A Deep Reinforcement Learning Approach for Anomaly Network Intrusion Detection System

Ying-Feng Hsu, Morito Matsuoka

Cybermedia Center, Osaka University, Osaka, Japan
Email: {yf.hsu, matsuoka}@cmc.osaka-u.ac.jp

***Abstract*—Network intrusion detection systems (NIDS) are essential for organizations to ensure the safety and security of their communication and information. In this paper, we propose a deep reinforcement learning-based (DRL) for anomaly network intrusion detection system. Our system has the ability of self-updating to reflect new types of network traffic behavior. This study includes three major contributions. First, 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. Second, to show the feasibility of our approach, we compared our method with three other classic machine learning methods and two related published results. Third, our model is capable of processing a million scale of network traffic on a real-time basis.**

***Keywords*—network intrusion detection system; NIDS; deep reinforcement learning; deep learning; random forest; support vector machine**

I. Introduction

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]. Due to increases in the volume and sophistication of network-based attacks, effective techniques and tools are required to avoid the threats of disclosure of sensitive information, disruption, and unauthorized access. 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.

Typically, NIDS can be classified into two categories: SNIDS (signature-based network intrusion detection system) and ANIDS (anomaly-based network intrusion detection system). Nowadays, the mainstream protection approaches rely on SNIDS, which are based on a pattern-matching algorithm to inspect and identify threats from the incoming packet by comparing a preinstalled attacked signature database against incoming network activities. SNIDS has good detection capabilities for known attacks, with fewer false alarms. However, it requires a pre-installed signature and is not aware of unknown threat patterns. When using SNIDS for network protection, frequent maintenance and routine updates are required. In contrast, ANIDS focuses 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 an anomaly network intrusion detection system (ANIDS) based on deep reinforcement learning (DRL). Our DRL anomaly detection engine has two modes, i.e., detection mode and learning mode, that can be flexibly switched. At any time when the performance of the detection mode under the predefined threshold, the system switch to the learning mode to learn new network traffics pattern. We evaluated our approach using two well-established benchmark network intrusion simulation datasets: NSL-KDD and UNSW-NB15. In addition, we also applied our method to our campus network environment, which consists of about 300 million daily network traffic records and is about 100 times larger than either of those two synthetic datasets. To further ensure the reliability of our proposed work, we include three classic ML methods and two recent published test results as baseline comparisons.

The rest of this paper is organized as follows: In Section II, we survey related approaches for developing network intrusion detection systems, especially for the reinforcement learning-based approaches. In Section III, we discuss the background of this study and the evaluation datasets. In Section IV, we describe our proposed methodology: we detail the system architecture and elaborate upon its components. To validate the advantages and performance of the proposed approach, we provide a comprehensive evaluation in Section V. We conclude this paper and discuss future directions for research in Section VI.

II. Related Work

Various machine learning anomaly detection or classification techniques have gained a great deal of attention, such as [2] , [3] and, [4]. Since there are only a few approaches that are based on the reinforcement learning method, we provide a detailed summary of those related works in this section. The work of [2] proposes an intrusion detection system by combining methods of log correlation, reinforcement learning, and association rule. In its work, reinforcement learning helps to detect the unknown attack by motivating the rewards activities to identify the known and unknown attacks. A context-adaptive IDS presented by [3] uses multiple independent deep reinforcement learning agents distributed across the network to enhance detection accuracy for the new and complex attacks. This proposed work was evaluated by the NSL-KDD, UNSW-NB15, and AWID dataset to show its capability of higher accuracy and low false positive rate (FPR) compared to other state-of-the-art solutions. An adversarial reinforcement learning approach [4] incorporates the environmental learning process with a modified reinforcement learning algorithm. Through its experimental results of using AWID and NSL-KDD datasets, the method is adequate for adjusting the RL to a surprised IDS intrusion classification problem with weighted accuracy (>0.8) and F1 (>0.79) metrics. Moreover, defining a reliable reward function is the most challenging step for a reinforcement learning-based IDS. As stated in [5], the approach modifies the concept of the class DRL live interaction paradigm by replacing it with a sampling function of recorded training intrusions. Another work [6] also propose an algorithm by modifying the Q-learning algorithm to learn optimum cut value for a different attribute of traffic data. The proposed algorithm is able to obtain a high classifying accuracy of 98% and has a faster processing speed for real-time prediction. In addition, several works apply Reinforcement Learning-based Intrusion Detection System (RL-IDS) for distributed network infrastructure. For example, [7] uses straightforward Q-learning with a simple exploration strategy in a hierarchical architecture network sensor agent. Each network sensor agent learns the local state and sends it to with central agent. By collecting all signals from local agents, the central agent learns the intrusion alarm. Similarly, another RL-IDS was proposed by [8] as the IDS engine to analyzing and monitoring a sensor network. This study conducts a simulation test to compare the proposed RL-IDS with an adaptive machine learning-based IDS (AML-IDS) and Clustered Hybrid IDS (ASCH-IDS). The experimental results show that the proposed RL-IDS outperforms other comparison models with about near 100% accuracy and precision-recall rate.

