustering number and the initialized clustering centers. Inappropriate initialized parameters will produce poor clustering results, which will further affect the generation of normal or attack pattern library and result in performance fluctuations of the NID system.

Since GMM can approximate any complex distribution model in theory, we adopt the GMM-based clustering method to learn the normal and attack network connection patterns. The expectation maximization (EM) algorithm provides a means of GMM parameter learning method with the guaranteed non decreasing log-likelihood of training set (Vlassis & Likas, 2002). However, theoretical evidence has demonstrated that traditional EM algorithm can be only guaranteed to converge to a locally optimal solution. One of the standard solutions to overcome this issue is the Monte-Carlo method, namely, executing the EM procedure with random initializations with adequate times and taking the resulting GMM model with the maximum likelihood on the training data set. Apparently, it is time-consuming and also has no guarantee to achieve a global optimal model.

More effective solutions are to incorporate an intelligent optimization algorithm in the EM training procedure to achieve an optimal GMM. Antonio, Francisco and Juan Manuel (2009) proposed a entropy-based EM algorithm, attempting to find the optimal number of mixing Gaussian components, Li and Barron (1999) have made a theoretical analysis and evidenced that it is possible to achieve a better mixed model by introducing a greedy searching paradigm. Then, Vlassis and Likas (2002) proposed a Greedy EM algorithm to achieve an incremental mixture density learning, which has the probability of achieving the global optimal model.

Inspired by Vlassis and Likas (2002) study, this paper presents a greedy algorithm (GA)-based global optimum GMM learning method, termed GA-GOGMM, to obtain highly-discernable and stable normal and intrusion patterns for the subsequent NID. GA-GOGMM can automatically determine the potentially global optimal clustering number and avoid the adverse effect of inappropriate initialization of cluster number and clustering centers in the pattern learning.

Abovementioned pattern learning methods are mainly aimed at processing network instances of low-dimensional feature space, which is prone to suffer the curse of dimension while processing high-dimensional network instances. Hence, network instances, having high dimensional features, pose a great challenge in the detection model training and online application (Raman, Kirthivasan & Sriram, 2017). In addition, the NID performance will be further exacerbated by the fact that many features are either irrelevant or redundant to other features, and even have noise, uncertainty and incompleteness (Liyanaraarachchi, Yang, Huang & Zhang, 2016).

High-dimensional network instances with redundancy and uncertainty make NID methods be intractable (Vijayanand, Devaraj & Kannapiran, 2018). Hence, it is necessary to make a preprocessing of the original highly-dimensional data. The well accepted preprocessing method is attribute reduction (Du & Hu, 2016), which can be implemented by feature projection or feature selection (Khammassi & Krichen, 2017).

The feature projection can be categorized into the linear method, e.g., principle component analysis (Ronao & Cho, 2016), factor analysis and the nonlinear mapping method, e.g., isometric
Feature mapping (Huang, Xu & Zuo, 2014), locally linear embedding (Roweis & Saul, 2000), embedding the original high-dimensional data into a low-dimensional manifold space. Either linear or non-linear methods have high computational complexity; hence most of them are unsuitable for the online NID.

Feature selection (FS) methods choose a subset of attributes by identifying and removing unnecessary, irrelevant and redundant attributes from the original attribute space. As addressed in the literature (Bolón-Canedo & Alonso-Betanzos, 2013), traditional FS methods can be categorized into two lines: filter and wrapper models (Crone & Kourentzes, 2010). Filter-based methods usually apply a statistical measure to assign a score to each feature, e.g., information theory, which are used to rank features and choose features one by one. Wrapper methods consider the selection of a representative subset of features as a searching problem. Generally, filter-based FS methods have low computational complexity though the latter may produce better results (Khansamak & Kirchen, 2017).

With respect to the redundancy and uncertainty as well as irrelevant information existing in network connection data, the rough set theory is an important mathematic tool for dealing with them. Rough set theory can achieve a subset of the whole attributes while preserving the discernible ability of original features without additional information. However, the classical rough set theory can only deal with nominal attributes (Tiwari, Shreevatsava, Som & Shukla, 2018), which has defects in dealing with network connection instances with most of attributes being continuous variables. Therefore, a fuzzy rough set (FRS) theory-based FS of network records is introduced to extract the optimal attribute subset of network connection data, inspired by Dai’s work (Dai & Xu, 2013).

In summary, this paper presents an ANID method based on the GA-GOGGM-based pattern learning with the FRS-based attribute selection. Differently, a FRS-based optimal attribute subset selection scheme, termed FRS-FS, based on an introduced information gain ratio (IGR) criterion is adopted in advance. Then, a greedy algorithm-based global optimal Gaussian mixture model learning method, termed GA-GOGGM, for the normal and attack network pattern learning is performed in the reduced attribute space. By combining FRS-FS and GA-GOGGM-based pattern learning, an ANID method with online model updating strategy is proposed and validated on a benchmark dataset NSL-KDD (Moustafa & Slay, 2015) and a Nidsbench-based network simulation platform.

The main contributions of this paper are summarized as follows:

1. The intrinsic and optimum attribute subset of network connection instances is automatically achieved based on the introduced IGR criterion, which can effectively improve the efficiency and stability of the NID system.
2. The optimum pattern representation of normal and attack network instances is achieved based on the GA-GOGGM clustering method, which can effectively avoid the negative influence of blindly-initialized cluster centers and achieve the optimum cluster number automatically.
3. An online NID model updating strategy is introduced based on the frequency pattern mining, which ensures the NID model to be adapted to the dynamically changing and retrofitting of attack modes and network environment.

The rest of the paper is organized as follows. Section 2 addresses the details of the proposed ANID method, including FRS-based optimum attribute subset selection and GA-GOGGM-based pattern learning with the online pattern updating scheme. Section 3 presents the validation and comparative results on the benchmark NSL-KDD dataset and on a self-built network simulation platform. Section 4 concludes the whole paper with possible further studies of this work.

2. Proposed ANID method

The proposed ANID system mainly includes four modules: (A) Network connection data preprocessing, including the optimum attribute subset selection and network connection pattern representation; (B) Pattern learning of normal and attack instances; (C) Pattern matching and online updating of the NID model and (D) Attack alarm. The schematic diagram of the proposed ANID system is shown in Fig. 1.

2.1. FRS-based optimum attribute subset selection

This section addresses the preliminaries of FRS theory followed by the introduced FRS-FS method.

A NID system can be represented as a four-tuple information system, $FS = (U, A, V, f)$, where $U$ denotes the data set consisting of the whole network connection instances, $A$ is a finite non-empty attribute set. The domain set is $V = U_x \cup V_o$, where $V_o$ is the value set of attribute $a$, also called domain of $a$: $U \times A \rightarrow V$ is an information function assigning specific values to objects from attribute domain. If we explicitly discriminate the conditional attributes and the decision attributes (label attribute, e.g., normal, DOS, Probe, etc.) as $C$ and $D$, respectively, a four-tuple information system can be expressed as,

$$FS = (U, C \cup D, V, f).$$

Since classic rough set theory can only deal with nominal attributes (Herawan, Deris & Abawajy, 2010; Jensen & Shen, 2009), which cannot be used directly in the NID system with mixed continuous and nominal attributes, a FRS-FS method is introduced for the optimum attribute subset selection.

2.1.1. Preliminaries of FRS

The fuzzy equivalence relation (FER) is the core of FRS, which can deal with the real-valued attributes rather than only the crisp rough set. Given a finite and non-empty set $X$ including $n$ instances, the FER $R$ on $X$, satisfying the properties of reflectivity, symmetry and transitivity, can be expressed as a fuzzy equivalence matrix $M \in \mathcal{X}^{n \times n}$.

It is nontrivial to generate a fuzzy equivalence matrix directly, which is usually achieved by calculating the transitive closure of a fuzzy similarity matrix $R \in \mathcal{X}^{n \times n}$, which satisfies the properties of reflectivity and symmetry and any element $r_{ij}$ in $R$ ranges in $[0, 1]$. Many approaches can be used to generate the fuzzy similarity matrix $R$, i.e., similarity coefficient, distance metrics. As stated in the literature (Dai & Xu, 2013), the fuzzy similarity coefficient of $x_i$ and $x_j$ can be calculated by the following methods,

$$r_{ij} = \max \left( \min \left( \frac{x_i - \sigma_a}{\sigma_a - \min(a)}, \frac{x_i + \sigma_a - x_j}{\min(a)}, 0 \right), 0 \right)$$

or

$$r_{ij} = \begin{cases} 1 - 4 \times \frac{|x_i - x_j|}{\max(a) - \min(a)} & |x_i - x_j| \leq 0.25 \\ 0, otherwise \end{cases}$$

where $x_i$, $x_j$ are the attribute values of two objects on the attribute $a$ with the maximal $\sigma_{max}$ and minimal $\sigma_{min}$ of all objects, and $\sigma_a$ is the standard deviation value of instances of attribute $a$.

Routinely, the transitive closure of a fuzzy relation matrix $R$ is $t(R) = \bigcup_{n=1}^{\infty} R^n$, which is inacculable since there is a union operation of infinite fuzzy matrices (Fu, 1992; Lee, 2001). In real applications, $M = t(R)$ can be simplified by the following theorems.
Theorem 1 (Bandler & Kohout, 1988; Fu, 1992; Lee, 2001). The transitive closure of a fuzzy matrix $R$ can be calculated as $\tau(R) = \bigcup_{n=1}^{\infty} R^n$. Theorem 1 gives a feasible computational approach of the transitive closure by a $n$ times union operation. And the computation can be further simplified by Theorem 2.

