Relational Deep Learning Detection with Multi-Sequence Representation for Insider Threats

Abdullah Alshehri
Department of Information Technology
Faculty of Computer Science and Information Technology
Al Baha University
Alaqiq 65779-7738, Saudi Arabia

***Abstract*—Insider threats are typically more challenging to be detected since security protocols struggle to recognize the anomaly behavior of privileged users in the network. Intuitively, an insider threat detection model depends on analyzing the audit data, representing trusted users' activity streams, on recognizing malicious behaviors. However, the audit data is high dimensional data in that it presents n dependent streams of activities where it establishes a complex feature extraction. In this context, the dependent streams represent user activities where each activity is represented by an ordered set of real variables that pertain to a specific occurrence, such as log-in records. As a result, multiple actions can be represented simultaneously, with one or more values being recorded at each timestamp. Moreover, the relations between dependent streams are typically neglected while detecting the anomaly behavior. Ideally, relation learning is commonly considered to recognize occurrence patterns in streaming data. Thus, the latent relations are thought to have insight for the accurate detection of anomaly behavior concerning insider threats. This study introduces a novel model to detect insider threats by representing audit data as multivariate time series to explicitly learn the existing inter-relations between activity streams using a Recurrent Neural Network (RNN). The model considers learning the latent relationships to effectively extract features for modeling the behavior profile where anomaly behavior can be detected accurately. The evaluation, using the CERT dataset has shown that the proposed model outperforms the comparator approaches to insider threats detection with AUC of 0.99.**

***Keywords*—Insider threats; machine learning; recurrent neural network; user behavior analytic**

I. Introduction

With the rapid advancement and growth in networking technology, cyber threats have become a significant issue for numerous companies and organizations worldwide [1]. A cyber threat can mainly be realized by breaching network security. Ideally, the primary option for malicious intent to breach network security is by using a malware [2]. In this context, the malware contravenes the secured network by an external malicious component such as rootkits, Trojan horses, viruses, and worms [3]. Thus, the ideal solution to secure the network from such external threats is by proposing a perimeter defense, e.g., firewalls, antivirus software, and intrusion prevention/detection systems.

However, a cyber attack can be triggered from an internal source in the network; it is well known as an insider threat [4]. A typical form of an insider threat is that the legitimate user may conduct harmful work at the network, such as leaking, altering, or disrupting sensitive data. Thus, an insider threat (malicious) is typically realized as an abnormal action, or behavior, in the network flows that is performed by the legitimate user [5]. Fig. **??** shows a conceptual illustration of the internal and external attacks intuition which can affect such a network in cyber space. It can be observed that the external attack is transparent to be prevented/detected by the perimeter defense protocols. On the other hand, the insider attack is commonly deceptive as the perimeter defense can hardly detect attacks conceived from inside the network. Therefore, insider attacks pose a critical challenge in the cyber security domain where the demand to propose practical solutions to detect insider threats remains a desirable solution for a tremendous number of organizations in the market [6].

To detect insider threats, the typical solution relies on developing systems that are capable of analyzing the user's behavior to discover anomalies in the network [7]. The idea is to observe the user's daily activities and tasks where these activities yield frequent network usage patterns. Ideally, the regular activities can underline insightful patterns to map a typical behavior for the legitimate user. In this context, the ideal method to analyze the user's behavior is accomplished using Machine Learning (ML)-based approaches, (see for instance [8], [9], [10], [11]). ML approaches, such as Hidden Markov Model (HMM) and Support Vector Machine (SVM), have been utilized to detect insider threats via modeling the behavioral profile from the audit data (daily activities) such as the log events. Typically, these ML approaches, which are well known as shallow learning methods [12], are subject to attentive feature engineering to model the behavior profile accurately. The reason is that the audit data is composed of a large volume of unstructured, high-dimensional, and sparsity instances, which makes extracting features a non-trivial task. The traditional approaches have modeled the user's behavior by aggregated data consisting of the user's activities within a single day. However, missing some features can cause unpredictable behavior where it imposes unbalanced detection alarms; for example, it can increase the false alarms in the system. Deep Learning (DL), a subset of ML approaches, has been employed to address the drawback mentioned earlier for insider threats detection [13], [4], [14], [15]. DL provides the advantage that features can be represented immediately from unstructured data [16]. Moreover, it has the advantage that features can be extracted sequentially to reduce missing temporal feature learning, a way that is not applicable in the case of shallow learning methods.

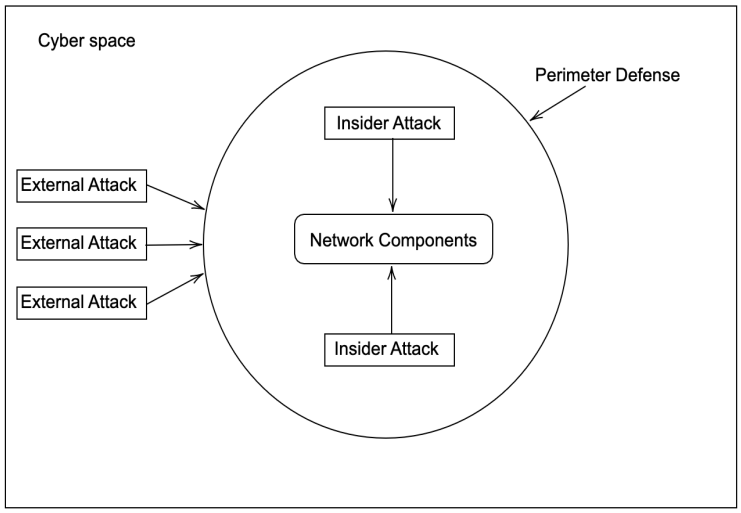


Fig. 1. A Conceptual Illustration Shows the Intuition of External and Internal Attacks in Cyber Space Networks.

