Training regime influences to semi-supervised learning for insider threat detection

Duc C. Le
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia, Canada
lcd@dal.ca

Nur Zincir-Heywood
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia, Canada
zincir@cs.dal.ca

Malcolm Heywood
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia, Canada
mheywood@cs.dal.ca

***Abstract*—A malicious insider is one of the most damaging threats to any organization from industry to government agencies. Many challenges from insider threat detection come from the fact that the ground truth is very limited and costly to acquire. This paper presents a semi-supervised learning approach to insider threat detection. We employ three machine learning methods under different real-world conditions. These include obtaining the initial ground truth training data randomly or via a certain type of insider malicious behavior or by anomaly detection system scores. Evaluation results show that the approach allows learning from very limited data for insider threat detection at high precision. 90% of malicious data instances are detected under 1% false positive rate.**

***Index Terms*—semi-supervised learning, insider threat, malicious behavior, anomaly detection, data availability**

I. Introduction

Insider threats are one of the most dangerous and prevalent security threats that companies, institutions and government agencies are facing. Malicious actions from insider threats are performed by authorized personnel inside the organization. A recent report shows that two thirds of organizations feel vulnerable to insider attacks, and 70% have experienced insider attacks within the last 12 months [1]. Reports also confirm that insider attacks are becoming more frequent, and harder to detect, with the migration to the cloud and remote working trend [2].

Many damaging attacks may originate from an insider threat, both intentionally and unintentionally, such as information system sabotage, intellectual property theft and disclosure of classified information, or careless use of computing resources [3]. The fact that (malicious) insiders are authorized to access computer systems of organizations and may have knowledge about the organization's security layers creates many challenges to detecting and mitigating an insider threat. Moreover, in organizational environments, while many types of monitoring data are recorded, from network traffic to email and web logs, processing and analyzing the data to detect malicious activities is another challenge. Reports indicate that lack of resources and too many false positive alerts are the biggest hurdles in insider threat detection using security information and event management systems [1].

In cyber security and especially insider threat detection, while collecting unlabeled data is relatively easy, the acquisition of labeled data is typically very costly and requires skilled cyber security analysts. This is even infeasible in some cases, especially when zero-day attacks are involved. This paper presents a system that focuses on the use of semi-supervised machine learning to maximize the effectiveness of limited labeled training data for insider threat detection. Furthermore, we explore and evaluate different situations, reflecting real-world cases. These situations include acquiring the labeled (ground truth) data for the semi-supervised learning from randomly selected data to insider threat behavior to using an unsupervised learning system's (anomaly detection) alerts (scores).

The rest of the paper is organized as follows. Section II summarizes related literature. Section III introduces the proposed system as well as the semi-supervised learning algorithms employed, and the data availability conditions. Section IV details the experimental settings and the evaluation results. Finally, conclusions are drawn and the future work is discussed in Section V.

II. Related Work

Recently, insider threat detection and mitigation has become increasingly important [1]. Hence, research into this issue has attracted interest from both research communities and cyber security firms. The CERT Insider Threat Center [4] publishes different guides and common practices to combat insider threats. Collins et al. described 20 practices for organizations to prevent insider threats, as well as case studies of malpractices [4]. Recent surveys by Homoliak et al. [5] and Liu et al. [6] address the definition, taxonomy and categorization of insider threats, and provide an overview of the countermeasures.

Different machine learning (ML) applications have been successfully introduced to solve cyber security problems, such as intrusion detection and anomaly detection [7]. The success mainly comes from the fact that ML techniques have the ability to learn from a large amount of data to detect patterns that reflect abnormal, attacks, and malicious behaviors. Based on that, many ML-based approaches have been proposed to insider threat detection. Most of them are based on unsupervised learning, where anomaly detection model(s) are built without using labeled data to detect deviation from

the dataset's normal behavior. Techniques introduced in this category include graph-based models [8], Hidden Markov Models [9], autoencoders, recurrent networks [10]–[12] and mixture models [13], [14].

Due to the nature of unsupervised learning, false alarms are unavoidable [6], which could be a deciding factor in the applicability of insider threat approaches [1]. Thus, other ML techniques have been introduced for this problem, in particular supervised learning [15], [16], stream online learning [10], and evolutionary computation [17] to deal with non-stationary environments and imbalanced data challenges.

