Network Abnormal Traffic Detection Model Based on Semi-Supervised Deep Reinforcement Learning

Article *in* IEEE Transactions on Network and Service Management · October 2021

DOI: 10.1109/TNSM.2021.3120804

CITATIONS
36

READS
218

3 authors:

Shi Dong
Zhoukou Normal University
60 PUBLICATIONS **965** CITATIONS
SEE PROFILE

Xia Yuanjun
Guilin University of Electronic Technology
6 PUBLICATIONS **54** CITATIONS
SEE PROFILE

Peng Tao
Wuhan Textile University
69 PUBLICATIONS **266** CITATIONS
SEE PROFILE

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

Network Abnormal Traffic Detection Model Based on Semi-Supervised Deep Reinforcement Learning

Shi Dong, Yuanjun Xia, Tao Peng

***Abstract*—The rapid development of Internet technology has brought great convenience to our production life, and the ensuing security problems have become increasingly prominent. These problems threaten users' privacy and pose significant security risks to the normal conduct of many aspects of society, such as politics, economy, culture, and people's livelihood. The growth of the information transmission rate expands the scope of attacks and provides a more attack environment for intruders. Abnormal detection is an effective security protection technology that can monitor network transmission in real-time, effectively sense external attacks, and provide response decisions for relevant managers. The development of machine learning has also led to the development of abnormal traffic detection technology. The goal has been to use powerful and fast learning algorithms to deal with changing threats and respond in real-time. Most of the current abnormal detection research is based on simulation, using public and well-known datasets. On the one hand, the dataset contains high-dimensional massive data, which traditional machine learning methods cannot be processed. On the other hand, the labeled data scale is far behind the application requirements, and the dataset's labels are all manually labeled, so the labeling cost is exceptionally high. This paper proposes a semi-supervised Double Deep Q-Network (SSDDQN)-based optimization method for network abnormal traffic detection, mainly based on Double Deep Q-Network (DDQN), a representative of Deep Reinforcement Learning algorithm. In SSDDQN, the current network first adopts the autoencoder to reconstruct the traffic features and then uses a deep neural network as a classifier. The target network first uses the unsupervised learning algorithm K-Means clustering and then uses deep neural network prediction. The experiment uses NSL-KDD and AWID datasets for training and testing and performs a comprehensive comparison with existing machine learning models. The experimental results show that SSDDQN has certain advantages in time complexity and achieved good results in various evaluation metrics.**

***Keywords*—Abnormal traffic detection; semi-supervised learning; machine learning; deep reinforcement learning**

I. Introduction

Due to the rapid development of computers, the Internet has penetrated all corners of social life, affecting people's work and life in different ways. With the increase in the number of users, the Internet's sensitive information also increases, which brings opportunities for attackers and threatens the culture and economy of the society and the country [1]. Also, there are many attacks, cases of abuse and network failures, and other abnormal behaviors on today's Internet. Port viewing, worms, and denial of service attacks [2] are prevalent. These abnormal behaviors in network traffic often waste network resources, reduce network equipment and end hosts' performance, and even threaten network users' information security.

To effectively respond to security threats during the development of the Internet, effective network dynamic management of network traffic [3] is needed to understand network operation accurately, monitor network performance, ensure network quality of service [4] (QoS), and meet the needs of the majority of users. Abnormal traffic detection [5] mainly detects the system's behavior and sends a notification to the system when the detected behavior deviates from normal behavior. Judging whether the traffic data is normal is the primary measure, mainly through technical means to detect and judge and then find abnormal behavior. Network abnormal traffic detection is a vital defense technology that has an indispensable position in ensuring the network's smooth operation and network security. The current network abnormal traffic detection methods are mainly divided into four classes: payload feature-based detection techniques [6], flow feature-based detection methods [7], statistical-based detection methods [8], and machine learning (ML)-based detection methods.

The payload feature-based detection method mainly analyzes the payload features in the packets. It determines whether there is abnormal traffic by matching them with the known payload features of the abnormal traffic. Although this method is simple and efficient with high recognition accuracy, it depends mainly on the payload feature library, which cannot detect new types of abnormal traffic and requires timely updating of the payload feature library. This method can not detect encrypted traffic because some of the payload features are lost when the traffic is

This paper is supported by Project supported by Key Scientific and Technological Research Projects in Henan Province (Grand No. 192102210125), Open Foundation of State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) (KLNST-2020-2-01), Hubei Provincial Department of Education Youth Project (Q201316), Hubei Provincial Department of Education Research Program Key Project (D20191708). In addition, the authors also will thank the anonymous reviewers for their comments and suggestions.

Corresponding author: Shi Dong is with the School of Computer Science and Technology, Zhoukou Normal University, Zhoukou, Henan, China; the School of Computer and Artificial Intelligence, Wuhan Textile University, Wuhan, Hubei, China; the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunication, Beijing, China (e-mail: njbsok@ 163.com).