Theorem 2 (Fu, 1992). The transitive closure of a fuzzy similarity matrix $R$ is a Fuzzy equivalence matrix, namely, $M = \tau(R)$, where $\tau(R) = R^\infty$.

Proof. We need to prove that the transitive closure of a fuzzy similarity matrix $R$, $\tau(R) = R^\infty$, and prove that $\tau(R)$ has the property of reflexivity and symmetry.

Reflexivity. Since fuzzy similarity matrix $R$ is reflexive, namely, $I_n \subseteq R$, and $R \subseteq R^2 \subseteq \cdots \subseteq R^n$, then we achieve $\tau(R) = \bigcup_{n=1}^{\infty} R^n = R^\infty \supseteq I_n$.

Symmetry. Since the fuzzy similarity matrix $R$ is symmetric, namely, $R^T = R$, then $(R^\infty)^T = R^\infty = (R^T)^\infty$.

To summary, since $\tau(R) = R^\infty$ is reflexive, symmetric and transitive, $\tau(R)$ is a Fuzzy equivalence matrix.

Theorem 3 (Fu, 1992). Given a fuzzy similarity matrix $R$, there exists a minimum natural number $k (k \leq n)$, that the transitive closure $\tau(R) = R^k$ and for any natural number $k > k$, that $R^k = R^k$ and $\tau(R) = R^k$ is a fuzzy equivalence matrix, namely, $M = \tau(R) = R^k$.

Based on the Theorem 3, the fuzzy equivalence matrix $M$ can be calculated as follows. When the fuzzy similarity matrix $R$ is achieved, we can calculate the fuzzy square of the fuzzy similarity matrix successively, namely, $R \rightarrow R^2 \rightarrow R^3 \rightarrow \cdots \rightarrow R^n$, when the equation $R^k = R^k$ meets at the first time, $\tau(R) = R^k$ and $M = R^k$.

Some basic definitions regarding FRS-FS are explained as follows.

Definition 1 (Fuzzy partition). Given a fuzzy information system, $FIS = (U, C \cup D, V, f)$, the fuzzy partition of an universe set $U$ based on a FER $R$ is defined as

$$U/R = \{ [x_i]_R \}_{i=1}^{n} \text{s.t.} [x_i]_R = \frac{r_{i1}}{x_1} + \frac{r_{i2}}{x_2} + \cdots + \frac{r_{in}}{x_n}$$

$[x_i]_R$ is the fuzzy equivalence class generated by $x_i$ and a fuzzy equivalence relation $R$. Due to the FER $R$, $U/R$ is a fuzzy partition and $[x_i]_R$ is a fuzzy set.

Definition 2 (FRS-based Information Quantity Measure Dai & Xu, 2013). The information quantity based on the FER $R$ is defined as

$$H(R) = -\frac{1}{n} \sum_{i=1}^{n} \log \frac{|[x_i]_R|}{n}$$

where $|[x_i]_R| = \sum_{j=1}^{n} r_{ij}$ is the cardinality of $[x_i]_R$.

Definition 3 (Joint entropy). Given a fuzzy information system, $FIS = (U, C \cup D, V, f)$, and assume that $P$ and $Q$ are two attribute subsets of $C$, the joint entropy of $P$ and $Q$ is defined as

$$H(PQ) = H(R_P R_Q) = -\frac{1}{n} \sum_{i=1}^{n} \log \frac{|[x_i]_P \cap [x_i]_Q|}{n}$$

Definition 4 (Condition entropy Dai & Xu, 2013). In a fuzzy decision system, $FIS = (U, C \cup D, V, f)$, assume $B$ is a subset of $C$, the conditional entropy of the decision attribute $D$ conditioned on $B$ is defined as

$$H(D|B) = -\frac{1}{n} \sum_{i=1}^{n} \log \frac{|[x_i]_B \cap [x_i]_D|}{|[x_i]_B|}$$

Definition 5 (Mutual information). Given a fuzzy information system $FIS = (U, C \cup D, V, f)$ and assume that $B$ is a subset of $C$, the mutual information of $B$ and $D$ is

$$I(B; D) = H(D) - H(D|B).$$

Definition 6 (Information gain (IG) Dai & Xu, 2013). Given a fuzzy decision system $FIS = (U, C \cup D, V, f)$, let $B$ is the current chosen subset of $C$ and $a$ is a new independent attribute excluded in $B$, namely, $a \in C - B$, the IG of attribute $a$, $Gain(a, B, D)$, tentatively incorporating it into the chosen attribute subset, can be defined as

$$Gain(a, B, D) = I(B \cup \{a\}; D) - I(B; D).$$
Definition 7 (Information gain ratio (IGR)). The IGR of an attribute \(a\), \(\text{gain Ratio}(a, B, D)\), can be defined as
\[
\text{Gain Ratio}(a, B, D) = \frac{\text{Gain}(a, B, D)}{H(a)} = \frac{I(B \cup \{a\}; D) - I(B; D)}{H(a)}.
\]

2.1.2. Detailed steps of FRS-FS

Either IG or IGR can be used to measure the importance of attribute \(a\), considering their decision ability in the FRS system. However, the IGR criterion is a comprehensive and more stable information criterion for the attribute selection. Hence, the IGR based-feature selection method is adopted in this work for the optimum attribute subset selection. Attribute with the maximum IGR is selected iteratively one by one until the IGR is equal to zero (or a suitable number of attributes) are properly chosen. Detailed steps of FRS-FS are summarized in Algorithm 1.

Algorithm 1 FRS-FS.

Input: Data instances \(X\) with conditional attributes \(C\) and dependence (decision) attribute \(D\).
Output: Optimum attribute subset \(B\).

Main steps:
1. Step 1. Initiate and empty \(B\).
2. Step 2. Calculate the fuzzy similarity matrix \(R\) for each attribute using the similarity function in (2) or (3) and achieve the corresponding fuzzy equivalence matrix \(M\) by calculating its transitive closure.
3. Step 3. For each attribute \(a \in C\), calculate the IGR of \(a\), \(\text{Gain Ratio}(a, B, D)\), using \(10\);
4. Step 4. Choose the attribute in \(C - B\) which has the maximum IGR value, namely, \(b = \arg \max \{\text{Gain Ratio}(a, B, D) | a \in C - B\}\);
5. Step 5. If the IGR of \(b\) is greater than 0, \(B \leftarrow B \cup \{b\}\), return to Step 2; otherwise, go to Step 6;
6. Step 6. Output \(B\) and \(B\) is the ultimate attribute reduction result.

2.1.3. Illustrative example and computational complexity analysis

Given a decision data set, including five continuous conditional attributes, \(C = \{c_1, \ldots, c_5\}\), and one discrete decision attribute \(d\) \(\in\{0, 1\}\), displayed in Table 1.

We first calculate the fuzzy similarity matrix \(R\) of each attribute by the relation equation as displayed in (3) and then we calculate the corresponding fuzzy equivalence matrix \(M\) by the transitive matrix computation. The fuzzy equivalence matrix \(M\) of each attribute is
\[
M_1 = \begin{bmatrix} 1 & 0 & 0 & 0.6357 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0.9185 \\ 0.6357 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0.9185 & 0 & 1 \end{bmatrix}.
\]
\[
M_5 = \begin{bmatrix} 1 & 0 & 0 & 0.9706 & 0.9723 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0.9706 \\ 0.9706 & 0 & 0 & 1 & 0.9706 \\ 0.9723 & 0 & 0 & 0.9706 & 1 \end{bmatrix}.
\]

Table 1

<table>
<thead>
<tr>
<th>(c_1)</th>
<th>(c_2)</th>
<th>(c_3)</th>
<th>(c_4)</th>
<th>(c_5)</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.057</td>
<td>7.850</td>
<td>7.932</td>
<td>9.614</td>
<td>232.006</td>
<td>0</td>
</tr>
<tr>
<td>7.112</td>
<td>21.778</td>
<td>4.936</td>
<td>8.494</td>
<td>45.843</td>
<td>0</td>
</tr>
<tr>
<td>0.960</td>
<td>13.740</td>
<td>9.753</td>
<td>11.774</td>
<td>300.829</td>
<td>1</td>
</tr>
<tr>
<td>3.485</td>
<td>7.648</td>
<td>21.057</td>
<td>23.034</td>
<td>552.444</td>
<td>1</td>
</tr>
<tr>
<td>0.832</td>
<td>7.752</td>
<td>8.237</td>
<td>28.032</td>
<td>382.955</td>
<td>1</td>
</tr>
</tbody>
</table>

The fuzzy equivalence matrix of the decision attribute \(d\) is
\[
M_d = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}.
\]

