2019 IEEE 12th International Conference on Cloud Computing (CLOUD)

Toward an Online Network Intrusion Detection System Based on Ensemble Learning

Ying-Feng Hsu[1], ZhenYu He, Yuya Tarutani[3], Morito Matsuoka[1]

[1] Cybermedia Center, Osaka University, Osaka, Japan
Email: {yf.hsu, matsuoka}@cmc.osaka-u.ac.jp
[2] Graduate School of Information Science and Technology, Osaka University, Osaka, Japan
Email: zhenyu@ist.osaka-u.ac.jp
[3] Okayama University, Okayama, Japan
Email: y-tarutn@okayama-u.ac.jp

***Abstract*—With information technology growing and rapidly increasing, ubiquitous networking technology generates a massive amount of data and is integrated into our daily life. Network intrusion detection systems (NIDS) are essential for organizations to ensure the safety and security of their communication and information. In general, there are two types of NIDS: signature-based (SNIDS) and anomaly-based (ANDIS). Most modern NIDS solutions are signature-based techniques, which require a routine signature update and cannot detect unknown types of attacks. However, ANDIS has been extensively studied and is considered a better alternative to NIDS. In this paper, we present a stacked ensemble learning based ANIDS that consists of autoencoder (AE), support vector machine (SVM), and random forest (RF) models. To show the overall applicability of our approach, we demonstrate our work through two well-known NIDS benchmark datasets: NSL-KDD and UNSW-NB15 and a real campus network log, which includes about 300 million daily records. We compare our method to three different machine learning classical models and two other reported study results. Our test result implies that our proposed method can also limit both false positive and false negative predictions.**

Keywords-network intrusion detection system; NIDS; ensemble learning; deep learning; autoencoder; random forest; support vector machine

I. INTRODUCTION

With information technology growing and its spread rapidly increasing, ubiquitous networking technology generates a massive amount of data and is integrated into our daily life. The total worldwide IP traffic is estimated to be about 120 exabytes per month in 2017 and to grow to about 400 exabytes per month in 2022 [1]. The number of cyberattack-related threats is growing and has become diversified, along with the increases in network traffic. The term "cyberattack" often refers to the unauthorized activities that attempt to threat, disable, destroy, steal, or expose other parties' information-related assets and to cause their loss. Network intrusion detection systems (NIDS) has become essential for many organizations. Such systems monitor network traffic and detect abnormal activities or cyber attacks to ensure the safety and security of their communication and information. In general, there are two types of NIDS: a signature-based network intrusion detection system (SNIDS), and an anomaly-based network intrusion detection system (ANIDS). Nowadays, most solutions operate under signature-based techniques, which are based on a pattern-matching algorithm to inspect and identify threats from the incoming packet by comparing a pre-installed signature against incoming network activities. SNIDS has good detection capability for known attaches with fewer false alarms; however, it requires a preinstalled signature and is not aware of unknown threat patterns. Frequent maintenance and routine updates are necessary when operating a SNIDS for network protection. In contrast, ANIDS focus on deviations of the traffic pattern and use those deviations to evaluate incoming traffic and determine the chance of anomaly, even when faced with unknown attacks.

In this paper, we propose a novel anomaly-based network intrusion detection system (ANIDS) for the task of intrusion detection in large-scale computer networks. Our implementation includes modules of network traffic data sniffing, data preprocessing, and anomaly detection. Threats or anomalies are detected based on the method of stacked ensemble learning, which consists of autoencoder (AE), support vector machine (SVM), and random forest (RF) models. We evaluated our approach using two well-established benchmark network intrusion simulation datasets: NSL-KDD [2] and UNSW-NB15 [3]. In addition, we also applied our method to our campus network environment log, the Palo Alto network [4] log dataset which consists of about 300 million daily network traffic records. This traffic log is about 100 times larger than either of those two synthetic datasets. To show the feasibility of our approach, we compare our proposed approach to five different machine learning methods i.e. 3 different classical machine learning models and two other related study results.