Nevertheless, insider threats detection approaches neglect considering the temporal representation of multivariate sequences to model the user's behavior, where this limitation has established a downside to developing an accurate detection model. Intuitively, the user activities are temporally recorded as sequences of dependent variables, i.e., at each timestamp, one or more variables, such as user id and log-in/off time, would be recorded simultaneously. Therefore, incorporating more extensive dependent variables can increase the chance of better modeling the user's behavior. It is worth mentioning that increasing data volume results in improving the learning accuracy as per Bonferonni's principle [17]. This, in turn, brings the motivation to structure the entire audit data as temporal sequences, i.e., multivariate time series representation, which thought it is fruitful to map all possible behavior patterns of the user. Moreover, as the audit data consist of multi-sequential actions, the relations between these sequences are not well considered in previous studies. Thus, the conjecture is that the sequences related to one user would underline strong relations to describe unique user-related patterns used for insider threat detection.

This study proposes a novel model that utilizes DL to learn the user's behavior for insider threats detection using a multivariate representation of audit data. The model represents the user activities as a set of dependent sequences in the temporal domain to where the hidden relations are extracted and learned. In concise, each activity is represented as an ordered set of actual values that refers to some event, such as log-in records. Thus several activities can be represented simultaneously such that one or more values are recorded at each timestamp, i.e. the user activities are represented as multivariate time series streams such that at each time tick, n values are recorded temporally. We then use Recurrent Neural Network (RNN) to learn the existing latent temporal relationships between sequences to map the hidden patterns. Thereafter, the model serves to extract features where the recognized behavior is classified as normal or anomaly, which indicates a possible insider threat.

The contributions of the proposed work can be summarized as follows:

i) The study has presented a novel method to model the user behavior in a multivariate time series structure. More specifically, the user behavior is represented using all sequences of events that denote the user activities. The intuition is that time series frequencies would present accurate, readable user behavior patterns because they incorporate exclusive features, not only aggregated features.

iii) The developed model has considered learning and extracting the relations between the multivariate sequences to extract patterns in training data. The existing sequences hold latent relations that can be extracted for accurate behavior modeling, leading to the accurate prediction of anomaly behavior.

iii) The proposed model has applied deep learning to extract features from inter-correlation streams that represent audit data of user activities. The importance is that user activities are stochastic and unstructured, so deep learning is ideal for extracting features from unstructured data.

The remainder of this paper is structured as follows. Section II presents an overview of the related work. This is followed by Section III where the proposed model has been introduced. In Section IV, we demonstrate the evaluation and results of the proposed model. Section V provides conclusions and future directions where this study can be extended.

II. Related Work

ML has been increasingly used for cyber threats detection throughout the previous decade [18]. Generally speaking, ML has shown to be advantageous in the identification and classification of anomalous occurrences in the network streams [19], [20]. Insider threat is a well-known type of cyber attack [21] that has received considerable work using ML approaches. The insider attack detection model heavily depends on the user's

daily activities where the behavior is recognized. Typically, the obtained data is unstructured and complex due to the diversity of the user's activities on the network. Thus, modeling the user's behavior is a relatively intricate task. In the literature, most ML approaches for insider threat detection are data-driven in that the user activities are aggregated to underline representative features where the behavior profile is modeled. In this context, insider threat detection would be proposed based on different examples of data instances where the detection model is handled as an anomaly detection problem. Accordingly, various ML methods have been developed for insider threat detection. For example, SVM is used as a one-class detection method for insider attack [22]. The study presented in [9] had proposed an HMM model to map the typical user behavior based on weekly activities. The insider attack is detected by computing a deviation score between sequences; a low probability score could indicate a probable attack. In [8] a set of supervised and unsupervised ML approaches have been evaluated for insider attack detection. The study had used Self Organization Map (SOM), HMM, and Decision Tree to model malicious behavior for anomaly detection. The features were extracted into two categories, including numerical and sequential features. Whatever category was being employed, the features were aggregated to a set of weekly representative instances. Thus the detection model was complex due to the need for extensive feature engineering.

DL methods have been proposed to tackle the issue of requiring feature extraction and handling the large volume of features to be learned in the detection model for insider attacks data. In [5] a comprehensive survey has introduced the state-of-the-art of DL with insider threat detection. In this context, a number of DL applications have been recruited such as deep feed-forward neural network [23], [24], recurrent neural network (RNN) [14], [25], conventional neural network[26], and graph neural network. [13], [27]. Due to the complexity of data structure, the majority of DL approaches have focused on representing sub-sequences, such as one-day activities, for detection granularity. In practice, each session is a subsequence that denotes a series of activities, i.e. "log-in" and "log-off" events. Whenever a subsequence contains malicious activity, the subsequence will be designated as a malicious subsequence where a possible attack could occur. Therefore, detecting abnormal actions is difficult due to the limited information (features) that can be leveraged. Moreover, the relation extraction between sequences is ignored; however, the extraction of latent relations can bring insight for better modeling of user's behavior.