Then we initiate subset \(B = \{\}\) and determine the first attribute by calculating the IGR, \(\text{gain Ratio}(a, B, D)\), for each attribute,
\[
\text{gain Ratio}(c_1, B, D) = \frac{I(\{\} \cup \{c_1\}; D) - I(\{\}; D)}{H(I(c_1))} = \frac{0.6870}{1.6620} = 0.4134.
\]
\[
\text{gain Ratio}(c_2, B, D) = \frac{I(\{\} \cup \{c_2\}; D) - I(\{\}; D)}{H(I(c_2))} = \frac{0.4282}{1.3878} = 0.3086.
\]
\[
\text{gain Ratio}(c_3, B, D) = \frac{I(\{\} \cup \{c_3\}; D) - I(\{\}; D)}{H(I(c_3))} = \frac{0.3578}{1.2972} = 0.2758.
\]
\[
\text{gain Ratio}(c_4, B, D) = \frac{I(\{\} \cup \{c_4\}; D) - I(\{\}; D)}{H(I(c_4))} = \frac{0.5967}{1.6180} = 0.3688.
\]
\[
\text{gain Ratio}(c_5, B, D) = \frac{I(\{\} \cup \{c_5\}; D) - I(\{\}; D)}{H(I(c_5))} = \frac{0.6494}{1.8265} = 0.3555.
\]

Thus we should choose \(c_1\) as the first attribute based on the IGR, since \(\text{gain Ratio}(c_1, \{\}, D)\) achieves the maximum value. Then \(B = \{c_1\}\). Since \(C-B\) is non-null and \(\text{gain Ratio}(c_1, \{\}, D)\) is not zero, we need to choose another attribute with the same IGR criterion.

\[
\text{gain Ratio}(c_2, B, D) = \frac{I(c_2 \cup \{c_1\}; D) - I(c_1; D)}{H(I(c_1))} = \frac{0}{1.6620} = 0,
\]
\[ \text{gain\_Ratio}(c_3, B = \{c_1\}) = \frac{H((c_3) \cup \{c_1\}) - H((c_1); D)}{H((c_3))} = \frac{0.2840}{1.2972} = 0.2189, \]

\[ \text{gain\_Ratio}(c_4, B = \{c_1\}) = \frac{H((c_4) \cup \{c_1\}) - H((c_1); D)}{H((c_4))} = \frac{0.2840}{1.6180} = 0.1755, \]

\[ \text{gain\_Ratio}(c_5, B = \{c_1\}) = \frac{H((c_5) \cup \{c_1\}) - H((c_1); D)}{H((c_5))} = \frac{0.2840}{1.8265} = 0.1555. \]

Thus, we will choose \( c_3 \) as the second attribute and \( B = \{c_1, c_3\} \). Since \( \text{gain\_Ratio}(c_3, B = \{c_1\}) \) is not equal to zero, we should execute the attribute selection procedure one more time. The IGs of the remaining attributes are

\[ \text{gain\_Ratio}(c_2, B = \{c_1, c_3\}) = H((c_2) \cup \{c_1, c_3\}) - H((c_1, c_3); D) = 0, \]

\[ \text{gain\_Ratio}(c_4, B = \{c_1, c_3\}) = H((c_4) \cup \{c_1, c_3\}) - H((c_1, c_3); D) = 0, \]

\[ \text{gain\_Ratio}(c_5, B = \{c_1, c_3\}) = H((c_5) \cup \{c_1, c_3\}) - H((c_1, c_3); D) = 0. \]

Since the IGs of all the remaining attributes are zero, the attribute selection procedure is terminated and the ultimate chosen result is \( B = \{c_1, c_3\} \).

The main computational cost of the proposed FRS-FS comes from the fuzzy equivalence matrix generation and the information gain ratio computation with comparison in each attribute selection.

For a decision data set, including \( n \) instances and \( c \) conditional attributes and one decision attribute, the computational complexity of the fuzzy similarity matrix for each attribute is \( O(n^2/2) \) and the classic computational complexity of the transitive closure for each attribute is \( O(n^3 \log n) \). Lee (2001) has proposed a highly-efficient computation algorithm with the time cost of \( O(n^2) \) for the max-min transitive closure computation. Hence, if we use the Lee’s algorithm for the transitive closure computation, the total computational complexity of the fuzzy equivalence matrix of all the conditional attributes and the decision attribute is \( O((c+1)n^2/2 + n^3) = O(cn^2) \).

During the attribute selection procedure, the computational cost of the IGR criterion is linear to the instance number with \( O(n) \). In each turn of attribute selection, the worst computational cost is \( O(cn) \). Thus, the total computation cost of the attribute selection procedure is not more than \( O(cn^2 + (c-1)n + \ldots + n) = O(cn^2) \).

Hence, the total computational complexity of the proposed FRS-FS method is \( O(cn^2 + cn^2) \), which is a quadratic polynomial time complexity to the number of instance in the decision data set. In addition, the FRS-FS procedure will only run one time before the online NID to determine the optimal attribute subset, which is relatively efficient and easy-to-implement.

### 2.2. GA-GOGMM-based offline pattern learning

This section addresses the GA-GOGMM-based pattern library learning method for the normal and attack pattern learning.

The GA searching strategy is introduced to automatically determine the optimal number of the mixed Gaussian components in the GMM by an incremental learning manner, namely, adding a new Gaussian component to the current GMM successively and determining the weighting coefficient and the new Gaussian component’s parameter by the maximum likelihood measuring of the training samples, inspired by the Vlassis’s study (Vlassis & Likas, 2002).  

#### 2.2.1. Preliminaries of GMM

GMM is one of the most popular clustering methods, which can be viewed as a linear combination of multiple Gaussian components. A \( K \)-component GMM can be expressed as

\[ p(x) = \sum_{k=1}^{K} \pi_k N(x | \mu_k, \Sigma_k) \]

where \( x \in \mathbb{R}^d \), \( \pi_k \) is the weighting coefficient of the \( k \)-th Gaussian component, satisfying \( \sum_k \pi_k = 1 \) is a Gaussian density function corresponding to the \( k \)-th Gaussian component with mean \( \mu_k \) and covariance matrix \( \Sigma_k \).

GMM-clustering assumes each point is generated from one of the \( K \) Gaussian components. If a GMM is learned, we can compute the probability of any point belonging to any component. For instance,

\[ p(l | x) = \frac{p(l) p(x | l)}{p(x)} = \frac{\pi_l N(x | \mu_l, \Sigma_l)}{\sum_k \pi_k N(x | \mu_k, \Sigma_k)} \]

where \( p(l | x) \) means the probability of point \( x \) belongs to the \( l \)-th component.

GMM parameters \( \Omega = \{ \pi_k, \mu_k, \Sigma_k \}_{k=1}^{K} \) can be learned by maximizing the log-likelihood of the training samples \( D = \{x_i | 1 \leq i \leq N \} \), namely,

\[ \hat{\Omega} = \arg \max_{\Omega} \{ \log \frac{\prod_{i=1}^{N} \sum_{k=1}^{K} \pi_k N(x_i | \mu_k, \Sigma_k)}{\sum_{k=1}^{K} \pi_k N(x_i | \mu_k, \Sigma_k)} \} \]

Since the optimization problem in (13) does not have the closed-form solution, GMM parameters are usually solved iteratively by the Expectation Maximization (EM) algorithm numerically. During the iterative procedure, in the E-step, it estimates the corresponding expected value of hidden variables given any observation \( x_n \) based on the last estimated GMM, namely,

\[ p(k | x_n, \Omega_{\text{old}}) = \frac{\pi_k \text{old} N(x_n | \mu_k \text{old}, \Sigma_k \text{old})}{\sum_{j=1}^{K} \pi_j \text{old} N(x_n | \mu_j \text{old}, \Sigma_j \text{old})} \]

where \( 1 \leq l \leq K \), \( 1 \leq n \leq N \) and \( N \) is the number of samples.

In the M-step, Given the expected hidden variable, it updates the GMM parameters \( \Omega_{\text{new}} = [\pi_k \text{new}, \mu_k \text{new}, \Sigma_k \text{new}]_{k=1}^{K} \) as

\[ \mu_k \text{new} = \frac{\sum_{n=1}^{N} x_n p(k | x_n, \Omega_{\text{old}})}{\sum_{n=1}^{N} p(k | x_n, \Omega_{\text{old}})}, \]

\[ \Sigma_k \text{new} = \frac{\sum_{n=1}^{N} p(k | x_n, \Omega_{\text{old}}) (x_n - \mu_k \text{new})(x_n - \mu_k \text{new})^T}{\sum_{n=1}^{N} p(k | x_n, \Omega_{\text{old}})}, \]

\[ \pi_k \text{new} = \frac{1}{N} \sum_{n=1}^{N} p(k | x_n, \Omega_{\text{old}}). \]

GMM-based clustering needs to provide the cluster number and the mean and covariance parameters of each component initially. Apparently, inappropriate initial value of model parameters may result in divergence of the EM iterative procedure or unreasonable clustering results. In this work, the GA is incorporated in learning the optimal Gaussian components in an incremental manner.
2.2.2. Principle of GA-GOGMM

As reported in the literature \((\text{Li} \& \text{Barron}, 1999; \text{Vlassis} \& \text{Likas}, 2002)\), GMM can be learned by adding a component to the pre-learned mixture model successively. Assume an additional Gaussian component, \(N(x|\Omega_{k+1})\), should be added to a pre-trained \(K\)-component GMM, \(p_k(x)\), the \(K+1\)-component mixture model can be expressed as \((\text{Vlassis} \& \text{Likas}, 2002)\):

\[
p_{K+1}(x) = (1 - \alpha)p_k(x) + \alpha N(x|\mu_{k+1}, \Sigma_{k+1})
\]  

(18)

where \((\mu_{k+1}, \Sigma_{k+1})\) is the parameters of the \(K+1\)-th Gaussian component, \(0 < \alpha < 1\) is a weight value.