Yuanjun Xia is with the School of Computer and Artificial Intelligence, Wuhan Textile University, Wuhan, Hubei, China; the School of Computer Science and Technology, Zhoukou Normal University, Zhoukou, Henan, China (e-mail: xiaujun@163.com).

Corresponding author: Tao Peng is with the School of Computer and Artificial Intelligence, Wuhan Textile University, Wuhan, Hubei, China (e-mail: pt@wtu.edu.cn).

encrypted. Besides, the detection needs to analyze each packet's payload features, which is expensive to detect and difficult to cope with the current complex network environment. The flow feature-based detection method mainly distinguishes the normal traffic from the abnormal traffic through the flow statistics information at the transport or network layers. Although this method has a higher detection accuracy and can identify abnormal encrypted traffic, it can only detect known abnormal traffic. For unknown abnormal traffic, the complexity is high, and detection is difficult to achieve. The statistical-based abnormal traffic detection method mainly analyzes traffic data sampling through time series. It uses statistical techniques to describe the traffic features (mainly including data distribution, traffic variation, CPU usage, memory occupation, etc.) to detect abnormal traffic. This method can identify abnormal traffic without relying on a priori knowledge features, and it can also detect new types of abnormal traffic. Although this method has significant advantages in abnormal traffic detection, it can only detect the existence of abnormal traffic. Because it can not accurately determine the type of abnormal traffic, it has certain limitations.

Currently, the most common abnormal traffic detection method is the ML-based detection method. Although traditional ML [9] has achieved good results in abnormal traffic detection, it relies too much on manual extraction of traffic features. Human intervention is more serious, limiting its robustness and generalization ability. Since traditional ML models are shallowly structured, limiting their learning and classification capabilities. Compared to traditional ML, deep learning (DL) [10,11] can automatically learn feature representations from complex data without human intervention and can cope with complex encrypted traffic. Although DL has its unique advantages, the internal structure is more complex and the training and prediction time complexity are relatively high, making it difficult to perform real-time detection. Compared with the existing traditional ML and DL, the optimization process of deep reinforcement learning (DRL) [12,13] is implemented by the non-distinguished reward function, making it flexible in applying various complex types of problems. Currently, DRL has used intrusion detection, computer vision, natural language processing, and other fields, which belong to supervised learning. After the data is labeled, the model can be trained. The higher the labeled data's quality, the better the model learning ability. However, today's labeled data scale is far from keeping up with application requirements, and many labeling costs are required.

Considering the above problems, the paper proposes a semi-supervised Double Deep Q-Network (SSDDQN)-based network abnormal traffic detection optimization method, which bases on the representative algorithm Double Deep Q-Network (DDQN) [14] in DRL. This method is implemented by combining two unsupervised learning algorithms (Auto Encoder (AE) [15] and K-Means [16]). Currently, traditional DRL requires interaction with the real environment. The paper uses a simulated environment instead of the real environment to apply it to abnormal traffic detection: traffic features as states and actions as labels. Comparing the label when the maximum Q-value function obtains with the real label of the dataset if it is consistent, it rewards; if not, it not rewards. The state of the next stage is provided directly by the dataset, without interaction with the real environment.

The paper chooses the public and well-known NSL-KDD [17] and AWID [18] datasets to implement the experiments. NSL-KDD and AWID datasets have been extensively studied, contains a sufficient number of samples, and are highly unbalanced in size between classes. In addition, the number of attack types in the test set and training set in the NSL-KDD dataset is inconsistent, and there are attack types in the test set that do not exist in the training set. To evaluate the performance of SSDDQN, the paper compares it with a series of well-known machine learning: decision tree (DT) [19], support vector machine (SVM) [20], random forest (RF) [21], multi-layer perceptron (MLP) [22], one-dimensional convolutional neural network (CNN) [23], gate recurrent unit (GRU) [24], Deep Q-Network (DQN) [25], DDQN, and Dueling DQN [26]. These comparative methods cover traditional ML, representative DL, and DRL to highlight our approach's differences. Finally, the paper not only describes the influence of different discount factors in SSDDQN on the F1-Score metric but also conducts ablation experiments on SSDDQN to describe the contribution of each component of SSDDQN.

In general, the main contributions of our work are as follows：

(1) The use of fast and simple policy and Q-value functions in SSDDQN allows it to be used in high-demand network environments.

(2) The optimization process in SSDDQN is adopted by a non-distinguish reward function, making it flexible in applying many complex types of problems.

(3) The target network uses the K-Means clustering algorithm, which only needs to input the next stage's traffic features and predict its feature labels, significantly reducing the manual labeling cost and can detect unknown attacks.

(4) The network uses for training and testing all use a simple autoencoder and deep neural network combines network structure, which can perform fast training and prediction to achieve real-time detection purposes.

The structure of the paper, as shown below: Section II reviews the related work. Section III introduces datasets for the experiment, the data preprocessing method, and the SSDDQN model the paper proposes. Section IV makes a comprehensive analysis of the experimental results. Finally, the paper is summarized and prospected.

