Deep Learning and Dempster-Shafer Theory Based Insider Threat Detection

Zhihong Tian [a], Wei Shi [b], Zhiyuan Tan [c], Jing Qiu [a, ∗], Yanbin Sun [a, ∗], Feng Jiang [d] and Yan Liu [e]

[a] *Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China*
[b] *School of Information Technology, Carleton University, Ottawa, Canada*
[c] *School of Computing, Edinburgh Napier University, EH10 5DT, UK*
[d] *School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China*
[e] *Department of Information Science and Engineering, Hebei University of Science Technology, Shijiazhuang 050000, China*
*Corresponding author: qiujing@gzhu.edu.cn; sunyanbin@gzhu.edu.cn

Abstract: Organizations' own personnel now have a greater ability than ever before to misuse their access to critical organizational assets. Insider threat detection is a key component in identifying rare anomalies in context, which is a growing concern for many organizations. Existing perimeter security mechanisms are proving to be ineffective against insider threats. As a prospective filter for the human analysts, a new deep learning based insider threat detection method that uses the Dempster-Shafer theory is proposed to handle both accidental as well as intentional insider threats via organization's channels of communication in real time. The long short-term memory (LSTM) architecture together with multi-head attention mechanism is applied in this work to detect anomalous network behavior patterns. Furthermore, belief is updated with Dempster's conditional rule and utilized to fuse evidence to achieve enhanced prediction. The CERT Insider Threat Dataset v6.2 is used to train the behavior model. Through performance evaluation, our proposed method is proven to be effective as an insider threat detection technique.

Keywords: Deep learning, Insider Threat, network security, recurrent neural networks.

1. Introduction

Threats posed by the insiders, such as employees, of an organization is among the greatest threats to information security. Only just the last decade, over 120 cases of malicious insider crime (espionage) that involve classified national security information were identified by the CERT Insider Threat Center (ITC) [1-4]. The latest case comes as the NSA has worked to reform security after the Edward Snowden disclosures, especially regarding insider threats. An insider threat is generally defined as a current or former employee, contractor, or other business partner who has or has had authorized access to an organization's network, system, or data and intentionally misused that access to negatively affect the confidentiality, integrity, or availability of the organization's information or information systems [5, 6].

Insider threats are not new. In May 2000, a Walt Disney CEO accidentally disclosed the quarterly earnings of the company to a reporter via an email prior a public announcement. Since information security has become very important in most organizations, there have been adversaries, enemies, and competitors trying to gain an advantage [7, 8]. To address the many problems arising from insider threat, recent research in cyber security has casted its focus on how to store, transmit, and process huge amounts of data. Unfortunately, these abilities cannot prevent illegitimate and

∗ Corresponding author.

Email addresses: tianzhihong@gzhu.edu.cn. (Zhihong Tian), wei.shi@carleton.ca (Wei Shi), z.tan@naier.ac.uk (Zhiyuan Tan), qiujing@gzhu.edu.cn. (Jing Qiu), sunyanbin@gzhu.edu.cn. (Yanbin Sun), fjiang@hit.edu.cn (Feng Jiang), apillowhill@gmail.com (Yan Liu)

unauthorized access to digital assets. Not all insider threats are the same; they differ in terms of their attack methods and objectives. Identifying insider threats and creating an effective mitigation strategy requires an understanding of threat types and representing the relationships [9, 10].

In this article, we propose a new deep learning based approach for insider threat detection that uses Dempster-Shafer theory to handle accidental and intentional insider threats through various monitored communication channels of an organization. The proposed *Dempster-Shafer and Deep Learning based Insider Threats Detection (DSDLITD)* is an online algorithm to filter network traffic or data traffic for analyst review in real time. Because insider threats contain different variations, we do not attempt to make direct models on threat behavior. Instead, we apply a Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) [11], which has been proved to have excellent performance in speech recognition and natural language processing over recent years, together with multi-head attention mechanism to aid modelling complex nonlinear relations by its well-designed architecture to capture different aspects of data representations which correspond to distinct levels of abstraction. To aid analysts in interpreting system decisions, DSDLITD deploys a fusion engine combination of multichannel classifiers to determine whether the data are a threat, which achieves high accuracy. Furthermore, in our proposed method, the selection of a neural network is guided by the characteristics of the data features. This equips our method with greater scalability and stronger applicability to other similar tasks.

In the following sections of this paper, we present some related work on data leakage prevention and insider threat detection in Section 2; the proposed DSDLITD model and the detection algorithms are presented in Section 3; we deliver the experimental results and analysis in Section 4; finally, we draw a conclusion in Section 5 with a discussion of future research.

2. Related Work

Over the past years, efforts have been invested to develop secure machine learning schemes for automated malware detection and cloud data analysis [12-16, 23-28]. This section focuses exclusively on the recent advances on insider threat detection using machine learning techniques.

CoBAn is a context-based model for prevention of accidental and intentional data leakage. The context-based model is comprised of two phases: training and detection. During the training phase, clusters of documents are generated and a graph representation of the confidential content of each cluster is created. During the detection phase, each tested document is assigned to several clusters and its content matched to each cluster's respective graph in an attempt to determine the document's confidentiality [17, 49]. Aruna et al. provided a means of detecting insider threat activities by leveraging patterns of usage and a tailored k-nearest neighbor (k-NN) algorithm in collaborative information systems [18]. In this work, the relational patterns of access logs are analyzed for nearest neighbors in terms of the number of subjects accessed. Parveen et al. conceptualized the insider threat detection problem as a stream mining problem and proposed a supervised learning solution based on one-class SVMs to cope with evolving concepts [19]. This method effectively addresses the rare instance issues, in which labeled training data fall short. Stolfo et al. [20] introduced a general-purpose Probabilistic Anomaly Detection (PAD) algorithm for the Windows environment, where anomalies or noise are assumed rare events in the training data. Hamedani et al. [21] introduced a novel method detecting attacks on smart grids with wind power generators using reservoir computing (RC). The proposed algorithms are shown to be more robust than MLP and SVE in dealing with different variables, such as the amplitude of the attack, attack types, and the number of compromised measurements in smart grids.

