Hindawi
Security and Communication Networks
Volume 2022, Article ID 6418420, 19 pages
https://doi.org/10.1155/2022/6418420

Research Article

Memory-Augmented Insider Threat Detection with Temporal-Spatial Fusion

Dongyang Li,[1,2] **Lin Yang**,[2] **Hongguang Zhang**,[2] **Xiaolei Wang**,[2] **and Linru Ma**[2]

[1]*Command and Control Engineering College, Army Engineering University of PLA, Nanjing 211101, China*
[2]*National Key Laboratory of Science and Technology on Information System Security, Institute of System Engineering, Academy of Military Science PLA, Beijing 100039, China*

Correspondence should be addressed to Dongyang Li; dongyangli_nj@126.com

Received 14 February 2022; Accepted 22 March 2022; Published 26 April 2022

Academic Editor: Robertas Damaševičius

Insider threat detection is important for the smooth operation and security protection of an organizational system. Most existing detection models establish historical baseline by reconstructing single-day and individual user behaviors, and then treat any outlier of the baseline as a threat. However, such methods ignore the temporal and spatial correlations between different activities, which result in an unsatisfying performance. To address such an issue, we propose a novel insider threat detection method, namely, Memory-Augmented Insider Threat Detection (MAITD), in this paper. Such an idea is motivated by the observation that the combination of individual model that focuses on historical baseline and group model that represents peer baseline can effectively identify the low-signal yet long-lasting insider threats, and reduce the possibility of false positives. To illustrate, our MAITD captures the temporal and spatial correlation of user behaviors by constructing compound behavioral matrix and common group model, and combines specific application scenarios to integrate the detection results. Moreover, it introduces the memory-augmented network into autoencoder to enlarge the reconstruction error of abnormal samples, thereby reducing the false negative rate. The experimental results on CERT dataset show that the instance-based and user-based AUCs of MAITD reach up to 87.54% and 94.56%, respectively, which significantly outperform previous works.

1. Introduction

With the frequent occurrence of data breaches and espionage incidents, insider threat has become one of the major challenges for system security. According to a recent survey, the number of security incidents caused by insiders has increased by 47% since 2018, and keeps increasing with increased economic uncertainty [1]. However, with so much at stake, only 33% organizations believe they are capable of detecting abnormal behaviors within the system [2]. Meanwhile, since the initiators of insider threats are typically authorized employees who clearly know the system framework and security measures, such insider damages are more harmful than external attacks. The Cybersecurity Insider organization even declared that "today's most damaging security threats do not originate from malicious outsiders or malware but from trusted insiders with access to sensitive data and systems—both malicious insiders and negligent insiders" [3]. Therefore, in the face of severe practical challenges, it is urgent to propose effective insider threat detection models to prevent such threats.

According to the latest definition given by the CERT Coordination Center, insider threats refer to threats that are carried out by malicious or unintentional insiders, whose authorized access to the organization's network, system, and data is exploited to negatively affect the confidentiality, integrity, availability, and physical well-being of the organization's information, information systems, and workforce [4]. The insiders generally consist of malicious traitors, hypocritical masqueraders, and unintentional perpetrators. Their attack methods include system damage, data breaches, intellectual property theft, etc. Although the topic of insider

threat detection has been studied for long, locating malicious behaviors precisely is still nontrivial and remains an open challenge.

Since threat scenarios are widely varying, it is impractical to explicitly model malicious threats. Consequently, most existing methods tend to convert the insider threat detection into user behavior anomaly detection problem [5]. To illustrate, security analysts build normal user behavior model by analyzing historical data, and regard the outliers (data out of distribution) as threats. Such methods are based on the assumption that adversarial activities do not follow past habitual patterns, which is followed by us in this paper. Many classical unsupervised learning algorithms have been used to model normal user behavior. Among them, autoencoder quickly became the mainstream insider threat detection algorithm because of their robustness on domain knowledge and strong anti-interference ability [6].

In fact, the performance of insider threat detection depends on not only the anomaly detection algorithm but also the representation quality of user behavior. In our previous work [7], we performed related studies on the feature extractions of user behavior, and categorized them into two types: (i) statistical features based on artificial definition; (ii) hidden features based on representation learning. Although previous methods [8–13] have their own unique insights, they are still faced with the following limitations:

> Existing methods ignore the temporal statistics when representing user behavior, thereby limiting the performance. Although some works [8–10] attempted to address this issue, they only capture the temporal characteristics from a single perspective, i.e., the potential sequence relationship and the variation tendency of activity frequency.
>
> Most baseline models do not consider the spatial correlation from their peers' data, which leads to the collective behavior changes caused by occasional factors such as service outages or environmental changes being misidentified as anomalies, thus increasing unnecessary manual investigation costs.
>
> Previous detection models focus on how to reasonably represent user behaviors, while ignoring the impact of optimizations on anomaly detection algorithms. Such a one-sided preference leaves much room for improvement.

To address the above limitations, we propose a novel insider threat detection model named Memory-Augmented Insider Threat Detection (MAITD). Our model first adopts the frequencies of daily activities as basic features, then employs the temporal-spatial fusion and an unsupervised learning algorithm to improve the overall performance. The so-called temporal-spatial fusion aims at capturing the temporal and spatial statistics of user behaviors by constructing compound matrix and common group model, and integrating the detection results *w. r. t.* different scenarios, thus allowing historical and peer baselines to work together. As for the unsupervised learning algorithm, we choose autoencoder as the baseline model, then additionally introduce a memory module based on attention weights to enlarge the reconstruction errors of anomalies. Note that the temporal and spatial statistics mentioned here are not actually related to time and space but an extension in a broad sense. Specifically, the temporal statistics not only denote the specific temporal information when user behavior occurs but also include the potential sequence relationships and variation tendency of activity frequencies. As for the spatial statistics, it refers to the correlations between peers' behaviors.

The main contributions of this paper are summarized as follows:

(i) We propose a novel user behavior temporal-spatial statistics fusion method to achieve the parallel historical and peer baselines. We perform comprehensive evaluations under different scenarios to obtain the final results. Our evaluation results clearly show the usefulness of MAITD.

(ii) We introduce a memory-augmented network into autoencoder to optimize the unsupervised learning algorithm. By enlarging the reconstruction errors of anomalies, it helps to reduce the false negative rate of detection. To the best of our knowledge, this is the first work to apply memory-augmented network on insider threat detection.

(iii) We perform extensive evaluations on CERT dataset [14] to demonstrate the superiority of our MAITD. The experimental results show that MAITD achieves the state-of-the-art performance on both instance-based and user-based settings.

The rest part of this paper is organized as follows: Section 2 summarizes the related work on insider threat detection, and outlines the differences between our approach and other similar works. Section 3 introduces the research motivation and basic idea. Section 4 presents the overall framework of MAITD and implementation details. Section 5 provides the detailed evaluation and analysis results. Section 6 discusses the limitations of MAITD and future work. Finally, Section 7 concludes this work.

2. Related Work

As an important component of system security protection, insider threat detection has attracted extensive attention from the research community. On the one hand, multiple insider-related projects like ADAMS, CINDER, and SCITE have been released successively by DARPA to prevent confidential data being stolen by insiders [15]. Among them, the latest work of SCITE project suggests that it is a feasible solution to detect insider threat by observing employee's reaction to tentative signals, which provides a new research strategy for reducing manual investigation burden [16]. The technical report released by CERT Insider Threat Center records various practical cases and corresponding mitigation

and preventive measures [17]. On the other hand, there are also many excellent surveys and solutions in the academia. Liu et al. [5] and Homoliak et al. [18] focus on the definition, taxonomy, and categorization of insider threats, and give a detail review of current research situation. Yuan et al. [19] discussed the opportunities and challenges of insider threat detection in the era of deep learning. The threat description models proposed by Pfleeger et al. [20] and Nurse et al. [21] believe that insider threat is the result of interaction of system environment, personal character, and historical behavior. However, it should be noted that the discussion of behavioral models that attempt to correlate insider threats with psychological profiles of users are outside the scope of this paper. Here, the insider threat detection is defined as: "At any given time instance, given their past online activities, how to predict if an employee is behaving abnormally either with respect to his past activity, or with respect to the behavior of his peers" [8]. That is, we simplify insider threat detection as anomaly behavior detection, and focus on how to effectively exploit the temporal and spatial characteristic of user behavior to improve detection performance. In view of the important impact of behavior representation and detection algorithm on solution performance, we will introduce the related studies from the perspective of temporal-spatial characteristic utilization and unsupervised detection algorithm optimization.

2.1. Temporal-Spatial Characteristic Utilization. Most insider threat detection schemes are based on historical baseline or peer baseline, in which the former represents the past habitual pattern of individual user while the latter focuses more on behavioral correlation between members in the same group. Normally, they were not in conflict but complementary, each with its own sphere of competence. However, most solutions only focus on one side (especially historical baseline), leading to the limited detection performance [8]. In order to more clearly elaborate the current research, we made a more fine-grained partition on the basis of previous works. Firstly, in the context of historical baseline schemes, some works tend to capture the temporal characteristics directly by means of the models with temporal learning capabilities. Examples of such models are statistical models [6, 8, 9, 22–24], Hidden Markov Model [11], Graph Embedding Model [25], Recurrent Neural Network [10, 12, 26], and Self-Attention Mechanism [27]. Gavai et al. [8] propose to capture time-varying characteristics by taking a weighted average of activity frequency feature to improve detection performance. Similarly, Chattopadhyay et al. [9] used the temporal indicators such as "Katz fractal dimension" and "total power corresponding to the top five frequencies in the power spectrum of the time-series signal" to generate time-series vectors, and combined with cost-sensitive undersampling technique to detect insider threats. In order to detect the low-signal yet long-lasting threats, Yuan et al. [6] constructed the compound behavioral deviation matrix to represent user behaviors. Different from the above schemes, Duc et al. [22–24] pay more attention on the impact of using the recent time window as a baseline comparison for each data instance, instead of designing new behavior representation. Excluding statistical methods, machine learning techniques have also been applied to building historical baseline models. Rashid et al. [11] used activity sequences as input, the hidden Markov model as modeling approach, and the deviation between predicted results and actual activities as judging criteria to detect anomalies. Liu et al. [25] developed an efficient anomaly detection system based on the graph embedding technique. Considering the powerful representation ability of deep learning models, Tuor et al. [12] and Sun et al. [26] used deep recurrent networks to detect insider threats. To further improve the accuracy of behavior model, Yuan et al. [10] combined the temporal point process with Long Short Term Memory (LSTM) to learn user's normal behavior pattern from four aspects: *activity duration*, *activity type*, *session duration,* and *session interval*. Inspired by position encoding [28], Yuan et al. [27] retained the absolute time information of user activities by calculating the minute offset, and used self-attention mechanism to construct the final behavior representation. Secondly, there are some other historical baseline schemes [13, 29, 30] that prefer to capture temporal characteristics in a round-about way. In these schemes, the model itself is only a modeling method, and training data and training mode are key. In other words, they simply take the individual historical behaviors as the model input. For example, Liu et al. [13] used the "4W" template to reorganize audit logs, and arranged them based on user id in chronological order to form training corpus. This indirect extraction method has the advantage of simplicity, but also confronts the challenge of limited performance.