II. RELATED WORK

To recognize and identify intrusion-related traffic, various anomaly detection or classification techniques have gained a great deal of attention. As previously mentioned, there are two categories of network intrusion detection systems (NIDS): a signature-based network intrusion detection system (SNIDS)

2159-6190/19/$31.00 ©2019 IEEE
DOI 10.1109/CLOUD.2019.00037

and an anomaly-based network intrusion detection system (ANIDS). Typically, SNIDS reads an incoming network packet and compare them against a large predefined signature (known threats). The pattern matching algorithms is the core component for the SNIDS approach, it may increase system overhead when performing large traffic record comparisons. Several GPU acceleration methods [5] [6] [7] have been proposed for increasing the performance of the multiple-pattern matching algorithm. The main limitation of SNIDS is that it can only detect known threats. In this study, we focus on investigating ANIDS and its related studies. Machine learning and deep learning have both been widely used to solve many classification problems, including those in the anomaly detection field. A recent survey from [8], compared to traditional signature-based NIDS, deep learning-based methods offer improved detection accuracy across a wide range of threat categories. Various machine learning methods have been proposed to develop NIDSs. A self-taught learning (STL) approach based on deep learning [9] was proposed, and consisted of unsupervised feature learning by an autoencoder and supervised labeled learning by adding a Softmax layer. [10] presents an SVM-based IDS that considered feature weights during SVM training and incorporated an information gain ratio (IGR) and a K-mean algorithm in SVM for intrusion detection. The work of [11] applied a random forest (RF) to detect four types of attacks (DDOS, probe, U2R, and R2L) and used an NSL-KDD dataset to evaluate the performance. This study reported that this approach had about a 7% higher accuracy than a decision-tree–based J48 algorithm. Another unsupervised learning approach by using a restricted Boltzmann machine (RBM) [12] focused on the noise and the removal of outliers from the input data.

III. PROPOSED METHODOLOGY

Figure 1 shows the structure of the proposed ANIDS, which contains three major components. First, this proposed system is designed to be scalable and continuously detect the future network dynamic of time series data streams. More specifically, in this study, the data stream is the incoming network traffic packets that are captured and logged by data sniffing. Second, a data preprocessing module is used that ensures the quality of data before the data is fed into intrusion detection. Finally, there is a stacked ensemble based anomaly detection engine for the intrusion detection task.

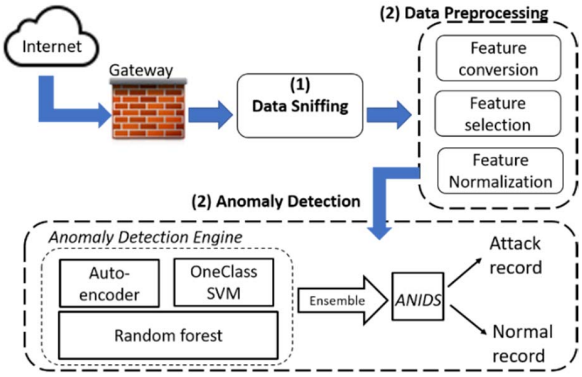

Figure 1. The system architecture of proposed ANIDS.

A. Data sniffing

The data sniffing module was designed to collect network traffic packets, such as tcpdump. In our case, we used the Palo Alto Panorama to collect our campus network traffic logs. The structure of the log includes 61 features and is divided into a threat log (which contains only threat-related records) and traffic (which contains all network records).

B. Data preprocessing

Time serial data often contains a certain number of discrepancies, especially in large-scale and high dimensional datasets like the ones found in this study. Data processing ensures the quality of model building and prediction, and as a result, we provide the solutions for common data discrepancy issues, such as data redundancy and data outliers. We considered three standalone network environment datasets (NSL-KDD, UNSW-NB15, and our Palo Alto system traffic log) when developing our system. Other than fixing the data discrepancy issue, we also include model training related preprocessing functions, such as the following.

Feature conversion:
It is common to have both character and numerical data present in the network data. To facilitate the machine learning process, we converted those character features into numeric values. For example: {TCP→ 1, UDP→2}.