Different from the one-level classification, numerous two-level classification methods have also been considered. Hodo et al. focused on the classification of normal and threat patterns and took a multilevel perceptron, a type of supervised artificial neural network, trained it using network traffic traces and then evaluated its capability in preventing distributed denial of service (DDoS/DoS) attacks [22]. Another such work proposed a hybrid method that used discriminative multinomial naïve Bayes as a base classifier and nominal to binary supervised filtering at the second tier along with 10-fold cross validation [29]. This mothed was later improved with END (ensembles of

balanced nested dichotomies) at the primary tier and apply Random Forest at the following tier [30]. As supposed, this improvement brings about a raise in detection rate and a decrease in false positive rate. One other two-class classification strategy, along with the 10-fold cross validation method, subsequent in promising detection accuracy with the unique training dataset and the full 41-feature set. The overall performance of the resultant model is enhanced [31]. Significant improvement had been shown that gained information can be used to rank features and assist the following behavioral-based feature selection, which reduces the number of features in the set down to 20. Consequently, the reported accuracy for using only the training dataset is improved. As oppose to depend on only the machine learning algorithms, the outcome of the proposed network-based intrusion detection system (IDS) is also influenced by the use of both training and testing datasets. Some studies researched on the division between training and testing data for obtaining better detection systems. Unsupervised clustering algorithms was applied in an important work had shown that as oppose to make use of only the training data, the performance using both training and testing data was drastically reduced [37]. Similarly, an implementation utilizing the *k*-point algorithm had displayed slight improved accuracy of detection and when training and test datasets were both employed [38], reduced the false positive rate.

Despite some positive result obtained from applying neural networks in this field, many researchers use deep learning [32-35, 42-44, 48] to obtain accuracy in the intrusion detection field [36]. A deep learning-based method on the implementation of an efficient and adaptable network IDS was proposed by Niyaz et al. Unlabeled network traffic data were captured from different sources in this work and an informative feature representation was crafted for these collected datasets with self-taught learning techniques. These well-crafted features can, then, be applied in supervised classification of a small, but labeled traffic dataset containing both normal as well as anomalous network traffic packets [38]. Kim et al. built an intrusion detection model, which essentially applies the LSTM architecture in a recurring neural network. The proposed intrusion detection model is trained using KDD Cup 1999 dataset [40].

3. The Proposed DSDLITD Model

DSDLITD is comprised of two phases: learning and detection. During the former phase, classifiers are generated based on Attention-LSTM. During the latter phase, input vectors to the feature extraction process will be fed into the multichannel classifier to calculate their confidentiality score. We introduce Dempster-Shafer theory in order to help to determine whether our input data can be considered confidential.

3.1. The learning phase

Figure 1 is the graphic representation of the learning phase. The objective of this phase is to generate an Attention-LSTM based classifier that represents each of the organization's fields of activity.

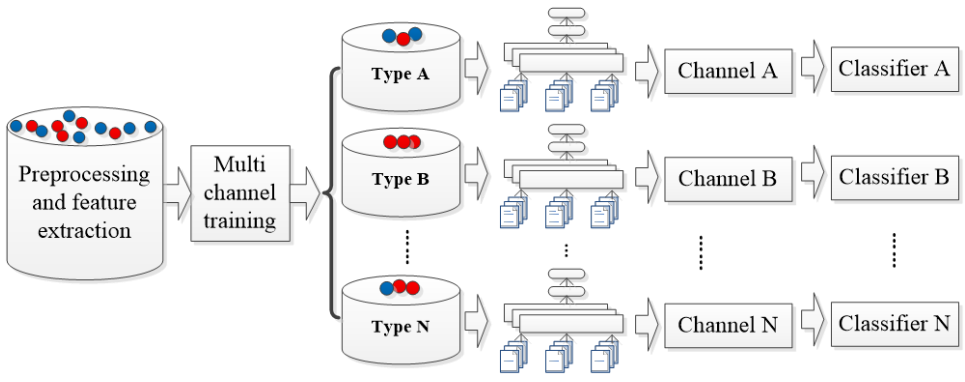

Figure 1. The learning phase.

3.1.1. The specific step of learning phase

The learning phase begins by a data preprocessing step which is the process of transforming format of the input data to a vectorized matrix. Data cleansing, data sampling, and data dimensionality are some common operations.

The next step is the extraction and learning of multi-feature. In the network, the traffic data tend to have various types of features as they are received from different sources. For instance, three different types of features: basic, content-based, and traffic-based can be categorized from the traffic data. The multi-feature is gathered by those categorized features which combined in different combination. Each multi-feature is named "Type φ". For example, as show in figure 1, the "Type A" multi-feature is composed of basic and content-based feature.

However, rather than directly concatenate the vector representation of those categorized features with different length to form multi-feature, we choose to use multi-head attention [50] to learn each of those features first. The multi-feature is finally formed by the learned vector of each feature, as shown in figure 2.