This study addresses the above-mentioned drawbacks by representing all activities (sequences) as multivariate time series streams where the relations between streams are also considered for building the behavior profile. The study also endeavors to leverage the advantage of using RNN for effective feature extraction from the temporal/sequential data. Recall that RNN has shown effective feature learning of the sequential data for anomaly detection [28], [29], [30].

III. Proposed Model

This section elaborates on the proposed insider threat detection model. Fig. **??** illustrates an overview of the model flows. As it can be seen from the figure that the model operates in several consecutive steps, including i) user activity representation, ii) sequence activity embedding, iii) latent relations learning, iv) feature learning, and v) anomaly detection. The following subsections give further detail of each step as follows.

A. User Activity Representation

The primary step in the proposed model is to represent the user's activities as multivariate time series. As noted in the introduction to this paper, the user's activities can be structured as a series of temporal activities recorded each tick of time. Several data points (characteristics values) should be recorded at each timestamp, such as log-in/off information, user's id, and HTTP data.

More formally, given a series of a single activity $\mathcal{A}_1$ it consists of a sequence of an ordered n data points such that $\mathcal{A}_1 = \{p_1^1, p_1^2, \ldots, p_1^n\}$, as for a given point p it maps an encoded real value of an event. The entire activities are expressed as a whole series $\mathcal{S}$ which represent set of m activities, i.e. $\mathcal{S} = (\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m) = (\mathcal{A}^1, \mathcal{A}^2, \ldots, \mathcal{A}^n)^\intercal \in \mathbb{R}^{m \times n}$, where $n \in \mathbb{N}$ is the length of time series, i.e. the number of data points in each single activity. Intuitively, $\mathcal{S}$ is structured as a matrix consisting the entire user's activities whose samples are denoted as $p_i^{(t)} (i = 1, \ldots, m; t = 1, \ldots, n)$.

$$\mathcal{S} = \begin{bmatrix} p_1^1, p_1^2, \ldots, p_1^n \\ p_2^1, p_2^2, \ldots, p_2^n \\ \cdots \\ p_m^1, p_m^2, \ldots, p_m^n \end{bmatrix}$$

B. Sequence Activity Embedding

The represented activity sequences have a wide range of features that can be related in diverse ways. For example, if we consider the log-in and user id sequences for two different users, the sequences for the same user likely have strong relations. Thus, the idea is to portray each sequence in a flexible fashion that captures the various characteristics that underpin its behavior in a multidimensional manner. To this end, each activity sequence $\mathcal{A}$ has been encoded as an embedding vector $\vec{v}$ such that $\vec{v}_i \in \mathbb{R}^d$, for $i \in \{1, \ldots, m\}$. Note that the encoded embedding sequences are randomly initialized before being trained with the remainder of the model. Moreover, sequences with comparable embedding values should have a strong inclination to be connected since similar embedding sequences indicate similar activities.

C. Latent Relations Learning

The subsequent step is to learn relations between the embedded vectors. The optimal method to conduct so is by using direct graph architecture. In this context, given embedded vectors $\mathcal{V} = \{\vec{v}_1, \ldots, \vec{v}_{|\mathcal{V}|}\}$, they are mapped to graph structure $\mathcal{V} \mapsto (\mathcal{N}, \xi)$ with nodes and edges; where nodes denote embedded vectors, such that $\vec{v}_i \in \mathcal{N}$ for $i = \{1, \ldots, |\mathcal{N}|\}$, and edges represent pairs $\varepsilon_i \in \xi \mapsto \varepsilon_i = \langle \vec{v}, \vec{v}_\prime \rangle \in \mathcal{N} \times \mathcal{N}$. In this study, we implement direct graph representation for latent relations learning as direct edges $\vec{v} \mapsto \vec{v}_\prime$ between nodes because the dependency patterns between vectors do not have to be symmetric. Thus, the mapped edges between vectors represent relative dependant relationship in that the first vector