Feature selection:
The principle of feature selection is that the data may contain irrelevant, redundant, or noisy features, which can be eliminated without a loss of information or without affecting the accuracy of the prediction model. For example, features with only one unique value should be eliminated from a future process. In addition, from the viewpoint of confidentiality and sensitivity, personal identification data should exclude certain items from feature selection, such as the IP address and user account. Table 1 summarizes the candidate variables (features) with a standardized naming convention in our system. In this study, we apply the feature selection for the Palo Alto system log dataset. We use all features from NSL-KDD and UNSW-NB15 since they are synthetic network traffic data; those features could possibly include particular meanings from data creators.

TABLE I. COMMON DATA TYPES

Data Types	Selected Features
Numerical type	Repeat Count, Bytes, Bytes Sent, Bytes Received, Source Port, Destination Port, Packets, Elapsed Time, Packets Sent, Packets Received
Character type	Threat/Content Type, Rule Name, Application, Source Zone, Destination Zone, Protocol, Action, Category, Source Location, Destination Location, Session End Reason, Action Source
Binary type	Flag

Feature normalization:
The operation variables of ICT equipment are heterogeneous, with different scale and units. In practice, it is recommended

that the data normalization method is applied to a high-dimensional dataset before the machine learning process [13]. In our approach, we used the normalized functions (1) and (2) to rescale the raw values into a proper range. For those high standard deviation related features such as Packets Sent and Packets Received, we converted to the logarithm of 10 and divided by its max value, as in (1) and for those low standard deviation related features, we normalize them by the max value, as in (2).

$$\bar{a} = \frac{\log(a+1)}{\log(a+1)_{max}} \quad (1) \qquad \bar{a} = \frac{a}{a_{max}} \quad (2)$$

; where a is the original raw value

C. Anomaly detection

To reflect the latest and most accurate information on intrusion detection, we investigated various machine learning models for the task of anomaly detection. There are two major considerations for implementation. First, a generated model that can apply in different network environments. Second, it should have a mechanism to adjust the detection threshold of normal and threat traffic. In other words, we expect our approach has the flexibility to dynamically adjust false positive and false negative predictions in accordance with the operational requirement from the system administrator.

To cope with this complicated scenario, we use the stacked ensemble learning method [14], which is a meta-algorithm that combines several machine learning techniques using the meta-classifier as our anomaly prediction engine. Different prediction models capture information from different angles with different advantages and disadvantages. By adequately leveraging the uniqueness of each model, in most cases, it is possible to obtain a higher prediction accuracy than by using a single learner model [15]. The bottom of Figure 1 illustrates our proposed anomaly detection engine, which consists of three models: an autoencoder (AE) with a support vector machine (SVM) and a random forest (RF) model.

1) Autoencoder with OneClass SVM (AE_SVM)

An autoencoder is a type of unsupervised deep neural network that is designed to train and replicate input as being similar to its output. An autoencoder is composed of an encoder and a decoder, which can have multiple layers. Training an autoencoder is unsupervised in the sense that no labeled data is needed. The training process is optimized through a cost function, which measures the error between the input x and its reconstruction at the output x'. Figure 2 describes this concept. Please note that in our implementation, we only use the normal traffic logs as the training dataset. By doing this, we expect the AE to be more sensitive to the anomaly traffic data by returning a higher loss result.

The loss value implies the quality of model behavior after each iteration of optimization. We adequately tuned the parameters, such as (1) the size of the hidden layer in the sparse autoencoder—for example, using 40 and 20 for the NSL-KDD; and (2) the sparsity value p was set to 0.05.

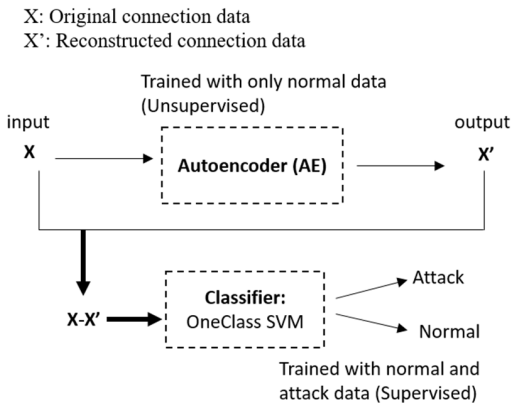

Figure 2. Implementation of autoencoder classifier.

