Reinforcement Learning for Intrusion Detection

Ahmed Mohamed Saad Emam Saad[1(✉)] and Beytullah Yildiz[2]

[1] Texas A&M University-Corpus Christi, Corpus Christi, TX 78412, USA
asaad1@islander.tamucc.edu
[2] Atilim University, Ankara, Turkey
beytullah.yildiz@atilim.edu.tr

AQ1

Abstract. Network-based technologies such as cloud computing, web services, and Internet of Things systems are becoming widely used due to their flexibility and preeminence. On the other hand, the exponential proliferation of network-based technologies exacerbated network security concerns. Intrusion takes an important share in the security concerns surrounding network-based technologies. Developing a robust intrusion detection system is crucial to solving the intrusion problem and ensuring the secure delivery of network-based technologies and services. In this paper, we propose a novel approach using deep reinforcement learning to detect intrusions to make network applications more secure, reliable, and efficient. As for the reinforcement learning approach, Deep Q-learning is used alongside a custom-built Gym environment that mimics network attacks and guides the learning process. The NSL-KDD dataset is used to create the reinforcement learning environment to train and evaluate the proposed model. The experimental results show that our proposed reinforcement learning approach outperforms other related solutions in the literature, achieving an accuracy that exceeds 93%.

Keywords: Reinforcement learning · Deep Q-learning · OpenAI Gym · Network security · Machine learning · Intrusion detection system

1 Introduction

Network based computer systems and technologies like web services, cloud computing, and Internet of Things (IoT) systems are becoming more popular. These technologies are prone to intrusion, and the growing popularity of network-based systems made the intrusion issue worse. We can get an estimate on the expansion magnitude of network-based technologies and in return the intrusion problem by examining the market of specific network-based technologies such as cloud computing services. There has been a massive increase in the market and the revenue of cloud services at around 54.9 [1] and 129 [2] billion US dollars from 2017 to 2020, respectively. It is evident from such an increase that network-based technologies are getting a lot of attraction, which increases the scale of the intrusion issue. This increasing scale comes with significant economic costs, which has

F. P. García Márquez et al. (Eds.): ICCIDA 2022, LNNS 643, pp. 1–14, 2023.
https://doi.org/10.1007/978-3-031-27099-4_18

been confirmed by two studies carried out by McAfee cyber-security firm. The two studies were conducted over a 6-year period and show the alarming increase in the economic cost of cyber-crimes. The first study shows that in 2014 the cost of cyber-attacks was around 475 billion US dollars [3], and the second study shows that in 2020 the cost of cyber-attacks was almost 1 trillion US dollars [4]. Since most of the network-based technologies and resources are obtained from a remote service provider that is not locally present through a medium, this raises the question of how to secure the medium used to obtain these services from intrusion, which in this case is the network system. The answer will significantly reduce the economical cost caused by intrusions and cyber-crimes. Securing that medium requires an absolute necessity of a modern solution to detect intrusions using a particular intrusion detection system (IDS).

Intrusion detection systems work in different ways and have many architectures. The most common two types of network intrusion detection systems are signature (misuse) based intrusion detection system and anomaly-based intrusion detection system [5]. Anomaly-based intrusion detection system attempts to deal with a novel attack by learning the normal pattern of network traffics, and any deviation from that normal pattern is considered as an intrusion [5]. The downside of this system is its high sensitivity that leads to a high false positive rate. Signature-based intrusion or misuse detection system works similarly to traditional rule-based intrusion detection systems like snort intrusion detection system [6]. Both were used to tackle and solve the intrusion problem. However, with the increase in novel attacks and the continuous change in the attack types and styles, rule-based intrusion detection systems are vulnerable. Even with rapid updates to their rules, they can not keep up with the continuous change in malicious attacks.

Therefore, creating a novel, scalable, and adaptive approach for detecting intrusions in network systems that copes with the new malicious attacks is a necessity. This is where machine learning comes into place with its adaptability and flexibility. It can provide a solution for the intrusion issue addressed before and solve the limitation of rule-based intrusion detection systems.