This work explores closing the gap between the approaches by using a semi-supervised learning approach to enhance the detection results based on very limited label (ground truth) availability. Thus, in this work, we examine different label availability conditions. One of which is to harness anomaly detection results using semi-supervised learning. In cyber security, semi-supervised learning has been employed in other problems, such as intrusion prevention [18], and distributed IoT attack detection [19].

III. Methodology

Figure 1 illustrates the overview of our proposed approach for insider threat detection using semi-supervised learning. The data processing steps are briefly described in Section III-A. Based on the available data, first anomaly detection steps or manual investigation is performed to obtain the initial labeled set of confirmed malicious and normal users' data. Then, based on the limited ground truth and extracted data, semi-supervised learning algorithms are employed in the next step to improve detection performance.

Presented in Section III-B, semi-supervised learning is a ML approach that permits harnessing the large amounts of unlabeled data in combination with typically smaller sets of labeled data, in order to improve the outcome [20]. Conceptually, semi-supervised learning falls between unsupervised learning, which does not need labeled training data, and supervised learning, which trains using only labeled training data. This motivates its use in cyber security applications, as obtaining a fully labeled training dataset is prohibitively costly or infeasible in many real world conditions [7].

In this paper, the focus is in using semi-supervised learning to improve upon the limited labeled data obtained from the initial detection step for identifying unknown malicious insiders. To achieve this, in III-C, we detail the explored approaches to present a limited labeled training set for semi-supervised learning algorithms using three different methods.

A. Data Pre-processing

After monitoring data collected in organizations, such as web access and email logs, the user activity data is first aggregated by time period, *daily* or *weekly*. The time periods for aggregation represent different levels of summarization and different amount of extracted data instances. More fine-grained data, e.g. session of user activities, could be extracted as well [16]. However, that might come at a cost of higher required workload to inspect alerts generated by the system, as fine-grained data has much higher data count. This may be explored in a future work, where further optimization or other detection techniques could be explored to reduce false positive rates.

Numerical features are then extracted from aggregated data to represent each day or week of a user's activities. We extract different types of numerical features from data: (i) Frequency features, i.e. the count of user actions over a day/week, e.g. *the number of external emails received, number of web access during weekend*, and (ii) Statistical features, i.e. the mean and standard deviation of varying records in user activities, e.g. *email size and file size*. Further details of the process to extract numerical features from log files with different information depicting machine and user interaction and/or action time can be found in [16].

Given that malicious insiders are essentially regular employees before they start performing malicious actions [4], approaches using temporal information in data representation have been proposed and showed significant improvements in detection performance. We employed the approach in this work with percentile comparison, where the goal is to highlight the changes from previously recorded activities of user behavior. Specifically, in this data representation, each data instance is further processed via percentile comparison to data points of the same user in the leading time window (30 days).

We note that in this work, while user activity logs are used (e.g. web and email logs), we extract only general counting / statistics of activities in each category. For example, we define sets of website (social networks, cloud storage, job search, ..) and file categories (documents, photos, ...) for extracting data features. As specific websites and files that users visited, as well as their contents are not inspected in data pre-processing, we believe that the employed feature extraction process can facilitate privacy preserving user monitoring.

B. Semi-supervised Learning Methods

In this work, we employ three popular semi-supervised learning methods from the literature [20]: Label Propagation (LP), Label Spreading (LS), and Self Training (ST).

1) Label Propagation: Label Propagation (LP) [21] is a graph-based method, which propagates given labels from a (typically small) subset of the data points to the whole dataset. In LP, label assignments $\hat{y}_i \in \mathbb{R}$ is obtained by propagating the estimated label at each node to its neighboring nodes based on the edge weights. A connection weight between node v_i and v_j is denoted by W_{ij}. The transition matrix A, $A_{ij} = W_{ij} / \sum_{v_k} W_{ik}$, allows the calculation of the new estimated label at each node as the weighted sum of the labels of its neighbors: $\hat{y}_i = A^T \cdot \hat{y}$. In this work, the radial basis function (rbf) is used as graph kernel for connection weights.