II. Related Work

At present, there is much literature on abnormal traffic detection using different datasets. We will first introduce and discuss various types of literature on machine learning used in abnormal traffic detection, then present representative literature on the application of NSL-KDD and AWID datasets. Finally, we will discuss related literature on the application of reinforcement learning in abnormal traffic detection.

ML applies to abnormal traffic detection: Lei et al. [27]

transformed the network abnormal traffic detection problem into a classification problem and proposed a hybrid PSO-SVM model. Kong et al. [28] proposed an abnormal traffic identification system (ATIS), and the SVM showed good performance by testing on the KDD CUP dataset. In [29], the authors proposed a new general equation for distance calculation and a PCA-based method for abnormal traffic detection, which was tested on the Kyoto 2006+ dataset, had low complexity, and could detect abnormal traffic quickly. In [30], the authors used a new deep learning method called Parallel Cross Convolutional Neural Network (PCCN) to improve unbalanced abnormal traffic detection results. Experimental results showed that this method could significantly reduce data detection time, which was superior to traditional ML algorithms and achieved better performance. Salman et al. [31] proposed a framework specifically for IoT device identification and abnormal traffic detection using different ML algorithms for comparison. Experimental results showed that RF achieved the best abnormal traffic detection results. In [32], a C-LSTM neural network model for Web abnormal traffic detection was proposed, mainly by combining CNN and long short-term memory network (LSTM). Gao et al. [33] designed a two-level abnormal traffic detection system based on DNN and correlation analysis (CA). DNN with different fully connected layers used for comparative experiments. Compared with representative ML methods, the four-layer fully connected hidden layer DNN obtained the best experimental results. In [34], the authors used LSTM to establish an abnormal traffic monitoring model. Sun et al. [35] used a hybrid network of CNN and LSTM to extract the Spatio-temporal features of network traffic. The experimental dataset used CICIDS2017, and the experimental results showed that the method had obtained better results. Khatouni et al. [36] studied ML-based network traffic classifiers as an effective means to classify encrypted multiple service channels accurately. In [37], the authors conducted a comprehensive analysis of the current ML technology to achieve traffic classification and hoped to outline the future direction of traffic classification based on ML. Shbair et al. [38] detailed the techniques used to monitor HTTPS traffic, from the most basic protocol identification to the finest identification of precise services. In addition, the author also described the actual solution.

The NSL-KDD dataset is a classical intrusion detection dataset. Currently, many researchers apply it to their research work: Safaldin et al. [39] proposed an intrusion detection system based on the SVM binary grey wolf optimizer. The experiment selected different numbers of features for training. In [40], the authors conducted a comprehensive analysis of the existing ML classifiers and analyzed feature selection, hyperparameter selection, and class imbalance issues. Tang et al. [41] proposed an intrusion detection model (SAAE-DNN) based on SAE, attention mechanism and DNN, and conducted binary and multi-classification experiments. Chaibi et al. [42] proposed two architectures for intrusion detection: the first method, the authors used artificial neural network (ANN) with information gain (IG) as feature selection methods; in the second method, RNN was applied to information gain (IG), grain ratio (GR) and related attributes (CA) as a feature selection method. In [43], a voting extreme learning machine (V-ELM) system was proposed, which selected 20000 training samples and 5000 test samples in NSL-KDD. In [44], the authors proposed a new IDS based on multimodal fusion to use Spark to protect medical data and introduced the idea of CBOA to provide more effective exploration. Jiang et al. [45] proposed a network intrusion detection algorithm that combines hybrid sampling and deep hierarchical networks. First, SMOTE was used to increase the number of samples, then CNN was used to extract spatial features, and BiLSTM was used to extract temporal features.

The AWID dataset is a relatively new intrusion detection dataset. Currently, many researchers also apply it to their research work: Kasongo et al. [46] proposed a wireless IDS system based on a feedforward deep neural network (FFDNN) based on Wrapper Based Feature Extraction Unit. In [47], the authors used Shapley Additive exPlanations (SHAP) with decision trees. Aminanto et al. [48] proposed a deep learning method based on a ladder network, which could effectively learn and classify abnormal attack traffic. In [49], the authors proposed a CNN-based WiFi network intrusion detection method.