Compared with plentiful historical baseline schemes, there are a few researches on peer baseline. One possible reason is that it is not easy to define the boundary of peers. Generally speaking, by comparing the difference between individual behavior and his peers' behavior, insider threat detection schemes can reduce the false positive rate in occasions (e.g. service outage and environmental change) where many users have common burst of events. The peers here do not simply refer to those users in the same department, but groups with similar behavioral trends. There are two broad approaches to divide the peers: role-based division [6, 31–33] and cluster-based division [34, 35]. The former arranges users into groups according to their roles, while the latter classifies users by clustering their behavior features. For example, Eldardiry et al. [34] divided users into different peer groups by calculating the similarity between behavioral data. Another issue closely related to peer baseline is how to represent the group's behavior pattern. A common solution is to extract behavior features automatically with the help of neural network model, and the key is that the training samples for network model should be the behavioral data of peer group rather than individual data [32, 36]. In addition, Yuan et al. [6] and Gavai et al. [8] used the statistical average to build peer baseline, but their implementation details are different. To sum up, the behavioral baseline model is closely related to the training mode, and how to effectively represent user behavior is still a problem worth exploring.

2.2. Unsupervised Detection Algorithm Optimization. Since the practical behavioral logs do not contain label information, unsupervised detection methods are the current mainstream study direction. Many classical unsupervised learning algorithms such as K-means Clustering [34], Isolated Forest (IF) [8] and One-class Support Vector machines (OCSVM) [37] have been applied in the field of insider threat detection. However, due to user activity spreading across multiple behavior domains (i.e. complexity), the above methods always achieved suboptimal performance. As a typical representative of reconstruction-based methods, autoencoder is favored by security practitioners because of its robustness on domain knowledge and stronger anti-interference ability. Briefly speaking, an autoencoder-based anomaly detection model only learns how to reconstruct normal samples, so the reconstruction error becomes higher for the abnormal samples. For example, Yuan et al. [6] and Liu et al. [38, 39] used fully connected autoencoders to learn normal behavior pattern, and regarded those whose reconstruction errors exceed predefined threshold as anomalies. However, although these reconstruction-based approaches have achieved fruitful results, there is still much room for improvement.

In order to improve the detection performance of reconstruction-based schemes, researchers have successively proposed various optimization methods. Zhou et al. [40] proposed a robust autoencoder model based on alternative optimization to strengthen the anti-noise capacity. Zong et al. [41] used a deep autoencoding Gaussian mixture model to detect anomalies, and optimized model parameters in an end-to-end way. Nguyen et al. [42] applied the variational autoencoder to anomaly detection, and attempted to explain anomaly from the aspect of gradient descent. Gong et al. [43] used memory-augmented network to optimize anomaly detection scheme, and achieved good results in multiple datasets. On this basis, Park et al. [44] further improved performance by introducing extra loss functions such as intra-class distance and inter-class distance, but its application scope was limited to the computer vision field. Rather than attempting to optimize model architecture, Mirsky et al. [45] chose to use multidetector integration to improve detectability. Besides, Yuan et al. [46] discussed the impact of different reconstruction methods (i.e. single-event prediction or sequence recomposition) on anomaly detection performance. Note that, although these works provide fruitful insights on optimizing unsupervised detection algorithm, it is difficult to apply their model directly on insider threat detection due to the requirement difference for the input data. Hence, how to choose the optimal unsupervised detection algorithm according to application scenarios is an open problem.

Compared with the existing works, our MAITD mainly focuses on the problem of temporal-spatial characteristics fusion, and at the same time chooses the memory-augmented autoencoder as an unsupervised detection algorithm to improve performance. In this regard, Acobe proposed in work [6] is similar to our MAITD, but its compound behavioral deviation matrix loses some critical behavior change information. Specifically, the main differences between MAITD and Acobe are as follows: (i) In terms of temporal characteristic analysis, Acobe generates the compound behavioral deviation matrix by concatenating multiple consecutive single-day feature vectors, while MAITD adds temporal indicators behind basic features to generate specific input for the temporal representation model. In other words, the compound matrix of MAITD contains the initial frequency information in addition to behavior variation information, so it has stronger representational capacity. (ii) In the aspect of spatial characteristic analysis, Acobe uses the same extraction method as temporal characteristics (that is, adding group feature to the compound deviation matrix), while MAITD builds an extra common group model to capture spatial characteristics. (iii) In terms of temporal-spatial fusion, the weighted summation mechanism gives MAITD more flexibility because it can adjust the weight factor according to application scenario, but Acobe is relatively rigid because it uses a single behavior model to simultaneously capture temporal and spatial characteristics. (iv) Unlike Acobe with full-connected autoencoder, MAITD chooses the memory-augmented autoencoder as unsupervised detection algorithm to improve performance.

3. Motivation

Although anomaly detection system is typically not used as a standalone solution, it plays an important role in assisting security analysts in selecting suspicious activities to be further scrutinized. However, the existing solutions are unable to satisfy growing practical demand, and how to improve insider threat detection performance has become a common goal in the research community. Driven by this goal, we first summarize the factors that affect detection performance in combination with application scenarios, and then propose the corresponding improvement measures for the existing problems.

Considering that false negatives and false positives are two critical evaluation metrics of anomaly detection system, we intend to elaborate the existing drawbacks from two aspects: malicious activities that are difficult to identify and normal activities that are easy to misidentify. First of all, some malicious activities will not be completed quickly in a short term, but there will be a long process of "commission." For example, in order to avoid exposure while stealing confidential data, the long-dormant spy usually does not steal numerous confidential documents at once but leaks sensitive data piece-by-piece in a long run. This means that the above threat scenario does not cause immediate behavioral deviation, and only long-term monitoring and analysis of user behavior can detect them. Many solutions [29, 30, 38, 39] only focus on the single-day and individual user behavior features, without considering the changes over multiple consecutive days, making it difficult to detect low-signal but long-term threats. Even though there are some works [6, 8, 9, 22, 24] in exploring the variation tendency of user behavior, they have different limitations, respectively. These schemes either design ideal—overly optimistic—time-varying indicators [9] or have trouble in keeping balance

between precision and overload [6]. Moreover, for autoencoder-based detection schemes, the assumption that anomaly incurs higher reconstruction error does not always hold in practice since those anomalies similar to normal activities can also be reconstructed well [43]. For example, when activity frequency is used as baseline model input, the number of connections to removable devices in malicious scenario (such as data breach) is similar to the frequency in normal scenarios (such as data migration result from device updates), which will make it difficult to distinguish whether the higher reconstruction error is from normal sample or anomaly.

To mitigate the above drawbacks, we plan to reduce the false negative rate of detection scheme from two perspectives. On the one hand, we hope to design more reasonable behavior representation to capture multidimensional temporal characteristics, and at the same time take the difference between different types of behaviors into account. This requirement inspires us to design new temporal indicators while retaining original activity frequency information. On the other hand, we intend to optimize the unsupervised detection algorithm to improve performance. It has been suggested that the memory-augmented network is beneficial to enlarge the reconstruction error of abnormal sample, so we plan to apply it in the field of insider threat detection.

Secondly, some normal behavior deviations can also be misidentified as anomalies. For example, due to the sudden service outage or environmental change, the employee's work pattern will inevitably change (such as longer work hours and more interaction, etc.), but these normal behavioral deviations cannot be correctly identified by the historical baseline model. This phenomenon fully indicates the importance of building a peer baseline. Generally speaking, there often exists certain behavioral correlation between an individual user and his peers, which provides a theoretical basis for the establishment of peer baseline [6, 8]. Given the probability that the whole group members are malicious is extremely small, we can make the following hypothesis: the greater behavioral correlation a user has with the group, the less likely the user is malicious. Based on this assumption, we plan to build an additional common group model to mitigate the above misreporting problem.

4. Methodology

The goal of this paper was to improve insider threat detection precision by analyzing the temporal and spatial characteristics of user behaviors. To this end, we propose a memory-augmented insider threat detection approach named MAITD. We first demonstrate the overall framework of MAITD, and summarize its basic idea and workflow. Then, we elaborate the temporal and spatial representation models and the improved unsupervised detection algorithm, respectively. Finally, we give the temporal-spatial fusion mechanism and corresponding implementation algorithm.

4.1. System Overview. Figure 1 shows the overview of MAITD, which mainly includes four modules: basic feature extraction, temporal characteristic analysis, spatial characteristic analysis, and comprehensive evaluation. The basic feature extraction module is responsible for multisource data collection and feature coding. Specifically, it extracts the behavior frequency features from multisource audit logs according to potential threat scenarios, and feeds them to the temporal and spatial characteristic analysis modules for further processing. In this process, the user's activities over a day are aggregated into a data instance to obtain better tradeoff between detection precision and response time. Based on these basic features, the temporal characteristic analysis module can calculate the temporal indicators according to the sliding window and predefined formulas, and then add them behind basic features to generate the compound behavioral matrix. With the help of historical compound behavioral matrixes and unsupervised detection algorithm, we can build an independent temporal representation model for each individual user. Similarly, the spatial characteristic analysis module will also build an additional spatial representation model for each group, but this model is shared with all the members within the same group. That is, the training space of spatial representation model is expanded from individual historical data to group's historical data, and at the same time the model input is changed from compound behavioral matrix to basic feature vector. After the temporal and spatial representation models are trained, we can obtain the anomaly scores of testing sample in historical and peer baseline, respectively. Finally, the comprehensive evaluation module integrates the above scores in combination with specific application scenarios to generate the final lists of anomaly instances and suspicious users.

Before getting into the details of this scheme, we will state the problem studied in this paper clearly and give the corresponding mathematical formulation. As mentioned above, the insider threat detection problem can be simplified as: "Given employee's past online activities, how to predict if an employee is behaving abnormally either with respect to his past activity, or with respect to the behavior of his peers." Let $\overline{X}$ be the space of all activities, and let $X \subseteq \overline{X}$ be the set of normal activities. Given a sample $S \subseteq X$, group affiliation P and employee's past normal activities, how to construct an unsupervised classifier $h_s(x)$: $\overline{X} \longrightarrow \{0, 1\}$ so that the formula $h_s(x) = 0 \Leftrightarrow x \in X$ is as valid as possible is the real crux of the matter.

Considering that the classifier is trained in an unsupervised manner and the method of providing only anomaly label is inconvenient to the subsequent investigation by security analysts, it is more preferable to generate an ordered list of suspicious users. Note that, the calculation of partial evaluation metrics such as detection rate and precision is closely associated with the organization's investigation budget. In practice, investigation budget represents the available human resources for analyzing the anomaly behavior instances, post-training of the detection system, and performing the necessary actions in response [22]. The more the investigation budget, the larger the range of the threshold that can be set. In addition, our analysis report will distinguish between anomaly instances and suspicious users to

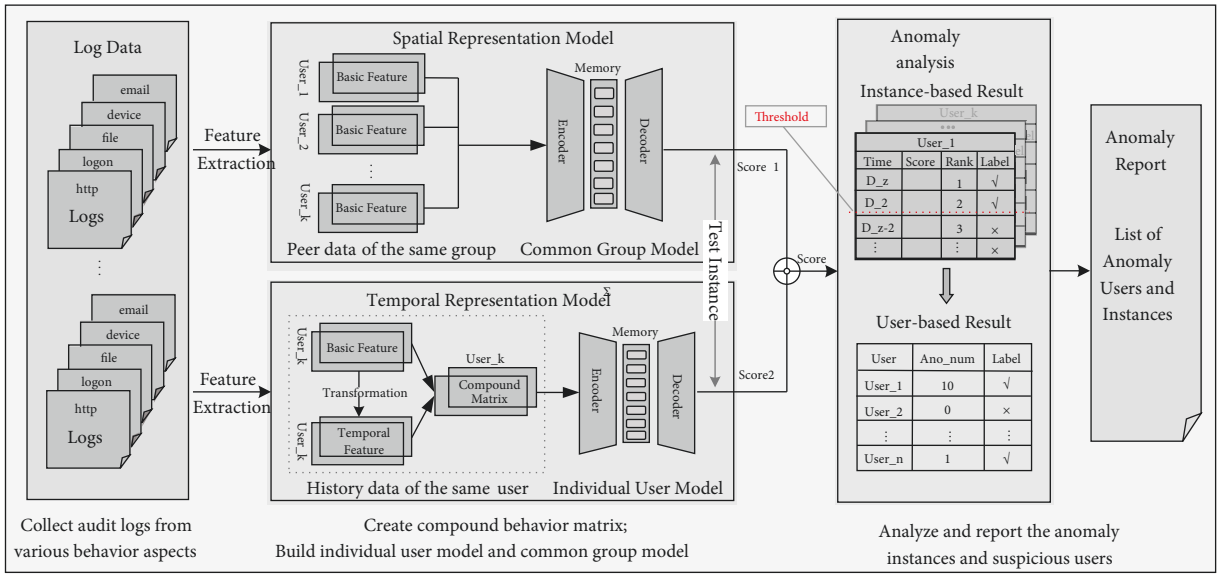

Figure 1: MAITD overview.

provide more comprehensive evaluation results. This is because high malicious instance detection rate is not synonymous with all malicious users being detected, and the latter is the ultimate goal of insider threat detection system. In this paper, suspicious users are defined as the users who have at least one anomaly behavior instances.