At the beginning, $\hat{y}$ is set to random for unlabeled data points and ground truth (known true labels) for the labeled points, respectively. Using this, the label propagation algorithm performs the two following steps repeatedly until converging (guaranteed) to obtain the final predictions [20]:

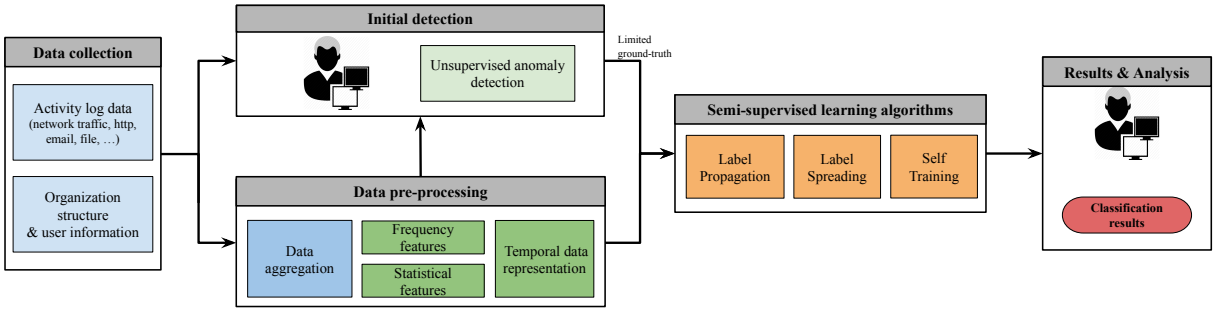

Fig. 1. Overview of the proposed system

i) Propagate labels from each node to the neighboring nodes ($\hat{y} = A^T \cdot \hat{y}$).
ii) Reset the predictions of the labeled data points to the corresponding true labels.

2) Label Spreading: Label Spreading (LS) [22] is also a graph based method with similar principles as Label Propagation. It was proposed to deal with two main drawbacks of Label Propagation: reducing label noise and regulation of influence of high degree nodes over the graph. To address the first issue, instead of assigning all the true labels to the labeled data points, LS relaxes the amount of true labels that need to be assigned, and uses the squared error between the true label and the estimated label in optimization. The second issue is addressed by using a modified version of the original graph and normalizing the edge weights via normalized graph Laplacian matrix ($\tilde{L}$). LS admits a closed-form solution and is a relatively efficient iterative approach to optimization [20]. In that, the label vector $\hat{y}_{t+1}$ at iteration $t+1$ is calculated based on that at iteration t, using the update rule $\hat{y}_{t+1} = \alpha \cdot \tilde{L} \cdot \hat{y}_t + (1-\alpha) \cdot y$, where y equals 0 for the unlabelled data points, and α balances the importance of the calculated label vector $\hat{y}$ and the base label vector y.

3) Self-Training: First proposed in [23], Self Training (ST) is a pseudo-labeling approach, which trains a supervised classifier iteratively using both the given true labels and the predicted labels (with high confidence) from previous iterations of the algorithm. Using only the labeled data at the beginning of the self-training procedure, a base classifier is trained and then used to obtain predictions for the unlabeled data points. Based on the classification result, a set of the most confident predictions are added to the labeled data set as pseudo-labels. The process is repeated with the supervised classifier retrained and new pseudo-labels obtained, until all samples have labels or the maximum number of iterations is reached. Self-training variations have been applied extensively to computer vision problems, especially with the use of deep learning, and demonstrated state-of-the-art performances [24]. In cyber security, ST variations are successfully used in intrusion detection and attack detection tasks [18], [19].

C. Label Availability for Semi-supervised Learning

In this paper, we explore different label availability conditions to semi-supervised learning. In real-world cyber security applications, the initial labeled set may not come from a random subset of data as assumed in many semi-supervised learning settings. For example, only some malicious actions are discovered by the analyst and labeled for further analysis, while other kinds of malicious actions remain undetected. In other cases, the results from unsupervised learning approaches for anomaly detection can be used as the basis for investigation. This in return provides an initial labeled data for training. Thus, the following conditions for label availability are examined in this paper:

(i) By random: a randomly selected small subset of data is labeled for training semi-supervised learning algorithms. This assumes that all users are equally suspicious.
(ii) By malicious action type: This case assumes that a certain type of malicious action is detected and labeled for training in combination with a random set of normal data. This condition examines whether learning methods can detect and generalize to other malicious behaviors.
(iii) By anomaly detection scores: This case uses anomaly detection scores (e.g. [11]) to label the unknown data. To this end, the highest (confidence) anomaly scores are provided to the semi-supervised learning.
(iv) As a variation of (iii), we assume the labeled data is obtained from all data points of a small set of *users* with highest anomaly scores.