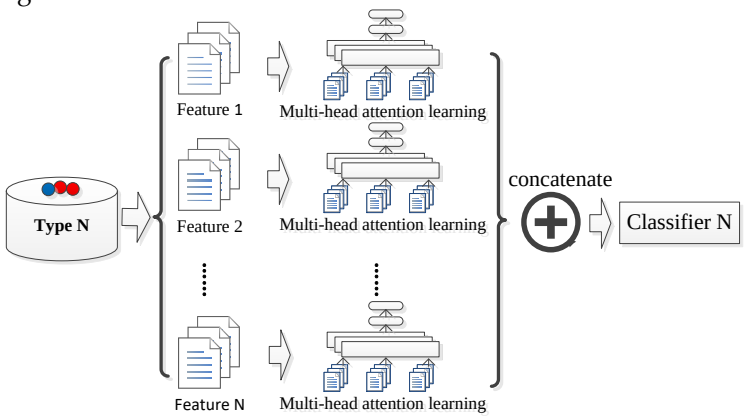

Figure 2. Use multi-head-attention to learn each feature of "Type N".

The reason we choose to use multi-head attention is under the consideration of the number of categorized features could be large. Multi-head attention is proposed by Google as an important component in their model Transformer. Multi-head attention has shown its high priority in dealing with neural language processing tasks for two advantages.

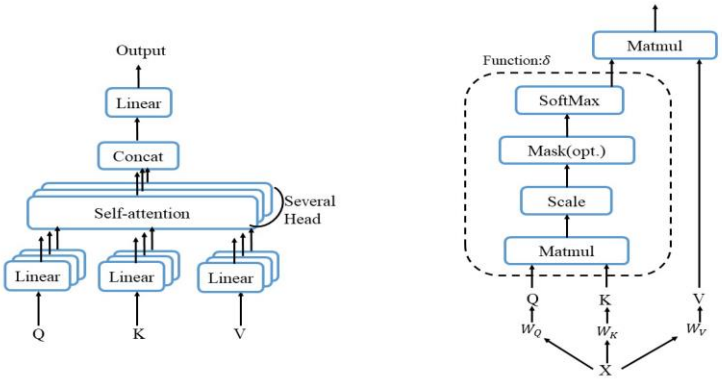

Figure 3. (left) Multi-Head Attention. (right) Scaled Dot-Product Attention..

The first advantage is adopting self-attention mechanism. As is depicted in figure 3 (left), multi-head attention is composed of several scaled dot-product attention, which is a specially type of self-attention. Self-attention is an attention mechanism that relating different position among a single sequence itself in order to generate the representation of the sequence. The second advantage is the parallelization compute capability, for it totally eschewing the recurrent neural network way. Recurrent neural network dealing with sequence along with time step to generate a sequence of hidden states and use the final hidden state as the representation of the whole sequence. However, the generate of current time hidden state rely on the previous generated hidden state, which is time

consuming and sequence length constrict. While using self-attention, a sequence representation could be generated in a unit time.

As depicted in figure3 (right), by using the embedding vector X multiply three different weight matrix: W_Q , W_K , W_V . A sequence $X:\{x_1, x_2, ..., x_n\}$ is represented in three ways $Q:\{q_1, q_2, ..., q_n\}$, $K:\{k_1, k_2, ..., k_n\}$, $V:\{v_1, v_2, ..., v_n\}$. According to attention function that mapping a query and a set of key-value pairs to an output, we obtain the output $O:\{o_1, o_2, ..., o_n\}$ as the learned vector of each feature. Specially, o_1 is obtained by:

$$W_{q_1}\{weight_1, weight_2, ..., weight_n\} = \delta(q_1, k_i), i = 1,2,..., n. \quad (1)$$

Where W_{q_1} is a set of values that weight the relevance between q_1 and k_i . Other W_{q_i} compute in the same way.

$$o_1 = \sum_{i=1,2,...,n} W_{q_1} \cdot v_i \quad (2)$$

Other output computed in the same way.

Multi-head attention consists of several self-attention layers running in parallel. Each self-attention represents a head. That allowed we set a smaller dimension of vectors to each head and finally concatenate them and project them to a higher dimension. A smaller dimension, apparently could improve calculation speed.

Now, the output literally contains more information than the originally embedding vector that could enhance other neural network. After gets the output of each feature, we concatenate them to form the final multi-feature. Each multi-feature is allocated to its own channel for next step training.

Based on the recurrent neural network, the last step is multichannel training. Each channel is trained to match with a neural network and followed by the generation of classifiers, which ultimately detect insider threats. The input for training the recurrent neural network is the multi-feature generate from the previous step. The number of channels can be varied, depend on the actual application. Considering that the traffic data are structured as sequences and the superiority for LSTM tackle with sequence, we choose it in our paper.

3.1.2. Learning algorithm

In the light of all derivations above, the proposed multichannel learning is described in following Algorithm 1.

Algorithm 1. The presented algorithm for learning

1: Input: A feature vector *X* extracted from a labeled given training dataset and the number of channel *I*

2: Initialization for the classifier result set C;

3: for fetch $c_i \in$C and set $i = 1$ to I do

4: Train Attention-LSTM model;

5: Save the Attention-LSTM model as a classifier c_i to a set of C;

6: end for

7: return: *C*;

3.2. The detection phase

Figure 4 provides an overview of the DLDS detection phase. First, traffic data in the network are fed into preprocessing and feature extraction components, where their counts are aggregated

and one vector is outputted for each distinct source. A feature vector is then fed into each Attention-LSTM classifier. These classifiers aim to predict the next vector in the sequence; effectively, they learn to model "normal" behavior. Anomalies occur in proportion to the prediction error and anomalous behavior, being marked, is sufficient enough for analysts to probe into. Finally, the results are passed to the D-S fusion engine, which determines the identity of the original traffic. Algorithm 2 is shown below in pseudocode for detailed procedures of the detection algorithm.