4.2. Temporal-Spatial Characteristics. The establishment of historical baseline and peer baseline is the key of insider threat detection scheme, directly influencing the quality of detection performance. In other words, how to make the best of the temporal and spatial characteristics of user behavior is a major challenge. To this end, our MAITD adopts two different methods to capture the temporal and spatial characteristics, respectively.

4.2.1. Temporal Characteristic Analysis. Inspired by Acobe [6], we find that the compound matrix is an effective way to capture the temporal characteristic. Because it can give the behavior model the ability to analyze temporal characteristics by adding time-varying elements to input. Motivated by this, MAITD also chooses the compound behavioral matrix as the input of the temporal representation model, but makes major changes in the element composition. As stated previously, the requirement that behavior representation not only contains multidimensional temporal information but also considers the difference between different types of behaviors gives us the inspiration to design new temporal indicators while retaining original activity frequency information. Therefore, we design a compound behavioral matrix with the structure shown in Figure 2(a) as the temporal representation model input. The compound behavioral matrix mainly consists of basic frequency features and time-varying features, and every feature can be further divided into two subparts (i.e. working hours 8 am to 6 pm and off hours 6 pm to 8 am) according to the occurrence time of user activity. In Figure 2(a), the basic feature set extracted in different behavior domains are arranged in the vertical direction. Given that the basic feature extraction is usually closely related to domain knowledge, here we take the CERT dataset as an example to design a series of basic features such as the number of copying file from other's PC during off hours and the number of visiting recruiting website on office computers during working hours. Here, note that the basic feature extraction is not the focus of this paper, and MAITD adopts the basic features proposed in our previous work, see literature [7] for details.

The horizontal direction in Figure 2(a) represents the different variants of basic behavioral features. The white area represents the normalized frequency information during working hours and off hours, while the blue area is filled with the feature variants (i.e. temporal indicators). As shown in Figure 2(b), the temporal indicators at the monitoring slot are calculated based on the historical values within the sliding window, and the reason behind it is that the sliding window mechanism ensures a smooth transition between the samples. Besides, we can also highlight that the decision of using all features from current and past observations is highly desirable since it may allow a near to optimal automatic weight assignment for past and current versions of each feature and provide a highly interpretable result [24]. In general, the compound matrix proposed in this paper does not simply concatenate multiple consecutive single-day features when capturing the temporal characteristics but relies on the property of temporal indicators. This is because simple data mergence cannot effectively reflect the variation tendency of user behavior but increase the unnecessary computational overhead [22]. Besides, a compound behavioral matrix represents the activity overview of a certain day, and "a certain day" here (e.g. the 5th day in Figure 2(a)) can be regarded as an index used to mark data instances.

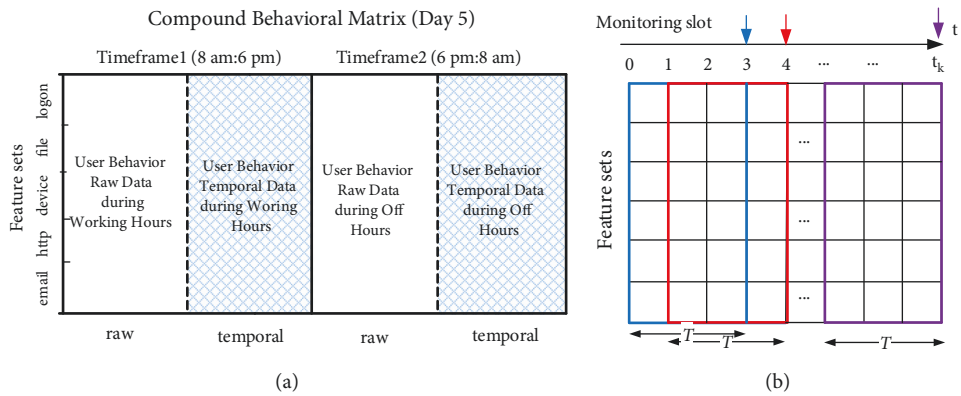

FIGURE 2: Compound behavioral matrix and sliding window mechanism. (a) The compound matrix can be divided into four main components according to activity timeframe and feature type. (b) The temporal indicators at the monitoring slot are calculated based on the historical values within the sliding window.

In fact, MAITD designs two different temporal indicators called standard factor and variation factor to capture the temporal characteristics of user behavior. The former mainly measures the relative size of user's basic features in the current time window, while the latter reflects the cumulative variation of the basic features within the current window. Let $g_{f,l,d}$ denote the numeric measurement of basic feature f in timeframe l on day d. $p_{f,l,d}$ and $q_{f,l,d}$ denote the standard factor and variation factor of basic feature f, respectively. $\overrightarrow{g}_{f,l,d}$ denotes the vector of history numeric measurements, and the detail can be expressed by (1), where l is the occurrence time of user behavior (0 or 1), T is the window size in days.

$$\overrightarrow{g}_{f,l,d} = \left[g_{f,l,i} | i\colon d - T + 1 \le i < d\right]. \tag{1}$$

Then, the standard factor $p_{f,l,d}$ and the variation factor $q_{f,l,d}$ can be calculated by the following transformation of $g_{f,l,d}$:

$$\begin{aligned} p_{f,l,d} &= \frac{g_{f,l,d} - \text{mean}\left(\overrightarrow{g}_{f,l,d}\right)}{\text{std}\, t\left(\overrightarrow{g}_{f,l,d}\right)}, \\ q_{f,l,d} &= (1-\beta) \cdot \sum_{j=1}^{T-1} \left(\beta^{T-j-1} \cdot \left(g_{f,l,d-T+j+1} - g_{f,l,d-T+j}\right)\right), \end{aligned} \tag{2}$$

where $\text{mean}(\overrightarrow{g}_{f,l,d})$ and $\text{std}(\overrightarrow{g}_{f,l,d})$ denote the mean and standard deviation of history measurements, respectively. β is the attenuation coefficient of feature variation, which is usually set as the reciprocal of window size. Actually, the standard factor can be regarded as the standardized transformation of basic feature within the current time window, and the variation factor is the exponentially weighted moving average of basic feature variation. Since the calculations of both indicators are closely relevant to the history measurements within the current window, they can represent the temporal correlation between user activities to some extent. Moreover, we set a lower bound for the standard deviation $\text{std}(\overrightarrow{h}_{f,l,d})$ to avoid worst cases where the standard factor is too large, and the related calculation equation is as follows:

$$\text{std}\left(\overrightarrow{g}_{f,l,d}\right) = \begin{cases} \varepsilon, & \text{std}\left(\overrightarrow{g}_{f,l,d}\right) < \varepsilon, \\ \text{std}\left(\overrightarrow{g}_{f,l,d}\right), & \text{std}\left(\overrightarrow{g}_{f,l,d}\right) > \varepsilon. \end{cases} \tag{3}$$

In addition to designing new temporal indicators, MAITD also takes the difference between different types of behaviors into account. In other words, we assign different weights for basic features and their variants to make the behavior model pay more attention on those features that are most helpful to improve the detection performance. In this regard, we use the same weight setting as Acobe scheme, that is, the weights are lower for chaotic features but higher for consistent features. The specific calculation method is shown in:

$$w_{f,l,d} = \frac{1}{\log_2\left(\max\left(\text{std}\left(\overrightarrow{g}_{f,l,d}\right), 2\right)\right)}. \tag{4}$$

After obtaining the weighted feature values, we should normalize the compound matrix to eliminate the adverse effects caused by different orders of magnitude. Finally, the temporal characteristic analysis module takes the compound behavioral matrix as input, the improved autoencoder as detection algorithm to build an independent temporal representation model for each user, thereby obtaining the anomaly score of behavior instance in historical baseline.

4.2.2. Spatial Characteristic Analysis. Different from temporal characteristic analysis, MAITD does not use the same way to capture spatial correlation between user behaviors, and instead achieves this goal by building an additional common group model. In short, the spatial characteristic analysis module first divides users into groups according to their roles, and then takes the basic feature vectors as input, the improved autoencoder as detection algorithm to build spatial representation model. It should be noted that this model is shared with all the members within same group. Since the group model captures the spatial characteristics based on neural network itself, it can be regarded as an end-to-end and data-driven method, which is similar to work [32, 36].

4.3. Detection Algorithm Optimization. As stated earlier, the autoencoder has become the mainstream insider threat detection algorithm because of its robustness on domain knowledge and stronger anti-interference ability. Therefore, this paper also chooses the deep autoencoder as the basic detector, and adds a memory module to improve detection performance. In general, the autoencoder consists of an encoder to obtain the compressed representation from the input and a decoder that can reconstruct the input purely based on the compressed representation. In the context of anomaly detection, the autoencoder is usually trained by minimizing the reconstruction error on the normal samples, and then uses the reconstruction error as an indicator of anomalies. Since the autoencoder only learns how to reconstruct the seen normal behaviors, those poor reconstructions result from behaviors that have not yet been seen. However, this does not mean that all anomalies can be detected effectively, because those abnormal samples that have many similarities with normal behaviors can also be reconstructed well. Inspired by work [43], we augment the deep autoencoder with a memory module based on attention weight to enlarge the reconstruction error of abnormal samples, thus achieving the goal of reducing the false negative rate. The intuition behind introducing a memory module is that there is a larger differential between the abnormal sample and the sample reconstructed from the normal behavior representations. With that in mind, we apply memory-augmented network to insider threat detection problem, and propose the improved unsupervised detection architecture shown in Figure 3.