In the following, the label availability conditions are referred to based on their numbers assigned in this section. For condition (ii), a number is used in conjunction, e.g. (ii-1), to specify the insider threat scenario – 1, 2, or 3 (see IV-A) – included in the initial label set.

IV. Experiments and Results

In this section, we present the datasets and experiments using the semi-supervised learning approach for insider threat detection. Section IV-A summarizes the dataset employed. Experiment settings and results are presented in sections IV-B and IV-C, respectively.

A. Dataset

The CERT insider threat test dataset (release 4.2) [25], [26] is employed for the evaluations performed in this paper. Specifically, the data simulates a company with 1000 employees, in which 70 are malicious insiders under three threat

scenarios. The malicious actions in the three insider threat scenarios are briefly described as follows [25]:

1) User begins logging in after hours, using a removable drive, and uploading data to wikileaks.org. The user leaves the organization shortly thereafter.
2) User begins surfing job websites and soliciting employment from a competitor. Before leaving the company, they use a thumb drive frequently to steal data.
3) System administrator becomes disgruntled. He downloads a keylogger and uses a thumb drive to transfer it to his supervisor's machine. The next day, he logs in as his supervisor and send out an alarming mass email.

The dataset consist of different logs of user activities (log on/off, email, web, file and thumb drive connect), and information about users and company structure. We manually examine the CERT data to approximately determine the missing information for feature extraction, such as user-user relationships, user-PC relationships, regular working hours, and website and file categories. Numerical features are then extracted from the data following the process presented in III-A[1]. A summary of the extracted data is presented in Table I.

TABLE I
SUMMARY OF THE DATASETS. SC: INSIDER THREAT SCENARIO

	Normal	Sc 1	Sc 2	Sc 3
day data	329466	85	861	20
week data	66840	52	254	10
# users:	930	30	30	10

B. Experiment Settings

In this work, we adopt realistic settings in previous work ([11], [16]) for training data, where data was obtained from only a restricted set of users over a given time period is used. Specifically, data from 200 users in the first half of the dataset's duration (37 weeks) is used to train the algorithms. The amount of label (ground truth) available is selected from the training data. In the main experiment, 20% of the small training data is labeled, based on the conditions provided in III-C. The labeled data amount is equivalent to that of 40 users (from the dataset of 1000 users). The trained classifiers are then used to test on the second half of the dataset to detect unknown malicious insiders. Results on the unlabeled portion of training data is also reported. In an additional experiment, we also explore the effect of the amount of initial labeling to semi-supervised learning performance. The experiments are repeated 5 times in each setting, and the averaged results are reported.

We implemented the data pre-processing and analysis steps using Python 3. Implementations from Scikit-learn [27] are used. For LP and LS, the rbf kernel is used with $\gamma = 10$. The parameters are chosen empirically. We tested different base classifiers for Self-Training, and found that the Random Forest classifier [28] achieved the best performance. From here on, Self-Training is referred as ST(RF) to reflect this. Anomaly scores are calculated using autoencoder, based on its advantages as shown in [11].

In this work, the insider threat detection performance is measured using ROC AUC metric. ROC (Receiving Characteristic Curve) plots the relationship between Detection rate (DR) and False Positive Rate (FPR) under different decision thresholds, and AUC (Area Under the Curve) summarizes the area trapped by ROC, which is useful for comparison between models.

$$DR = \frac{TruePositive}{TruePositive + FalseNegative} \quad (1)$$

$$FPR = \frac{FalsePositive}{FalsePositive + TrueNegative} \quad (2)$$

C. Results

The main experiment results are presented in Table II. The table reports AUCs on both unlabeled train data and test data, and DR on test data at different critical FPR levels. Overall, the results achieved using Self-Training are the most promising, on all data availability conditions. For example, on day data, using top anomaly scores for initial labels, this combination was able to detect 77% and 90% of the malicious instances in test data at FPRs of only 0.01% and 1%. The results demonstrate the ability of semi-supervised learning to improve upon the available limited ground-truth or anomaly scores to detect insider threats.