Although machine learning offers a solution, not all machine learning approaches are created equal. Machine learning can be categorized into three major topics: the first is reinforcement learning (RL), the second is unsupervised learning and the third is supervised learning. Many solutions were developed for intrusion detection using the three different machine learning approaches. Most of the researched and implemented approaches are based on supervised and the unsupervised learning. Supervised learning is based on the idea of recognizing attacks from captured and labeled network traffic attack data upon which it can detect relatively similar attacks. The unsupervised learning approach uses unlabeled datasets to learn and classify attacks based on their common features and patterns.

Both approaches have downsides. Most of the real traffic data are present as unlabeled data. Labeling data is an extremely cumbersome and costly process. In addition, the continuously developed attacks and changes in attack patterns

render the supervised learning approaches inefficient and lead to a high false positive rate. On the other hand, the unsupervised learning approaches are inferior to the supervised ones in performance when presented with numerous features. Moreover, the feature engineering process for the large number of features in unlabeled data is a laborious task.

In this research, we propose a novel approach called OpenAI Gym Env-DQN (OGE-DQN) using reinforcement learning because, in theory, it is superior to other machine learning approaches in intrusion detection for the following reasons. First, it can go beyond the dataset by solving the labeling issue. Second, it can generalize and approximate when dealing with large observation space or features. Third, it can scale and adapt to numerous attack patterns, and it is not volatile to changes. The novel machine learning approach introduced utilizes a reinforcement learning-based algorithm called Deep Q-Network (DQN).

Reinforcement learning utilizes an environment to train an agent. We use OpenAI Gym library [7] to build the environment. The Gym environment guides the learning process through positive and negative rewards and makes use of the NSL-KDD intrusion detection dataset as a source of network traffic. In other words, the reinforcement learning agent learns from its previous actions by observing the states and rewards from the environment, so it can perform better actions in the future by maximizing the reward it gets from the environment. The reinforcement learning agent uses deep neural networks as a function approximator for Q-values associated with decision (action) making. For the NSL-KDD dataset, it is the largest and the most diverse in terms of attack types, and it fits the eleven criteria for an appropriate IDS dataset [8]. It is well-suited and serves the goal of this research. The dataset goes through a preprocessing stage before being used as the source of encoded network traffic.

The main contributions of this study are as follows:

1. We present the first attempt to create a custom-built standardized intrusion detection reinforcement learning environment using OpenAI Gym framework.
2. Our novel reinforcement learning approach offers a significant improvement in terms of metrics, such as accuracy, recall, and precision, compared to other relevant works from the literature.
3. A novel Gym environment is proven to be more expressive about intrusions and has more resolution. This is one of the key elements for the success of the reinforcement learning model.

The rest of this paper is organized as follows. Sections 2 and 3 provide discussion of the related work and a brief background, respectively. The research methodology is explained in Sect. 4. In Sect. 5, evaluation and benchmarking are explored. Finally, we conclude with the outcomes in Sect. 6.

2 Related Work

Numerous intrusion detection solutions using machine learning approaches and techniques were developed.

Liang et al. [9] suggested a hybrid approach of a multi-agent reinforcement learning model consisting of three parts: data management, analysis and response modules, and data collection. The analysis modules are based on deep learning to detect anomalies from the transport layer in the network. The dataset used for evaluation was the NSL-KDD dataset. The anomaly detection accuracy of 98% was achieved in an IoT environment. The ability for the proposed model to classify different types of attacks was accurate by 97%. Nevertheless, the model was only tested in an IoT environment where the types of attacks are very limited.

Koduvely [10] proposed making a Gym environment based on the OpenAI Gym environment concept to detect network intrusions using reinforcement learning and policy gradient model, which inspired us into building a Gym environment as part of the approach suggested in this paper. The proposed approach works by solving the environment, and for evaluation, a receiver operating characteristic (ROC) curve is used. The proposed solution's performance was not evaluated, and the False Positive (FP) and False Negative (FN) rates were unknown. The research also suggested implementing other techniques such as deep neural network and deep and wide neural network, but there was no continuation on this proposal.