Although DL has made substantial progress in network abnormal traffic detection, it overly depends on computer performance. Due to its complex internal structure, it is difficult to train and predict quickly, making it difficult to cope with dynamic and complex network environments. RL uses an adaptive learning strategy that enables detection in dynamic network traffic environments. DRL can learn high-dimensional data spaces due to the combination of DL and RL learning and is easier to deploy in real-time network environments. Therefore, many researchers apply reinforcement Learning (RL) or even DRL to abnormal traffic detection. The authors [50] used a Cooperative Fuzzy Q-Learning (Co-FQL) optimization algorithm to identify abnormal traffic to protect the network's wireless nodes and target nodes from distributed denial of service attacks. Servin et al. [51] proposed a new method of training multi-agent reinforcement learning (MARL), using the architecture of distributed sensors and decision-making agents to guide RL to identify the normal and abnormal state of the network. In [52], a new DRL-based wireless sensor network (WSN) and Internet of Things (IoT) intrusion detection system were proposed. In [53,54], the authors proposed adversarial reinforcement learning methods that enhanced the learning of a smaller number of abnormal traffic and improved overall accuracy, with both reaching 80% accuracy. In [55], several deep reinforcement learning algorithms (including DQN, DDQN, Policy Gradient (PG), and Actor-Critic (AC)) were applied to a supervised intrusion detection framework, and the experiments used NSL-KDD and AWID datasets to train the models, DDQN obtained the best results. Sethi et al. [56] proposed a context-adaptive IDS, which mainly used multiple independent deep reinforcement learning agents to detect and classify new and complex attacks.

Although DRL has achieved excellent network abnormal

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

traffic detection results, it is hugely dependent on network traffic feature labels. The higher the quality of the dataset labeling is, the more accurate the classification results are. The paper uses the NSL-KDD dataset, which has a severely unbalanced number of classes. In addition, there are attack types that do not exist in the training set in the test set, which brings particular challenges to the classification of the classifier. Inspired by the above work, we combine DRL with unsupervised learning algorithms to propose a semi-supervised DRL model, effectively reducing the manual label cost and improving network abnormal traffic detection accuracy.

III. Work Description

This Section presents our work in the following aspects: (1) Subsection *A* introduces the abnormal traffic detection experiment datasets. (2) In subsection *B*, a brief description of RL, DQN, and DDQN related theories. (3) In subsection *C*, the NSL-KDD and AWID datasets preprocessing methods are introduced. (4) The SSDDQN model is described in detail in Subsection *D*.

A. Dataset

The work considers datasets that satisfy the following aspects: (1) A public and well-known dataset with many researchers using the dataset for experiments. (2) A highly unbalanced dataset with varying degrees of imbalance is closer to the real network environment. (3) A dataset divided into training and test sets. (4) A dataset contains enough samples to facilitate training and testing.

Therefore, the paper chooses the NSL-KDD dataset as the main dataset for the experiment. In addition, the paper also used the newer AWID dataset to verify the reliability and effectiveness of the proposed method.

The NSL-KDD dataset is an improved version of KDD Cup99 [57], which produces more realistic results in the simulation experiment. There are many redundant records in the KDD Cup99 dataset, reaching 78% in the training sample and 75% in the test sample. Due to the higher number of redundant records, the model learns multiple times for repeated samples during training. The experimental results are biased, with higher accuracy generally biased towards more frequent records. To address these issues, researchers in the literature [17] removed redundant records from KDD Cup99 and created a more efficient and accurate NSL-KDD dataset.

The NSL-KDD dataset contains 125,973 training samples and 22,544 test samples, containing 41 features, as shown in TABLE I. There are 38 continuous features and three categorical variables (discrete values) in these 41 features. Besides, the attack type distribution of the dataset is severely unbalanced. There are 1 normal type and 22 attack types in the training set and 1 normal type and 37 attack types in the test sample. There are 21 common types in the training set and test set, 2 of them are unique to the training set, and 17 are unique to the test set. Because there are unknown attack types in the test set, it brings certain challenges to the learning of ML models. The attack types in the training set, the test set, and the unknown attack types in the test set are shown in TABLE II.

The NSL-KDD contains five classes of traffic, including one class of normal traffic and four classes of abnormal traffic, and the proportion of each traffic shows in Fig. 1. The four classes of abnormal traffic include Denial of Service Attack (DoS), Probe Attack, Remote to Local Attack (R2L), and User to Root Attack (U2R), as shown in TABLE III.

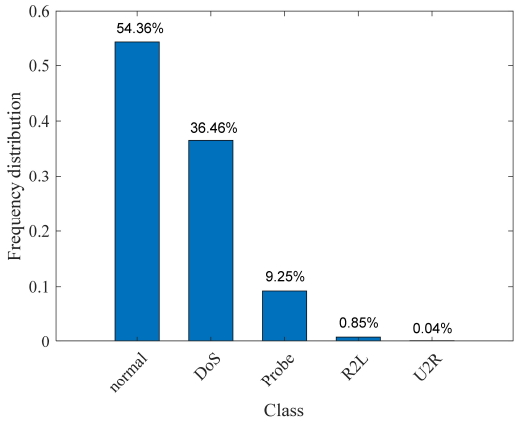

Fig. 1. Distribution frequency of each class of NSL-KDD

TABLE I
NSL-KDD Feature Variables

Type	Feature
Nominal	Protocol_type, Service, Flag
Binary	Land, logged_in, root_shell, su_attempted, is_host_login, is_guest_login
Continuous	Duration, src_bytes, dst_bytes, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate

TABLE II
NSL-KDD Attack Type

Set	Attack type
Training	back, land, neptune, pod, smurf, teardrop, ipsweep, nmap, portsweep, satan, ftp_write, guess_passed, imap, multihop, phf, spy, warezclient, warezmaster, buffer_overflow, loadmodule, perl, rootkit
Test	neptune, ipsweep, portsweep, teardrop, nmap, satan, smurf, pod, back, guess_passwd, ftp_write, multihop, rootkit, buffer_overflow, imap, warezmaster, phf, land, loadmodule, spy, perl, saint, mscan, apache2, snmpgetattack, processtable, httptunnel, ps, snmpguess, muilbomb, named, sendmail, xterm, worm, xlock, xsnoop, sqlattack, udpstorm
Unknown attack	saint, mscan, apache2, snmpgetattack, processtable, httptunnel, ps, snmpguess, muilbomb, named, sendmail, xterm, worm, xlock, xsnoop, sqlattack, udpstorm

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

TABLE III
CLASSES AND DESCRIPTIONS OF ABNORMAL TRAFFIC

Class	Specific attack form
DoS	back, udpstorm, processtable, land, pod, mailbomb, neptune, smurf, apache2, teadrop
Probe	nmap, satan, saint, ipsweep, port, mscan, sweep
R2L	phf, spy, imap, worm, xlock, named, sendmail, ftp_write, xsnoop, warezclient, guess_passwd, warezmaster, snmpguess, snmpgetattack, multihop
U2R	ps, perl, xterm, sqlattack, http tunnel, loadmodule, buffer_overflow

(1) DoS: The attacker occupies lots of system resources, destroying other users' regular access to resources such as the network and servers so that normal users cannot get services.
(2) Probe: Scan the target network or server to obtain information such as port number, IP address, and operating system vulnerability.
(3) R2L: The attacker can gain local access to the target host without the target user's account information.
(4) U2R: A regular local users on the target host have illegally gained super administrator privileges on the system.

The AWID dataset [18] comes from Kolias and is the largest and most comprehensive dataset collected in a real WiFi network environment. According to the attack class level, the dataset can be divided into two data subsets: the ATK dataset of 16 sub-attack classes and the CLS dataset of 4 large attack classes. Most researchers choose the AWID-CLS-R dataset, which includes 154 features, divided into continuous features and categorical features. In addition, the training and test sets include 1795474 and 675642 samples, respectively. Among them, the dataset includes one class of normal traffic and three classes of abnormal traffic. The distribution frequencies of samples for each class of traffic are shown in Fig. 2. Among them, the three classes of abnormal traffic are injection, impersonation, and flooding.

B. Models Description

In the Subsection, we present our work in the following aspects: (1) A brief introduction to the idea of the Markov Decision Making (MDP) model. (2) A brief introduction to the idea of RL. (3) An introduction to the idea of Q-Learning, a representative algorithm for RL. (4) A brief introduction to the idea of DQN and DDQN, representative algorithms in DRL, and their advantages.

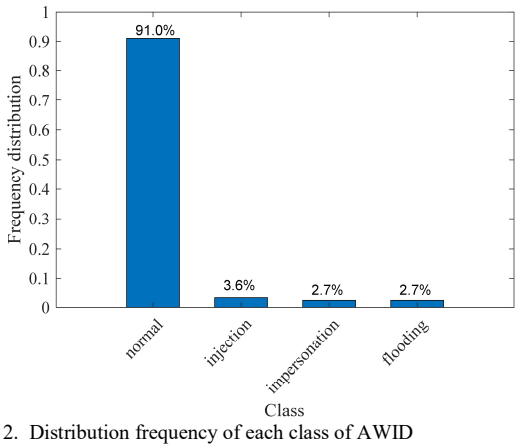

Fig. 2. Distribution frequency of each class of AWID

RL [58] and other ML algorithms' difference is that RL is a making a decision process to obtain the maximum reward or return by continuously learning the optimal strategy through the agent's interaction with the environment. The MDP model can describe the above interaction process. An MDP represented by a quintuple $M = (S, A, \varphi, T, R)$. Among, S represents the state set of the environment; A represents the action set that the agent can perform; φ represents the set of executable state-action pairs. T is a transfer function, which represents the probability of transitioning to state s' after performing action a in state s; R is a reward function, representing the expected reward value obtained when performing action a in state s and then transitioning to state s'.

The agent and the environment's specific interaction process are divided into two parts: (1) The agents perceive the current state of the external environment $s_t \in A$ at time t. It then decides to select the appropriate action $a_t \in A$ to execute according to the strategy π.（2）The environment responds to agent and updates the state s_t, generates a reward or return $r_t \in R$, and then enters the next MDP process, which shows in Fig. 3.