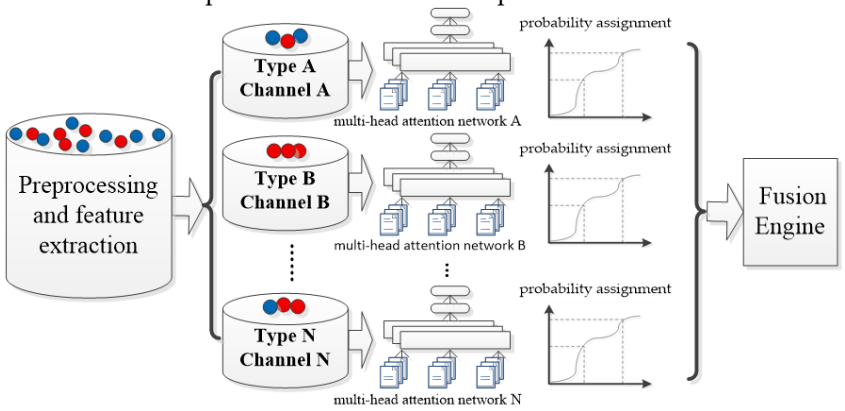

Figure 4. The detection phase.

Algorithm 2. The presented algorithm for detection

1: Input: A feature vector X extracted from test dataset with labeled information
2: Initialization:
3: for channel c_i set $i = 1$ to I do
4: Load Attention-LSTM model as a classifier;
5: Get the result vector R of the classifier;
6: end for
7: Insider threat detection:
8: Select the appropriate element v in vector R as the detection result using D-S fusion engine;
9: return v;

3.3 D-S fusion engine

In 1976, Shafer proposed a theory that is generally considered a generalized Bayesian approach and is named the Dempster-Shafer theory [45]. On the basis of the Dempster-Shafer, the frame of discernment, that is, the set of all possible complete facts or events $\Theta = \{\theta_i \mid 1 \le i \le N\}$, which are mutually exclusive, is involved in the D-S fusion engine. It consists of all hypotheses whose evidence is provided by the information sources. Θ contains hypothesis H as subsets, in which a value is assigned by an observation from a probability mass function m defined as follows:

$$m : 2^{\Theta} \to [0,1] \tag{3}$$

and meets the requirements as follows:

$$\begin{aligned} &m(\phi) = 0 \\ &m(H) \ge 0, \forall H \subseteq \Theta \\ &\sum_{H \in 2^{\Theta}} m(H) = 1 \end{aligned} \tag{4}$$

The degree to which the evidence supports the hypothesis H can be revealed by the probability that is within the interval determined by the plausibility and belief of H.

$$\left[Belief(H), Plausibility(H)\right] \tag{5}$$

where

$$\begin{aligned} Belief(H) &= \sum_{B \subseteq H} m(B) \\ Plausibility(H) &= 1 - \sum_{B \cap H \neq \phi} m(B) \end{aligned} \tag{6}$$

Belief(*H*) and *Plausibility*(*H*) denote the lower limit and the upper limit, respectively. The belief function can be distinguished from the plausibility function by the ignorance and both $Belief(\Theta) = 1$ and $Plausibility(\Theta) = 1$. The ignorance of the hypothesis can be indicated by the gap $\left[Plausibility(H) - Belief(H)\right]$. In D-S theory, the probability is calculated according to whether a hypothesis is supported by the evidence instead of the hypothesis itself. Thus, the probability being calculated is not bound to some particular hypothesis being correct but rather is bound to the confidence that a specific series of evidence is being correctly interpreted.

In DLDS, the D-S fusion engine collects the classification results from each classifier, and the evidence from all observers is combined by the D-S fusion engine to deduce the true state of the system. Next, with Dempster's rule of combination, multiple evidence can be gathered together. When A and B are used to calculate the new belief function for the focal element *H*, Dempster's rule of combination is represented as follows:

$$m(H) = \frac{\sum_{A \cap B = H} m(A)m(B)}{1 - \sum_{A \cap B = \varnothing} m(A)m(B)} \tag{7}$$

Dempster's rule is independent of a priori probability distributions on the possible system states, which is considered a useful property. Moreover, even without a priori knowledge about the system, it is still applicable. Actually, Dempster's rule can be generalized by iteration. If evidence A is denoted as the classification result, which is generated by classifier $i\,(i = 1...n)$, we are inclined to know whether A is true or false, and the frame of discernment is $\Theta = \{A, \neg A\}$. Then, the possible hypotheses are as follows:

$$2^{\Theta} = \{\varnothing, \{A\}, \{\neg A\}, \{A, \neg A\}\} \tag{8}$$

It is known that:

$$m(\{A, \neg A\}) = 0,\ m(\{\neg A\}) = 1 - m(\{A\}) \tag{9}$$

Then, the Dempster' rule is transformed as:

$$\begin{aligned} m(\{A\}) &= \frac{P(\{A\})}{P\{A\} + P\{\neg A\}} \\ m(\{\neg A\}) &= \frac{P(\{\neg A\})}{P\{A\} + P\{\neg A\}} \end{aligned} \tag{10}$$

In the detection algorithm, we employ the confidence $(1 - \tau)$ as the probability mass function $m_i(A)$. Then, as described in Eq. 7, all the evidence from classifier $i\,(i = 1...n)$ is recursively gathered.

$$m_{final} = m_1 \oplus m_2 \oplus \cdots \oplus m_n \tag{11}$$

where $\oplus$ shows Dempster's rule of combination. When determining this two-classification problem (true or false), it is not necessary to address the increasing computational complexity of D-S theory [47]. Then, the final result can be obtained by employing threshold t to m_{final}:

$$Result = \begin{cases} True\ positive, & if \quad m_{final} \geq t \\ False\ positive, & otherwise \end{cases} \tag{12}$$

4. Experimentation and Evaluations

This section begins with the explanation of our experiments (implemented it in TensorFlow (v1.2) [41]) performed. We then analyze the effectiveness of DSDLITD. Furthermore, we compare our experimental results against those of other known methods (including, RBNN, PNN, GRNN, Bayesian, SVM and KNN). For better demonstrating the efficacy of our proposed method, a single channel was used for the purpose of training and detection only. All results generated by the compared methods were get from running the respective pieces of source code indicated in their original papers with the best suitable parameters.