Deep Autoencoder can be a classification model by adding a softmax layer for the task of classification. However, based on our investigation and related work from [16] , it does not carry out good classification result. Thus, we replace the softmax by OneClass SVM to be the intrusion discriminator. SVM often has a higher time complexity than other machine learning models. Since the preceding AE reduces the number of features before inputting to the SVM model, we do not observe significant latency in this study.

2) Random Forest (RF)

To evaluate the intrusion behavior from a different point of view, we adopted the random forest (RF) algorithm. The RF mechanism is based on an ensemble classifier, which builds multiple decision trees and merges them together to get a more accurate and stable prediction. Several studies [11] [17] [18] have reported that the RF algorithm outperforms other traditional classifiers in the effective classification of anomaly traffic, and we adopted the concept of those studies to implement our own RF model.

3) Stacked ensemble learning

The final intrusion prediction result comes from the optimal parameters of the ensemble model, as shown in (3).

$$\text{Intrusion}_{\text{prediction}} = \alpha(\text{AE}_{\text{SVM}}) + \beta\ (\text{RF}) \quad (3)$$
$$; \textit{where } \alpha + \beta = 1$$

To obtain the optimal parameters of [α, β], we emulate all possible combinations with a step of 0.01. We obtained optimal [α, β] values for our three standalone network testing scenarios. For the NSL-KDD, UNSW-NB15, and our campus network data, the values are [0.86, 0.14], [0.43, 0.57], and [0.08, 0.92] respectively.

IV. Experiment

In this section, we evaluate our proposed anomaly network intrusion detection system (ANIDS), based on the methodology discussed in the previous section. We conducted three experimental tests based on three network intrusion

evaluation datasets: NSL-KDD, UNSW-NB15, and our campus network traffic records (Palo Alto system log). To enable the fair comparison, our control group includes 5 different methods and they are:

1) 3 different classical machine learning models; i.e., the random forest (RF), support vector machine (SVM), and multiplayer perception (MLP)
2) 2 other reported approach's test results from NSL-KDD and UNSW-NB15 datasets.

Our evaluation method is based on the confusion matrix described in Table II, with metrics for accuracy, precision, and recall.

TABLE II. CONFUSION MATRIX FOR NIDS

Positive = A (attack) Negative = N (normal)		actual result	
		A	N
Predicted class	A	TP	FP
	N	FN	TN

A. Evaluation of NSL-KDD

It is evident from Figure 3 that our proposed ensemble model yielded a high accuracy of 91.7% and a recall of 92.93% to classify the attacks. The three classical models from our study (RF, SVM, and MLP) behave a similar classification pattern. All of these obtain a higher precision because they have a lower false-positive ratio; however, they all show a relatively lower recall rate (around 65% to 71%), which may be caused by their high number of false negative predictions. The false negative represents incorrectly classifying anomalous traffic as normal traffic. In this case, it decreases system reliability and incurs the risk of network intrusion. The experimental result shows that our approach achieves a balance of about 92% across the evaluation metrics of accuracy, recall, and precision. Note that two other recent related models, self-taught learning [9] and RBM [19], both have an accuracy and precision rate of less than 90%. For the recall rate, RBM has a rate of 78.8%, which is higher than our three studied classical models, but it is significantly lower than our proposed ensemble method (92.4%). The recall rate from self-taught learning (96%) is slightly higher than our proposed model (92.4%), but both its accuracy and precision are lower than our approach.

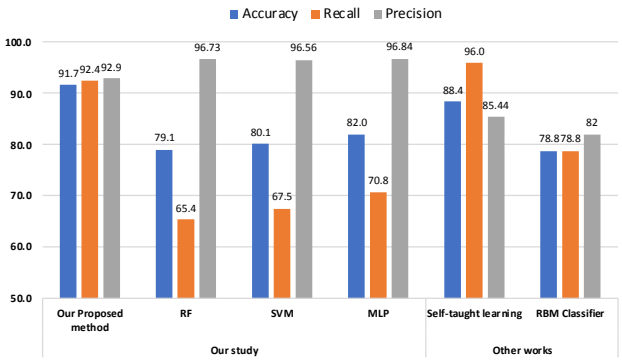

Figure 3. Test results comparison from NSL-KDD dataset .