Compared with the traditional autoencoder, the improved autoencoder (i.e. Autoencoder-Mem) adds a memory module between the encoder and the decoder to reprocess the compressed representation. In a way, the memory module can be regarded as a storage component used to record prototypical normal behavior patterns. It is through this component that MAITD can map the abnormal samples to the most relevant normal behavior patterns for reconstruction, resulting in an output significantly different to the anomaly input. Based on this mechanism, the reconstruction errors of abnormal samples can be further enlarged. Specifically, given an input x, the encoder Φ first obtains the initial compressed representation $\Phi(x)$. By using the compressed representation $\Phi(x)$ as a query, the memory module retrieves the most relevant items m_i in the memory M via the attention-based addressing operator [43] to generate the reprocessed representation $\Phi(x)'$, which is then delivered to the decoder for reconstruction. After decoding the reprocessed representation $\Phi(x)'$, the decoder Ψ can obtain the reconstructed sample $\hat{x}$, and calculate the mean square error between $\hat{x}$ and x as the model output. In this process, the essence of mapping is to reconstruct the compressed representation based on the prototypical normal behavior patterns recorded in memory, and it is actually realized by using attention-based memory addressing. Let query z denote initial compressed representation $\Phi(x)$, M denote the memory network with prototypical normal behavior patterns, and w denote the attention weight row vector of each item in M for the query z. Then, the weight entry w_i of w and reprocessed query $\hat{z}$ can be obtained by the following equations:

$$w_i = \frac{\exp\left(zm_i^T / \|z\| \|m_i\|\right)}{\sum_{j=1}^{N} \exp\left(zm_j^T / \|z\| \|m_j\|\right)}, \quad \forall i \in \{1, 2, \ldots, N\}. \tag{5}$$

$$\begin{aligned} &\hat{z} = w \cdot M = \sum_{i=1}^{N} w_i \cdot m_i, \\ &\text{s.t.} \quad \sum_{i=1}^{N} w_i = 1, \end{aligned} \tag{6}$$

where row vector m_i is any item of memory network M, representing a prototypical normal behavior pattern. The dimension of item m_i is same to query z, and N is the capacity of the memory network M.

In addition, we also applied the sparse addressing method proposed in Ref. [47] when reconstructing the query z to make the memory network learn more accurate normal behavior patterns:

$$\widetilde{w} = \frac{\overline{w_i}}{\|\overline{w}\|_1}, \overline{w_i} = \frac{\max(w_i - \lambda, 0) \cdot w_i}{|w_i - \lambda| + \epsilon}, \tag{7}$$

where λ is the lower bound of the weight m_i in the sparse addressing process, and ϵ is a very small positive scalar to avoid divide-by-zero exception. After finishing the above sparse addressing operation, we can obtain the reprocessed query $\hat{z}$ based on (6). Simply speaking, the sparse addressing encourages the model to represent an example using fewer but more relevant memory items, leading to learning more informative representations in memory.

Due to the introduction of the memory module, we adjust the objective function used in the training process. In addition to minimizing the reconstruction error on each sample, we take the sparsity characteristic of memory-augmented network into account, and add the sparsity regularizer on attention weight $\widetilde{w}$. The final objective function is shown in (8), where K is the size of the training set, and α is a hyper-parameter in training. Note that, despite using a new objective function in the training stage, we still use the l_2-norm based mean square error, i.e., score $= \|x - \hat{x}\|_2^2$, to measure the anomaly score of test sample.

$$\text{Loss} = \frac{1}{K} \sum_{i=1}^{K} \left(\|x_i - \hat{x}_i\|_2^2 - \alpha \cdot (\widetilde{w}_i \cdot \log(\widetilde{w}_i)) \right). \tag{8}$$

The basic explanation of this optimization mechanism can be summarized as follows: during training, the encoder and decoder are dedicated to minimizing the reconstruction error, and the memory module is simultaneously updated to record the prototypical normal behavior patterns. At the test stage, the learned parameters of encoder, decoder, and memory module are fixed, and the reconstruction is obtained from a few selected memory items of normal behaviors. Thus, the reconstruction tends to be close to the normal sample, resulting in larger errors in abnormal behavior instances.

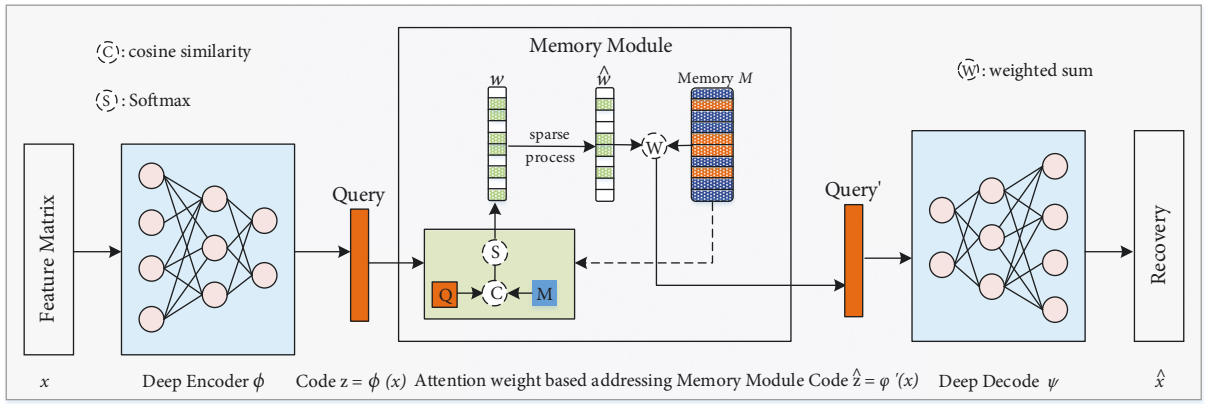

Figure 3: The improved anomaly detection model architecture.

4.4. Fusion Analysis. After building the temporal and spatial representation models, we can obtain the anomaly scores of test sample in historical and peer baseline, which are denoted by sco_t and sco_s, respectively. Subsequently, the comprehensive evaluation module calculates the final anomaly score of test sample by integrating the above detection results, and generates the final lists of anomaly instances and suspicious users according to whether it exceeds the predefined threshold. As for the specific fusion method, MAITD chooses the classical weighted summation way:

$$\text{sco} = \xi \cdot \text{sco}_s + (1 - \xi) \cdot \text{sco}_t, \tag{9}$$

where ξ is a hyper-parameter, representing the proportion of the spatial characteristic analysis module in the whole detection model. As mentioned previously, the setting of threshold is closely associated with the organization's investigation budget. In this paper, we adopt the following decision strategy:

$$\text{thre} = \text{mean}_{\text{train}} + \sigma \cdot \text{std}_{\text{train}}, \tag{10}$$

where $\text{mean}_{\text{train}}$ and $\text{std}_{\text{train}}$ denote the mean and standard deviation of anomaly scores of training samples, respectively. σ is the system input (i.e. investigation budget), which is set to 3 in this paper. Based on the above mechanism, MAITD can obtain the list of anomaly instances for each user, and label the users who have at least one anomaly behavior instance as suspicious users. Algorithm 1 shows the details of the whole insider threat detection scheme. Specifically, after the initialization work is completed, we can build the temporal representation model by constructing compound matrix and training the memory-augmented autoencoder (lines 2–10). Then in line 11, we split users into different groups according to group affiliation **P**. Next, the spatial representation model is built based on the basic features of group members (lines 12–22). Finally, our comprehensive evaluation module can generate the final anomaly scores of test samples according to the fusion mechanism and report the suspicious behavior set Λ and suspicious user set $\mathbf{U_a}$ (lines 23–33).

5. Evaluations

To verify the effectiveness and feasibility of MAITD, we performed extensive experiments on the CERT insider threat dataset [14]. We first introduce the dataset, evaluation metrics, and the experimental setting used in this paper and then compare MAITD with other representative schemes in detail. Next, we analyze the effectiveness of spatial-temporal fusion mechanism and the improved anomaly detection algorithm. Finally, we discuss the impact of the parameters on the detection performance.

5.1. Dataset. The CERT dataset released by Carnegie Mellon University is the most widely used public dataset in the field of insider threat detection. It contains multiple versions that simulate the daily behaviors of internal employees in different organizations. In this paper, we choose the latest r6.2 release as the primary dataset to evaluate the detection performance of MAITD, and at the same time use the classical r4.2 release as the secondary dataset to verify its generalization performance. These two datasets record the daily behaviors of 4000 and 1000 employees of different organizations within 516 days, and provide 5 predefined insider threat scenarios as detection objects. Specifically, these activities cover five behavior domains: logon, device, file, http, and e-mail, and the threat scenario can be regarded as a specific combination of the above activities. Since the malicious activities are usually rare in the real world, the class-imbalance problem is also embodied fully in these datasets. Such a phenomenon explains why supervised classification methods are not suitable for insider threat detection to a certain degree. Besides, due to the excessive overhead of processing the entire dataset (200G), we select several user groups to form a subset to conduct performance evaluation. During this process, in addition to the necessary groups of anomaly users, we also randomly select multiple groups without anomaly users to simulate the class-imbalance situation. Table 1 lists the main information of the dataset used in this paper.

Like most insider threat detection schemes [6, 9, 22, 39], we chose the following evaluation metrics, namely, detection

(i) Input: user set $\mathbf{U}$, behavior instance set Ω, group affiliation $\mathbf{P}$, threshold σ
(ii) Output: suspicious behavior set Λ , suspicious user set $\mathbf{U_a}$
(1) $\varnothing \leftarrow \Lambda$, $\varnothing \leftarrow \mathbf{U_a}$, $\varnothing \leftarrow \Omega_{\text{train}}^{\text{group}}$
(2) for $u \in \mathbf{U}$ do
(3) $\quad \Omega^u = \Omega_{\text{train}}^u + \Omega_{\text{test}}^u$
(4) $\quad$ for $x \in \Omega_{\text{train}}^u$ do
(5) $\quad\quad \widehat{x_t^u} \leftarrow x_t^u \leftarrow x$ //calculate the basic feature x_t^u and compound matrix $\widehat{x_t^u}$
(6) $\quad$ end for
(7) $\quad$ while not converged do
(8) $\quad\quad$ train the memory-augmented temporal model Model_t on $\widehat{x_t^u}$
(9) $\quad$ end while
(10) end for
(11) $\mathbf{G} \leftarrow \mathbf{U}, \mathbf{P}$ //split users into different groups according to group affiliation $\mathbf{P}$
(12) for group $\in \mathbf{G}$ do
(13) $\quad$ for $u \in$ group do
(14) $\quad\quad \Omega_{\text{train}}^{\text{group}} = \Omega_{\text{train}}^{\text{group}} \cup \Omega_{\text{train}}^u$
(15) $\quad$ end for
(16) $\quad$ for $x \in \Omega_{\text{train}}^{\text{group}}$ do
(17) $\quad\quad x_t^{\text{group}} \leftarrow x$ //calculate the basic feature vector x_t^{group}
(18) $\quad$ end for
(19) $\quad$ while not converged do
(20) $\quad\quad$ train the memory-augmented spatial model Model_s on x_t^{group}
(21) $\quad$ end while
(22) end for
(23) for $u \in \mathbf{U}$ do
(24) $\quad \text{thr} \leftarrow (\text{sco}_{\text{train}}, \sigma)$
(25) $\quad$ for $x \in \Omega_{\text{test}}^u$ do
(26) $\quad\quad \text{sco}_s \leftarrow (\text{Model}_s, x^{\text{group}}), \text{sco}_t \leftarrow (\text{Model}_t, x^u), \text{sco} \leftarrow (\text{sco}_s, \text{sco}_t)$
(27) $\quad\quad$ if sco > thr then
(28) $\quad\quad\quad \Lambda_u = \Lambda_u \cup \{x\}$
(29) $\quad\quad\quad \mathbf{U_a} = \mathbf{U_a} \cup \{u\}$
(30) $\quad\quad$ end if
(31) $\quad$ end for
(32) end for
(33) return suspicious behavior set Λ and suspicious user set $\mathbf{U_a}$

Algorithm 1: Memory-augmented insider threat detection approach with temporal-spatial fusion

rate (DR), precision (PR), F1-score, and the area under the receiver operating characteristic curve (AUC). Their calculation methods are as follows:

$$\begin{aligned} DR &= \frac{TP}{TP + FN}, \\ PR &= \frac{TP}{TP + FP}, \\ FPR &= \frac{FP}{TN + FP}, \\ F1 &= \frac{2}{PR^{-1} + DR^{-1}}, \end{aligned} \tag{11}$$

where true (false) positive (TP/FP) represents the number of malicious (normal) samples that are correctly recognized as "malicious," and false (true) negative (FN/TN) denotes the number of malicious (normal) samples that are incorrectly recognized as "normal." Among these metrics, AUC plays a more important role in evaluating solution performance because it is independent of the predefined threshold. In general, the larger the area under the curve, the better the performance of detection scheme. For the sake of brevity, all the performance metrics except AUC are reported in percent (%). Moreover, since the performance evaluation is reported in terms of both anomaly instance detection and suspicious user identification, there are two kinds of performance metrics: Instance-based (IDR, IPR, IF1, IAUC) and User-based (UDR, UPR, UF1, UAUC).