The CERT Insider Threat Dataset v6.2 is used. All experiments were executed on a PC with Inter(R) Core(TM) i5-7200 U CPU 2.50 GHz, 4 Gb memory, and an NVIDIA GeForce 920 MX GPU. For evaluation, the metrics defined below are employed. True positive (TP) - Attack samples that are correctly flagged as attacks. False positive (FP) - Normal samples that are incorrectly flagged as attacks. True negative (TN) - Normal samples that are correctly classified as normal. False negative (FN) - Attack samples that are incorrectly classified as normal. Detection Rate (DR) - the ratio of the attack occasions discovered by the presented method. Precision Rate (PR) and False Alarm Rate (FAR) - the ratio of misclassified normal occasions. The DR increases together with the FAR decreases made the performance of the method increases. The accuracy measures the proportion of the overall number of correct classifications over all classifications.

$$DR = TP / (TP + FN), \tag{13}$$

$$FAR = FP / (TN + FP), \tag{14}$$

$$PR = TP / (TP + FP) \tag{15}$$

$$Accuracy = (TP + TN) / (TP + FN + FP + TN), \tag{16}$$

As shown in Table 1, three datasets are generated to match each channel for training and testing purposes. In these datasets, the features of data are divided in the same way as described in [46]. The following features are used as the inputs for each channel: channel 1: basic, content-based, time, and traffic-based features; channel 2: basic and traffic-based features; and channel 3: basic and content-based features.

Table 1: Input features selected in each channel for training and detection purposes.

CHANNEL	DATASET INPUT FEATURES
# 1	basic, content-based, traffic-based, time
# 2	basic, traffic-based
# 3	time, content-based

In the CERT Insider Threat Dataset v6.2, which is composed of event log entries from the computer network of a virtual organization that was simulated with advanced user models. External storage device usage, email traffic, file operations, http traffic and Logon/logoff activity are the five sources of events that we employ. 135,117,169 events (log lines) were generated by 4,000 users over the course of 516 days. Events were manually driven by domain experts, representing the given insider threat scenarios. Moreover, it includes the user attribute metadata. For instance, the six categorical attributes are listed as follows:

Table 2: CERT insider threat dataset statistics**.**

# Device Events	# Email Events	# File Events	# HTTP Events	# Logon Events	Total Events
1,551,828	1,926,528	2,014,883	117,025,216	3,530,285	35,117,169

The unified Attention-LSTM, used in the training as well as the detection channels, composed both the priority of LSTM and multi-head attention mechanism. All the weights of Attention-LSTM are initialized with the numbers obtained from executing a stochastic gradient descent algorithm. Attention-LSTM is trained for 5 epochs, each of which of size 128. The learning rate is decayed from 0.01 to 0.0001 over 5 epochs.

For our proposed method, we study the performance using different input features on one single channel to adjust manually the channel numbers. As we can see in Figure 5, the learning rate degraded from 0.01 to 0.0001, while the accuracy follows a growing trend as the learning rate increases. Different results of single channel are produced for using different input features. In this case, combining basic and traffic-based features results in a higher detection accuracy.

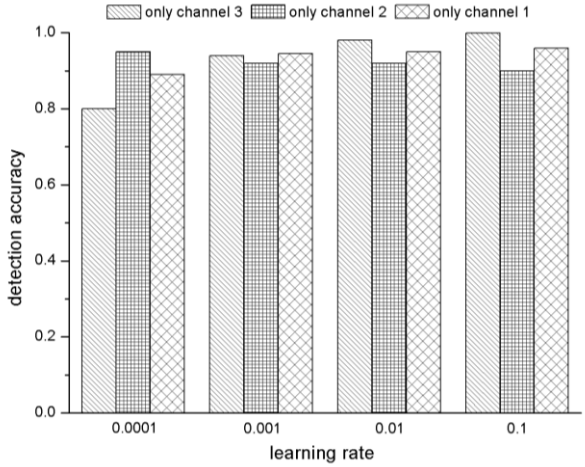

Figure 5. Impact of learning rate on detection accuracy when using a single channel

We study the impact of various parameters on the performance of DSDLITD, including the number of layers N and window size of h. Figure 6 plots the detection results curves. In each experiment, the value of one parameter is varied while the values of others are set default. Figure 6 shows that with $N = 4$, DSDLITD obtains the best results of the precision rate and detection rate. It also shows that the precision rate and detection rate are highly sensitive to changes in the window sizes, and a larger h value leads to a higher detection rate.