We should learn the model parameter \(\Omega = (\alpha, \mu_{k+1}, \Sigma_{k+1})\) in the formula \((18)\) to achieve the \(K+1\)-component GMM, which is equivalent to maximize the logarithm of the new likelihood function of the \(K+1\)-component GMM, namely,

\[
\hat{\Sigma}_{ML} = \arg \max \{ \ell_{K+1}(D|\Omega) \}
\]  

(19)

where the log-likelihood function \(\ell_{K+1}(D|\Omega)\) is

\[
\ell_{K+1}(D|\Omega) = \sum_{i=1}^{N} \log p_{K+1}(x_i|\Omega) = \sum_{i=1}^{N} \log \{ (1 - \alpha)p_k(x) + \alpha N(x|\mu_{k+1}, \Sigma_{k+1}) \}
\]  

(20)

\(p_{K+1}(x)\) in the formula \((18)\) can be regarded as a two-component mixture model, the pre-learned \(K\)-component GMM model \(p_k(x)\) and a newly additional Gaussian component, \(N(x|\Omega_{k+1})\). Hence, the EM algorithm can be also employed to learn the model parameters.

Since \(p_{K+1}(x)\) is a two-component GMM, the EM-based iterative procedure can converge fast to achieve the local maxima of \(\ell_{K+1}(D|\Omega)\). As addressed in Vlassis’s work \((\text{Vlassis} \& \text{Likas}, 2002)\), since the EM-based iterative procedure is a locally searching, the GMM often is still sensitive to the initialization of the model parameters. Hence, a global searching strategy should be introduced to alleviate the adverse influence of the initializations of these parameters \((\text{Vlassis} \& \text{Likas}, 2002)\).

If we make a second-order Taylor decomposition of \(\ell_{K+1}(D|\Omega)\) with respect to the parameter \(\alpha\) on a specific point \(\alpha_0\) (i.e., 0.5 or 0),

\[
\ell_{K+1} \approx \ell_{K+1}(\alpha_0) + \ell_{K+1}'(\alpha_0)(\alpha - \alpha_0) + \ell_{K+1}''(\alpha_0)(\alpha - \alpha_0)^2 / 2
\]  

(21)

Then the maximum point of \(\ell_{K+1}\) should locate at \(\frac{\partial \ell_{K+1}}{\partial \alpha} = 0\), where the maximum of \(\ell_{K+1}\) is

\[
\hat{\ell}_{K+1} = \ell_{K+1}(\alpha_0) - \frac{\ell_{K+1}''(\alpha_0)}{2 \ell_{K+1}'(\alpha_0)}
\]  

(22)

where \(\hat{\alpha}_{K+1}\) stands for the maximum of \(\ell_{K+1}\) and the maximum point \(\hat{\alpha} = 0 = \frac{\ell_{K+1}'(\alpha_0)}{\ell_{K+1}''(\alpha_0)}\). Specifically, if we set \(\alpha_0 = 1 / 2\),

\[
\hat{\ell}_{K+1} = \sum_{i=1}^{N} \log p_{K+1}(x) + \frac{N(x_i|\mu_{k+1}, \Sigma_{k+1})}{2} - \frac{\sum_{i=1}^{N} N(x_i|\mu_{k+1}, \Sigma_{k+1})p_{K+1}(x_i)}{\sum_{i=1}^{N} p_{K+1}(x_i) N(x_i|\mu_{k+1}, \Sigma_{k+1})}^2 
\]  

(23)

and

\[
\hat{\alpha} = \frac{1}{2} - \frac{1}{\frac{2}{N} \sum_{i=1}^{N} \left[ N(x_i|\mu_{k+1}, \Sigma_{k+1}) - p_{K+1}(x_i) \right]} - \frac{p_{K+1}(x_i) N(x_i|\mu_{k+1}, \Sigma_{k+1})}{\left[ p_{K+1}(x_i) N(x_i|\mu_{k+1}, \Sigma_{k+1}) \right]^2}
\]  

(24)

As can be seen from the formula \((22)\) or \((23)\), \(\hat{\ell}_{K+1}\) is independent to the model parameter \(\alpha\), hence the partial EM iteration procedure will be no long influenced by the initialized parameter of \(\alpha\). Based on the formula \((24)\), \(\alpha\) can be initialized as \(\hat{\alpha} = 0.5\), when \(K = 1\) and \(\hat{\alpha} = 2/(K + 1)\), when \(K \geq 2\) \((\text{Vlassis} \& \text{Likas}, 2002)\).

Though we have derived a global optimum of the log-likelihood \(\hat{\ell}_{K+1}\), independent to the parameter \(\alpha\), which is still depended to the other parameters \(\mu_{k+1}\) and \(\Sigma_{k+1}\). Worse still, it is nearly infeasible to search the whole spanning space of \((\mu_{k+1}, \Sigma_{k+1})\), because the covariance matrix \(\Sigma_{k+1}\) involves too many parameters. However, the optimum function \(\hat{\ell}_{K+1}\) is actually a quadratic function of \(\mu_{k+1}\) (or a function of the Mahalanobis Distance between \(\mu_{k+1}\) and each sample \(x_i\)).

To make a global search, every point in the training samples are considered as a candidate center for the new component as addressed in Vlassis and Verbeek’s works \((\text{Verbeek, Vlassis \& Kröse}, 2003; \text{Vlassis} \& \text{Likas}, 2002)\), which can be used to optimize \(\hat{\ell}_{K+1}\) to achieve the other matrix parameter as expressed in the formula \((23)\). Hence, during the partial EM iteration procedure, if we fix the covariance matrix to optimize the mean vector \(\mu_{k+1}\), we can compute the pairwise distance \(k_{i,j}\) between any two samples in advance, which only needs to be carried out once at the beginning of the iteration. In Vlassis’s work \((\text{Verbeek et al., 2003})\), a nonparametric multivariate density estimator is introduced as

\[
k_{i,j} = N(x_i, x_j, \Sigma^2) = \frac{1}{(2\pi)^{d/2}} \exp \left\{ -\frac{\|x_i - x_j\|^2}{2\sigma^2} \right\}
\]  

(25)

where \(d\) is the dimensionality of \(x\) and \(\sigma\) should depend on the dimension of \(x\) and the size of training set. In Vlassis’s work \((\text{Verbeek et al., 2003})\), a nonparametric multivariate density estimator is introduced as

\[
\sigma = \gamma \left( \frac{4}{(d + 2)n} \right)^{1/(d+4)}
\]  

(26)

where \(\gamma\) is a predefined constant and \(D_{ij}\) with \(\sigma\) can be defined as different kernel forms with a proper distance definition and density expression, which will not affect the final estimation of the GA-GOGMM as addressed in Vlassis’s work \((\text{Verbeek et al., 2003})\).

2.2.3. Effectiveness and complexity analysis of GA-GOGMM

GA-GOGMM-based network pattern representation uses the GA-based global maximum searching method (with regard to the log-likelihood) for the GMM learning to achieve the effective representation of either normal instances or attack instances. GA-GOGMM determines the optimal number of mixing components by inserting optimally one additional (the \(K+1\)-th) components to the optimally-learned \(K\)-component GMM. The main steps of GA-GOGMM-based pattern learning method are summarized in Algorithm 2.

Fig. 2 gives an example of the dynamic GMM learning results (from one to six Gaussian components) based on the GA-GOGMM algorithm. Experimental results in Fig. 2 show that the GA-GOGMM learning procedure can effectively converge to the GMM of the optimum number (generally the true number) of mixing components with their true parameters.

The convergence to the true probability density model of the GA-based GMM learning procedure has been analyzed in the work \((\text{Verbeek et al., 2003})\), where the author has evidenced that there exists a number \(C\) such that the Kullback-Leibler divergence between the GA-based learning model and the same \(K\)-component GMM tend to be zero at the rate of \(\mathrm{C}/K\), where \(C\) is independent to the number component \(C\).

With regard to the computational complexity, for any data set, it cost \(O(nk)\) times in the updating of an additional new component (the \(k\)-component), hence the total run time of a
Fig. 2. Example of the dynamic clustering learning results (from 1-component to 6-component GMM) of GA-GOGMM method on a simulating data set with a 6-component GMM. The true GMM is \( p_d(x) = \sum_{k=1}^{K} \frac{1}{6} \mathcal{N}(x|\mu_k, \Sigma_k) \), where \( \mu_1=(1,2)^T \), \( \mu_2=(13, -5)^T \), \( \mu_3=(7, 5)^T \), \( \mu_4=(4, 10)^T \), \( \mu_5=(-4, -3.5)^T \), \( \mu_6=(14, 10)^T \); \( \Sigma_1=[1 -0.5; 0.5 1] \), \( \Sigma_2=[1 0; 0 1] \), \( \Sigma_3=[0 1; 1 0] \), \( \Sigma_4=[1 0; 0 1] \), \( \Sigma_5=[0 1; 1 0] \), \( \Sigma_6=[1 0; 0 1] \).

Algorithm 2: GA-GOGMM-based pattern learning.