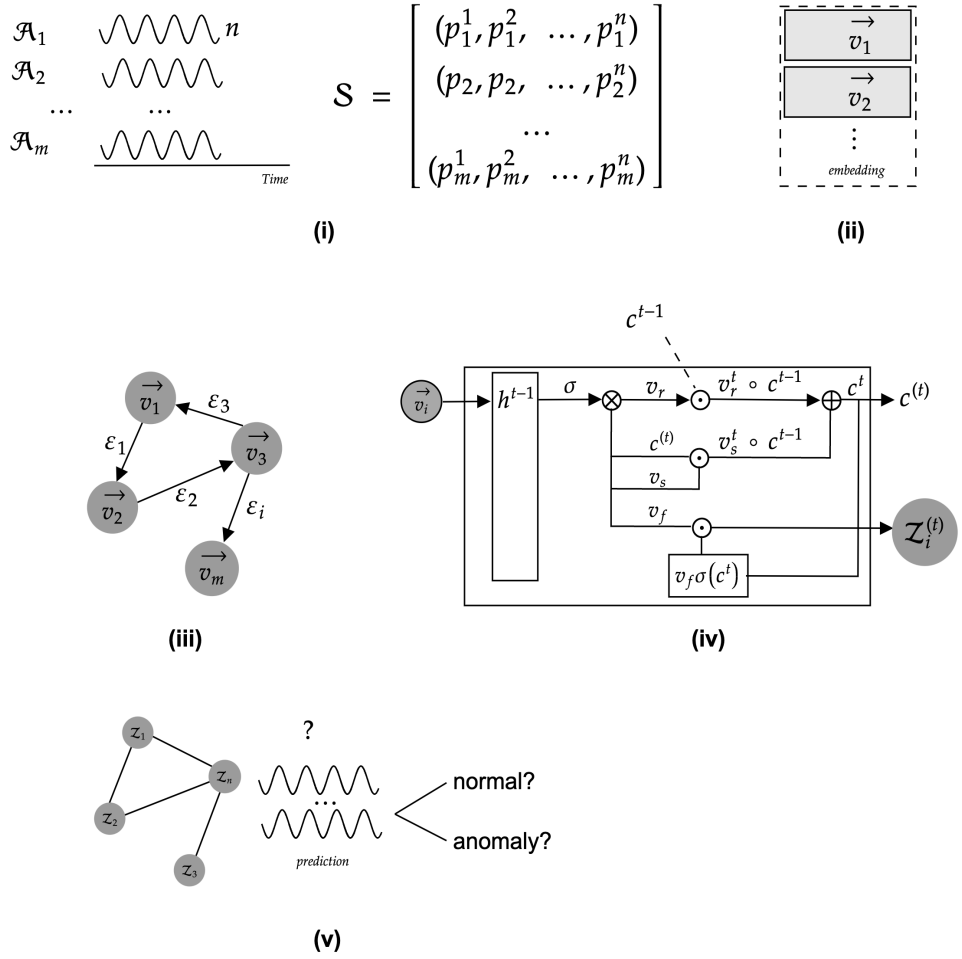


Fig. 2. Illustration of the Proposed Insider Threat Prediction Model.

is used to model the behavior of the second vector. Recall that given a node vector $\vec{v}$ is denoted by $h_{\vec{v}} \in \mathbb{R}^d$. Each node has given a label $\ell_{\vec{v}} \in \{1, \ldots, \mathcal{L}_{\mathcal{N}}\}$ that indicates the activity type, e.g. the log-in activities, where the edge has also been given label such that $\ell_{\varepsilon} \in \{1, \ldots, \mathcal{L}_{\xi}\}$ for each given ε_i.

Note that when an edge connects two vectors, it means the first vector is utilized to underline the behavior of the second vector. The dependency is represented between vectors as a set of candidate relations $\mathcal{R}_i$ for each vector such that $\mathcal{R}_i \subseteq \{1, \ldots, m\} \backslash \{\vec{v}_i\}$. The selection of which dependencies related to $\vec{v}_i$ is conducted by the search for the most similar candidate. To this end, we compute the similarity θ between ε_i edge node and the embeddings of its candidate relation $j \in \mathcal{R}_i$ using dot product measure. Equation 1 shows θ similarly measurement.

$$\theta(\varepsilon_i, \varepsilon_j) \frac{\vec{v}_i \cdot \vec{v}_j}{||\vec{v}_i|| \cdot ||\vec{v}_j||} \quad \text{for} \quad j \in \mathcal{R}_i \tag{1}$$

D. Feature Learning

Having represented relations between embedded vectors (graph nodes), the next step is concerned with extracting and learning features. The idea is to establish an abstracted feature space using RNN to fuse a node's information with its neighbors. In the proposed model, feature extraction includes the vector embedding $\vec{v}_i$, which describes the various behaviors of various vector kinds. Nevertheless, feature extraction has been accomplished using Long-Short Term Memory (LSTM), a well-known set of RNN architecture. The main benefit of adopting an LSTM unit is that the cell state averages activities over time, which helps to avoid disappearing gradients and better capture long-term time series relationships.

At each tick of time, an individual entry node $\vec{v}$ will map a hidden layer $h^{(t)}$. Each hidden unit has a memory cell $c^{(t)}$ to obtain long-term dependencies. The intuition is that $c^{(t)}$ serves to remember the effect of the prior input layer. In the proposed model, the mapping function use three non-linear gates to manage the access to $c^{(t)}$ cell as follows: i) remember vector v_r, save vector v_s, and focus vector v_f. The following equations express the mathematical notation of gated vector v_r, v_s, v_f respectively (2, 3, 4), memory cell control $c^{(t)}$ (5), and mapping function $h^{(t)}$ (6).

$$v_r = \sigma(W_r[h^{(t-1)}; v^{(t)}] + \epsilon_r) \tag{2}$$

$$v_s = \sigma(W_s[h^{(t-1)}; v^{(t)}] + \epsilon_s) \tag{3}$$

$$v_f = \sigma(W_f[h^{(t-1)}; v^{(t)}] + \epsilon_f) \tag{4}$$

$$c^{(t)} = v_r \odot c^{(t-1)} + v_s \odot \sigma(W_c[h^{(t-1)}; \vec{v}^{(t)}] + \epsilon_c) \tag{5}$$

$$h^{(t)} = v_f \odot \sigma(c^{(t)}) \tag{6}$$

where $\left[h^{(t-1)}; v^{(t)}\right] \in \mathbb{R}^d$ is the sum of the prior hidden state $h^{(t-1)}$ and the current vector $v^{(t)}$ along with some bias ϵ, $\odot$ is the element-wise multiplication, and σ is the non-linear Rectified Linear Unite (ReLU) activation function [31]. Here, the resulted output is an aggregated representation $\mathcal{Z}_i$ of hidden layers at time (t) from such input node $\vec{v}_i$.

$$\mathcal{Z}_i^{(t)} = \sigma(h^{(t)}, \vec{v}_i^{(t)}) \tag{7}$$

where $\vec{v}_i^{(t)}$ is the input for a given node at time t, with ReLU activation σ.