We implement the MAITD with Pytorch, in which both the temporal and spatial representation models adopt the architecture of 6-layer fully connected autoencoder plus a memory module. The related parameters are set as follows: the number of hidden units at each layer in the encoder and decoder are 256, 128, 64 and 64, 128, 256, respectively. The hyper-parameters $N, \lambda, \alpha, \epsilon$ in the memory module and ε in the compound matrix are, respectively, set as 100, 0.02, 0.002, 10^{-4}, and 0.01 according to the reference works [6, 43]. In addition, for each anomaly group, the training set includes the data from the first collection day until roughly one month before the date of the labeled anomalies, and the testing set includes the dates from then until roughly one month after the labeled anomalies. For normal groups, the user's behavior dataset is split into a training set and a testing

Table 1: Summary of dataset.

Dataset	Feature count	Mal_user: Nor_user	Mal_instances: Nor_instances
R6.2	112	5 : 812	45 : 51720
R4.2	112	70 : 866	966 : 55194

set in chronological order, and the splitting ratio is set to 30%. During training, we set the batch size and epochs as 32 and 40, and use Adam under the default parameters to optimize the detection model. All experiments are performed on compute nodes with Gold 5118 CPU, GTX 2060GPU, and 128 GB RAM, and the averaged results are reported after repeating 10 times.

5.2. Comparison with Other Works. In order to verify the superiority of MAITD, we compare it with other representative insider threat detection schemes. In this section, we select four similar works [6, 8, 9, 38] as comparison objects, and discuss their detection performance on the following three aspects: temporal representation model, spatial representation model, and the whole detection scheme. Among them, Acobe [6] is also designed to build the historical baseline and peer baseline simultaneously, but it adopts different spatial and temporal feature extraction methods (see related work for details). Gavai [8] and Pratik [9] design their own indicators to capture the temporal characteristics, respectively, and Liu [38] propose a simple autoencoder-based insider threat detection scheme. Prior to analysis, we first introduce the naming rules of experimental schemes to facilitate understanding. The name of experimental scheme consists of two parts: the former part "**" means the initial feature extraction method (i.e. basic features), and the latter part "##" represents the temporal or spatial representation model. For example, MAITD_Acobe_S denotes the detection scheme which adopts the MAITD's basic features and Acobe's spatial representation method.For the purpose of comparing temporal and spatial representation models, we design the following experiment scenarios. Under the conditions of same dataset, unsupervised detection algorithm, and parameter settings, we generate the comparative schemes by combining different basic features, temporal and spatial representation models, and then compare their performance to verify the superiority. Figure 4 shows the performance comparison results of different temporal and spatial representation methods on the r6.2 dataset. It can be seen from the left subgraph that the temporal representation method of MAITD has the best performance among all the temporal representation methods, and Acobe is better than Pratik and Gavai. This result further verifies the previous conclusion that simple data mergence cannot effectively reflect the variation tendency of user behavior. As for the poor performance of Partik and Gavai schemes, we think that the method of applying temporal indicators in other fields on insider threat detection directly is too ideal to get remarkable results. Likely, the experimental results in the right subgraph show that even if the basic features are changed, the common group model of TSDIM also performs better than the compound behavioral deviation matrix of

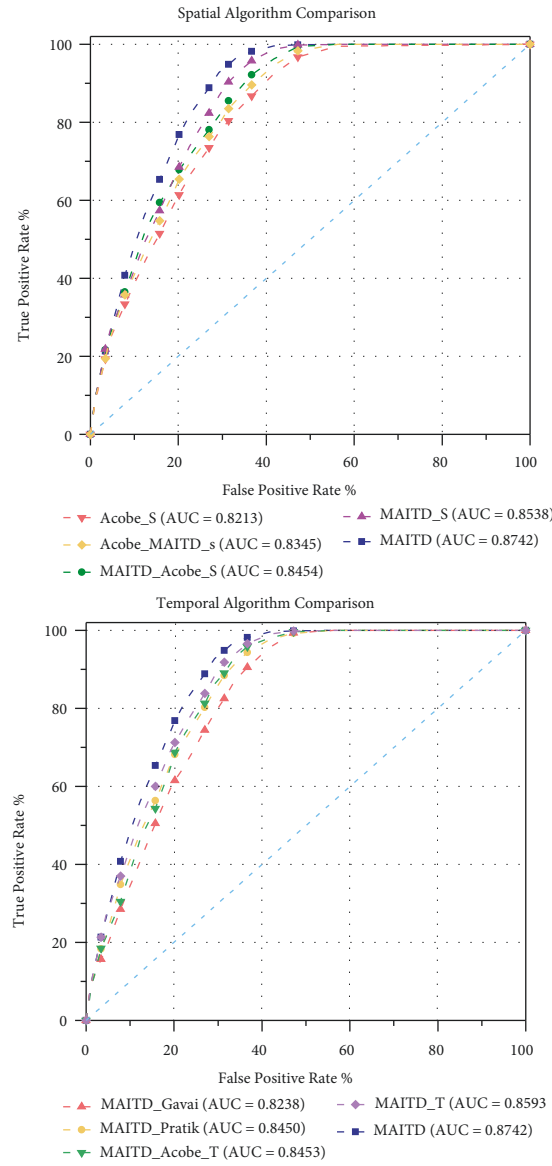

Figure 4: ROCs on r6.2 with different temporal and spatial representation methods.

Acobe in capturing the spatial characteristics. This is because Acobe only relies on the average of users' basic features in the same group when capturing the spatial characteristic, which loses much correlation information between peer's behaviors.Subsequently, we compare the whole detection scheme with other solutions. In addition to the recommended parameter setting, we also use some parameter tuning tools such as *hyperopt* [48] to optimize detection model when reimplementing comparative schemes. Figure 5 presents the experiment results based on the r6.2 dataset. As can be seen from this figure, MAITD outperforms other

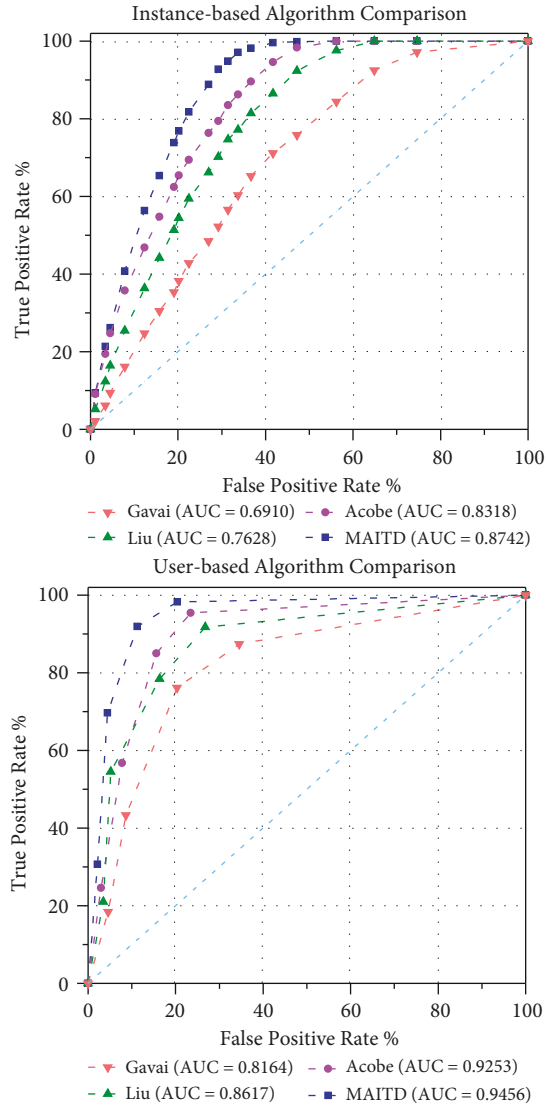

Figure 5: ROCs on r6.2 with different insider threat detection schemes.

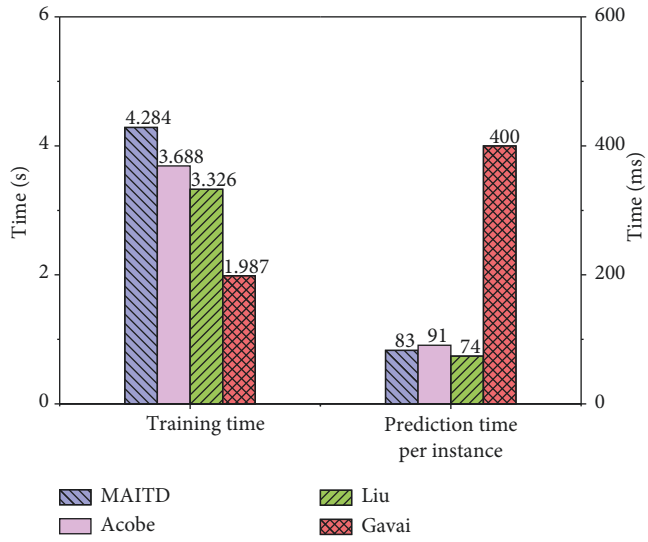

Figure 6: Average training time and prediction time per instance of different detection algorithms on r6.2.

three detection schemes in either case, and the instance-based and user-based AUC are, respectively, improved by 3.94% and 2.03% than the suboptimal scheme, which directly demonstrates the superiority of our scheme. Acobe gets suboptimal performance despite some defects in the aspect of temporal and spatial characteristic analysis. Surprisingly, the Gavai scheme with the ability of temporal characteristic analysis is weaker than the simple autoencoder-based scheme Liu. We think one possible reason is that the isolation forest used in Gavai is not suitable for insider threat detection. Because the success or failure of IF-based detection scheme is heavily dependent on the choice of good features and proper predefined contamination parameters, this prior knowledge usually is not known for security practitioners.

Moreover, we also make a comparison of model training time and prediction time per instance. As shown in Figure 6, the prediction time per instance of MAITD is shorter than Acobe despite opposite result in terms of training time. It can be explained through different baseline model construction methods, where MAITD adopts two independent representation models but others only build one individual model. But it should be noted that MAITD is not penalized in prediction time per instance as the individual and group models can be run in parallel. Besides, the size of the compound matrix of Acobe is significantly larger than MAITD, which invisibly introduces a large amount of computational overheads, thereby leading to longer prediction time. Although other schemes perform better than our TSDIM in training and prediction time, their poor detection performance also mean much human resources and additional investigation overheads. In general, our MAITD not only improves the insider threat detection performance but also takes into account the real-time requirement as much as possible.