III. Background

In this study, we evaluated our approach using two well-established benchmark network intrusion simulation datasets: NSL-KDD and UNSW-NB15. In addition, we also applied our method to our campus network environment, which consists of about 300 million daily network traffic records and is about 100 times larger than either of those two synthetic datasets. The background of those three datasets is provided in the following.

A. NSL-KDD dataset

The NSL-KDD [9] dataset was originally taken from the Kdd99 dataset for a data science competition. The training dataset consisted of 125,973 TCP connection records, while the testing dataset consisted of 22,544 records. There were 41 features, including 9 basic features of individual TCP connections, 13 content features within a connection, 9 temporal features computed within a two-second time window, and 10 network traffic statistics related features. In the training dataset, there are normal types and 22 attack types, while in the testing dataset, there are 37 attack types (15 types not in the training dataset); this model was designed to handle unknown attacks. Other than normal and abnormal detection, it is common to see a 5-class classification problem (normal, DoS, R2L, U2R, and probing)

B. UNSW-NB15 dataset

The UNSW-NB15 dataset [10] was created by the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). Similar to the NSL-KDD dataset, the UNSW-NB15 dataset is also a simulation-based network traffic dataset. There are 10 categories of data types: normal, analysis, backdoor, DoS, Exploits, Fuzzers, Generic, reconnaissance, shellcode, and worms. Note that only normal and DoS type are common features with this dataset and NSL-KDD. This dataset consists of 49 features, including 5 flow features, 13 basic features, 8 content features, 9 time features, and 14 other features, and that there are only 5 common features with NSL-KDD. There are 257,673 traffic records, among which 175,341 are used for the training set, and the remaining 82,232 are used for testing.

C. Palo Alto Networks (PANW) system log dataset

The Palo Alto network [11] log dataset was obtained from commercial hardware-based firewalls designed to provide protection for medium to large business networks. Most commercial firewalls are complex and often require special training to understand the operations. The Palo Alto system traffic log has 61 features; as compared to both the NSL-KDD and UNSW-NB15 datasets, it can be converted to 8 common features, such as duration, Protocol type, service, src_bytes, dest_bytes, src_port, dest port, and dst_host_count (dst_host_srv_count). There are 3 types of threat labels: spyware, virus, and vulnerability. Spyware and vulnerability appear in a similar amount and account for 99% of the threats logged. Fig. 1 shows the campus network traffic statistics on weekdays. The maximum and average network traffics per second is 5,520 records/s and 3912 records/s, respectively.

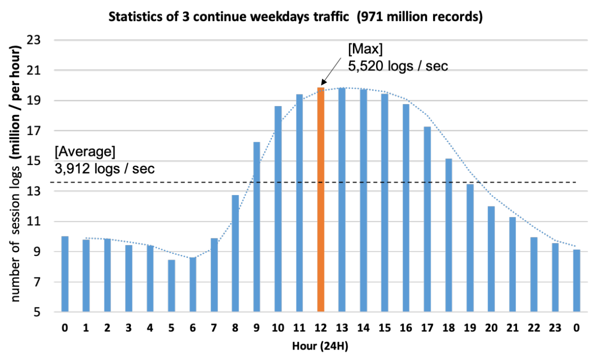

Fig. 1. Campus network traffic statistics on weekdays

IV. Methodology

In this section, we elaborate on our proposed anomaly network intrusion detection system (ANIDS) system. Fig. 2 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. Lastly, there is a deep reinforcement learning (DRL) based anomaly detection engine for the intrusion detection task.

(1) 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. Regarding the size of our campus traffic logs, it contains about

300 million records on weekdays and about 30% to 40% less during the holiday and weekend. The total data size of a single weekday's traffic log is about 120 GB, including a threat log of around 200 MB to 1.5 GB; i.e., about 98.5% of data is considered to be normal traffic data.