E. Anomaly Detection

When the relations are learned as per the previous subsection, the next step is to determine how the anomaly behavior can be detected accordingly. The idea is to determine how such an unseen behavior deviates from learned relations for each user. Thus, the proposed model attempts to calculate a similarity score Φ between the observed behavior of the user, that has resulted from the learned relations, and the new abstracted stream $\hat{\mathcal{Z}}$:

$$\Phi_i^{(t)} = |\mathcal{Z}_i^{(t)} - \hat{\mathcal{Z}}_i^{(t)}| \tag{8}$$

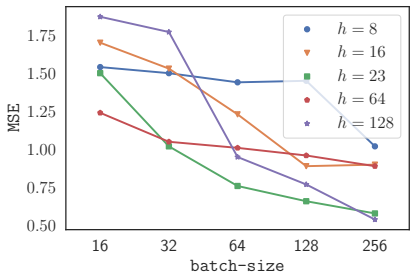
To assure a robustness calculation of the similarity score, we normalize the score for each input node using the median u of the difference between 1st and 3rd quartiles of distribution α.

$$\varphi_i^{(t)} = \frac{\Phi_i^{(t)} - u_i}{\alpha_i} \tag{9}$$

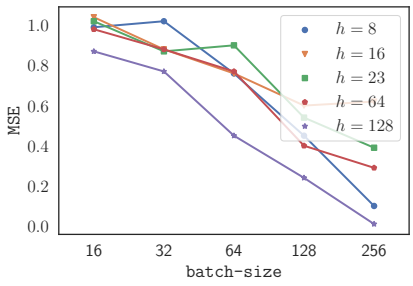
Recall that the use of inter-quartile range shows an effectiveness calculation of the distribution's spread for anomaly detection in stream data [32]. Then, the anomaly score $\texttt{Anom}_{\texttt{sc}}$ is the max value of φ that is computed at time (t) as follows:

$$\texttt{Anom}_{\texttt{sc}}^{(t)} = \max(\varphi_i^{(t)}) \tag{10}$$

Hence, the stream is classified as either normal or anomaly activity at some fixed threshold. The user can configure the threshold value; however, in the evaluation of this study, the value is set to max over the validation data. Thus, the stream is labeled as an anomaly whenever the similarity score exceeds the max $\texttt{Anom}_{\texttt{sc}}^{(t)}$ value.



(a)



(b)

Fig. 3. The MSE Performance over the Grid Search Concerning different Parameters: **(a)** Considers the Grid Search with epochs = 50, and **(b)** Considers the Grid Search with epochs = 100

IV. Performance Evaluation

This section provides the evaluation of the proposed model to determine its efficacy. The main objective of the evaluation is to show the effectiveness of detecting insider anomalous in network flows based on relation-based learning with LSTM. Moreover, the evaluation examines the model's performance under different LSTM parameters tuning to determine how they would affect the detection accuracy. Finally, the evaluation shows how well the performance of the proposed model compared to baseline methods of detecting insider attacks.

A. Evaluation Settings

1) Dataset: The evaluation experiments have been conducted over CERT dataset [33]. CERT is a public released insider threat dataset. It consists of activities data for more than 1k users, with 32 million events (log lines) generated over 502 days. There are around 7k log lines representing anomaly actions among the total recorded activities; these logs were manually placed into the data records by specialists. The data pertaining to log-in, log-off, device, and HTTP is stored in the logline. Each user action is parsed into a vector in the experiment, including the id, date, user, computer, and activity type as multivariate time series sequences. The dataset has been spillted to training (70%) and testing (30%) for all conducted experiments.

2) Metrics: Confusion Matrix (CM) has been used to compute a number of metrics to evaluate the model's performance. Recall that CM shows the classification performance of the proposed model under different parameters settings and with compared to other comparators baselines.Thus, CM is used to measure: i) F1-score (F1 = Equation 11), Precision (Pre =