1. Prepare the normal data set and the attack data sets \( D_0, D_1, \cdots, D_t \) for the network connection pattern learning, i.e., \( D_0 \) stands for the normal data set and each of the others denotes the data set of a specific attack mode.
2. For any data set \( D_n, 0 \leq n \leq T \), execute the GA-based optimal GMM learning procedure.

Step 2.1. Initialize a \( k \)-component GMM (\( k=1 \)) with \( \mu_{0,k} = \mathbb{E}(D_n) \) and \( \Sigma_{0,k} = \text{cov}(D_n) \), and precompute the distance-based kernel matrix \( k=[k_{ij}] \) by the formula (20), where \( \sigma \) is estimated by the formula (16) with the setting of \( \gamma \) as the half of the maximum eigenvalue of \( \Sigma_{ij} \).

Step 2.2. Start the GA-based \( k=k+1 \)-component GMM learning as described in (18).

Step 2.2.1. Treat each point \( x_i \) as a potential clustering center of the new Gaussian component, namely, set \( \mu_{0,k} = x_i \) and use the precomputed kernel matrix \( k \) to approximate the Gaussian component \( \mathcal{N}(x|\mu_i, \sigma^2 \Sigma) \) to achieve the maximum of the log-likelihood function \( \ell_k \) as described in the formula (23).

Step 2.2.2. Prepare the initialization of \( a \) by the formula (24) with the estimated \( \mu_{0,k} \) and \( \Sigma_{i,j} = \sigma^2 \).

Step 2.2.3. Execute the partial EM-based iteration procedure until them converge and then we achieve the new \( K \)-component GMM.

Step 2.2.4. If \( \ell_{k+1} \leq \ell_k \) (or sufficient numbers of Gaussian mixing components are achieved for the GMM learning), then terminate the GA-GOGMM learning procedure. Otherwise go to Step 2.2.

Step 3. For each data set \( D_n, 0 \leq n \leq T \), output its final number \( K \) of mixing components and achieve the \( K \)-component GMM-based pattern representation of normal and attack network instances, i.e., use the clustering centers with their corresponding sample number (as a survival number) of each Gaussian component for the pattern representation.

2.3. Pattern matching-based NID and online pattern updating

If the normal and the attack network pattern libraries are learned, a pattern matching based intrusion detection procedure can be conducted to detect the network anomaly. Attack modes and network environment are dynamically changing or retrofitting constantly, since intruders will update their intrusion skills or strategies at any time. Hence, a kind of automatic model updating mechanism is also adopted to improve the adaptability of the NID model. Schematic diagram of the pattern matching-based NID with online pattern updating module is displayed in Fig. 3.

Online model updating occurs accompanying with the online NID procedure by introducing a new cache pattern library (CPL) to discover the unforeseen network connection patterns but frequently occurred in the network connection instances. Each pattern in normal or attack pattern libraries has a survival value (SV) to maintain the pattern updating.

Model updating occurs simultaneously with the online NID. During the pattern updating, if a new network connection instance can correctly match a learned pattern (in the normal pattern library (NPL) or the attack pattern library (APL)), whose corresponding SV will increase by \( s \) (i.e., 1 in this work) and the SVs of all other patterns will decline by \( e(e << 1) \). When the SV of a pattern is less than a predefined threshold, \( \lambda \), it means that this pattern is an obsolete pattern, which can be discarded to ensure the adaptability of the proposed NID method.

If a new network connection instance neither matches normal patterns nor attack patterns, we will check whether it is a frequent pattern (FP) by matching it with patterns (samples with corrected pattern representation results by the FRS-FS) in CPL. If it correctly matches a pattern in CPL, the CPL will be updated analogously to the NPL or APL. Otherwise, it will be added into the CPL with an initial SV. Patterns with SVs less than \( \lambda \) will be discarded to ensure that the CPL would not be growing indefinitely. Patterns with SVs greater than a predefined threshold, \( r \), are regarded as frequent patterns (FPs).
Given a frequent pattern, \( p \), we can determine its real network connection mode (either normal or an attack mode) by measuring its similarities to normal patterns and attack patterns, or determine its real network connection mode by experts. Then, append it to the corresponding pattern library as a rational pattern for the successive NID with an initial SV (i.e. a small value). The cosine similarity is used as the pattern matching or pattern similarity measuring criterion, given by

\[
d(x, y) = \frac{x^T y}{\|x\| \|y\|}.
\]

(27)

The smaller value of \( d(x, y) \) means smaller angle of \( x \) and \( y \) and also means greater similarity of \( x \) and \( y \).

In summary, the proposed ANID system combines FRS-FS and GA-GOGMM with an online updating mechanism to ensure the adaptability of the NID system on various attacks, especially for the unforeseen attacks under dynamic changing network environment. Main steps of the proposed ANID method are summarized as follows.

Step 1: **Initialization.** Initialize the pattern attenuation coefficient (AC), \( \epsilon \), the frequent pattern threshold (FPT), \( \tau \), and the obsolete pattern threshold (OPT), \( \lambda \). Empty the NPL, APLs (including multiple libraries according to the number of attack modes, e.g., DOS, U2R, R2L and Probe pattern libraries) and CPL. Prepare the training dataset and test dataset.

Step 2: **FRS-FS-based optimum attribute selection.** Achieve the optimum attribute subset by Algorithm 1.

Step 3: **GA-GOGMM-based NPL and APL learning.** Achieve the optimum clustering model to generate the normal and attack pattern libraries by Algorithm 2.

Step 4: **Online NID and online pattern updating.** Given a test network instance \( x_t \), determine its network connection type by either a normal or one of attack modes by pattern similarity measuring and update pattern libraries, as displayed in Fig. 3.

Step 4.1: **Normal pattern matching.** Use the cosine similarity criterion to determine whether \( x_t \) can well match normal patterns. If it is matched, increase the SV of the matched pattern in the normal pattern library (NPL) by \( s \) and decline that of others by \( \epsilon \) and discard the obsolete patterns with SVs less than \( \lambda \) in NPL, otherwise go to Step 4.2.

Step 4.2: **Attack pattern matching.** If \( x_t \) can well match an attack pattern, perform alarm and increase the corresponding SV of the matched attack patterns by \( s \) and decline that of others in the APL by \( \epsilon \) and discard the obsolete patterns with SVs less than \( \lambda \) in the APL, otherwise go to Step 4.3.

Step 4.3: **Cache pattern matching.** If \( x_t \) can well match an instance (pattern) in CPL, update the SV of the corresponding matched pattern in the CPL analogous to the normal pattern matching or attack pattern matching and discard the obsolete patterns in CPL and go to Step 4.4, else go to Step 4.5.

Step 4.4: **FP checking.** Check whether the largest SV of patterns in CPL is greater than \( \tau \), if it meets the constraint, the pattern with largest SV in CPL is a FP, go to Step 4.6.

Step 4.5: **Append \( x_t \) to the CPL.** Append \( x_t \) to the CPL as a new pattern with an initial SV.

Step 4.6: **Determine \( p \)'s network connection type (by similarity measuring or determine interactively by experts).** If \( p \) is more similar to a normal pattern append it to the NPL with an initial SV. If \( p \) is an attack pattern, achieve its most similar attack pattern and append it to the corresponding APL.

Step 4.7: Return to Step 4 to process a new network connection instance \( x_{t+1} \).

Fig. 3. Pattern matching-based NID and online pattern updating.
3. System implementation and experimental validation

In this section, we use python to implement the software prototype, including three modules: the FRS-based attribute selection module, which actually only run one time based on the training data set offline, the incremental normal and attack network connection pattern library generation module based on the GA-GOMM approach and the online pattern matching-based intrusion detection with pattern library updating module.

The Performance analysis has been performed on a desktop running with Intel (R) Core (TM) i7-8700, CPU 3.20 GHz, and 16-GB RAM hardware platform with the Ubuntu16.04 64 bit operating system by python3.6 and PyCharm programming environment.

We validate the NID performance of the proposed method by verification and comparative experiments conducted on a benchmark NSL-KDD data set and a Nidsbench-based network simulation platform.

3.1. Experimental results on NSL-KDD data set

NSL-KDD is a modified version of the widely-used computer intrusion detection dataset of KDD99. The original KDD99 is a relatively large data set (about 50,000 records in the training set and test set, respectively). However, as analyzed in Taylorae's study [Taylorae, Bagheri, Lu & Ghorbani, 2009] that KDD99 data set has a huge number of redundant records in the training set and test set (about 75% and 75% of the records are duplicated in the train and test set, respectively), which will cause the classifier to be biased towards the more frequent records and harmful to the sparse attack types.

The NSL-KDD data set has a reasonable number of records in train and test sets, which includes 125,973 and 22,544 records in the full training and test set, respectively. It also provides a subset of 20% subset of the NSL-KDD full training set including 25,192 records, which can be used for the prototype NID system experiment. Analogously to the KDD99 data set, each single connection instance contains 41 attributes, which can be categorized to be three groups, basic connection attributes (9 attributes), network traffic features (9 attributes) and network content features (13 attributes). Detailed description can be found in the literature [Lin, Ying, Lee & Lee, 2012].