B. Evaluation of UNSW-NB15

Figure 4 demonstrations our test results from the UNSW-NB15 dataset. Similar to the NSL-KDD test result, our proposed approach achieves a balance of about 92% across the evaluation metrics of accuracy, recall, and precision. Compare these results to our three studied classical methods and two other recent related models, self-taught learning [9] and cascade ANN [20]. Those five models have higher recall and lower accuracy rates (around 83% to 87%) and lower precision (around 78% to 82%) than our proposed method. This result indicates that those models have a higher false positive ratio, which tends to treat normal traffic as an anomaly. Users may find it highly problematic if system administrators enforce the prediction results to deny normal traffic. In addition, our proposed method obtained the highest accuracy rate (91.8%) and recall rate (91.7%) among all the models that were compared. Based on the above assessment, we conclude that our proposed method also provides a promising test result in this UNSW-NB15 use case.

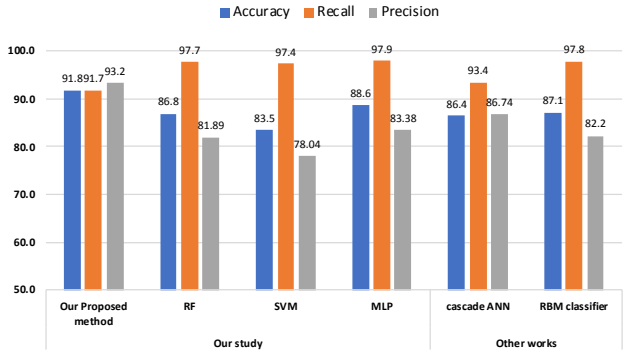

Figure 4. Test results comparison from UNSW-NB15 dataset .

C. Evaluation of Palo Alto system log datasets

In other words, in this test, we tried to apply our machine learning method to replicate the capability of intrusion detection from Palo Alto's mechanism. Please note that this is data from a real network and that Palo Alto's behavior does not guarantee to provide the absolute truth of anomaly detection. We continued to use those three evaluation metrics, but this scenario indicated that our approach could act as a real-time intrusion detection system and to inspect large-scale network traffic logs. Figure 5 shows the prediction results from our proposed model, as well as from those three classical machine learning models. Obviously, our method outperforms those three models across all evaluation metrics. The RF algorithm is highly like our proposed ensemble learning method. We consider that this finding is due to our campus network is closer to a real use case, which the threat records do not find to be as complicated and artificially created as the simulation-based benchmark datasets of NSL-KDD and UNSW-NB15. In this case, a well-tuned RF model is enough for threat detection. In fact, the RF model also looks to be three times faster than our proposed stacked ensemble approach.

The balanced test results in term of accuracy, precision, and recall from those three evaluation datasets implies that while having high accuracy, our method can also limit both false positive and false negative predictions. We believe that our proposed model provides an ideal solution for implementing an effective ANIDS system.

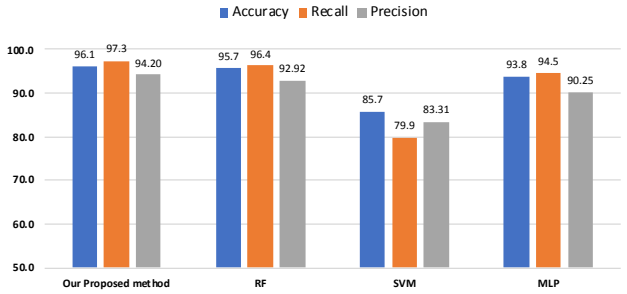

Figure 5. Test results comparison from Palo Alto log dataset .