Table III shows the test results from unsupervised learning (using autoencoder), and supervised learning (using Random Forest) for comparison purposes. The comparison results are obtained under the same training/testing data setting (IV-B), i.e. data from 200 users in the first half of the dataset is used in training, but with fully unlabeled or labeled training set, depends on the learning algorithm. We note that the results as presented in Table III are state-of-the-art for corresponding learning methods on the CERT dataset [11], [16]. From the tables, it is clear that ST(RF) on percentile data shows much better detection performances than unsupervised learning and LS or LP. Furthermore, ST(RF) results are approaching that of classification approaches in some cases.

Comparing results by the semi-supervised learning algorithms, it is apparent that LP and LS lag behind ST(RF) under all conditions. This can partially be explained by the learning mechanisms of the methods. LS and LP assume neighboring relationships to label unlabeled data. Hence, their effectiveness is improved on condition (i) – randomly selected training set, in that the labeled training set is more likely to evenly distributed throughout the data. On other conditions (ii)-(iv), the initial labeled training set may have tendency to focus on specific regions of data (describing specific malicious actions, or with high anomaly scores). In these cases, neighborhood information as in LP and LS might not be effective in labeling other regions. On the other hand, ST(RF) performs well in most cases, except (ii-1) and (ii-3), where two scenarios with

[1]Feature extraction code available at https://github.com/lcd-dal/feature-extraction-for-CERT-insider-threat-test-dataset

TABLE II
DETECTION RESULTS (AUC AND DR) OF THE SEMI-SUPERVISED LEARNING ALGORITHMS UNDER DIFFERENT DATA AVAILABILITY CONDITIONS

Ground-truth availability	Algorithm	Week data					Day data				
		Unlabelled train AUC	Test AUC	Test DR @ 0.1% FPR	Test DR @ 1% FPR	Test DR @ 5% FPR	Unlabelled train AUC	Test AUC	Test DR @ 0.1% FPR	Test DR @ 1% FPR	Test DR @ 5% FPR
(i) Random	Label Propagation	0.781	0.779	9.48%	20.76%	38.65%	0.821	0.801	17.75%	31.27%	47.65%
	Label Spreading	0.767	0.728	10.92%	20.52%	35.53%	0.812	0.769	21.01%	32.39%	46.12%
	Self-training (RF base)	0.923	0.942	24.55%	42.75%	74.49%	0.978	0.983	64.99%	76.74%	92.80%
(ii-1) Scenario 1	Label Propagation	0.594	0.622	14.04%	16.69%	22.69%	0.474	0.477	8.93%	10.54%	14.12%
	Label Spreading	0.415	0.539	10.92%	14.88%	19.93%	0.381	0.418	8.89%	9.70%	12.35%
	Self-training (RF base)	0.558	0.709	21.56%	31.74%	34.40%	0.532	0.656	11.51%	32.88%	35.21%
(ii-2) Scenario 2	Label Propagation	0.614	0.747	5.76%	15.25%	33.73%	0.485	0.784	14.29%	26.88%	48.05%
	Label Spreading	0.542	0.719	4.20%	13.20%	30.85%	0.535	0.781	12.76%	25.19%	44.10%
	Self-training (RF base)	0.830	0.951	23.71%	41.92%	66.95%	0.914	0.992	78.99%	91.99%	96.74%
(ii-3) Scenario 3	Label Propagation	0.561	0.625	4.32%	13.45%	22.93%	0.348	0.390	2.25%	2.90%	5.59%
	Label Spreading	0.457	0.494	3.36%	6.48%	12.36%	0.340	0.349	1.57%	2.13%	3.94%
	Self-training (RF base)	0.507	0.542	3.95%	9.58%	13.23%	0.513	0.536	3.11%	11.58%	15.12%
(iii) Top anomaly scores	Label Propagation	0.670	0.630	11.84%	17.61%	23.26%	0.800	0.651	18.99%	24.64%	32.14%
	Label Spreading	0.601	0.605	13.12%	19.42%	23.79%	0.815	0.647	20.47%	26.23%	33.08%
	Self-training (RF base)	0.970	0.968	49.15%	60.43%	79.15%	0.985	0.996	77.57%	90.47%	98.26%
(iv) Users with highest anomalous scores	Label Propagation	0.722	0.729	11.53%	22.20%	32.13%	0.807	0.718	17.25%	26.16%	38.19%
	Label Spreading	0.677	0.691	12.06%	18.79%	33.40%	0.810	0.674	18.55%	26.12%	35.65%
	Self-training (RF base)	0.856	0.930	27.13%	40.32%	67.98%	0.990	0.992	57.64%	78.04%	97.54%

TABLE III
ANOMALY DETECTION AND CLASSIFICATION PERFORMANCES (AUC) FOR COMPARISON

	Unsupervised learning	Classification
Week data	0,874	0.9826
Day data	0.902	0.9995

very few malicious instances are labeled in initial training set. It seems that the iterative training process, in combination with ensemble of decision trees in random forest, was able to effectively partition the data space to identify / isolate regions of data describing malicious actions in these cases.