(2) Data preprocessing

Time series data usually contains a certain amount of discrepancies, especially in large-scale and high-dimensional data sets (such as those found in this study). We perform data preprocessing for common data discrepancies (such as data redundancy and data outliers) to ensure the quality of the prediction model. Three standalone network environment datasets (NSL-KDD, UNSW-NB15, and our Palo Alto system traffic log) are used to develop our system. Besides fixing the data discrepancy issue, our procedure also includes model training related preprocessing functions, such as the following.

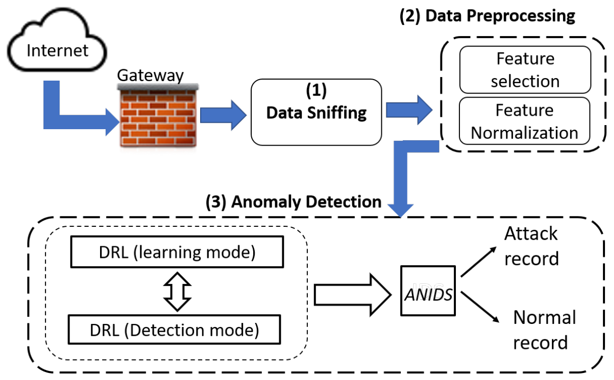

Fig. 2. The system architecture of proposed ANIDS.

- *Feature selection:*

The concept of feature selection is that the data may contain irrelevant, redundant or noisy features, which can be eliminated without losing information or affecting prediction accuracy. For example, a feature with only one unique value should be deleted from the subsequent process. Besides, from the perspective of confidentiality and sensitivity, personal identification should be excluded from the selected features, such as IP addresses and user accounts. The original PANW log includes 61 features and is divided into a threat log (which contains only threat-related records) and traffic (which contains all network records). Those two traffic log datasets can be joined by session id. We use all similar features from NSL-KDD and UNSW-NB15 since those features imply particular important information. Such as "Bytes Sent", "Bytes Received" …etc.

- *Feature normalization:*

Network traffic trace log is heterogeneous with different scales and units. In practice, it is recommended to apply data normalization before the machine learning process [12]. In our approach, we first standard categorical features such as IP address, protocol by using one-hot encoding. For example, the IP address is converted to 32bits (8bits x 4) binary vector. For numerical features, 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)

$$\bar{a} = \frac{\log(a+1)}{\log(a+1)_{max}} \tag{1}$$

; where a is the original raw value

For those low standard deviation related features, we normalize them by the max value, as in (2).

$$\bar{a} = \frac{a}{a_{max}} \tag{2}$$

; where a is the original raw value

(3) Deep Reinforcement Learning (DRL) Model

The core of our proposed ANIDS is the reinforcement learning-based anomaly detection engine; it is designed to achieve self-updating of the detection model to automatically learn new network traffic behavior, especially for new types of attack. Unlike the other IDS studies that are mainly based on the simulated dataset, our proposed model is designed for real-time operation in the network environment. Thus, our system considers both accuracy and process speed. In our approach, network traffic data is treated as environment state variables in RL, the RL agent is the intrusion detection engine, the action is equal to the result of intrusion detection, and the reward is determined by whether the recognition result is correct. Fig. 3 shows the structure of this system. In order to achieve the self-update function, reinforcement learning has two modes: learning mode and detection mode. The flow of these two modes is organized as follows:

Learning Mode:

1. The RL agent process the state variables, which is converted from raw network traffic data, and gives an action.
2. The reward function module calculates the reward based on action and label, feedback to the RL agent.
3. The RL agent updates its policy (intrusion detection model) based on the reward and states.
4. Back to step 1.

Detection mode:

1. RL agent process the state variables, which is converted from raw network traffic data, and gives an action.
2. The reward function module gives a dummy reward to the RL agent, keep the process working.
3. Back to step 1.