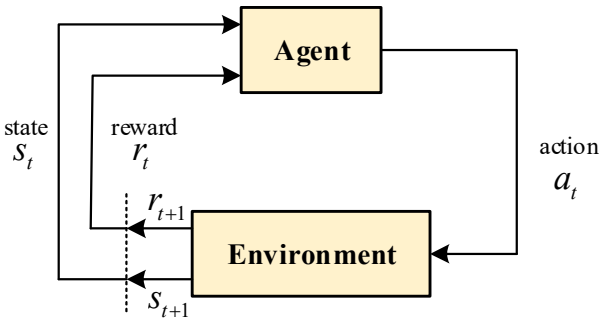

Fig. 3. The interaction model between the agent and the environment

The agent's primary goal is to strive to accumulate as many rewards as possible in the process of interacting with the environment, assuming that after time t, the agent interacts with the environment and continuously learns to maximize the expected return. The process is shown in formula (1).

$$R_t = r_{t+1} + \gamma . r_{t+2} + \gamma . r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k . r_{t+k+1} \quad (1)$$

Where γ is a discount factor and $\gamma \in [0,1]$, the discount factor determines the contribution of future rewards to the expected return. When $\gamma = 0$, the agent lacks foresight because it only considers maximizing the present rewards. The closer γ gets to 1, the more foresight the agent is because it focuses more and more on future rewards. $r_{t+1}, r_{t+2}, r_{t+3}, \cdots$ is the random variable reward sequence received by the agent after time t.

RL establishes the connection between the optimal criterion and the optimal policy by introducing a value function divided into a state-value function and a state-action-value function. The state-value function is the expected return obtains by an agent employing a policy π in state s. The equation is shown as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

$$V^{\pi}(s) = E_{\pi}\{R_t \mid s_t = s\} = E_{\pi}\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\} \quad (2)$$

Where $E_{\pi}\{\}$ denotes the expected value under the strategy π. There is an optimal state-value function when the optimal strategy is adopted among all these possible state-value functions. The expressions of the optimal strategy and the optimal state-value function are defined as below:

$$\pi^* = \arg\max_{\pi} V^{\pi}(s) \quad (3)$$

$$V^*(s) = \max_{\pi} V^{\pi}(s) \quad (4)$$

Similarly, the state-action-value function can define as the expected return of the agent performing action a in state s, and the equation is expressed as follows:

$$\begin{aligned} Q^{\pi}(s,a) &= E_{\pi}\{R_t \mid s_t = s, a_t = a\} \\ &= E_{\pi}\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a\} \end{aligned} \quad (5)$$

To obtain the optimal Q-value function, the expression of the optimal strategy is as follows:

$$\pi^* = \arg\max_{\pi} Q^{\pi}(s,a) \quad (6)$$

To further express the optimal Q-value function, the Bellman optimization equation is used to solve it further. The final expression is as follows:

$$\begin{aligned} Q^{\pi}(s,a) &= R + \gamma E_{s'}[V^*(s')] \\ &= R + \gamma \sum_{s' \in S} P(s' \mid s,a) V^*(s') \end{aligned} \quad (7)$$

Where R is the immediate expected reward obtained by the agent after transitioning to the state s' after performing action a in the state s, $E_{s'}[V^*(s')]$is the cumulative expected reward after transitioning from state s to s', γ is the discount factor.

Watkins et al. [59] proposed the Q-learning algorithm, which is also a breakthrough in the reinforcement Learning field. It is a model-free generalized strategy iteration algorithm based on offline strategy temporal difference (TD), and the algorithm mainly updates the Q-value function iterated, with the equation as follows:

$$\begin{aligned} Q(s_t,a_t) \leftarrow\ & Q(s_t,a_t) + \\ & \alpha[r_{t+1} + \gamma \max_{\pi} Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \end{aligned} \quad (8)$$

Where α is the learning rate and $r_{t+1} + \gamma \max_{a} Q(s_{t+1}, a_{t+1})$ is the update target. Besides, the core of the Q-Learning algorithm is the construction of states and actions into a table of Q-value functions, as shown in TABLE IV.

Although Q-Learning is a powerful learning algorithm based on value, it relies on a series of data in the sample when learning the optimal strategy and is greatly affected by the sample. Therefore, it is greatly affected by the training variance and even the convergence of the Q-value function. It can not estimate future state, change the environment's infinite state space, and not calculate certain Q-value functions. For TD algorithm, like Q-Learning, are very flexible for small RL learning problems. However, in the era of big data, the unusually complex states and optional actions make the table of Q-value functions to be maintained very largely, wasting many computer resources and even maintaining the table of Q-value functions beyond the memory size.