On three training conditions using different insider threat scenarios – (ii-1), (ii-2), and (ii-3) –, the AUC is higher only when scenario-2 is used as initial labels. We think this is due to a greater amount of malicious data from this scenario (Table I), as well as different data describing those types of malicious actions (IV-A). This also shows a weakness of this type of detection system, where supervised and semi-supervised learning may find it harder to generalize from previously seen attacks to detect novel unseen attacks. Unsupervised learning based anomaly detection may be better suited for this case.

Using condition (iii) – top anomaly scores, ST(RF) achieves the best AUCs and DRs in most cases (Table II). This indicates that learning upon anomaly detection results is beneficial in this case, as high anomaly scores may be indicative of malicious and unusual activities. On the other hand, employing data from users with the highest anomalous scores – condition (iv) – offers no further advantages in almost all of the cases.

Finally, on day and week data types, it appears that the performance (AUC) achieved on day data is better than that on week data. However, this comes at a cost of higher amount of alerts, as day data has five times the number of instances in week data (Table I).

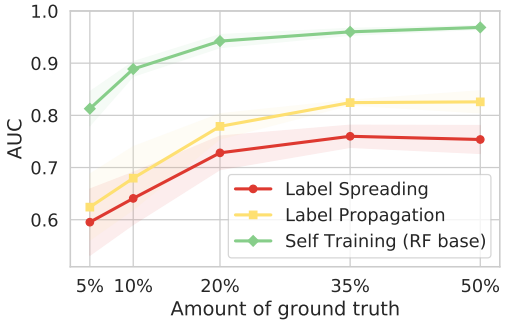

Fig. 2. AUC (on week data) by the amount of initial labels randomly selected

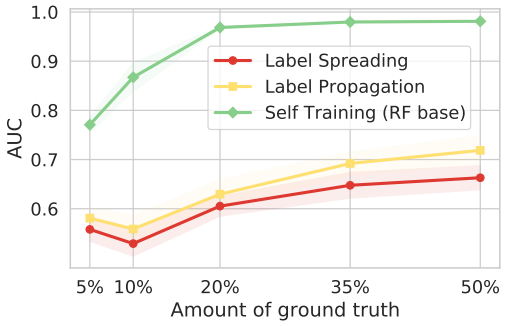

Fig. 3. AUC (on week data) by the amount of initial labels selected using anomaly scores

1) Effect of the initial labeled training set size: To understand the effect of the initial labeled set, we conducted further evaluations. To this end, we vary the amount of data selected as the initial labeled set for semi-supervised learning, in two selection settings: (i) Random and (iii) Anomaly Scores. 5% to 50% of training data (from 200 users) is labeled, which is

equivalent to the amount of data from 10 to 100 users'. AUCs by the three algorithms are presented in Figures 2 and 3.

Again, similar trends can be observed in both cases, where ST(RF) shows much better performances than the other algorithms and the AUCs are higher with better label availability. The improvement is significant up to when 20% of training is labeled. After that, detection performance is only slightly increased. This seems to indicate that 20% is the sweet spot of labeled data for performance gains versus the labeling costs in this case. Furthermore, with very small initial labeled set, condition (i) – randomly selected – yields better results (using ST(RF)). But as the amount of given labels increases, initial label selection by anomaly scores – condition (iii) – provides better AUCs, i.e. detection performances.

V. Conclusion

In this paper, we present a semi-supervised machine learning approach for insider threat detection. Three different semi-supervised learning algorithms (LP, LS, ans ST) are used in conjunction with different labeled data availability conditions. These were designed to emulate real-world situations representing the availability of various scenarios of ground truth. The proposed approach demonstrates the ability to learn from very limited ground truth to support cyber security analysts in detecting malicious insider behaviors in new data. Specifically, the semi-supervised learning approach using the Self-Training with Random Forest algorithm as the base classifier achieves the best results in most conditions. This approach successfully improves upon anomaly detection results using limited training labels to detect insider threats. The obtained test AUC is 0.992, with 90% of malicious instances detected at only 1% false positive rate, which is very competitive with the performance of supervised learning that uses all the labels of the training set. In the future work, other semi-supervised learning methods including deep learning based techniques can be examined for further improvements, and under other conditions, such as labeling mistakes and concept drift, and different training data duration.