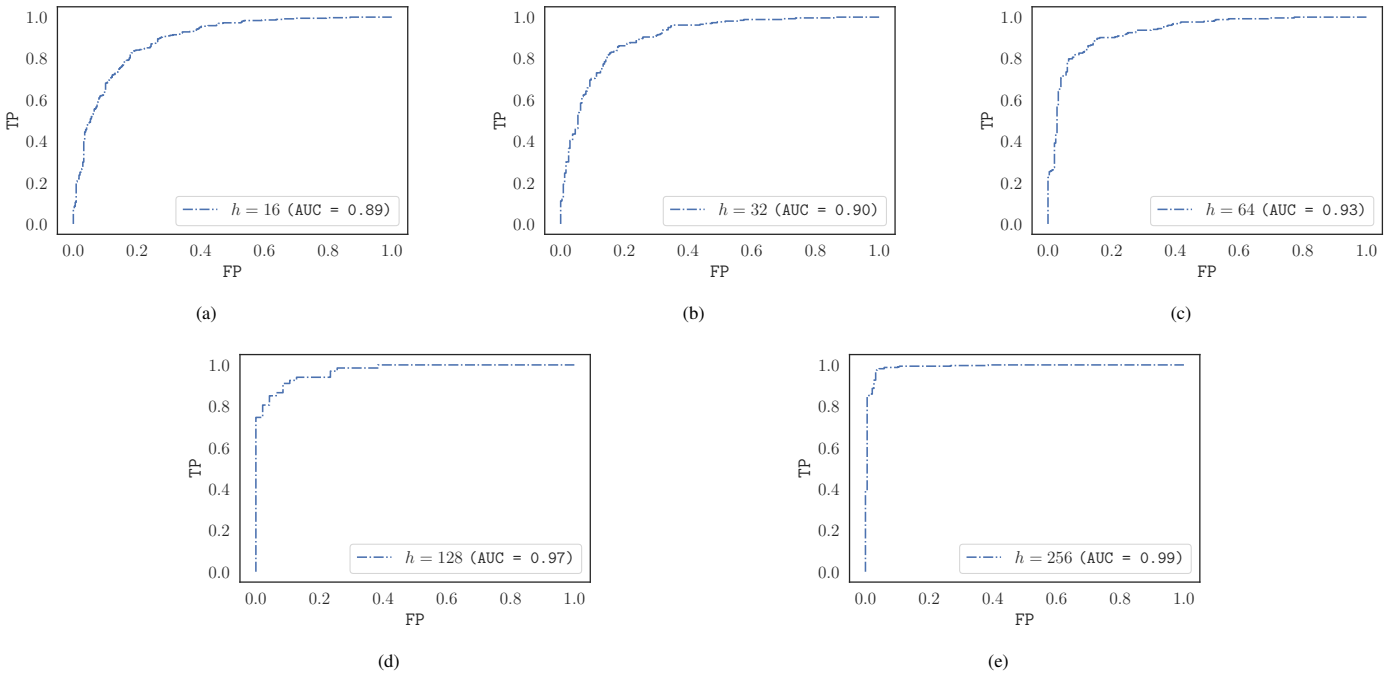


Fig. 4. ROC Curves of the Proposed Model with different `Hidden-Layer` Size Setting as Follows: **(a)** $h = 16$, **(b)** $h = 32$, **(c)** $h = 64$, **(d)** $h = 128$, and **(e)** $h = 256$,

Equation 12), and iii) Recall (`Rec` = Equation 13), Area Under Cure (AUC), and Receiver Operator Characteristics (ROC) curve.

$$\texttt{F1} = \frac{\texttt{TP}}{\texttt{TP} + \frac{1}{2}(\texttt{FP} + \texttt{FN})} \tag{11}$$

$$\texttt{Pre} = \frac{\texttt{TP}}{\texttt{TP} + \texttt{FP}} \tag{12}$$

$$\texttt{Rec} = \frac{\texttt{TP}}{\texttt{TP} \ + \ \texttt{FN}} \tag{13}$$

where `TP` is true positive samples, `TN` is true negative samples, `FP` is false positive samples, and `FN` is false negative samples.

Moreover, Mean Squared Error (`MSE`) is used to calculate the error rate between predicted values and the actual values. Considering n anomalous sequences inside the dataset, `MSE` computes the mean of the sum of all the squared errors of each sequence individually (Equation 14).

$$\texttt{MSE} = \sum_{i=1}^{n} \frac{(a_i - p_i)^2}{n} \tag{14}$$

where a_i is the actual value, and p_i is the predicted value.

3) Experiment Settings: The experiments have been conducted using TensorFlow[1]. A number of LSTM parameters, including the `epochs`, `batch-size` and `hidden-layers`, have been tested while learning relations and extracting features. Further detail concerning the choice criterion is given in the following section. Recall that `batch-size` refers to the size of the embedding vector $\vec{v}_i$ of an activity stream as proposed in the model. The model is trained using Adam optimizer with 0.01 learning rate.

[1]https://www.tensorflow.org/

B. Results

The model has been evaluated under different LSTM parameter setting to determine the best performance of relation learning and feature extraction. To this end, we determine the performance of the model under different parameter-setting including `epochs`, `batch-size`, and `hidden-layer` size. To determine the optimal setting for `batch-size` we conduct a grid search over $\{32, 64, 128, 256\}$. For `hidden-layer` we also conduct a grid search to tune the best performance over hyper-values of $\{8, 16, 32, 64, 128\}$. The model is run over `epochs` $= 50$ and `epochs` $= 100$, for the entire grid searches, respectively. Fig. 3 illustrates the performance of the proposed model in terms of `MSE` value for different parameter setting. The figure shows that `epochs` $= 100$ has generally produced better performance than `epochs` $= 50$ with different scales of `batch-size` and `hidden-layer`. The best results of `MSE` are recorded with `batch-size` $= 256$. It can be seen that the `batch-size` has an influence on obtaining better results whenever the value get larger although we found that the time complexity is increased accordingly. However, the efficiency on terms of time complexity scale is beyond the scope of this study despite it is interesting for consideration in other simulations such as the case of online feature extraction. Moreover, the model has yield best results whenever `hidden-layer` $= 128$ get larger. Figure 4 shows the ROC curves of the proposed model with respect to different `hidden-layer` sizes. The best result is recorded with AUC = 0.99 at `hidden-layer` = $h = 256$.