The first approach that integrated a reinforcement learning framework as intrusion detection solution was introduced by Caminero et al. [11]. The approach was named Adversarial Environment using Reinforcement Learning (AE-RL). They created an environment that provides network traffic samples to the agent and also act as a second adversarial agent by increasing the classifier's incorrect predictions. Moreover, they implemented a new mechanism for dynamically over-sampling/under-sampling during training from the dataset to overcome the issue of the under represented classes. Their approach was tested on two datasets: NSL-KDD and AWID. The approach was compared with several other approaches that implemented supervised and reinforcement learning algorithms. They achieved an accuracy of 80%.

A novel approach using deep reinforcement learning to detect attacks in a network without requiring to solve an environment by directly using batches from two datasets (NSL-KDD and AWID) separately was suggested by Lopez-Martin et al. [12]. This approach proposes a new reward method for the model in the training process, whether the detection was correctly performed or not. They used four different approaches to implement their proposal, which are Policy Gradient, Double Deep Q-Network (DDQN), Actor-Critic, and Deep Q-Network (DQN). They claimed that the top performance was achieved by the Double Deep Q-Network approach and that they were able to decrease the overall computational time required compared to traditional machine learning approaches.

Suwannalai and Polprasert [13] proposed a multi agent deep reinforcement learning model to detect intrusions. They used the NSL-KDD dataset to train and test the proposed model. They created a multi-class model to evaluate their model capabilities on detecting each attack type in the dataset. The accuracy

of the model was 80% and the F1 score was 79%. The model showed very low accuracy in detecting minority attack types.

An intrusion detection system that utilizes a reinforcement learning approach based on Deep Q-learning was implemented by Hsu and Matsuoka [14]. The system uses two modes: learning mode and detection mode. They tested their model on a real captured traffic from their campus network, NSL-KDD dataset, and the UNSW-NB15 dataset. Moreover, the model achieved an accuracy of 97.95%, 91.4%, and 91.8% respectively. Additionally, they compared the performance of their approach with three other traditional machine learning approaches and two published research papers.

Ma and Shi [15] suggested building a network intrusion detection system based on a deep reinforcement algorithm known as Deep Q-learning. They tested their approach using the NSL-KDD benchmark dataset. Moreover, a suggestion was made to increase the number of less represented classes (R2L and U2R) and reduce the number of over represented classes (NORMAL, DoS, and PROBE) in the dataset by creating synthetic data using various over and under sampling techniques like Synthetic Minority Over-Sampling Technique (SMOTE), Random Over Sampling (ROS), NearMiss1, and NearMiss2. The SMOTE was the highest performing technique when combined with a reinforcement learning framework with an accuracy of 82%.

The most relevant work from the literature are [12,14,15] since they used a reinforcement learning approach and tested their approach on the same dataset used in this research. Therefore, these research papers' results will be used later for performance comparison purposes.

3 Deep Q-Learning

Reinforcement learning is an evolutionary machine learning approach that simulates the learning process in living organisms [16]. Learning is a process in which living organisms increase their knowledge in the scope of different tasks by accumulating knowledge that enhances their capability on how to perform certain tasks better as their knowledge increases. Similarly, machine learning algorithms can emulate living organisms' learning process through their exposure to data, which in this scenario simulates the accumulative knowledge living organisms acquire over time. In reinforcement learning, the algorithm does not need to have prior domain knowledge and can learn over time by trial and error [17]. The algorithm in that scenario represents the brain in living organisms. The main purpose in reinforcement learning is trying to develop that brain, which in this case is called an agent, and this is where the algorithm resides. The agent's duty is to sum up the reward R_t over time t and get as many positive rewards as possible. This process simulates doing correct actions in certain tasks given the prior knowledge accumulated in the brain of a living organism. The agent interacts with an observation acquired from the environment (see Fig. 1), which is called a state S_t. In living organisms, such state represents the interaction with the real world whereby experience or knowledge is gained. Building a well-structured reinforcement learning environment is paramount for optimizing the

agent's learning process and performance in a specific domain. For the reinforcement learning agent to take any action, it follows a strategy called an epsilon greedy strategy. We start by setting an exploration rate which is initially $\epsilon = 1$ and will decay by a certain rate at the beginning of each training loop, or in this context called episode. Then, a random number between "0" and "1" will be produced as a threshold. If that threshold is greater than ϵ, the agent will start exploitation to select its next action [18].