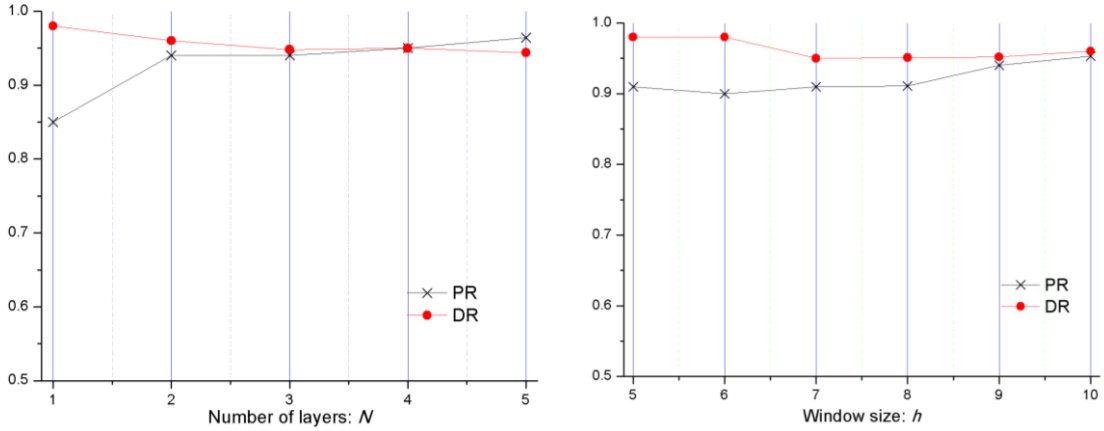

Figure 6. Performance with different parameters.

In Figure 6, we set the hyperparameters $N = 4$ and $h = 10$ in order to train the Attention-LSTMs. For test purpose, we set the DSDLITD input to be the same as what is introduced in Section 4. We choose DR, FAR and accuracy as the comparison factors against other methods [40]. Table 3 illustrate all comparison results on DR, FAR and accuracy. We observe that DLDS achieves the best

results, namely, DR of 95.79%, FAR of 4.67% and an accuracy of 95.47%, compared to the dataset baselines.

Table 3: DR, FAR and Accuracy results of all competitive methods

METHODS	DR (%)	FAR (%)	Accuracy (%)
GRNN	59.12	12.46	87.54
PNN	96.33	3.34	96.66
RBNN	69.83	6.95	93.05
KNN	45.74	46.49	90.74
SVM	87.65	6.12	90.4
Bayesian	77.6	17.57	88.46
DLDS	95.79	4.67	95.47

Figure 7 compares the CPU and memory usage rates of DLDS. Obviously, with a large dataset, the overhead of DLDS will become increasingly large. This is because when using -AttentionLSTM for modeling, a given word is mapped from a one-hot vector to an embedding vector following an input-embedding matrix. Then, to predict the probability of the next word, the top hidden layer is projected onto a probability distribution over all the words in the vocabulary following an output-embedding matrix.

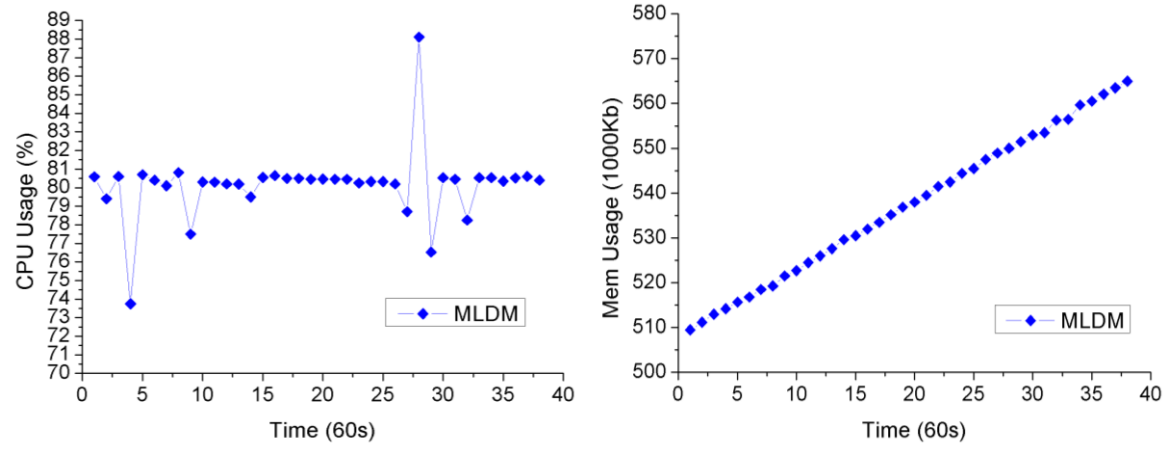

Figure 7. The CPU and memory usage of DLDS

5. Conclusions and future work

This paper has proposed an effective insider threat detection method based on neural network and Dempster-Shafer theory. The contributions of this paper are threefold: First, the proposed DLDS models "normal" behavior and indicates anomaly as potential malicious behaviors. Second, Attention-LSTM has been introduced in multichannel processing to build classifiers, which separate the attack traffic data from the normal ones. This allows the preservation of attack features in input traffic. A unified optimization method is used to train the Attention-LSTMs. Finally, we introduce Dempster-Shafer theory to determine whether a given set of input data forms an attack. The CERT Insider Threat Dataset v6.2 is used to perform tests that are used to evaluate our proposal algorithms. The results show that DLDS models outperform six standard baseline anomaly detection techniques (based on GRNN, PNN, RBNN, KNN, SVM and Bayesian). Experimental results demonstrate that state-of-the-art performance on running time comparing to most postprocessing algorithms is achieved by the proposed intelligent attack detection method.

Detecting an insider threat is still a long and complex investigative process. First, we will further improve the performance of DSDLITD on its execution time and memory usage. Second, we will consider the role that users and content metadata can play in both attack and defense

perspectives. Finally, another area of future work will be on conducting an implementation on a distributed filing system such as Hadoop HDFS using large datasets containing terabytes of log files.

Acknowledgments: This work is partially funded by the National Natural Science Foundation of China under Grant No. U1636215, 61871140, 61872100 and No. 61572153. And the National Key Research and Development Plan under Grant No. 2018YFB0803504. And the Guangdong Key Research and Development Plan under Grant No. 2019B010137004.