TABLE I. THE RESULTS OF ANOMALY BEHAVIOR DETECTION IN TERMS OF F1, PRE AND REC OF THE PROPOSED MODEL COMPARED WITH BASELINES

Method	F1	Pre	Rec
SVM	0.28	34.20	21.20
HMM	0.35	55.98	49.20
NN	0.74	89.76	58.10
Rel-RNN	**0.80**	**99.12**	**67.12**

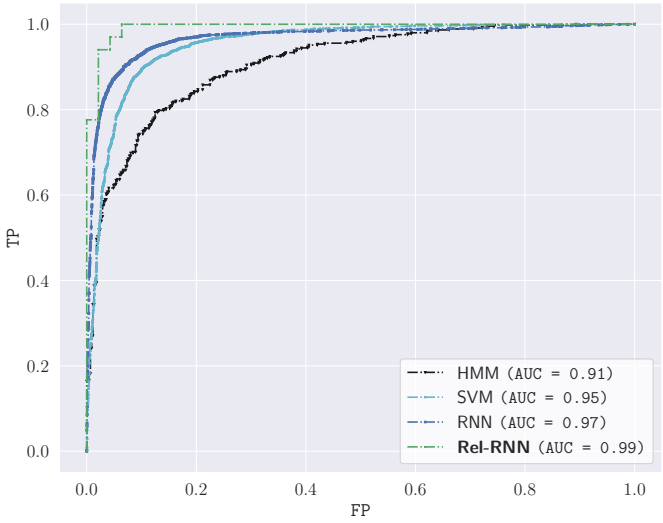


Fig. 5. The ROC Curves that Show the Performance of the Proposed Model along with Baseline Methods.

The proposed model has been evaluated to determine the detection accuracy in terms of F1, Pre and Rec compared with baseline methods. The evaluation explores the detection of anomaly behavior with relation-based feature learning of temporal streams, as in the proposed model, and the aggregated features based on statistician abstraction of audit data. We adopt the following baselines: SVM, HMM, and shallow Neural Network (NN). Note that the later considers feedforward structure with one layer of batch-size $= 265$ and hidden-layer $= 128$. Table I shows the obtained results of all methods; for clarity, we denote the proposed method as **Rel-RNN**. It can be seen from the table that the **Rel-RNN** has recorded best results with 0.80, 99.12, 67.12 for F1, Pre, and Rec respectively. To further demonstrate the obtained results, Fig. **??** shows the ROC curve for the proposed model compared with baseline methods. It can be observed that the performance of **Rel-RNN** has obtained a better AUC value of 0.99.

V. CONCLUSION

This study has proposed a novel model for insider threats detection. The proposed model structures the audit data, which represents the daily activities, as a multivariate time series covering broader characteristics for better user behavior learning. Thus, the temporal sequence of exclusive events is considered rather than an abstract set of features. The represented sequences are fed into an RNN model to learn hidden relations for feature extraction. The relations between representative features can be learned to identify latent patterns in the sequences for recognizing malicious behavior. To maintain the consecutive temporal lags between the set of features, LSTM has been used thus to avoid the vanishing gradient problem. The evaluation on the CERT dataset has shown that the proposed model has outperformed the comparator baselines for insider threat prediction. In the future, the plan is to incorporate Spatio-temporal dependencies to determine whether it affects modeling the latent relations to profile the user's behavior. This is desirable when users have the authorization to access the network from different places remotely. This case is observed during the COVID-19 pandemic when most companies and organizations allow employees to access networks from distant locations.

ACKNOWLEDGMENT

The author would like to thank Al Baha University, Saudi Arabia, for supporting this work under the fund 8/1440.

REFERENCES

[1] A. B. Pandey, A. Tripathi, and P. C. Vashist, "A survey of cyber security trends, emerging technologies and threats," in *Cyber Security in Intelligent Computing and Communications*. Springer, 2022, pp. 19–33.

[2] J. Jang-Jaccard and S. Nepal, "A survey of emerging threats in cybersecurity," *Journal of Computer and System Sciences*, vol. 80, no. 5, pp. 973–993, 2014.

[3] Ö. A. Aslan and R. Samet, "A comprehensive review on malware detection approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020.

[4] F. Yuan, Y. Cao, Y. Shang, Y. Liu, J. Tan, and B. Fang, "Insider threat detection with deep neural network," in *International Conference on Computational Science*. Springer, 2018, pp. 43–54.

[5] S. Yuan and X. Wu, "Deep learning for insider threat detection: Review, challenges and opportunities," *Computers & Security*, p. 102221, 2021.

[6] M. N. Al-Mhiqani, R. Ahmad, Z. Zainal Abidin, W. Yassin, A. Hassan, K. H. Abdulkareem, N. S. Ali, and Z. Yunos, "A review of insider threat detection: Classification, machine learning techniques, datasets, open challenges, and recommendations," *Applied Sciences*, vol. 10, no. 15, p. 5208, 2020.