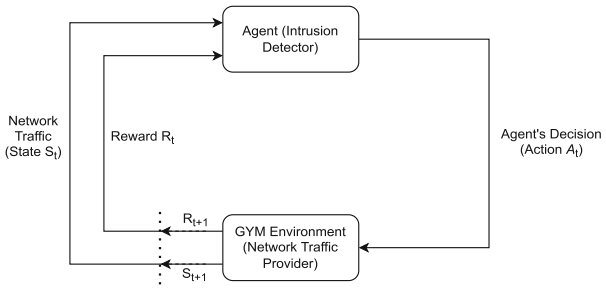

Fig. 1. Reinforcement learning workflow

The environment includes the dataset of captured network traffic features. It also evaluates the agent's actions (right or wrong). The observation represents the network traffic, and the agent represents the detection mechanism that decides whether the network traffic is an intrusion or not. The agent chooses its best action trajectory using a sequential decision-making process called Markov Decision Process (MDP). The sequential process starts by the agent observing a state from the environment. Depending on the state, the agent selects an action to perform. Afterwards, the agent gets a reward from the environment, and then another state is initiated. This entire process can be optimized in order for the agent to get the maximum accumulative reward, not just the immediate reward. Simply put, we are trying to map state-action pairs to rewards as represented in Eq. 1.

$$f(S_t, A_t) = R_{t+1} \tag{1}$$

The quality of the agent action in each step of the process is calculated by using the Bellman equation (Eq. 2). Q-Function is an indication of the quality of an action taken by the agent given a certain state. Therefore, it is referred to as the action-value function. The output of the function for any state-action is known as the Q-value which is the left hand side of the equation. The Bellman equation right hand side consists of two parts: the immediate reward R_{t+1} and the future Q-Values. A discount actor γ is added to make the process finite.

The $\max q_*(s', a')$ is the maximum discounted future reward we can get from some states onwards if we are going to follow this particular behavior. In our approach, we used neural networks to work as a function approximator to obtain the optimal Q-values [19]. This approach is known as Deep Q-learning. We use neural networks to map states to action-Q-value pairs as shown in Fig. 2.

$$q_*(s, a) = E\left[R_{t+1} + \gamma \max_{a'} q_*(s', a')\right] \tag{2}$$

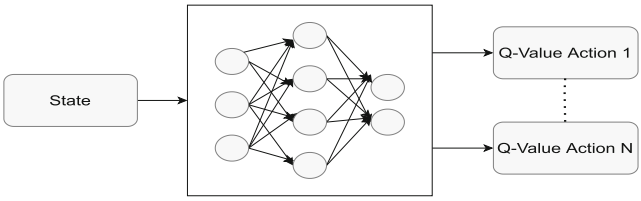

Fig. 2. Deep Q-learning

4 Methodology

This research proposes a novel approach for intrusion detection in network systems, which utilizes a deep reinforcement learning algorithm and network traffic present in the NSL-KDD intrusion detection dataset. The deep reinforcement learning approach used for learning is Deep Q-Learning. This approach includes building an RL agent using a reinforcement learning algorithm known as Deep Q-Network. It is used as part of the agent's structure to detect intrusions. Moreover, the approach includes creating and setting up an environment to guide the agent's learning process. The environment created uses the network traffic in the NSL-KDD dataset and was inspired by an environment proposed by Hari Koduvely [10]. The environment was custom-built to suit the developed RL approach.