References

1. X. Du and H. H. Chen, Security in Wireless Sensor Networks, IEEE Wireless Communications Magazine, Vol. 15, Issue 4, pp. 60-66, Aug. 2008.
2. Gupta B B, Agrawal D P, Yamaguchi S. Call for Chapters: Handbook of Research on Modern Cryptographic Solutions for Computer and Cyber Security. 2014..
3. X. Xu, Z. Rong, Z.X. Wu, T. Zhou, and C. K. Tse. Physical Review E, 95(5), id.052302, 2017.
4. Yajun Mao, Xiongrui Xu, Zhihai Rong and Zhi-Xi Wu. The emergence of cooperation-extortion alliance on scale-free networks with normalized payoff. EPL (Europhysics Letters), 122(5), 2018.
5. Atat R, Liu L, Chen H, et al. Enabling cyber-physical communication in 5G cellular networks: challenges, spatial spectrum sensing, and cyber-security. IET Cyber-Physical Systems: Theory & Applications, 2017, 2(1):49-54.
6. X. Du, M. Guizani, Y. Xiao and H. H. Chen, Transactions papers, A Routing-Driven Elliptic Curve Cryptography based Key Management Scheme for Heterogeneous Sensor Networks, IEEE Transactions on Wireless Communications, Vol. 8, No. 3, pp. 1223 - 1229, March 2009.
7. Plageras A P, Stergiou C, Psannis K E, et al. Efficient IoT-based Sensor BIG Data Collection-Processing and Analysis in Smart Buildings. Future Generation Computer Systems, 2017.
8. Zhihong Tian, Mohan Li, Meikang Qiu, Yanbin Sun, Shen Su. Block-DES: A Secure Digital Evidence System using Blockchain, Information Sciences. 491(2019) 151-165. DOI: 10.1016/j.ins.2019.04.011.
9. Liehuang Zhu, Meng Li, Zijian Zhang, Zhan Qin, ASAP: An anonymous smart-parking and payment scheme in vehicular networks, IEEE Transactions on Dependable and Secure Computing (2018). DOI:10.1109/TDSC.2018.2850780
10. Y. Tian, J. Guo, Y. Wu, and H. Lin, Towards Attack and Defense Views of Rational Delegation of Computation, IEEE Access, DOI: 10.1109/ACCESS.2019.2908858, 2019.
11. Zhihong Tian, Wei Shi, Yuhang Wang, Chunsheng Zhu, Xiaojiang Du, Shen Su, Yanbin Sun and Nadra Guizani. Real Time Lateral Movement Detection based on Evidence Reasoning Network for Edge Computing Environment. IEEE Transactions on Industrial Informatics. 2019. DOI: 10.1109/TII.2019.2907754.
12. Zhihong Tian, Shen Su, Wei Shi, Xiang Yu, Xiaojiang Du, and Mohsen Guizani. A Data-driven Model for Future Internet Route Decision Modeling. Future Generation Computer Systems. 2019. DOI: 10.1016/j.future.2018.12.054
13. Zhihong Tian, Yu Cui, Lun An, Shen Su, Xiaoxia Yin, Lihua Yin and Xiang Cui. A Real-Time Correlation of Host-Level Events in Cyber Range Service for Smart Campus. IEEE Access. vol. 6, pp. 35355-35364, 2018. DOI: 10.1109/ACCESS.2018.2846590.
14. Qingfeng Tan, Yue Gao, Jinqiao Shi, Xuebin Wang, Binxing Fang, and ZhiHong Tian. Towards a Comprehensive Insight into the Eclipse Attacks of Tor Hidden Services. IEEE Internet of Things Journal. 2018. DOI: 10.1109/JIOT.2018.2846624.
15. Liehuang Zhu, Xiangyun Tang, Meng Shen, Xiaojiang Du, Mohsen Guizani, Privacy-preserving DDoS attack detection using cross-domain traffic in software defined networks, IEEE Journal on Selected Areas in Communications, Vol 36, No.3, pp628-643, March 2018.
16. Zhihong Tian, Yuhang Wang, Yanbin Sun and Jing Qiu. Location Privacy Challenges in Mobile Edge Computing: Classification and Exploration. IEEE Network. 2019.
17. Gilad Katz, Yuval Elovici, Bracha Shapira. CoBAn: A context based model for data leakage prevention. Journal Information Sciences: an International Journal archive Volume 262, March, 2014 Pages 137-158.