Finally, to eliminate the adverse effects of accidental factors (such as the dataset is atypical or too small) on performance evaluation, we use r4.2 dataset to conduct the same comparative experiments. Compared with r6.2 release, r4.2 release has more positive samples and different organization background, so we think it is reasonable to verify the generality based on r4.2 release. More specifically, we record and compare multiple performance metrics to make a fair assessment, and the detailed information can be seen in Table 2 and Figure 7. It can be seen that the TSDIM scheme outperforms other methods even if the dataset used is changed. In addition, although the overall performance of MAITD on r4.2 is better than that on r6.2, the performance gap with other detection schemes is reduced. That is, we think that MAITD has more advantages in dealing with complex threat scenarios.

5.3. Ablation Study. In the previous section, we made a comprehensive comparison with other representative detection schemes. In the following, we will conduct several further ablation studies to verify the effectiveness of two

Table 2: The summary of insider threat detection results.

Data	Type	DR ($\sigma = 3$)				PR ($\sigma = 3$)				AUC			
		MAITD	Acobe	Liu	Gavai	MAITD	Acobe	Liu	Gavai	MAITD	Acobe	Liu	Gavai
r6.2	Instance	**69.06**	68.79	65.26	64.18	**50.34**	44.26	40.64	38.16	**0.8742**	0.8318	0.7628	0.6910
	User	**100**	**100**	80	80	**67.56**	63.15	59.91	47.43	**0.9456**	0.9253	0.8617	0.8164
r4.2	Instance	75.48	**75.61**	68.13	67.74	**53.72**	49.23	42.67	35.48	**0.8936**	0.8646	0.8127	0.7267
	User	**86.42**	83.36	80.62	72.64	**62.13**	59.86	55.37	49.35	**0.9551**	0.9434	0.9042	0.8673

The threshold used to calculate DR and PR is set as mean + 3*std, and the unit of DR/PR is percent. The bold values represent the maximum value i.e., the best performance among these methods.

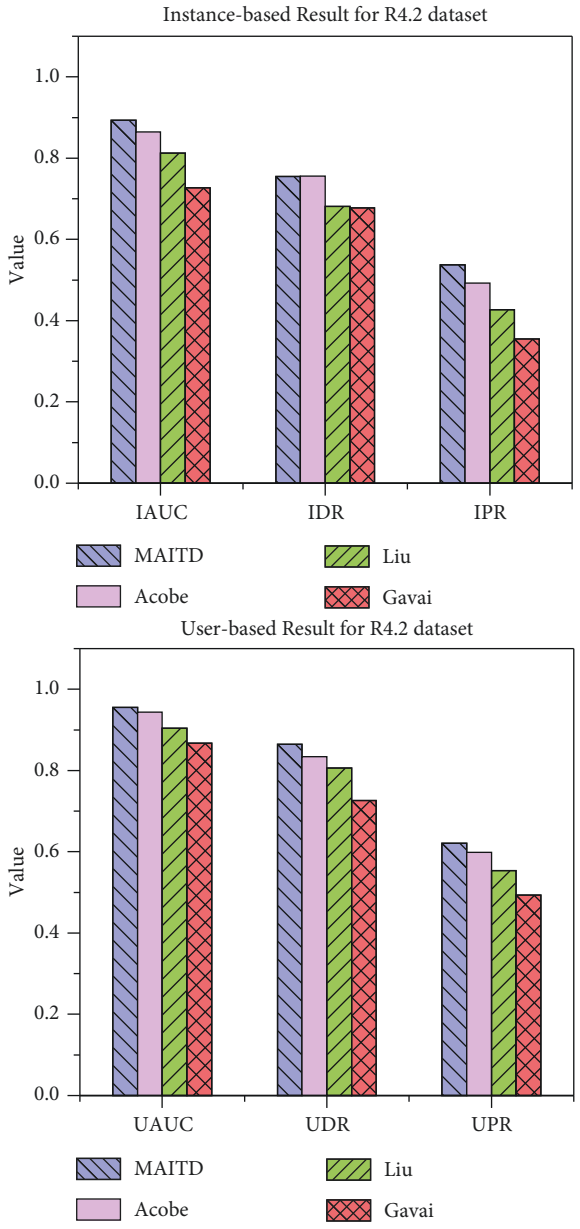

Figure 7: Performance comparison results of different insider threat detection schemes on r4.2.

optimization components. The naming rules of detection scheme are similar to the previous ones, but the latter part "##" can also represent the different unsupervised detection algorithms. For example, "MAITD-S″ refers to the detection scheme with the basic features and spatial representation model used in the MAITD method, and "MAITD-Vae" refers to the detection scheme which adopts the MAITD's temporal-spatial representation model and variational autoencoder-based detection algorithm.

The first experiment is used to evaluate the effectiveness of temporal and spatial characteristic analysis components. In this experiment, we choose the memory-augmented autoencoder as the unsupervised detection algorithm, and achieve the experimental goal by removing temporal or spatial representation models. Figure 8 records the instance-based and user-based ROCs on r6.2 dataset with different behavior representation models. It can be seen that removing either the temporal representation model or spatial representation model will degenerate the performance. Without the temporal (spatial) representation model, the insider threat detection scheme cannot capture the temporal (spatial) correlation between user activities, which may lead to the missing (false) alarms of the low-signal yet long-lasting threats (the collective behavior changes caused by occasional factors). Moreover, we also notice that the temporal representation model and the spatial representation model have the similar detection performance (the gap in IAUC is only 0.0055) when deployed separately, but there is still a certain gap (the gap in IAUC is 0.02) between them and the fusion scheme. This phenomenon indicates the necessity of temporal-spatial characteristic fusion in the field of insider threat detection. However, although the detection scheme with single representation model is worse than the fusion scheme, it performs better than the scheme without any representation model, thereby verifying the effectiveness of temporal and spatial characteristic analysis modules.

The second experiment is used to evaluate the effectiveness of the improved unsupervised detection algorithm. To have a fair comparison, all detection schemes leverage the temporal-spatial fusion component to capture potential correlation between user behaviors, and we choose two other detection algorithms as comparison objects. Here, we refer to the detection scheme with a fully connected autoencoder as Baseline, and generate new schemes by adding a variational mechanism (denoted as MAITD-Vae) and memory module (denoted as MAITD-Mem). Figure 9 shows the ROCs on r6.2 with different unsupervised detection algorithms. Experimental data show that the variational autoencoder provides limited performance improvement (0.0079), but the improvement brought from memory module (0.0236) is 3 times the former. This is because the variational autoencoder is designed to strengthen the ability to resist noise data, and it plays an important role in

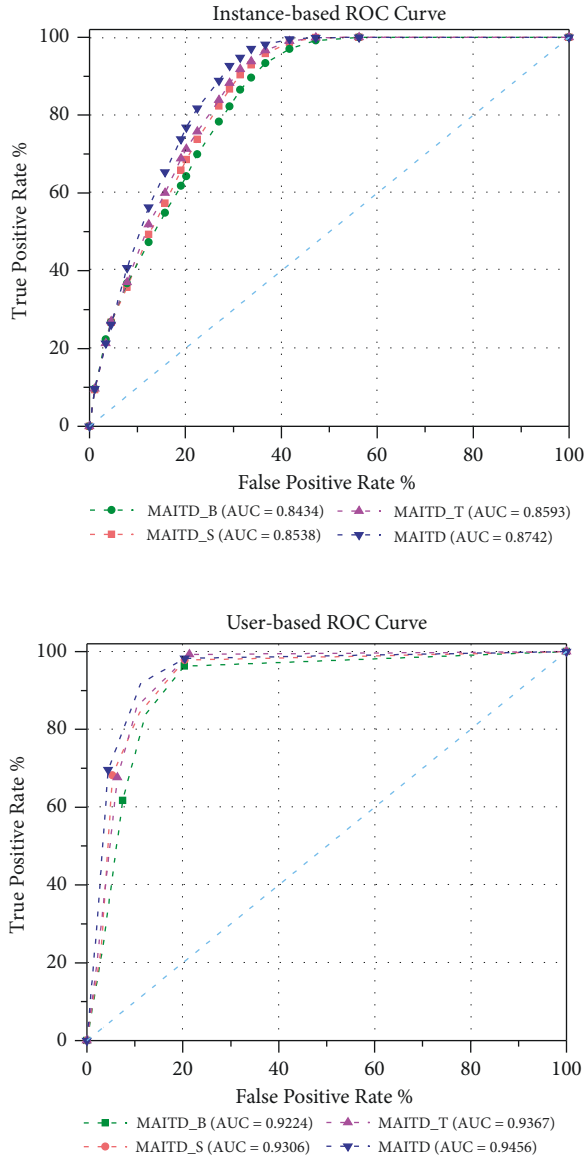

Figure 8: Rocs on r6.2 with different behavior representation models.

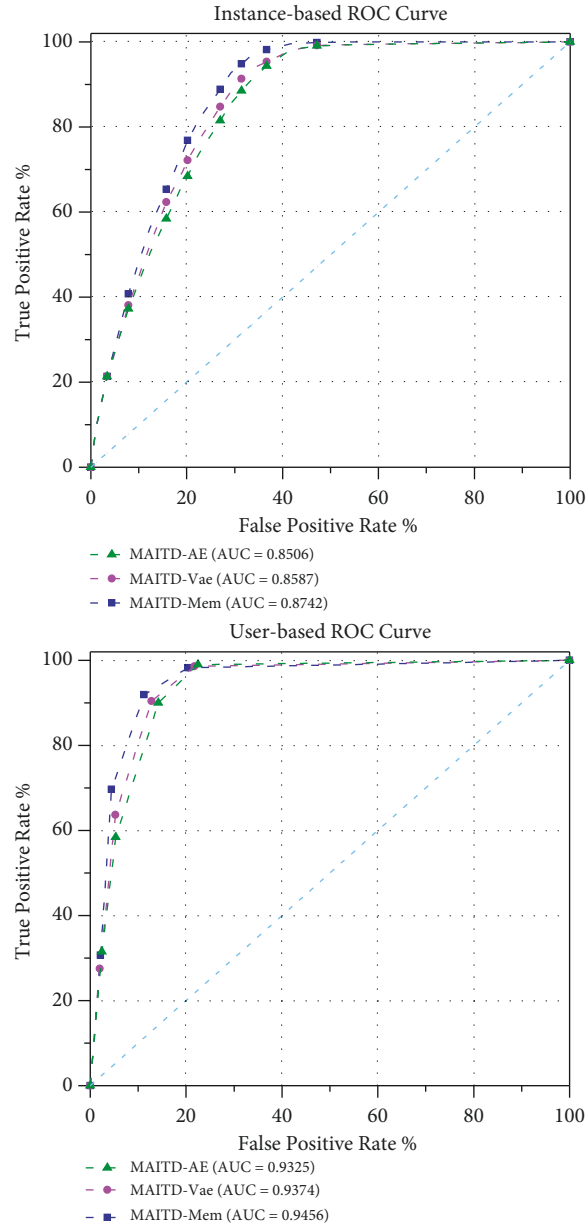

Figure 9: ROCs on r6.2 with different detection algorithms.

decreasing reconstruction errors of normal samples. Although this method is beneficial to distinguish anomalies in a way, it still faces the problem that some anomalies can also be reconstructed well. Instead, the memory-augmented autoencoder alleviates this problem by enlarging the construction errors of anomalies, which is more in line with the realistic demand of insider threat detection.