4.1 Deep Reinforcement Learning Agent

A reinforcement learning agent is the brain where the algorithm resides. The agent's purpose is to select the appropriate action given a certain state to maximize the overall reward. This indicates that it is taking the best action available. Deep Q-learning is the chosen approach used in this research. Deep Q-learning uses two separate neural networks: the first one is called the main network or the policy network, and the second one is called the target network. The reason for using two separate neural networks is that when we feed a state to the neural network in order to obtain the appropriate action-Q-value pairs, the weights

of the network are updated. With that constant change, this mapping process will be impossible because we are chasing a dynamic target that contradicts the whole purpose of mapping states to action-Q-value pairs. Instead, we use a second neural network with initially the same weights as those of the main network. The target network is used to obtain the target Q-values as shown in Fig. 3, and then, the weights will be updated to match the main network ones every certain number of iterations to ensure that the learning process is functioning properly [20].

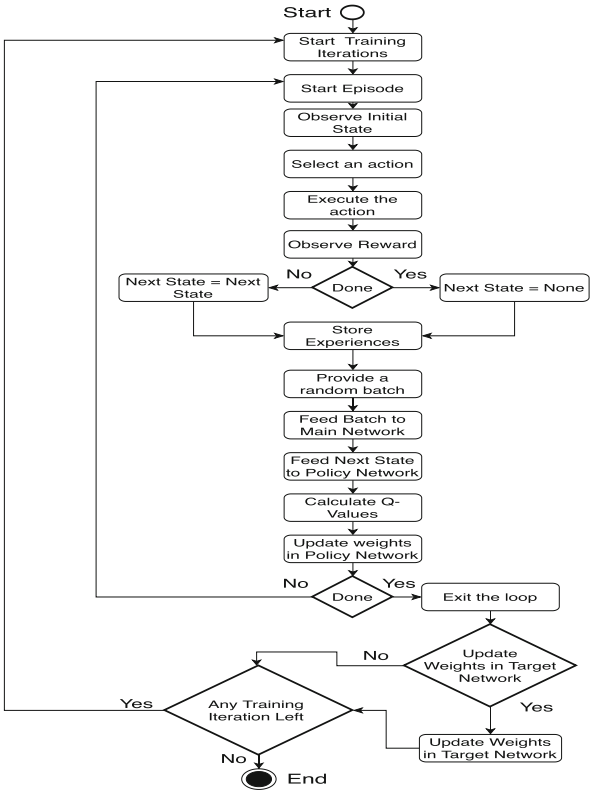

Fig. 3. Activity Chart of RL Agent

4.2 NSL-KDD Dataset

The NSL-KDD dataset, which is used by the Gym environment to provide observations to the agent, was obtained from the website of the Canadian Institute of Cyber Security, University of New Brunswick [21]. This dataset possesses the criteria to serve as a benchmark for today's modern intrusion detection systems [8]. Its training set contains 125,973 records of which 67,343 represent normal network traffic and 58,630 represent intrusions. Its testing set contains 22,544

records of which 9,711 represent normal network traffic and 12,833 represent intrusions [22]. The dataset contains 4 main attack categories (DoS, Probe, U2R, and R2L) and 39 different subcategories. Moreover, each record of the dataset represents 41 features of network traffic and a label.

4.3 Gym Environment

The environment is a very important element of the reinforcement learning process. Building an expressive environment is a necessity for the agent to be properly trained and perform the assigned task efficiently. In order to implement this environment concept, a specifically developed toolkit called OpenAI Gym toolkit is used.

The Gym environment structure is based on four different functions that are all in the same class: initialization, step, reset, and render. The environment will be referred to as an object called *env*. The initialization function is composed of several parameters. The two most relevant parameters are the action space, which is the available actions for the agent to take, and the observation space, which includes network traffic's features that will be observed by the agent. The step function will observe the action taken by the agent and return four different parameters: *next state*, *reward*, *done*, and *info*. These parameters represent the next state that the agent will observe, the current reward, a Boolean value indicating whether this episode is over or not, and some additional diagnostic information. The reset function provides a new random initial state. The render function outputs a graphical representation of the current situation in the environment. In this context, however, it is irrelevant because we do not have a graphical representation for the Gym environment. Moreover, all the possible actions, state labels, and rewards are provided in Table 1 to deepen the understanding of the previous concepts.