Acknowledgment

This research was enabled in part by support provided by the Natural Science and Engineering Research Council of Canada (NSERC) and Compute Canada (www.computecanada.ca). Duc C. Le gratefully acknowledges the support by the Killam Trusts and the province of Nova Scotia. The research is conducted as part of the Dalhousie NIMS Lab at: https://projects.cs.dal.ca/projectx/.

References

[1] Cybersecurity Insiders, "2020 insider threat report," Gurucul, Tech. Rep., 2020, https://www.cybersecurity-insiders.com/wp-content/uploads/2019/11/2020-Insider-Threat-Report-Gurucul.pdf.

[2] M. Doering, "Combating insider threats in the age of remote work," Security Magazine, 2020, https://www.securitymagazine.com/articles/94156-combating-insider-threats-in-the-age-of-remote-work.

[3] Crowd Research Partners, "2018 insider threat report," CA Technologies, Tech. Rep., 2018, https://crowdresearchpartners.com/insider-threat-report.

[4] M. L. Collins *et al.*, "Common sense guide to mitigating insider threats, fifth edition," The CERT Insider Threat Center, Tech. Rep., 2016, CMU/SEI-2015-TR-010.

[5] 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. 30:1–30:40, Apr. 2019.

[6] 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*, 2018.

[7] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[8] F. Liu, X. Jiang, Y. Wen, X. Xing, D. Zhang, and D. Meng, "Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise," in *Proc. ACM Conference on Computer and Communications Security*, vol. 18, 2019, pp. 1777–1794.

[9] T. Rashid, I. Agrafiotis, and J. R. Nurse, "A new take on detecting insider threats," in *Int. Workshop on Managing Insider Security Threats*, 2016.

[10] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," in *AAAI Workshop on Artificial Intelligence for Cyber Security*, 2017.

[11] D. C. Le and N. Zincir-Heywood, "Exploring adversarial properties of insider threat detection," in *2020 IEEE Conference on Communications and Network Security (CNS)*, 2020.

[12] L. Liu, C. Chen, J. Zhang, O. De Vel, and Y. Xiang, "Unsupervised insider detection through neural feature learning and model optimisation," in *Lecture Notes in Comp. Sci.*, vol. 11928 LNCS. Springer, 2019.

[13] T. E. Senator *et al.*, "Detecting insider threats in a real corporate database of computer usage activity," in *ACM SIGKDD Conf. (KDD)*, 2013.

[14] B. Bose, 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*, 2017.

[15] G. Gavai *et al.*, "Supervised and unsupervised methods to detect insider threat from enterprise social and online activity data," *J. Wireless Mobile Netw., Ubiquitous Comput., & Depend. Appl.*, vol. 6, no. 4, 2015.

[16] 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.

[17] D. C. Le, S. Khanchi, N. Zincir-Heywood, and M. I. Heywood, "Benchmarking evolutionary computation approaches to insider threat detection," in *ACM Genetic and Evolutionary Computation Conf.*, 2018.

[18] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Information Sciences*, vol. 378, pp. 484–497, 2017.

[19] S. Rathore and J. H. Park, "Semi-supervised learning based distributed attack detection framework for iot," *Applied Soft Computing*, vol. 72, pp. 79–89, 2018.

[20] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.

[21] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Carnegie Mellon University, Tech. Rep., 2002, CMU-CALD-02–107.

[22] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," *Advances in neural information processing systems*, vol. 16, no. 16, pp. 321–328, 2004.

[23] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *33rd Annual Meeting of the Association for Computational Linguistics*, 1995, pp. 189–196.

[24] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 687–10 698.

[25] B. Lindauer, "Insider threat test dataset," Carnegie Mellon University, Tech. Rep., 2020, https://doi.org/10.1184/R1/12841247.v1.

[26] J. Glasser and B. Lindauer, "Bridging the gap: A pragmatic approach to generating insider threat data," in *IEEE SPW*, 2013.

[27] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, 2011.

[28] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, 2001.