Each network instance is labeled as either a normal or a specific type of 22 attacks [Al-Yaseen, Othman & Nazari, 2017]. These attacks fall under one of the four categories: DoS, Probe, U2R, and R2L. DoS is an attack attempting to prevent legitimate users from using a service, e.g., syn flooding. Probe is a cyber-attack refers to the surveillance and other probing, e.g., port scanning. U2R refers the unauthorized access from a remote machine, e.g. guessing password. U2L denotes the attacks that unauthorized access to local superuser (root) privilege, e.g., various buffer overflow attacks.

We validate the effect of the attribute selection method by using two different training sets provided in the NSL-KDD data library, namely the full NSL-KDD training set (NSL-KDDTrain) and the 20% subset of the NSL-KDDTrain (NSL-KDDTrain_20), with two different test sets, the full NSL-KDD test set (NSL-KDD test) and KDDTest-21. Beside the normal instances, the corrected test data set includes 18 additional attack types. Structures of training and test sets are shown in Table 2.

Nominal attributes (the second, third and fourth attributes in the train and test data set) in each original network connection instance are assigned corresponding numeric labels according to the number of labels in the signal reconstruction experiments. For instance, since there are three potential values of the second feature (protocol_type), including TCP, UDP and ICMP, symbol TCP is changed to 1 to represent the first kind of protocol_type. The min-max-normalization is used to normalize the training and test samples.

Five criteria, true positive rate (TPR), detection accuracy (DA), False alarm rate (FAR) (also termed false positive rate, FPR), Missing rate (MR) (also termed false negative rate), F1-measure (F1M) are adopted for the performance evaluation. For an experiment with $N_p$ actual positive instance and $N_n$ actual negative instances, the four outcomes of a detector (classifier) can be formatted in a confusion matrix as displayed in Table 3. Based on the confusion matrix, the evaluation criteria can be defined as (Tharwat, 2018)

$$TPR = \frac{TP}{N_p},$$

$$DA = \frac{TP + TN}{N_p + N_n},$$

$$FAR = \frac{FP}{TP + TN} = \frac{FP}{N_p},$$

$$MR = \frac{FN}{TP + FN} = \frac{FN}{N_p},$$

$$F1M = \frac{2TP}{2TP + FP + FN}.$$  

3.1.1. Effect of attribute selection

Feature redundancy is usually determined by the correlation between features. Figure 4 displays the scatter plot of the pairwise relations of 7 attributes of the NSL-KDDTrain_20 data set. As can be seen from Figure 4, some attributes have evident dependency. For instance, A13 and A16 have linear correlation, A23 and A24 have strong dependency. Hence, attribute selection is beneficial to the subsequent anomaly detection.

We compare the attributes selection results of the introduced FRS-FS method with some classic attribution reduction methods, including the rough set-based FS (RS-FS) methods (Jiang, Sui & Cao, 2013), one is based on the significance of attributes and the other is based on the dependency of attributes, the hypergraph-based genetic algorithm (HG-GA)-based SVM-based FS method (HG-GA-SVM) (Raman, Somu, Kirthivasan, Liscano & Sriram, 2017), linear regression and genetic algorithm-based wrapper algorithm (LR-GA wrapper) (Khammassi & Krichen, 2017), the intelligent agent-based attribute selection algorithm (IAASA) (Ganapathy, Yogesh & Kannan, 2012) and the cost-sensitive attribute selection algorithm using histograms (CASH) (Weiss, Elowici, & Rokach, 2013). The FS results of these methods are listed in Table 4.

To validate the validity of the discriminant ability of the reduced attributes, Table 5 lists the NID performance indexes based on different training and test data set. To achieve relatively stable evaluation results, thirty repeated experiments are conducted and the statistical mean and standard deviation value of these repeated experiments with replacement are listed in the table. In this experiment, all the comparative experiments are treated with the same detection model introduced in this work except the feature selection method. The model parameters $\alpha$ and $\lambda$ are both fixed to be a low value of 0.001 and $\tau$ is set as 100.

As can be seen from Table 5, by introducing the FRS-FS method, the proposed ANID method can achieve higher detection accuracies based on the evaluation criteria, DA and F1M, than comparative methods as well as low false alarms based on the FAR criterion on the all four test data sets. With regard to the criterion of MR, the proposed method can also achieve lower missing rates than all the comparative methods on data sets TD1 and TD4. On data sets TD1 and TD3, though the missing rates based on the proposed FRS-FS are a little higher (only higher about 0.18%) than the GA-LR Wrapper method and RS-FS (dependency of attributes).
Table 2
Structures of data sets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total</th>
<th>Normal</th>
<th>Attack</th>
<th>Attack types</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSL-KDDTrain_20</td>
<td>25,192</td>
<td>13,448</td>
<td>11,743</td>
<td>neptune, warezclient, ipsweep, portsweep, teardrop, nmap, satan, smell, pod, back, guess_pass, wd, ftp_write, multihop, rootkit, buffer_0, erlfw, imap, warezmaster, phf, land, loadmodule, spy, perl</td>
</tr>
<tr>
<td>NSL-KDD Train</td>
<td>125,973</td>
<td>67,342</td>
<td>58,630</td>
<td>neptune, warezclient, ipsweep, portsweep, teardrop, nmap, satan, smell, pod, back, guess_pass, wd, ftp_write, multihop, rootkit, buffer_0, erlfw, imap, warezmaster, phf, land, loadmodule, spy, perl</td>
</tr>
</tbody>
</table>

Fig. 4. Scatter plots of partial attributes of NSL-KDDTrain_20, where A13: num_compromised, A16: num_root, A23: count, A24: srv_count, A25: serror_rate, A28: srv_error_rate and A29: same_srv_rate. Each row show the relations of an attribute with the other attributes, i.e., where, for instance, the cell in the first row and third column represents the relation between A13 (i.e. first row) and A23 (i.e. third column). Whereas, cells in the diagonal lines records the histogram distributions of the 7 attributes.

Table 3
Confusion matrix.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive</td>
<td>True positive (TP)</td>
<td>False positive (FP)</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>False negative (FN)</td>
<td>True negative (TN)</td>
</tr>
</tbody>
</table>

method, respectively, the FARs and the detection performance, including the DAs, F1Ms are obviously superior to the comparative methods.

Combining the feature selection results in Table 4 and intrusion detection results in Table 5, though GA-LR Wrapper method achieves the minimum attribute number of 16 compared to the 18 attributes of the FRS-FS method on the training data set of NSL-KDDTrain_20, the detection accuracy of the GA-LR Wrapper-based ANID method is lower over 8 percent than the FRS-FS-based ANID method on the test data set NSL-KDD test. That means the introduced FRS-FS method can achieve most intrinsic and least redundant attributes for the intrusion detection, hence it can achieve high intrusion detection accuracies and low false alarms as well as low missing reports.

3.1.2. Effect of parameter setting in the intrusion detection and updating model

In this section, we investigate the effect of the different parameter setting in the ANID model. The proposed ANID method will be mainly affected by three parameters, AC (ε), OPT (λ) and FPT (τ).

Intuitively, both ε and λ should be set relatively low values to ensure the stability of learned patterns, because larger values of ε and λ will result in important but occasionally-occuring patterns being obsolete patterns with high probability and being removal from the pattern libraries, which may lead to a low detection
The accuracy of network connection instances and results in unstable NID results.

Large value of $\tau$ will thwart the timely updating of normal and attack patterns. Oppositely, too small valued $\tau$ may lead to the rapid expansion of the normal and attack libraries, which will increase the time cost of the online NID and even results in detection instability.

We design two experiments to validate the effect of these parameters. The first experiment attempts to validate the effect of different values of the same setting of $\varepsilon$ and $\lambda$ with a fixed $\tau$ ($\varepsilon$ and $\lambda$ are set to be a common value). Fig. 5 displays the average DA of these four data sets under different logarithmic values of $\varepsilon$ or $\lambda$.

As can be seen from Fig. 5, the NID accuracy on the test set (NSL-KDD test is used as the test set and NSL-KDDTrain is the training set) will generally decline with the increasing of $\varepsilon$ or $\lambda$. When the logarithmic value of $\varepsilon$ or $\lambda$ ranges in $[-7.9 - 7.6]$ (i.e., $\varepsilon$ or $\lambda$ ranges in $[0.00035, 0.00505]$), the NID accuracy retains in a relatively high and flat value. And then with the continuous increase of $\varepsilon$ or $\lambda$, the NID accuracy will decline gently until the logarithmic value of $\varepsilon$ or $\lambda$ reaches over the value of $-5.3$.

In terms of the experimental results, $\varepsilon$ or $\lambda$ should be assigned a relatively small value, preferably no larger than 0.0005, consistent with our preceding analysis. Hence, to achieve high NID accuracy, we set $\varepsilon$ and $\lambda$ to be 0.0004 (mean value of the first flat and high DA region) for the subsequent NID validation and comparative experiments.

The other experiment attempts to validate the effect of different value of FPT ($\tau$). Experimental results are shown in Fig. 6. As can
be seen from Fig. 6, the smaller value of $\tau$, the relatively higher NID accuracy we can achieve; however, smaller value of $\tau$ also achieves generally higher FAR. Apparently, smaller $\tau$ leads to rapid updating of normal and attack patterns and results in more abundant and timely updating normal and attack patterns, which ensure relatively higher intrusion detection accuracy. However, occasionally, some ambiguous and frequent instances may be mistakenly introduced into the pattern libraries, which will result in higher false alarm in the subsequent NID, though these patterns can be removed when their SVs are over a predefined small OPT $\lambda$. Opposite results will be achieved if relatively larger $\tau$ is assigned.