18. Aruna Singh, Smita Shukla Patel.Applying Modified K-Nearest Neighbor to Detect Insider Threat in Collaborative Information Systems. International Journal of Innovative Research in Science, Engineering and Technology. Vol. 3, Issue 6, June 2014.
19. Pallabi Parveen, Zackary R Weger, Bhavani Thuraisingham, Kevin Hamlen and Latifur Khan. Supervised Learning for Insider Threat Detection Using Stream Mining. In proceeding of 23rd IEEE International Conference on Tools with Artificial Intelligence. 2011.
20. Stolfo, Salvatore J. and Apap, Frank and Eskin, Eleazar and Heller,Katherine and Hershkop, Shlomo and Honig, Andrew and Svore, Krysta,A comparative evaluation of two algorithms for Windows Registry Anomaly Detection, Journal of Computer Security, volume 13, issue 4,July, 2005.
21. Hamedani, Kian & Liu, Lingjia & Rachad, Atat & Wu, Jinsong & Yi, Yang. (2018). Reservoir Computing Meets Smart Grids: Attack Detection Using Delayed Feedback Networks. IEEE Transactions on Industrial Informatics. 14. 734-743. 10.1109/TII.2017.2769106.
22. Elike Hodo, Xavier Bellekens, Andrew Hamilton, Pierre-Louis Dubouilh. Threat analysis of IoT networks using artificial neural network intrusion detection system. International Symposium on Networks, Computers and Communications (ISNCC). 17 November 2016.
23. Y. Xiao, V. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway, A Survey of Key Management Schemes in Wireless Sensor Networks, Journal of Computer Communications, Vol. 30, Issue 11-12, pp. 2314-2341, Sept. 2007.
24. X. Du, Y. Xiao, M. Guizani, and H. H. Chen, An Effective Key Management Scheme for Heterogeneous Sensor Networks, Ad Hoc Networks, Elsevier, Vol. 5, Issue 1, pp 24–34, Jan. 2007.
25. Y. Ma, Y. Wu, J. Li, and J. Ge, APCN: A Scalable Architecture for Balancing Accountability and Privacy in Large-scale Content-based Networks, Information Sciences, DOI: 10.1016/j.ins.2019.01.054, In Press.
26. X. Cheng, Y. Wu, G. Min, A.Y. Zomaya, Network Function Virtualization in Dynamic Networks: A Stochastic Perspective, IEEE Journal on Selected Areas in Communications, vol. 36, no. 10, pp. 2218-2232, 2018.
27. Y. Zuo, Y. Wu, G. Min, L. Cui, Learning-based Network Path Planning for Traffic Engineering, Future Generation Computer Systems, vol. 92, pp. 59-67, DOI: 10.1016/j.future.2018.09.043, 2019.
28. Y. Ma, Y. Wu, J. Ge, J. Li, An Architecture for Accountable Anonymous Access in the Internet-of-Things Network, IEEE Access, vol. 6, pp. 14451-14461, 2018.
29. Panda, M., Abraham, A. & Patra, M. R. Discriminative multinomial naive bayes for network intrusion detection. 2010 6th International Conference on Information Assurance and Security, IAS 2010 5–10 (2010). doi:10.1109/ISIAS.2010.5604193
30. Panda M, Abraham A, Patra M R. A Hybrid Intelligent Approach for Network Intrusion Detection. Procedia Engineering, 2012, 30(4):1-9.
31. Heba F E, Darwish A, Hassanien A E, et al. Principle components analysis and Support Vector Machine based Intrusion Detection System. In Proceedings of International Conference on Intelligent Systems Design and Applications. IEEE, 2010:363-367.
32. Eid H F, Salama M A, Hassanien A E, et al. Bi-Layer Behavioral-Based Feature Selection Approach for Network Intrusion Classification. 2011.
33. Al-Ayyoub M, Nuseir A, Alsmearat K, et al. Deep learning for Arabic NLP: A survey. Journal of Computational Science, 2017.
34. Elmisery A M, Sertovic M, Gupta B B. Cognitive Privacy Middleware for Deep Learning Mashup in Environmental IoT. IEEE Access, 2017, PP(99):1-1.
35. Al-Smadi M, Qawasmeh O, Al-Ayyoub M, et al. Deep Recurrent Neural Network vs. Support Vector Machine for Aspect-Based Sentiment Analysis of Arabic Hotels' Reviews. Journal of Computational Science, 2017.
36. Feng Jiang, Yunsheng Fu, Brij B. Gupta, Fang Lou, Seungmin Rho. Deep Learning based Multi-channel intelligent attack detection for Data Security. IEEE Transactions on Sustainable Computing (2018).
37. Syarif I, Prugelbennett A, Wills G. Unsupervised Clustering Approach for Network Anomaly Detection. 2012.
38. Gogoi P, Bhuyan M H, Bhattacharyya D K, et al. Packet and Flow Based Network Intrusion Dataset. In Proceedings of International Conference on Contemporary Computing. Springer Berlin Heidelberg, 2012:322-334.

39. Javaid A, Niyaz Q, Sun W, et al. A Deep Learning Approach for Network Intrusion Detection System. In Proceedings of International Conference on Bio-Inspired Information and Communications Technologies. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016:21-26.
40. Kim J, Kim J, Thu H L T, et al. Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection. In Proceedings of International Conference on Platform Technology and Service. IEEE, 2016:1-5.
41. Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 2002, 5(2):157-166.
42. Mozer M C. A focused backpropagation algorithm for temporal pattern recognition. Backpropagation. L. Erlbaum Associates Inc. 1995:349-381.
43. Werbos P J. Backpropagation through time: what it does and how to do it. Proceedings of the IEEE, 1990, 78(10):1550-1560.
44. Deutsch. Supervised Sequence Labelling with Recurrent Neural Networks | Springer. Springer-Verlag Berlin Heidelberg, 2012.
45. Abadi, M.; Agarwal, A.; TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. 2015.
46. Hee-su Chae, Byung-oh Jo, Sang-Hyun Choi, Twae-kyung Park, Feature Selection for Intrusion Detection using NSL-KDD, Recent Advances in Computer Science, 2013.
47. JamesInglis. A Mathematical Theory of Evidence. Technometrics, 1976, 20(1):106-106.
48. Jing Qiu, Yuhan Chai, Yan Liu, ZhaoQuan Gu, Shudong Li, Zhihong Tian. Automatic Non-Taxonomic Relation Extraction from Big Data in Smart City. IEEE Access. vol. 6, pp. 74854-74864, 2018. DOI: 10.1109/ACCESS.2018.2881422.
49. Y. Xiao, X. Du, J. Zhang, and S. Guizani, Internet Protocol Television (IPTV): the Killer Application for the Next Generation Internet, IEEE Communications Magazine, Vol. 45, No. 11, pp. 126–134, Nov. 2007.
50. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In Proceedings of Advances in Neural Information Processing Systems. 2017: 5998-6008.