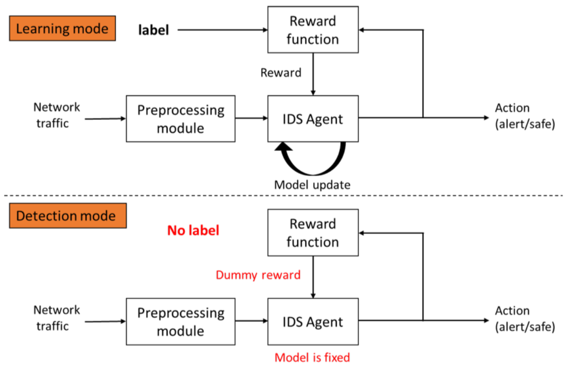

Fig. 3. Two operation modes of the proposed reinforcement learning

In the learning mode, the DRL agent evaluates the performance of the detection results by monitoring the reward. If the reward drops, it will update the detection model with current data to improve the performance of intrusion detection. In the detection mode, the RL agent will use a fixed detection model to process network traffic, and the reward is a dummy reward; it is only used to keep the operation process. The difference between these two modes is whether the reward function calculates the true reward with the label. We set a switch flag to make the system can switch between the two modes flexibly. This setting allows the system to evaluate and update the detection model at any time. Please note that the DRL agent will give a certain action as the intrusion detection result; our system is able to extend for the task of intrusion prevention.

The design of our DRL model is based on a deep Q network (DQN agent), which uses a deep neural network to calculate the expected value of rewards. Besides, to ensure the learning stability of the learning process, we fixed the maximum expected reward and train the model with an equal amount of 50 normal and 50 attack log data in each episode. Other parameters settings include 100 steps per episode (2,100,000 full training steps) and Adadelta optimizer (with learning rate = 0.001, rho=095, and epsilon =1e-07). Fig. 4 shows an example of the fluctuation of the accuracy when using the learning mode for 10 connective days' network traffic data. We observe many phenomena that the accuracy drops for a while and then bouncing back shortly. This shows that the system continuously updates the model during processing network traffics to improve its detection performance.

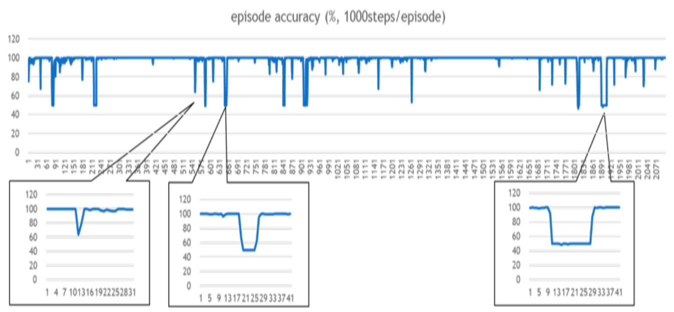

Fig. 4. The fluctuation of accuracy for the learning mode

V. Experiment

In this section, we evaluate our proposed DRL based ANIDS by using the methodology discussed in the previous section. Our system is designed to have the ability of self-updating to continue to detect the abnormal incoming network traffic. It should be noted that in the learning mode, the model is continuously updated until it stops learning. Thus, in the evaluation, we use the prediction result from detection mode. We conducted three experimental tests based on two well-known synthesis intrusion evaluation datasets NSL-KDD, UNSW-NB15, and our real campus network traffic records (Palo Alto system log). To ensure the feasibility of our proposed method, we include a total 5 different baseline approaches in the following 2 categories.

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.

Besides, there are two considerations for our experimental setup. First, we considered a binary classification. In a real-time NIDS practice, successfully identifying the normal and anomaly traffic is enough to secure the network, and the finely granulated threat subcategories can be analyzed offline when there is a need to do so. Additionally, there are many ways to compare the performance of the result; for example, a 2-class classification (normal and anomaly) or a 5-classification (normal and 4 subtypes of attacks in NSL-KDD), the 2-class classification test is more intuitive and has been reported in many related studies. Second, we used pure testing data. Some studies combined training and testing datasets and randomly selected a certain number of samples. For example, Ramp-KSVCR [13] obtained an accuracy of 93.52% for UNSW-NB15, while [14] proposed a geometric area analysis method and reported an accuracy of 99.7% and 92.8% for NSL-KDD and UNSW-NB15, respectively. Both approaches are considered to be one of the highest results in both datasets. Although a more accurate result testing result can be obtained by performing this process, we believed it is not the original intention of creating training and testing datasets from both datasets creators.

A. *Evaluation metrics*

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

- Accuracy:

In our case, accuracy represents the system able to identify traffic in anomalous and normal conditions correctly. It is the percentage of correctly classified traffic records over the total number of records.

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}$$

Accuracy may not properly evaluate an imbalanced data label scenario like the network traffic records, where there is always a significantly higher portion of normal traffic data. As a result, we include both recall and precision as our evaluation metrics.

- Recall:

The recall is the percentage of correctly identified positive traffic, i.e., the threat-related records divided by the total number of relevant traffic records. Depending on the research area, there are other synonyms for this term, including detection rate (DR), sensitivity, and true positive rate (TPR)

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- Precision:

The precision measures the degree of correctly detecting the threat-related records over the total number of threat records predicted by the model.