To explore the possible explanation for the optimization components, we analyze the trends of anomaly scores of two different users, and give the related case study. The detailed information can be seen in Figure 10, in which CMP2946 is a malicious user and LYB3419 is a normal user. The black curve denotes the anomaly scores of MAITD, and the gray curve denotes the anomaly scores of the MAITD-B scheme. The star markers at the bottom indicate the actual anomaly days. The false negative, false positive, and true positive are depicted by red, purple, and blue points, respectively. By comparing the number of red points in Figure 10(a), we can conclude that our two optimization components effectively reduce the false negatives in the detection results. Here, we take the behaviors of user CMP2946 on February 24, 2011 as a case, and briefly analyze the reasons for different results in two detection schemes. By studying the action sequence of the user on the day, we find that the number of visitors to the recruiting website is much less than the previous few days, and the number is even similar to the frequence during the normal period. This makes it difficult for detection scheme based on basic features to identify such anomalies. However, in addition to the initial frequence information, the temporal representation model of the MAITD scheme can capture the time-varying information hidden in user behaviors, so as to detect the above anomalies accurately. Furthermore, we can observe that the anomaly scores of anomaly user (CMP2946)

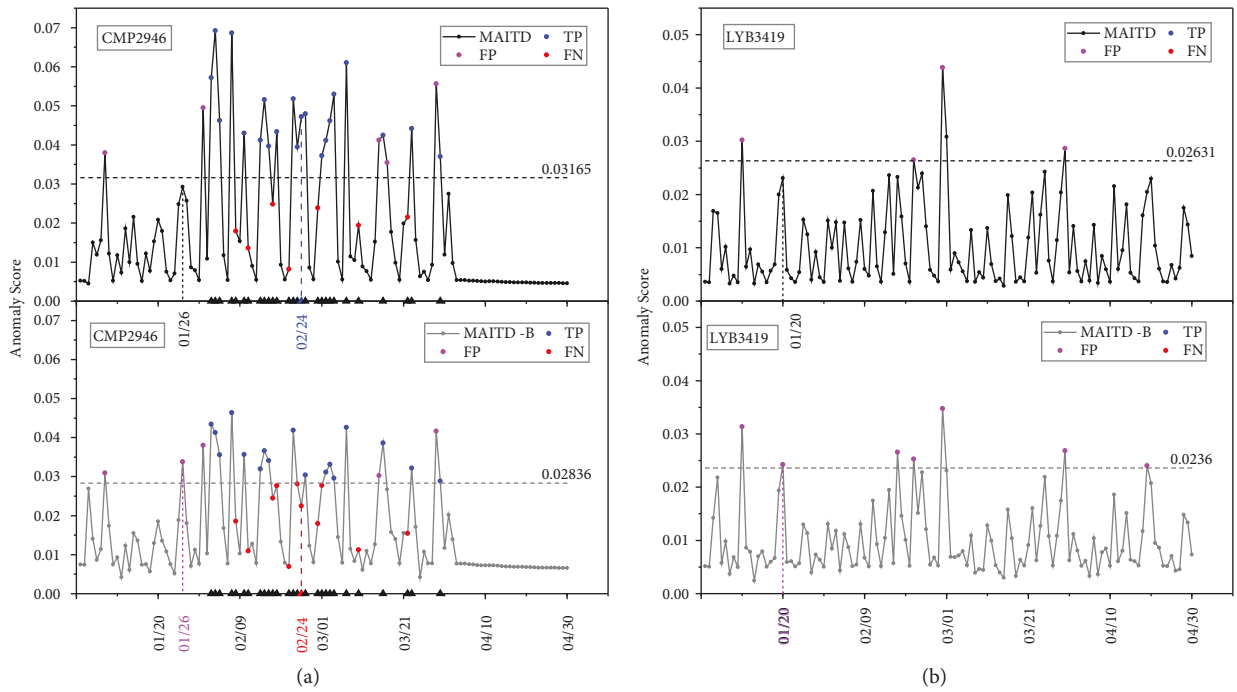

Figure 10: Trends of anomaly scores of different users on r6.2. (a) Malicious user CMP2946. (b) Normal user LYB3419.

in MAITD is obviously higher than that in MAITD-B scheme, while the normal user (LBY3419) has nearly the same anomaly scores. This phenomenon is consistent with the desired purpose of optimization components, thus verifying the feasibility of applying the memory-augmented network on insider threat detection.

Combined with the variation of purple points in Figure 10(b), we believe that two optimization components also play an important role in reducing the false positives. For example, two normal instances (the behaviors of user CMP2946 on January 26, 2010 and the behaviors of user LBY3419 on January 20, 2010) are successfully transformed from false positive samples to true negative samples. However, since there are no specific descriptions about occasional factors in the CERT dataset, we cannot conduct in-depth analysis to explain these phenomena. We think the application of spatial representation model is the main reason, and work [6] makes the same guess. In summary, the temporal-spatial fusion mechanism and the improved unsupervised detection algorithm are practical and feasible optimization measures.

5.4. Parameter Analysis. When describing the MAITD, we emphatically introduce two parameters, the size of sliding time window T and the model weight coefficient ξ, to help optimize the detection performance. The size of sliding window T is related to the scale of historical data used by the temporal representation model, and the weight coefficient ξ is responsible for adjusting the balance between historical baseline and peer baseline. Generally speaking, the larger the window size, the more historical information the temporal representation model can leverage, and the more advantageous for the detection of low-signal yet long-term anomalies. However, an overlarge window size will also weaken the short-term variation of user behavior, thereby lowering the ability to detect the sudden appearing threats. Meanwhile, there is no one-fit-all weight coefficient for every threat, and its value usually depends on the specific threat scenario. For example, in organizations with relatively obscure roles and functions, the behavioral patterns of members in the same group are not similar, so the importance of peer baseline in the whole detection system should be weakened. For these reasons, we prefer to obtain the best values of these parameters in the CERT dataset through numerical experiments.

Firstly, we evaluate the impact of window size T on the detection performance. Figure 11(a) shows that when other parameters are fixed, multiple evaluation metrics vary with different window size T. Note that the y-axis of Figure 11 represents the value of multiple evaluation metrics. It can be seen from the figure that the detection precision of anomaly instances and suspicious users shows an upward trend with the increase of window size, but this trend goes into reverse when the window size reaches a relatively large value. This phenomenon is line with our expectation, that is, the increase of window size expands the scale of historical data that the temporal representation model can leverage, but overlarge window size also hinders the acquisition of weak variation feature. Moreover, we also observed that other metrics such as detection rate show similar trend despite weak amplitude of variation, and it can be explained through the following two reasons. First, there are only a few anomaly instances and fewer malicious users in the dataset, which limits the variable range of evaluation metrics. Second, some rare anomalies are inherently difficult to detect based on

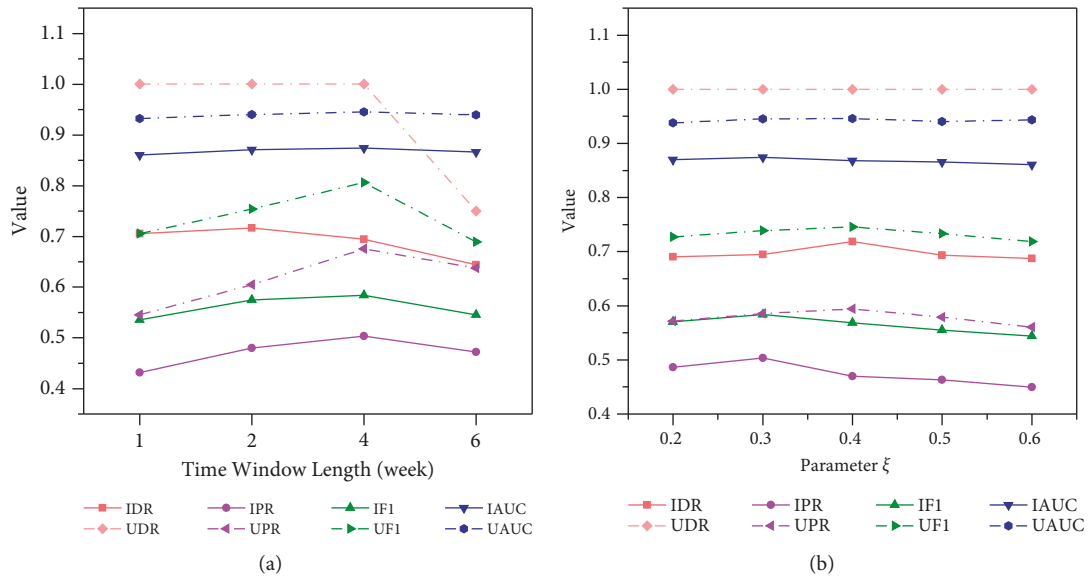

Figure 11: Parameter comparison results on r6.2. (a) The relationship between window size T and performance. (b) The relationship between parameter ξ and performance.

Table 3: The detection results of different weight coefficients.

Para ξ	Instance-based results				User-based results			
	IDR	IPR	IF1	IAUC	UDR	UPR	UF1	UAUC
0.1	69.04	48.61	57.05	0.8701	100	57.15	72.73	0.9382
0.2	69.46	**50.34**	**58.37**	**0.8742**	100	58.56	73.86	0.9456
0.3	**71.86**	49.01	58.28	0.8693	**100**	**59.45**	**74.57**	**0.9461**
0.4	69.34	48.31	56.95	0.8657	100	57.91	73.34	0.9406
0.5	68.74	45.98	55.10	0.8608	100	56.05	71.83	0.9438

The bold values represent the maximum value i.e., the best performance among these methods.

historical baselines. Therefore, we believe that the whole performance of insider threat detection scheme is at a relatively high level when the size of sliding time window is set to be 4 weeks (i.e. one month).

Secondly, we also design a numerical experiment to evaluate the impact of model weight coefficient on detection performance. Figure 11(b) and Table 3 show that when time size is set to be 4 weeks, the evaluation metrics vary with different weight coefficient ξ. It is observed that compared to the window size T, the impact of parameter ξ on detection performance is not so significant, but the variation trend of evaluation metrics is similar. Specifically, when the weight of the spatial representation model is at a relatively small level, increasing the value of parameter ξ is beneficial to improve the detection performance. But, it is undeniable that the temporal representation model plays a more important role in the whole detection process. From the data in Table 2, it can be concluded that when the weight coefficient ξ is set to be 0.3, multiple evaluation metrics are generally high.

6. Discussion and Future Work

Below we discuss limitations and future works. First of all, MAITD is still an insider threat detection scheme based on feature engineering in the traditional sense, and its many improvement measures are based on the premise of good basic features design. In other words, it is necessary for MAITD to enhance the detection ability of completely unknown threats, and this is also a common drawback of the traditional insider threat detection schemes based on feature engineering. To solve this, one option is to obtain the abstract representation of user behaviors by means of natural language processing technology. That is, we need to design a feature extraction scheme which can capture the potential semantic properties in the original audit logs without relying on any domain knowledge. Another possible solution is to encode discrete event logs into activity sequences, and build user's behavior profile to detect insider threat by utilizing the process mining method. Secondly, given that the MAITD detection model is trained on a fixed set of historical data, the online and system evolution in reality may cause significant performance degradation. Therefore, how to update detection model incrementally on newly arriving data to achieve consistently good performance with negligible cost is another important research direction. In this regard, Parveen [49] and Sun [26] provide an important reference to achieve this goal. Moreover, like most insider threat detection schemes, the results of MAITD lack intuitive interpretability and require further artificial investigation. In response to this problem, we believe that improving the

detection granularity of anomaly instances may be another feasible solution except for model interpretability study. In a way, the result itself has a certain interpretability when the detection granularity reaches the event level. We must admit that there is no one insider threat detection scheme which can detect all anomalies accurately without artificial investigation, and all that we can do is to reduce the investigation overhead as much as possible. In summary, although our MAITD suffers from some weaknesses, it provides an effective reference for other anomaly detection problems in the security field. Meanwhile, we will implement and verify the above possible solutions in future work, and positively contribute to a successful application of the proposed system in real-world scenarios.