Table 1. State, Action and Reward as represented in the environment

State label	Action	Reward
Normal	0	1
Normal	1	–1
Intrusion	0	–1
Intrusion	1	1

First, the reset function is used to obtain the initial observation from the environment, which is then passed to the agent. Second, the agent will choose an action from the available ones. Third, the action chosen by the agent is passed to the environment step function. The step function will return the next observation, the reward for the agent's past action, a Boolean value indicating whether the training episode is over or not, and some diagnostic information.

This training process continues until it is terminated. Table 1 also explains the reward system given by the environment in correspondence with the action taken by the agent. When the agent observes a state provided by the environment, it responds by choosing an action from the available actions. The agent can either choose"0" or "1" which indicates whether this network traffic is a normal or an intrusion, respectively. Given this action, the environment judges the agent behavior by matching it with the label associated with each network traffic in the dataset. Thereupon, the environment assigns either a positive reward "1" or a negative reward "–1" in response to the correct or incorrect action of the agent, respectively.

5 Results

In this section, the evaluation approach, metric, and the results of the experiments will be discussed. The experiments were conducted using different number of training iterations, various neural network structures including different numbers of hidden layers and neurons, and multiple batch sizes. Additionally, we used 50% of the data in some experiments and 100% in others to prove that reinforcement learning can handle more attack representations and function properly with the increase in the number of attack types represented in the dataset, leading to the conclusion that RL can generalize. The reason for the variation proposed in the experiments is that obtaining an improved model is not an exact formula but rather based on a trial-and-error approach. At the end of the section, a general comparison with other reinforcement learning approaches implementing the NSL-KDD dataset from the literature is discussed in order to demonstrate performance comparison.

5.1 Evaluation Metrics

The most appropriate evaluation methods used for classification prediction models are confusion matrix, F1 score, accuracy, precision, and recall [23].

The confusion matrix [23] can be used for classification problems to show the actual classification (label) and the predicted one. Since all of the approaches in this research use binary classification, confusion matrix is considered the optimal option for evaluation. Although using confusion matrix is not suitable for reinforcement learning model evaluation, we can utilize the confusion matrix for our reinforcement learning approach since the data is labeled. Metrics were derived from the confusion matrix, which represent accuracy, precision, recall (sensitivity), and F1 score of the evaluated model in terms of True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN) predictions.

5.2 Reinforcement Learning Model Evaluation

Reinforcement learning model evaluation differs from other machine learning approaches because of its unique structure. In order to evaluate a reinforcement

learning model, we have two approaches: the performance of the policy estimated by the model and the learning curve that indicates the improvement of the agent's actions over time. The learning curve approach was followed in this paper. It can be identified as the accumulative reward as a function of the number of episodes. In other words, we are plotting the accumulative reward acquired by the RL agent over time and showing if it is increasing or decreasing. If the accumulative reward is increasing, it indicates that the RL agent's actions are improving during the learning process, which is the desired output.

5.3 Reinforcement Learning Experimental Results

In Fig. 4 below, the learning curve of the RL agent is plotted. It is showing a great improvement in the agent's performance over time, which validates the model. This means that the accumulative reward that the agent is receiving is increasing. This indicates that the agent's ability to choose the most appropriate action by maximizing its gain is improving over time.