V. Conclusion and Future Works

In this research, we propose an anomaly network intrusion detection system (ANIDS) based on a stacked ensemble learning model that consists of models of an autoencoder (AE), support vector machine (SVM), and random forest (RF). Our proposed methodology includes steps for data collection and data preprocessing and is applicable to different network environments. We show that our approach is applicable in different standalone networks by first evaluating our approach through two well-known NIDS benchmark datasets, NSL-KDD and UNSW-NB15. In addition, we further apply our model to a hundred-million scale of a Palo Alto system network logs collected from our campus networking environment. To show the feasibility of our approach, we compared our proposed approach to three different machine learning models, as well as to reported results from two other related studies. Our experimental results show that our proposed ensemble model provides a high classification accuracy and is also accurate for the metrics of precision and recall. This result implies that we also limit both false positive and false negative predictions.

References

[1] C. Whitepaper, "Cisco Visual Networking Index: Forecast and Trends, 2017–2022," Cisco, 2018.

[2] M. Tavallaee, E. Bagheri, W. Lu and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009.

[3] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective,* vol. 25, no. 1-3, 2016.

[4] P. A. Networks, "TechDoc Palo Alto," [Online]. Available: paloaltonetworks.com.

[5] X. Bellekens, C. Tachtatzis, R. Atkinson, C. Renfrew and T. Kirkham, "A Highly-Efficient Memory-Compression Scheme for GPU-Accelerated Intrusion Detection Systems," in *Proceedings of International Conference of Security of Information and Networks (SIN)*, 2014.

[6] N.-F. N.-F. Huang, H.-W. Hung, S.-H. Lai, Y.-M. Chu and W.-Y. Tsai, "A GPU-based Multiple-pattern Matching Algorithm for Network Intrusion Detection Systems," in *International Conference on Advanced Information Networking and Applications*, 2008.

[7] G. Vasiliadis, S. Antonatos, M. Polychronakis, E. P. Markatos and S. Ioannidis, "Gnort: High Performance Network Intrusion Detection Using Graphics Processors," *International Workshop on Recent Advances in Intrusion Detection,* pp. 116-134, 2008.

[8] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *Proceedings of IEEE International Conference on Communication Software and Networks (ICCSN)*, 2016.

[9] Q. Niyaz, W. Sun, A. Javaid and M. Alam, "A Deep Learning Approach for Network Intrusion Detection System," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT)*, 2015.

[10] J. Jha and Leena Ragha, "Intrusion Detection System using Support Vector Machine," in *Proceeding of International Conference & workshop on Advanced Computing (ICWAC)*, 2013.

[11] N. Farnaaz and M.A.Jabbar, "Random Forest Modeling for Network Intrusion Detection System," *Procedia Computer Science,* vol. 89, pp. 213-217, 2016.

[12] S. Seo, S. Park and J. Kim, "Improvement of Network Intrusion Detection Accuracy by Using Restricted Boltzmann Machine," in *Proceedings of International Conference on Computational Intelligence and Communication Networks (CICN)*, 2016.

[13] I. H. Witten, E. Frank and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques (fourth edition), Elsevier, 2017.

[14] A. Dixit, Ensemble Machine Learning: A beginner's guide that combines powerful machine learning algorithms to build optimized models, 2017.

[15] J. Sill, G. Takacs, L. Mackey and D. Lin, "Feature-Weighted Linear Stacking," arXiv:0911.0460.

[16] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," *IEEE Transactions on Emerging Topics in Computational Intelligence,* vol. 2, no. 1, 2018.

[17] Y. Chang, W. Li and Z. Yang, "Network Intrusion Detection Based on Random Forest and Support Vector Machine," in *Proceedings of IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, 2017.

[18] J. Zhang, M. Zulkernine and A. Haque, "Random-Forests-Based Network Intrusion Detection Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) ,* vol. 38, no. 5, 2008.

[19] J. Yan, D. Jin and P. Liu, "A Comparative Study of Off-Line Deep Learning Based Network Intrusion Detection," in *International Conference on Ubiquitous and Future Networks (ICUFN)*, 2018.

[20] M. M. Baig, M. M. Swais and E.-S. M. EI-Alfy, "A multiclass cascade of artificial neural network for network intrusion detection," *Journal of Intelligent & Fuzzy Systems,* vol. 32, pp. 2875-2883, 2017.