TABLE IV
Q-VALUE FUNCTION TABLE

Q Table	a_1	a_2	⋯	a_n
s_1	$Q(s_1,a_1)$	$Q(s_1,a_2)$	⋯	$Q(s_1,a_n)$
s_2	$Q(s_2,a_1)$	$Q(s_2,a_2)$	⋯	$Q(s_2,a_n)$
⋯	⋯	⋯	⋯	⋯
s_n	$Q(s_n,a_1)$	$Q(s_n,a_2)$	⋯	$Q(s_n,a_n)$

To solve the above problems, related researchers have combined neural networks with Q-Learning and proposed DRL methods represented by DQN and DDQN, which mainly approximate the Q-value function by using states and actions as the neural networks' inputs. Experience replay is also used to store the transfer data (current state, action, reward, and next stage state) and randomly sample the data for training. This storing sampling method breaks the correlation and non-smoothness between data and improves the algorithm's stability and efficiency. Although DQN has achieved excellent results in the artificial intelligence area, it does not address the inherent drawback of overestimating Q-Learning. The maximum Q-value function is predicted in the DQN algorithm for a given state and all possible action cases. Although this approach allows the Q-value function to converge toward the optimization goal quickly, it tends to cause overestimation and eventually bias the algorithm. To solve this problem, DDQN eliminates overestimation by decoupling the two steps of target Q-value action selection and target Q-value calculation. The maximum Q-value function is predicted mainly by the next stage's state and all possible actions using the current neural network, finding the maximum action under this Q-value function and predicting the target network's target Q-value function. Because of the constraint between the two, they can eliminate the maximum deviation.

C. Dataset Preprocessing

Using ML learning to train and test datasets, the common

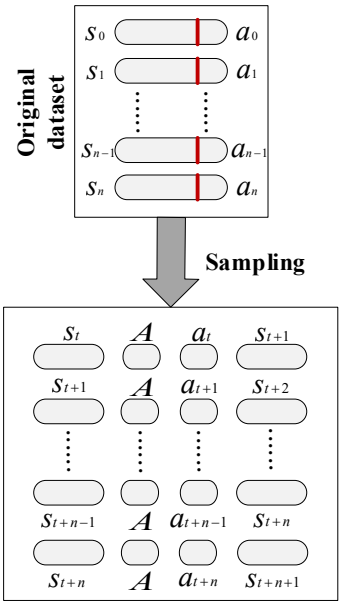

Fig. 4. Dataset preprocessing

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

feature types are numbers or Boolean values. Moreover, in the original NSL-KDD dataset, some features are strings or characters. Therefore, we need to convert all the features. For continuous variables, scale them to the range of [0-1]. For categorical variables, use one-hot encoding to convert them into dummy variables. After data transformation, the dataset has 122 features, containing 38 continuous variables and 84 binary variables. Among them, protocal_type has 3 features, service has 70 features, and the flag has 11 features.

In the AWID dataset, some features have null values, constant values, and network addresses. After discarding them, the 154-dimensional features are reduced to 49-dimensional. Because the value range of different features is quite different, in order to eliminate the influence of this difference on the model, continuous variables are also scaled to the interval [0-1]. In addition, for categorical variables, one-hot encoding is also used to convert them into dummy variables.

The current DRL input contains two parts: state and action. Furthermore, the dataset we use also includes two parts: network traffic features and feature labels. To assimilate the network traffic elements to the DRL concept, we consider the network traffic features as states and the feature labels as actions. As shown in the original dataset in the top half of Fig. 4, $S = \{s_0, s_1, \cdots, s_{n-1}, s_n\}$ is the network traffic feature and $A = \{a_0, a_1, \cdots, a_{n-1}, a_n\}$ is the feature label.

Reading the original dataset by random sampling and starting sampling from the time t. Each sample contains four parts: (1) Network traffic features s_t at the time t. (2) All traffic features label A. (3) Traffic real labels a_t at the time t. (4) Network traffic features s_{t+1} at the next time. As shown in the bottom half of Fig. 4, we divide the original data into $n+1$ random samples and perform random permutations before random sampling.

D. Proposed Method

In this Subsection, the details of the SSDDQN model are described.

SSDDQN model is mainly based on DDQN theory, a free-environment interaction model, where the state of the next stage is not generated by interaction with the environment but is directly provided by the dataset, as shown in Fig. 5.

During each training iteration, a new training sample is formed by sampling the original training sample, which contains the state s_t at time t, the set of all actions $A = \{a_0, a_1, \cdots, a_{n-1}, a_n\}$, the action a_t at time t and the state s_{t+1} at the next time. First, all possible Q-value functions are predicted by inputting the state s_t and the set of actions A in the neural network at the time t : $Q(s_t, A) = \{Q(s_t, a_0), (s_t, a_1), \cdots (s_t, a_{n-1}), (s_t, a_n)\}$. Using the policy algorithm π to filter the maximum Q-value function $\max_{\pi} Q(s_t, A)$ for the state s_t. Finally, the action of which is found by this maximum Q-value function: $a_t^* = arg_{a_t^*} \max_{\pi} Q^*(s_t, A)$. The $\varepsilon - greedy$ policy algorithm is further used in selecting the actions. In the $\varepsilon - greedy$ policy algorithm, the action a_t^* selects with probability ε, and other random actions select with probability $(1 - \varepsilon)$.