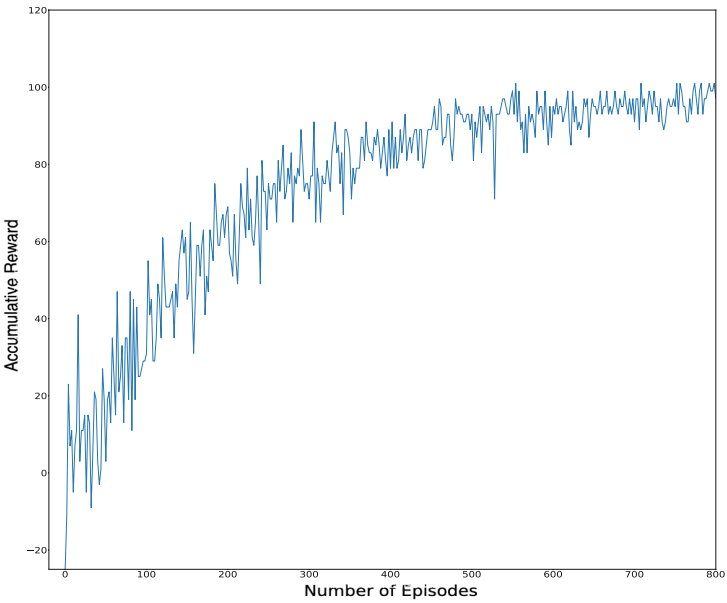

Fig. 4. RL agent learning curve

The results of the Deep Q reinforcement learning model are shown below. Although many experiments were conducted, only the top-performing ones are expressed. The other experiments were attempts to reach the optimal hyper-parameters. An experiment with only 50% of the dataset is performed to show the generalization property of RL when compared to the full dataset experiment.

Another experiment is shown using different hyper-parameters to prove that the RL model's hyper-parameters and neural network structure can affect its performance drastically.

We started by using 50% of the data with 400 training iterations, 100 steps per episode, and a batch size of 64. We used two hidden layers of size 50*10. The accuracy reached 86.80%. In the second conducted RL experiment, we used 100% of the data with 800 training iterations, 100 steps per episode, and a batch size of 64. We used two hidden layers of size 50*10 and the accuracy was 93.12%.

An aggregation of all the conducted experiments' results can be shown in Table 2 to give an overview.

Table 2. Aggregated results for the reported experiments

Experiment	**F1** (%)	**Recall** (%)	Precision (%)	Accuracy (%)
RL - Experiment 1	84.45	86.55	82.45	86.8
RL - Experiment 2	96.36	95.9	96.83	93.12

5.4 Performance Comparison with Relevant Work

Table 3 is an aggregation of the top-performing experimental results of the proposed RL agent alongside state-of-the-art results collected from related researches in the literature. The related researches implemented RL approaches such as Deep Q-Network (DQN) [12,14], and Double Deep Q-Network (DDQN) [15] alongside the same NSL-KDD dataset. The table also expresses the value added by our approach (OGE-DQN) in enhancing and improving the reinforcement learning agent's performance in intrusion detection as shown.

Table 3. Aggregated Results Comparison with Relevant Work

Approach	F1 (%)	Recall (%)	Precision (%)	Accuracy (%)
OGE-DQN (Developed in this research)	96.36	95.9	96.83	93.12
DQN (Manuel Lopez-Martin et al. [12])	89.35	89.37	89.33	87.87
DQN (Hsu and Matsuoka [14])	–	90.2	92.8	91.4
DDQN (Ma and Shi [15])	82.43	82.09	84.11	82.09

6 Conclusion

Due to the massive shift to network-based technologies, cloud computing-based services are becoming the ultimate replacement and solution for handling and

providing the infrastructure of computer systems, storage space, and computational power needed by agencies, companies, organizations, institutions, and governments. This transition introduced major concerns regarding the medium or network systems used to deliver these services and resources. Securing the network systems poses a challenge to customers and users of cloud computing services and other network-based technologies. The new attack types and continuous change in attack patterns introduced frequently pose a threat that makes it very difficult for traditional cyber-security and intrusion detection systems to keep up with those developments in attacks and the increasing scale of network systems.

In this research, we provided an answer to the intrusion problem in modern network systems by using deep reinforcement learning. We developed a novel approach for intrusion detection, and this approach had two key parts. The first part was the agent, which was built using Deep Q-learning algorithm and a deep neural network. The second part was the custom-built Gym environment that guided the learning process and provided network traffic from the dataset to the reinforcement learning agent. The dataset used to fuel the learning process was the NSL-KDD dataset, as it is used as a benchmark for most modern intrusion detection systems. The Gym environment presented is the first attempt to create a standardized Gym environment for intrusion detection using OpenAI Gym. Additionally, the custom-built Gym environment, which is our main contribution, contributed to the model's superiority by having expressive and inclusive features of intrusions. The superiority of our approach was validated and confirmed by comparing its results with other relevant work from the literature. The proposed Deep Q-learning solution achieved the highest performance with an accuracy of more than 93%, and it appears to be the most efficient solution among those compared. This high accuracy was achieved as a result of using an expressive Gym environment and a well-tuned model. Several experiments were carried out using several hyper-parameters and configurations by trial and error to obtain the optimal results. The experiments performed were reported and classified according to their configurations.