[7] A. Kim, J. Oh, J. Ryu, J. Lee, K. Kwon, and K. Lee, "Sok: A systematic review of insider threat detection." *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 10, no. 4, pp. 46–67, 2019.

[8] D. C. Le and A. N. Zincir-Heywood, "Evaluating insider threat detection workflow using supervised and unsupervised learning," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 270–275.

[9] 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*, 2016, pp. 47–56.

[10] B. Böse, B. Avasarala, S. Tirthapura, Y.-Y. Chung, and D. Steiner, "Detecting insider threats using radish: A system for real-time anomaly detection in heterogeneous data streams," *IEEE Systems Journal*, vol. 11, no. 2, pp. 471–482, 2017.

[11] 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.

[12] E. R. DeLancey, J. F. Simms, M. Mahdianpari, B. Brisco, C. Mahoney, and J. Kariyeva, "Comparing deep learning and shallow learning for large-scale wetland classification in alberta, canada," *Remote Sensing*, vol. 12, no. 1, p. 2, 2020.

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

[14] J. Lu and R. K. Wong, "Insider threat detection with long short-term memory," in *Proceedings of the Australasian Computer Science Week Multiconference*, 2019, pp. 1–10.

[15] S. Paul and S. Mishra, "Lac: Lstm autoencoder with community for insider threat detection," in *2020 the 4th International Conference on Big Data Research (ICBDR'20)*, 2020, pp. 71–77.

[16] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, p. 122, 2019.

[17] A. L'heureux, K. Grolinger, H. F. Elyamany, and M. A. Capretz, "Machine learning with big data: Challenges and approaches," *Ieee Access*, vol. 5, pp. 7776–7797, 2017.

[18] A. McCarthy, E. Ghadafi, P. Andriotis, and P. Legg, "Functionality-preserving adversarial machine learning for robust classification in cybersecurity and intrusion detection domains: A survey," *Journal of Cybersecurity and Privacy*, vol. 2, no. 1, pp. 154–190, 2022.

[19] E. T. Ogidan, K. Dimililer, and Y. Kirsal-Ever, "Machine learning for cyber security frameworks: a review," *Drones in Smart-Cities*, pp. 27–36, 2020.

[20] R. Geetha and T. Thilagam, "A review on the effectiveness of machine learning and deep learning algorithms for cyber security," *Archives of Computational Methods in Engineering*, vol. 28, no. 4, pp. 2861–2879, 2021.

[21] A. Bendovschi, "Cyber-attacks–trends, patterns and security countermeasures," *Procedia Economics and Finance*, vol. 28, pp. 24–31, 2015.

[22] A. Sanzgiri and D. Dasgupta, "Classification of insider threat detection techniques," in *Proceedings of the 11th annual cyber and information security research conference*, 2016, pp. 1–4.

[23] D. C. Le and A. N. Zincir-Heywood, "Machine learning based insider threat modelling and detection," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 1–6.

[24] Y. Wei, K.-P. Chow, and S.-M. Yiu, "Insider threat detection using multi-autoencoder filtering and unsupervised learning," in *IFIP International Conference on Digital Forensics*. Springer, 2020, pp. 273–290.

[25] M. N. Al-Mhiqani, R. Ahmad, Z. Z. Abidin, W. Yassin, A. Hassan, and A. N. Mohammad, "New insider threat detection method based on recurrent neural networks," *Indones. J. Electr. Eng. Comput. Sci*, vol. 17, no. 3, pp. 1474–1479, 2020.

[26] T. Hu, W. Niu, X. Zhang, X. Liu, J. Lu, and Y. Liu, "An insider threat detection approach based on mouse dynamics and deep learning," *Security and Communication Networks*, vol. 2019, 2019.

[27] J. Jiang, J. Chen, T. Gu, K.-K. R. Choo, C. Liu, M. Yu, W. Huang, and P. Mohapatra, "Anomaly detection with graph convolutional networks for insider threat and fraud detection," in *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*. IEEE, 2019, pp. 109–114.

[28] R. K. Pathan, M. Biswas, and M. U. Khandaker, "Time series prediction of covid-19 by mutation rate analysis using recurrent neural network-based lstm model," *Chaos, Solitons & Fractals*, vol. 138, p. 110018, 2020.

[29] S. Jeon and J. Moon, "Malware-detection method with a convolutional recurrent neural network using opcode sequences," *Information Sciences*, vol. 535, pp. 1–15, 2020.

[30] K. Kayama, M. Kanno, N. Chisaki, M. Tanaka, R. Yao, K. Hanazono, G. A. Camer, and D. Endoh, "Prediction of pcr amplification from primer and template sequences using recurrent neural network," *Scientific reports*, vol. 11, no. 1, pp. 1–24, 2021.

[31] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[32] C. Wang, K. Viswanathan, L. Choudur, V. Talwar, W. Satterfield, and K. Schwan, "Statistical techniques for online anomaly detection in data centers," in *12th IFIP/IEEE international symposium on integrated network management (IM 2011) and workshops*. IEEE, 2011, pp. 385–392.

[33] J. Glasser and B. Lindauer, "Bridging the gap: A pragmatic approach to generating insider threat data," in *2013 IEEE Security and Privacy Workshops*. IEEE, 2013, pp. 98–104.