Hence, a compromise should be taken is to set a proper $\tau$ neither too great nor too small to ensure higher NID accuracy simultaneously with relatively low false alarm ratio. In terms of preceding analysis and the NID results in Fig. 6, to achieve high NID detection accuracies, a small value, e.g., 100, of $\tau$ should be assigned to allow more patterns that can be added into the pattern library. With the expansion of the normal and APL, pattern libraries store abundant and rich enough patterns. Then it can increase $\tau$, e.g., increase it to 500, to thwart the dramatic expansion of pattern libraries, which will absolutely generate negative effect on the time cost of online NID and may incur negative effect on the NID accuracy since online updated patterns in pattern libraries may have conflicted.

3.1.3. NID performance validation and comparison

To validate the effectiveness of the proposed ANID method, we conduct experiments on the test set by setting both $\epsilon$ and $\lambda$ to be 0.0004 and $\tau$ to be 100 as a fixed value in each experiment. The intrusion performance criteria are displayed in Table 6.

As can be seen from Table 6, the proposed ANID method can achieve relatively high TPRs on all attack types except the U2R attack modes on almost all four test sets. This is because network connection data instances of U2R are similar to that of normal type, which may be mistakenly recognized as normal mode. However, the overall detection accuracies of the known attacks on both normal and attack instances, DAs, are almost over 98%, 97%, 95% and 93% on the four test sets, respectively, and the DAs for novel attacks are also over 90% on all the four data sets with an average value of about 92.75%.

Experimental results in Table 6 indicate that the proposed method can effectively detect almost all four attack behaviors. Since the proposed method extracts the intrinsic and remove the irrelevant and redundant attributes of network connection instances, it can achieve high detection accuracy on known intrusion types. With respect to the novel attacks, the proposed method can ensure the adaptability by online updating normal and attack patterns. Hence, it can also achieve relatively high detection rates on novel attack types.

To further evaluate the NID performance of the proposed method, it is compared with seven classic and representative NID methods. These comparative methods are Cluster Center and Nearest Neighbor (CANN) (Lin, Ke & Tsai, 2015), fuzziness based semi-supervised learning approach (FBSLADS) (Ashfaq, Wang, Huang, Abbas & He, 2017), intrusion detection using hypergraph-Genetic algorithm for parameter optimization and feature selection in support vector machine (HG-GA-SVM) (Raman, Kirthivasan, et al., 2017), an intelligent intrusion detection method using SVM and simulated annealing (SA) for feature selection and the detection tree for the anomaly detection reported in Lin’s study (Lin et al., 2012) (SVM-SA-DT, for short), intrusion detection methods based on firefly algorithm-based feature selection and C4.5 or Bayesian networks (BN) (Selvakumar & Muneeswaran, 2019), shortened as Firefly-C4.5 and Firefly-BN, respectively, and CAI (Construction with adaptive increments extreme learning machine)-based intrusion detection method.

Comparatively, CANN adopts a new feature representation approach combined clustering and nearest neighbor matching for the NID; FBSLADS introduces a fuzziness-based semi-supervised NID; HG-GA-SVM employs a hypergraph-based GA for feature selection; SVM-SA-DT adopts the support vector machine and simulated annealing algorithm for the feature selection; Firefly-C4.5 takes the firefly algorithm for the feature selection with a decision tree model as the classifier; Firefly-BN also uses the firefly algorithm for the feature selection but it uses the Bayesian Networks as the classifier; CAI introduces an adaptively incremental learning-based optimal extreme learning machine for the NID. Comparative experiment results on the experimental data set TD1 are listed in Table 7.

As can be seen from Table 7, the proposed method outperforms comparative methods obviously in terms of the criterion of detection accuracies. The performance evaluation criterion DA is over 2.5 percent than that of the second-ranked results from HG-GA-SVM. F1M is a more representative evaluation criterion for the imbalanced dataset, which is a harmonic mean of the precision and recall (Tharwat, 2018). The proposed method can also achieve higher F1M than these comparative methods, hence it can draw a conclusion that the proposed method can achieve high detection accuracies.

With regard to the missing reports, the proposed method can achieve the lowest MR results, which means the proposed method has lower missing reports than all of the comparative methods. In terms of the criterion of FAR, the proposed method can achieve lower false alarms than almost all comparative methods, except the Firefly-BN. Though the FAR of the proposed method is slightly higher than that of Firefly-BN (a little higher over 0.04%), the detection accuracy of the proposed method is higher than that of Firefly-BN, and the detection performance is much stable than that of Firefly-BN in terms of the standard deviation value of the detection criterion.
To summarize, the proposed intrusion detection method has a higher detection accuracy than these comparative methods with relatively lower FAR and MR. Hence, the proposed ANID method has obvious performance superiority in the NID. The superiority of the proposed method mainly results from the optimal attribute selection based on the FRS-FS and the online updating of the pattern learning of both normal and intrusion network connection records. FRS-FS can remove the redundant, uncertain information in the original instances. The GA-GOCMM-based optimum pattern representation of normal and attack connection instances can effectively avoid the negative effect of the empirical initialization of the cluster number with the random initialization of the clustering centers on clustering-based pattern learning. In addition, the online model updating strategy introduced in this work can ensure the ANID model be adaptive to the retrofitting and change of attack modes and the network environment. Hence, on both known attacks and unforeseen attacks, the proposed ANID method can achieve higher detection accuracies with lower false alarm rates and missing reporting rates.

### 3.2. Experimental results on Nidsbench-based network simulation system

To make a further evaluation of the proposed ANID method on a more real network environment, a network simulation experiment platform is built by using the Nidsbench Test Platform (Lippmann, Haines, Fried, Korba & Das, 2002). Nidsbench is a well-used network testing software package developed by Anzen, including Tcpreplay and Fraqrouter (Lippmann et al., 2002).

To simulate real network running states, Tcpreplay replays the data packets captured by the simulation system. In order to test the performance of an intrusion detection system, Fraqrouter constructs a series of attacks attempting to avoid being detected. Nidsbench-based test platform can simulate various intrusion instances, especially the unforeseen attacks, to evaluate the applicability of the proposed ANID method.

Fig. 7 displays the topological diagram of the network simulation system. The network simulation system consists of three routers (router1, router2, and router3, where RIP protocol is adopted), three switches (switch1, switch2, and switch3), four servers (database server, email server, FTP server and Web server), and six Ethernet subnets (subnet1, subnet2, subnet3, subnet4, subnet5 and subnet6).

Network traffic data are analyzed over different time periods. Based on this model, each event of each network protocol is simulated. Each host of each subnet runs database services, email services, FTP services, and HTTP services according to a certain traffic mode in the network simulation system.

Network simulation platform simulates normal operation and network attack activities in an intranet. Simulation data include training data and actual test data, both of which contain data records of normal activities and intrusion activities and test data contain attack types that do not appeared in the training data. Main network connection types include Normal, Probe and R2L, Vulnerability, Dos, and U2R. Each instance is a 42-dimensional data (including 41-dimensional conditional attributes and 1-dimensional decision-making attributes).

Through a four-day network data collection, 30,000 network requests are filtered on every day and a total 120,000 instances are

### Table 6

Intrusion detection performance of known and unknown attacks.

<table>
<thead>
<tr>
<th>Attack types</th>
<th>TPR Known</th>
<th>TPR Novel</th>
<th>FMR Known</th>
<th>FMR Novel</th>
<th>TPR Known</th>
<th>TPR Novel</th>
<th>FMR Known</th>
<th>FMR Novel</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS</td>
<td>98.52 ± 0.23</td>
<td>96.52 ± 0.21</td>
<td>90.64 ± 0.34</td>
<td>98.26 ± 0.32</td>
<td>99.46 ± 0.28</td>
<td>97.34 ± 0.28</td>
<td>94.06 ± 0.34</td>
<td>96.52 ± 0.21</td>
</tr>
<tr>
<td>U2R</td>
<td>85.22 ± 0.45</td>
<td>82.24 ± 1.23</td>
<td>80.45 ± 1.23</td>
<td>84.05 ± 1.02</td>
<td>84.56 ± 0.88</td>
<td>83.24 ± 0.84</td>
<td>86.43 ± 0.86</td>
<td>81.34 ± 0.24</td>
</tr>
<tr>
<td>R2L</td>
<td>97.25 ± 0.80</td>
<td>94.35 ± 1.45</td>
<td>95.34 ± 1.23</td>
<td>97.47 ± 1.23</td>
<td>97.66 ± 1.01</td>
<td>94.73 ± 0.86</td>
<td>97.12 ± 0.84</td>
<td>93.33 ± 0.68</td>
</tr>
<tr>
<td>PROBE</td>
<td>92.73 ± 1.50</td>
<td>92.87 ± 0.44</td>
<td>93.46 ± 1.02</td>
<td>92.19 ± 0.23</td>
<td>92.66 ± 1.02</td>
<td>92.66 ± 0.24</td>
<td>92.46 ± 0.04</td>
<td>91.80 ± 0.14</td>
</tr>
<tr>
<td>Overall</td>
<td>95.23 ± 1.02</td>
<td>94.15 ± 1.40</td>
<td>94.01 ± 0.45</td>
<td>93.92 ± 1.23</td>
<td>94.34 ± 0.80</td>
<td>92.46 ± 0.24</td>
<td>92.67 ± 1.02</td>
<td>90.68 ± 0.56</td>
</tr>
</tbody>
</table>

### Table 7

Performance comparison of different NID methods.