References

1. Cloud infrastructure market share. https://www.statista.com/chart/7994/cloud-market-share/ (2021)
2. Cloud infrastructure market share. https://www.statista.com/chart/18819/worldwide-market-share-of-leading-cloud-infrastructure-service-providers/ (2021)
3. Armin, J., Thompson, B., Ariu, D., Giacinto, G., Roli, F., Kijewski, P.: 2020 cybercrime economic costs: no measure no solution. In: 2015 10th International Conference on Availability, Reliability and Security, pp. 701–710. IEEE (2015)
4. The hidden costs of cybercrime. https://www.csis.org/analysis/hidden-costs-cybercrime (2021)
5. Scarfone, K.A., Mell, P.M.: Sp 800-94, guide to intrusion detection and prevention systems (IDPS). Tech. Rep., Gaithersburg, MD, USA (2007)
6. Roesch, M., et al.: Snort: Lightweight intrusion detection for networks. In: Lisa, vol. 99, pp. 229–238 (1999)

7. Brockman, G., et al.: OpenAI Gym. arXiv preprint arXiv:1606.01540 (2016)
8. Sharafaldin, I., Gharib, A., Lashkari, A.H., Ghorbani, A.A.: Towards a reliable intrusion detection benchmark dataset. Softw. Netw. **2018**(1), 177–200 (2018)
9. Liang, C., Shanmugam, B., Azam, S., Jonkman, M., De Boer, F., Narayansamy, G.: Intrusion detection system for internet of things based on a machine learning approach. In: 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), pp. 1–6. IEEE (2019)
10. Koduvely, H.: Anomaly detection through reinforcement learning (2018). https://doi.org/10.13140/RG.2.2.33673.29283
11. Caminero, G., Lopez-Martin, M., Carro, B.: Adversarial environment reinforcement learning algorithm for intrusion detection. Comput. Netw. **159**, 96–109 (2019)
12. Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A.: Application of deep reinforcement learning to intrusion detection for supervised problems. Expert Syst. Appl. **141**, 112963 (2020)
13. Suwannalai, E., Polprasert, C.: Network intrusion detection systems using adversarial reinforcement learning with deep Q-network. In: 2020 18th International Conference on ICT and Knowledge Engineering (ICT&KE), pp. 1–7. IEEE (2020)
14. Hsu, Y.F., Matsuoka, M.: A deep reinforcement learning approach for anomaly network intrusion detection system. In: 2020 IEEE 9th International Conference on Cloud Networking (CloudNet), pp. 1–6 (2020). https://doi.org/10.1109/CloudNet51028.2020.9335796
15. Ma, X., Shi, W.: AESMOTE: adversarial reinforcement learning with SMOTE for anomaly detection. IEEE Trans. Netw. Sci. Eng. 8, 943–956 (2020)
16. Hougen, D.F., Shah, S.N.H.: The evolution of reinforcement learning. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1457–1464. IEEE (2019)
17. Qiang, W., Zhongli, Z.: Reinforcement learning model, algorithms and its application. In: 2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC), pp. 1143–1146. IEEE (2011)
18. 5 things you need to know about reinforcement learning. https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html (2021)
19. Yildiz, B.: Reinforcement learning using fully connected, attention, and transformer models in knapsack problem solving. Concurrency and Computation: Practice and Experience (2021). https://doi.org/10.1002/cpe.6509
20. A hands-on introduction to deep Q-learning using openAI GYM in python. https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/ (2021)
21. NSL-KDD dataset. https://www.unb.ca/cic/datasets/nsl.html (2021)
22. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD cup 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pp. 1–6. IEEE (2009)
23. Vihinen, M.: How to evaluate performance of prediction methods? measures and their interpretation in variation effect analysis. In: BMC genomics, vol. 13, pp. 1–10. BioMed Central (2012)