7. Conclusion

Most existing insider threat detection schemes only focus on the historical behavior baseline while ignoring the peer baseline, resulting in poor detection performance. To solve this problem, we propose a novel insider threat detection scheme named MAITD, which adopts two different optimization measures to improve detection performance. First, it captures the temporal and spatial characteristics of user behaviors by constructing a compound behavioral matrix and common group model, and combines specific application scenarios to integrate the detection results, so as to enable both historical and peer baselines to work together. Second, it adds a memory module based on attention weight to autoencoder to enlarge the reconstruction error of the anomalies, and alleviate the false negatives. The experimental results on CERT datasets show that MAITD outperforms the latest insider threat detection scheme, and improves the instance-based and user-based AUC by 3.94% and 2.04%, respectively.

Data Availability

All the data used during the study were provided by Carnegie Mellon University CERT Insider Threat dataset. The dataset can be downloaded at https://kilthub.cmu.edu/articles/dataset/Insider_Treat_Test_Dataset/12841247/1.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by a research grant from the National Science Foundation of China under Grant nos. 61772271 and 62106282.

References

[1] P. Institute, "2020 cost of insider threats global report," https://www.proofpoint.com/us/resources/threat-reports/2020-cost-of-insider-threats.

[2] Gurucul, "2020 insider threat survey report," https://gurucul.com/2020-insider-threat-survey-report.

[3] C. Insiders, "2020-cyber-threat-intelligence-report," https://www.cybersecurity-insiders.com/portfolio/2020-insider-threat-report-darktrace/.

[4] D. L. Costa, M. J. Albrethsen, and M. L. Collins, *Insider Threat Indicator Ontology*, Carnegie-Mellon Univ Pittsburgh Pa Pittsburgh United States, Pittsburgh, PA, USA, Tech. Rep, 2016.

[5] L. Liu, O. De Vel, Q.-L. Han, J. Zhang, and Y. Xiang, "Detecting and preventing cyber insider threats: a survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1397–1417, 2018.

[6] L.-P. Yuan, E. Choo, T. Yu, I. Khalil, and S. Zhu, "Time-window based group-behavior supported method for accurate detection of anomalous users," in *Proceedings of the 2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 250–262, IEEE, Taipei, Taiwan, JJune 2021.

[7] D. Li, L. Yang, H. Zhang, X. Wang, L. Ma, and J. Xiao, "Image-based insider threat detection via geometric transformation," *Security and Communication Networks*, vol. 2021, Article ID 1777536, 2021.

[8] G. Gavai, K. Sricharan, D. Gunning, R. Rolleston, J. Hanley, and M. Singhal, "Detecting insider threat from enterprise social and online activity data," in *Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats*, pp. 13–20, Colorado, DN, USA, October 2015.

[9] P. Chattopadhyay, L. Wang, and Y.-P. Tan, "Scenario-based insider threat detection from cyber activities," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 3, pp. 660–675, 2018.

[10] S. Yuan, P. Zheng, X. Wu, and Q. Li, "Insider threat detection via hierarchical neural temporal point processes," in *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)*, pp. 1343–1350, IEEE, Los Angeles, CA, USA, December 2019.

[11] T. Rashid, I. Agrafiotis, and J. R. Nurse, "A new take on detecting insider threats: exploring the use of hidden Markov models," in *Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats*, pp. 47–56, Vienna, Austria, October 2016.

[12] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," in *Proceedings of the Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, March 2017.

[13] L. Liu, C. Chen, J. Zhang, O. De Vel, and Y. Xiang, "Insider threat identification using the simultaneous neural learning of multi-source logs," *IEEE Access*, vol. 7, pp. 183 162–183 176, 2019.

[14] J. Glasser and B. Lindauer, "Bridging the gap: a pragmatic approach to generating insider threat data," in *Proceedings of the 2013 IEEE Security and Privacy Workshops*, pp. 98–104, IEEE, San Francisco, CA, USA, May 2013.

[15] A. P. Moore, D. A. Mundie, and M. L. Collins, "A system dynamics model for investigating early detection of insider threat risk," in *Proceedings of the 31st International Conference of the System Dynamics Society*, Pittsburgh, PA, USA, July 2013.

[16] S. Wasko, R. E. Rhodes, M. Goforth et al., "Using alternate reality games to find a needle in a haystack: an approach for testing insider threat detection methods," *Computers & Security*, vol. 107, Article ID 102314, 2021.

[17] M. Collins, *Common Sense Guide to Mitigating Insider Threats*, Carnegie-Mellon Univ Pittsburgh Pa Pittsburgh United States, Pittsburgh, PA, USA, Tech. Rep, 2016.

[18] I. Homoliak, F. Toffalini, J. Guarnizo, Y. Elovici, and M. Ochoa, "Insight into insiders and it: a survey of insider threat taxonomies, analysis, modeling, and countermeasures," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–40, 2019.

[19] S. Yuan and X. Wu, "Deep Learning for Insider Threat Detection: Review, Challenges and Opportunities," *Computers & Security*, vol. 104, Article ID 102221, 2021.

[20] S. L. Pfleeger, J. B. Predd, J. Hunker, and C. Bulford, "Insiders behaving badly: addressing bad actors and their actions," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 169–179, 2009.

[21] J. R. Nurse, O. Buckley, P. A. Legg et al., "Understanding insider threat: a framework for characterising attacks," in *Proceedings of the 2014 IEEE Security and Privacy Workshops*, pp. 214–228, IEEE, San Jose, CA, USA, May 2014.

[22] D. C. Le and N. Zincir-Heywood, "Anomaly detection for insider threats using unsupervised ensembles," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1152–1164, 2021.

[23] P. Ferreira, D. C. Le, and N. Zincir-Heywood, "Exploring feature normalization and temporal information for machine learning based insider threat detection," in *Proceedings of the 2019 15th International Conference on Network and Service Management (CNSM)*, pp. 1–7, IEEE, Halifax, Canada, October 2019.

[24] D. C. Le and N. Zincir-Heywood, "Exploring adversarial properties of insider threat detection," in *Proceedings of the 2020 IEEE Conference on Communications and Network Security (CNS)*, pp. 1–9, IEEE, Avignon, France, August 2020.

[25] F. Liu, Y. Wen, D. Zhang, X. Jiang, X. Xing, and D. Meng, "Log2vec: a heterogeneous graph embedding based approach for detecting cyber threats within enterprise," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1777–1794, London, UK, November 2019.

[26] D. Sun, M. Liu, M. Li, Z. Shi, P. Liu, and X. Wang, "Deepmit: a novel malicious insider threat detection framework based on recurrent neural network," in *Proceedings of the 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 335–341, IEEE, Dalian, China, May 2021.

[27] S. Yuan, P. Zheng, X. Wu, and H. Tong, "Few-shot insider threat detection," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 2289–2292, Virtual Event, Ireland, 2020.

[28] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5998–6008, Vancouver, Canada, 2017.

[29] D. C. Le, N. Zincir-Heywood, and M. I. Heywood, "Analyzing data granularity levels for insider threat detection using machine learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 30–44, 2020.

[30] T. E. Senator, H. G. Goldberg, A. Memory et al., "Detecting insider threats in a real corporate database of computer usage activity," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1393–1401, Chicago,USA, August 2013.

[31] K. Nance and R. Marty, "Identifying and visualizing the malicious insider threat using bipartite graphs," in *Proceedings of the 2011 44th Hawaii International Conference on System Sciences*, pp. 1–9, IEEE, Kauai, HI, USA, January 2011.

[32] L. Liu, C. Chen, J. Zhang, O. De Vel, and Y. Xiang, "Doc2vec-based insider threat detection through behaviour analysis of multi-source security logs," in *Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 301–309, IEEE, Guangzhou, China, January 2020.

[33] D. Zhang, Y. Zheng, Y. Wen et al., "Role-based log analysis applying deep learning for insider threat detection," in *Proceedings of the 1st Workshop on Security-Oriented Designs of Computer Architectures and Processors*, pp. 18–20, Toronto, Canada, October 2018.

[34] H. Eldardiry, E. Bart, J. Liu, J. Hanley, B. Price, and O. Brdiczka, "Multi-domain information fusion for insider threat detection," in *Proceedings of the 2013 IEEE Security and Privacy Workshops*, pp. 45–51, IEEE, San Francisco, CA, USA, May 2013.

[35] A. Coden, W. Lin, K. Houck et al., "Uncovering insider threats from the digital footprints of individuals," *IBM Journal of Research and Development*, vol. 60, no. 4, pp. 8–1, 2016.

[36] P. A. Legg, O. Buckley, M. Goldsmith, and S. Creese, "Automated insider threat detection system using user and role-based profile assessment," *IEEE Systems Journal*, vol. 11, no. 2, pp. 503–512, 2015.

[37] L. Lin, S. Zhong, C. Jia, and K. Chen, "Insider threat detection based on deep belief network feature representation," in *Proceedings of the 2017 International Conference on Green Informatics (ICGI)*, pp. 54–59, IEEE, Fuzhou, China, August 2017.

[38] L. Liu, O. De Vel, C. Chen, J. Zhang, and Y. Xiang, "Anomaly-based insider threat detection using deep autoencoders," in *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 39–48, IEEE, Singapore, November 2018.

[39] L. Liu, C. Chen, J. Zhang, O. De Vel, and Y. Xiang, "Unsupervised insider detection through neural feature learning and model optimisation," in *Proceedings of the 13th International Conference on Network and System Security*, pp. 18–36, Sapporo, Japan, December 2019.

[40] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 665–674, Halifax, Canada, August 2017.

[41] B. Zong, Q. Song, M. R. Min et al., "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada, February 2018.

[42] Q. P. Nguyen, K. W. Lim, D. M. Divakaran, K. H. Low, and M. C. Chan, "Gee: a gradient-based explainable variational autoencoder for network anomaly detection," in *Proceedings of the 2019 IEEE Conference on Communications and Network Security (CNS)*, pp. 91–99, Washington, DC, USA, August 2019.

[43] D. Gong, L. Liu, V. Le et al., "Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1705–1714, Montreal, Canada, 2019.

[44] H. Park, J. Noh, and B. Ham, "Learning memory-guided normality for anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14360–14369, Seattle, WA, USA, March 2020.

[45] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," 2018, https://arxiv.org/abs/1802.09089.

[46] L.-P. Yuan, P. Liu, and S. Zhu, "Recompose event sequences vs. predict next events: a novel anomaly detection approach for discrete event logs," in *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pp. 336–348, Virtual Event, Hong Kong, May 2021.

[47] B. Zhao, L. Fei-Fei, and E. P. Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *Proceedings of the CVPR 2011*, pp. 3313–3320, IEEE, Colorado Springs, CO, USA, June 2011.

[48] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the International Conference on Machine Learning. PMLR*, pp. 115–123, Edinburgh, Scotland, July 2013.

[49] P. Parveen and B. Thuraisingham, "Unsupervised incremental sequence learning for insider threat detection," in *Proceedings of the 2012 IEEE International Conference on Intelligence and Security Informatics*, pp. 141–143, IEEE, Washington, DC, USA, June 2012.