<table>
<thead>
<tr>
<th>Model</th>
<th>DA (%)</th>
<th>FIM (%)</th>
<th>FAR (%)</th>
<th>MR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed method</td>
<td>96.86 ± 0.62</td>
<td>96.32 ± 1.23</td>
<td>0.88 ± 0.04</td>
<td>1.02 ± 0.20</td>
</tr>
<tr>
<td>CANN</td>
<td>93.88 ± 2.34</td>
<td>94.67 ± 2.34</td>
<td>1.16 ± 0.02</td>
<td>2.46 ± 0.25</td>
</tr>
<tr>
<td>FBSLADS</td>
<td>92.83 ± 1.06</td>
<td>93.45 ± 1.08</td>
<td>1.31 ± 0.12</td>
<td>2.63 ± 0.30</td>
</tr>
<tr>
<td>HG-GA-SVM</td>
<td>94.33 ± 2.45</td>
<td>94.67 ± 1.68</td>
<td>0.87 ± 0.12</td>
<td>1.89 ± 0.10</td>
</tr>
<tr>
<td>SVM-SA-DT</td>
<td>90.34 ± 1.56</td>
<td>91.24 ± 2.34</td>
<td>2.23 ± 0.23</td>
<td>3.45 ± 0.12</td>
</tr>
<tr>
<td>Firefly-C4S</td>
<td>92.45 ± 2.45</td>
<td>91.45 ± 2.68</td>
<td>2.45 ± 0.89</td>
<td>2.67 ± 0.34</td>
</tr>
<tr>
<td>Firefly-BN</td>
<td>93.45 ± 1.08</td>
<td>94.56 ± 1.68</td>
<td>0.84 ± 0.45</td>
<td>2.25 ± 0.37</td>
</tr>
<tr>
<td>CAI</td>
<td>94.20 ± 2.34</td>
<td>95.34 ± 2.03</td>
<td>1.48 ± 0.14</td>
<td>2.88 ± 0.09</td>
</tr>
</tbody>
</table>

To make a further evaluation of the proposed ANID method on a more real network environment, a network simulation experiment platform is built by using the Nidsbench Test Platform (Lippmann, Haines, Fried, Korba & Das, 2002). Nidsbench is a well-used network testing software package developed by Anzen, including Tcpreplay and Fraqrouter (Lippmann et al., 2002).
used for the NID model training. Among the training network instances, 30% of them contain network attacks. Another four-day network data collection, including 146,000 instances for the NID performance test. The sample composition of the test samples is listed in Table 8, where including 98,000 normal samples and a total of 48,000 attack samples, where there are 4000 instances are unknown attack types (belonging to the four major attack types, but are not included in the training data, such as DDoS attacks and ARP attacks).

Based on the FRS-FS method, 19-dimensional features are achieved for normal and attack pattern learning and the subsequent online NID. The NID detection details of the test data set from the network simulation platform based on the proposed method are listed in Table 8. The comparative NID results with different methods are listed in Table 9.

Combining Tables 8 and 9, the overall detection accuracies of the proposed method on the simulation platform is 96.52%, which is obviously higher than these comparative methods (over 4 percent than the suboptimal method, CANN).

If we make a differential comparison of the known and unknown attacks, the proposed method can achieve better detection performance on the known attacks, including the higher F1M, lower FAR and MR than these comparative NID methods. With regard to the unknown attacks, the proposed method can make a true positive rate of the unknown attacks as high as 85% and achieve the highest overall evaluation criterion of F1M as 87.67% with a quite low false alarm rate as 0.25%. With respect to the missing reports, the proposed method can achieve a comparable low MR result as the optimal detector by the Firefly-C4.5, whose MR is only lower 0.12% than that of the proposed method. Hence, we can draw a conclusion that the proposed method can achieve higher NID accuracies on the Nidsbench-based network simulation platform with lower false alarm rates than these state-of-art comparative NID methods.

Table 8
NID results of the proposed method on simulation system.

<table>
<thead>
<tr>
<th>Type</th>
<th>Test samples</th>
<th>Detection results</th>
<th>Evaluation criterion (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Normal</td>
<td>Probe</td>
<td>R2L</td>
</tr>
<tr>
<td>Normal</td>
<td>98,000</td>
<td>97,000</td>
<td>650</td>
</tr>
<tr>
<td>R2L</td>
<td>6500</td>
<td>30</td>
<td>90</td>
</tr>
<tr>
<td>Vulnerability</td>
<td>10,000</td>
<td>15</td>
<td>120</td>
</tr>
<tr>
<td>DOS</td>
<td>12,000</td>
<td>281</td>
<td>154</td>
</tr>
<tr>
<td>U2R</td>
<td>10,000</td>
<td>100</td>
<td>90</td>
</tr>
<tr>
<td>Unknown</td>
<td>4000</td>
<td>200</td>
<td>24</td>
</tr>
</tbody>
</table>

Fig. 7. Topological diagram of the simulation network system.
4. Conclusions

This paper presents an ANID method in combination of FRS-FS and GA-GOCMM-based pattern learning. FRS-FS selects the optimum attribute subset by measuring the IGR of each potentially sensitive attribute based on the theory of the FRS. GA-GOCMM can automatically determine the number of clusters based on an incremental learning scheme and avoid the negative influence of the initial cluster centers, to realize the automatic extraction of optimum pattern features of both normal and attack network connection instances for the subsequent pattern matching-based NID. Simultaneously, an online pattern updating strategy is introduced to update the learned normal and attack pattern libraries by mining the frequent patterns and discarding obsolete patterns accompanied with the online NID results. Hence, the proposed NID model can be adaptive to the dynamic change of the network environment and achieve high NID accuracies on both known and unforeseen attacks with low false alarms and missing reporting rate. Extensive validation and comparative experiment results on benchmark data set NSL-KDD and a self-built Nidsbench-based network simulation system show the effectiveness and superiority of the proposed ANID method. It can achieve high detection accuracy on real network environment in term of the physical simulation experiments with low false alarms, which can be widely spread to the real cybersecurity monitoring. Further work could be done by deploying the proposed ANID approach as a multi-agent model on a real distributed physical network environment to enhance the power of the intrusion model on more complex and real network environments.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Credit authorship contribution statement


Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under grant nos. 61501183, 61771492, NSFC for Distinguished Young Scholars under grant no. 61725306 and the joint fund of NSFC and Guangdong provincial government under grant no. U1701261, in part by the Hunan Natural Science Fund under grant no. 2018J3349, the key laboratory of minister of education for image processing and intelligence control (Huazhong University of Science and Technology) under grant no. IPIC2017-03 and the postgraduate student research and innovation projects of Hunan Province under grant no. CX2018B31.

Appendix. List of Acronyms or Abbreviations

<table>
<thead>
<tr>
<th>Acronyms or abbreviation</th>
<th>Full name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NID</td>
<td>Network intrusion detection</td>
</tr>
<tr>
<td>ANID</td>
<td>Adaptive network intrusion detection</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian mixture model</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>Density-based spatial clustering of</td>
</tr>
<tr>
<td></td>
<td>applications with noise</td>
</tr>
<tr>
<td>FRS</td>
<td>Fuzzy rough set</td>
</tr>
<tr>
<td>FS</td>
<td>Feature selection</td>
</tr>
<tr>
<td>FER</td>
<td>Fuzzy equivalence relation</td>
</tr>
<tr>
<td>IGR</td>
<td>Information gain rate</td>
</tr>
<tr>
<td>DOS</td>
<td>Denial of service attack</td>
</tr>
<tr>
<td>U2R</td>
<td>User to root attack</td>
</tr>
<tr>
<td>R2L</td>
<td>Remote to local attack</td>
</tr>
<tr>
<td>Probe</td>
<td>Probing attack</td>
</tr>
<tr>
<td>FP</td>
<td>Frequent pattern</td>
</tr>
<tr>
<td>SV</td>
<td>Survival value</td>
</tr>
<tr>
<td>AC (\tau)</td>
<td>Attenuation coefficient</td>
</tr>
<tr>
<td>OPT (\lambda)</td>
<td>Obsolete pattern threshold</td>
</tr>
<tr>
<td>FPT (\tau)</td>
<td>Frequent pattern threshold</td>
</tr>
<tr>
<td>NPL</td>
<td>Normal pattern library</td>
</tr>
<tr>
<td>APL</td>
<td>Attack pattern library</td>
</tr>
<tr>
<td>CPL</td>
<td>Cache pattern library</td>
</tr>
<tr>
<td>DA</td>
<td>Detection accuracy</td>
</tr>
<tr>
<td>GA</td>
<td>Greedy algorithm</td>
</tr>
<tr>
<td>FAR</td>
<td>False alarm rate</td>
</tr>
<tr>
<td>CAI</td>
<td>Greedy algorithm-based global optimal</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
</tr>
<tr>
<td>NSL-KDD</td>
<td>A new version of the KDD99 data set</td>
</tr>
</tbody>
</table>

Supplementary materials


References