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

B. *Evaluation of NSL-KDD*

It is evident from Fig. 5 that our proposed DRL model outperforms other baseline approaches with high accuracy of 91.4% and a recall of 90.2% 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 [15] and RBM [16], 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 still lower than our proposed DRL method (92.4%). The recall rate from self-taught learning (96%) is slightly higher than our proposed model (90.2%), but both its accuracy and precision are lower than our approach. Based on the above comparison, we can conclude that our proposed method is promising in this use case.

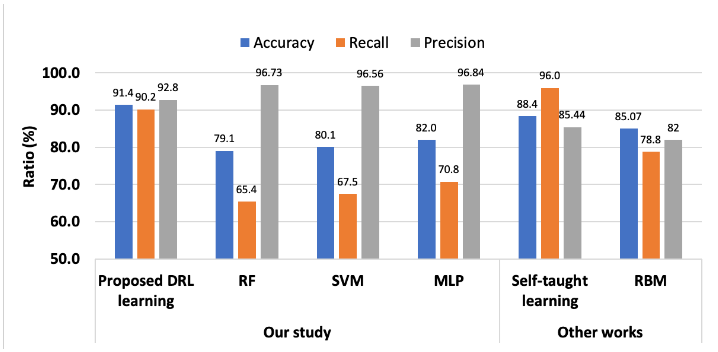

Fig. 5. Test results comparison from NSL-KDD dataset

C. *Evaluation of UNSW-NB15*

Fig. 6 demonstrates our test results from the UNSW-NB15 dataset. Similar to the NSL-KDD test result, our proposed DRL 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 [15] and cascade ANN [17]. Those five models have the detection pattern of higher recall (around 97%), lower accuracy (around 83% to 87%), and lower precision (around 78% to 82%) than our proposed DRL 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 precision (93.2%) among all the models that were compared. Based on the above assessment, we concluded that the proposed DRL method also provides promising test results in the UNSW-NB15 use case.

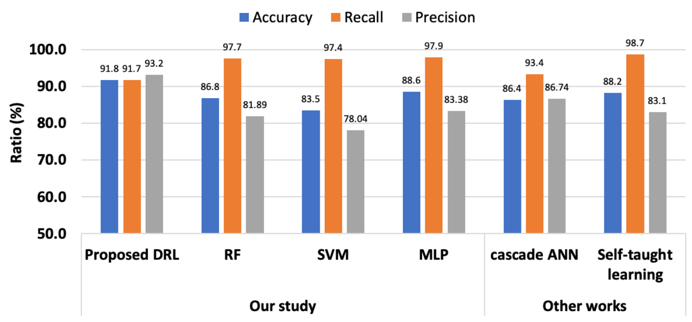

Fig. 6. Test results comparison from UNSW-NB15 dataset

D. *Evaluation of real-time campus network traffic*

The log dataset is collected and abstract from 14 consecutive days' campus network traffic log and sorted in time sequence. The data from the first 10 days is used as the training dataset, and the last 4 days is used as the testing dataset. The training dataset consisted of 2,218,703 connection records, while the testing dataset consisted of 613,337 records. Since this is our campus network and contains confident information, in this experiment, we only provide the baseline comparison from those three classical ML methods. Fig. 7 shows the test results of each model. All models achieved very high precision (about 99%), which means the false positive predictions are very few in these models. The highest accuracy and precision score is obtained by our DRL method. For the recall, all models show a similar score of 96%. This result demonstrates the effectiveness of continuously update the model. Overall, all the models obtain good results in this test; we consider that this is due to our campus network log data is a real network traffic dataset; the normal traffic and attack traffic do not have many patterns in a short period of time. Even so, continuously updating the model still shows its effect.

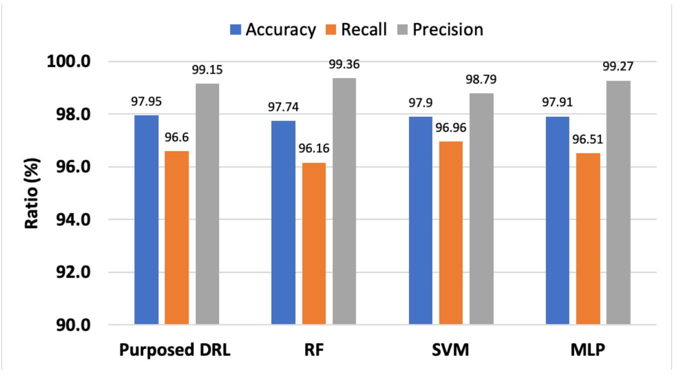

Fig. 7. Performance scores for all models (14 days' Palo Alto system log dataset result)

In addition, we measured the processing speed of the two modes. We first test the processing speed from a single CPU Core i7 7700 and obtain the result of about 494 logs/sec and 4,522 logs/sec for the learning mode and detection mode, respectively. The processing speed of the detection mode is 9 times faster than the learning mode. We further evaluated this DRL model to an HPC server, which includes dual Intel Xeon E5-2690 v4 processors (56 cores in total), 768 GB of main memory. Fig 8 shows the result of processing speed. By utilizing the multiple CPU parallel computing, the processing speed of the detection mode is far beyond our campus network traffic (maximum and average traffic log per second is 5,520 records/s and 3912 records/s as shown in Fig. 1). In fact, it is applicable for even large-scale networking traffic environment.

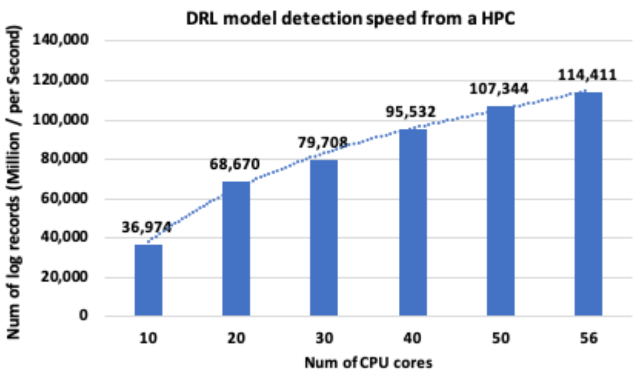

Fig. 8. The processing speed of the DRL model by using parallel computing

VI. Conclusion

A NIDS monitors network traffic and detects abnormal activities or cyber-attacks to ensure the safety and security of communication and information. In this research, we propose an anomaly network intrusion detection system (ANIDS) based on deep reinforcement learning (DRL). Our proposed methodology includes steps for data collection and data preprocessing and is applicable to different network environments. This reinforcement learning method can be operated in two modes. The learning mode is designed to continuously learn and update the model to maintain high detection accuracy for continue coming network traffic while detection mode is decided for the performance of high processing speed. To show the feasibility of our approach, we further apply our model to a hundred-million scale of a Palo Alto system network logs collected from our campus networking environment. We compared our proposed DRL to three different machine learning models. Our proposed approach model shows both the highest detection accuracy, high performance of the processing speed, and effectiveness of continuous model updating.

References

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

[2] B. Deokar and A. Hazarnis, "Intrusion Detection System using Log Files and Reinforcement Learning," *International Journal of Computer Applications,* vol. 45, no. 19, 2012.

[3] K. Sethi, E. S. Rupesh, R. Kumar, P. Bera and Y. V. Madhav, "A context-aware robust intrusion detection system: a reinforcement learning-based approach," *International Journal of Information Security,* 2019.

[4] G. Caminero, M. Lopez-Martin and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Computer Networks,* vol. 159, 2019.

[5] M. Lopez-Martin, B. Carro and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Systems With Applications,* vol. 141, 2020.

[6] N. Sengupta, J. Sen, J. Sil and M. Saha, "Designing of on line intrusion detection system using rough set theory and Q-learning algorithm," *Neurocomputing,* vol. 111, 2013.

[7] A. Servin and D. Kudenko, "Multi-agent Reinforcement Learning for Intrusion Detection," in *Proceedings of Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, 2006.

[8] S. Otoum, B. Kantarci and H. Mouftah, "Empowering Reinforcement Learning on Big Sensed Data for Intrusion Detection," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2019.

[9] 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.

[10] 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.

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

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

[13] W. H. Seyed Mojtaba Hosseini Bamakan and S. Yong, "Ramp loss K-Support Vector Classification-Regression; a robust and sparse multi-class approach to the intrusion detection problem," *Knowledge-Based Systems,* vol. 126, pp. 113-126, 2017.

[14] N. Moustafa, J. Slay and G. Creech, "Novel Geometric Area Analysis Technique for Anomaly Detection using Trapezoidal Area Estimation on Large-Scale Networks," *IEEE Transactions on Big Data,* 2017.

[15] 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.

[16] 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.

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