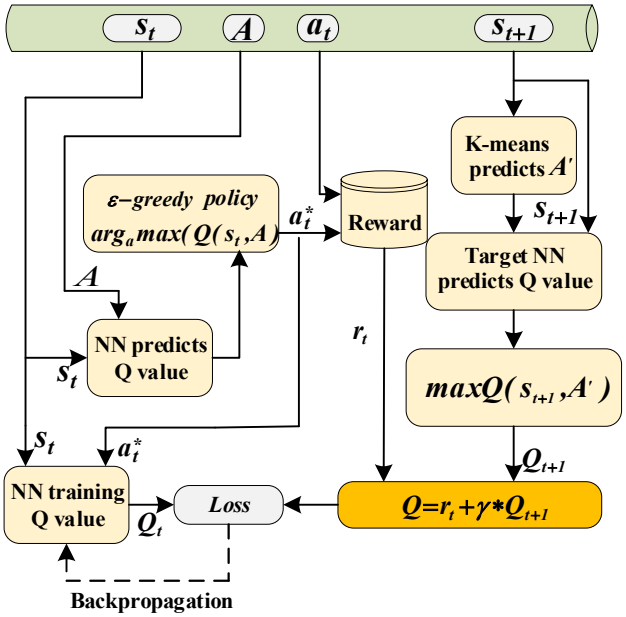

Fig. 5. Schematic diagram of SSDDQN model

TABLE V
SSDDQN ALGORITHM

Algorithm SSDDQN

Input: Exploration rate ε, discount factor γ, experience replay memory M, sampling batch k, dataset D, episode T.
1: Initialize $Q(s, a)$ arbitrarily.
2: **for** $i = 0,1, \cdots, T$ **do :**
3: Initialize the state value: $s = random\ sampling(D)$.
4: Input state s_t and all actions A in the current neural network to predict all Q value function: $Q(s_t, A)$.
5: Use $policy\ \pi$ to obtain the maximum Q value function in all Q value function.
6: Use the $\varepsilon - greedy$ strategy to select the action under the maximum the maximum Q value function: $a^* = arg_a \max_{\pi} Q(s_t, A)$.
7: Calculate the current Q function value: $Q_t = Q(s_t, a^*)$.
8: Get rewards by judging whether a and a^* are equal:r_t.
9: Store transition $M = [s_t, a, r_t, s_{t+1}]$.
10: Take the next stage state s_{t+1} from M, and use the **K-Means algorithm** to predict all possible labels A'.
11: Take samples from M to calculate the objective Q value function: $Q = r_t + \gamma * \max_{\pi} Q(s_{t+1}, A')$.
12: Use the gradient descent algorithm in Equation (9) to calculate the loss function.
13: Update neural network parameters through backpropagation algorithm.
14: **end for**

The reward function is composed of 1/0 rewards predicted to be correct/incorrect. The above-predicted action value a_t^* compared with the real action value a_t under state s_t and the reward or return is calculated. If the two are equal to get the reward, the reward is 1; if the two are not equal, the reward is 0.

The next stage aims to predict the Q-value function by the state s_{t+1} at the time $t+1$. Since the test samples in the NSL-KDD dataset contain unknown attack samples, to enhance the model's ability to detect unknown attacks, an unsupervised learning algorithm is used to predict its feature labels at the next moment. Therefore, the paper first uses the unsupervised learning K-Means algorithm to predict the action to which they belong. In the K-Means algorithm, the paper chooses the

number of clustering actions $k = 5$, representing the five traffic classes in NSL-KDD: normal, DoS, Probe, R2L, U2R. In the algorithm, only the state s_{t+1} at the time $t + 1$ needs input and sampled its sample as the prime vector μ_i with the same sample size k. The samples clusters by calculating the Euclidean Distance between the prime vector μ_i and the sample s_j: $d_{ij} = \|s_j - \mu_i\|_2$. The minimum distance of the sample s_j to the centroid vector μ_i predicts the action a'_j to which it belongs and finally outputs the action $A' = \{a'_1, a'_2 \cdots a'_n\}$ to which all samples belong. Finally, the maximum Q-value function at the time $t + 1$ predicts by the target neural network: $Q_{t+1} = \max Q(s_{t+1}, A')$.

Also, since the transfer samples (s_t, a_t, r_t, s_{t+1}) not independently distributes, the experience replay used to store the transfer samples in the above process. The model can randomly take out transfer samples through experience replay during training, which breaks the correlation between data and non-static distribution. Besides, experience replay allows the transferred samples to be reused, thereby improving efficiency.

Finally, the training of the neural network is carried out. The current Q-value function calculates by inputting the state s_t and the action a_t^* in the current neural network: $Q_t = Q(s_t, a_t^*)$. Moreover, calculate the target Q-value with the maximum Q-value function at the time $t + 1$ and the reward r_t: $Q = r_t + \gamma * Q_{t+1}$, and use the current Q-value function Q_t and the target Q-value function Q to calculate the Loss function. The Huber loss function is used in SSDDQN because the Huber loss function has the advantages of both mean square error (MSE) and squared absolute error (MAE) loss functions, as shown in formula (9).

$$L(Q_t - Q) = \begin{cases} \frac{1}{2}(Q_t - Q)^2 & |Q_t - Q| \le \delta, \\ \delta|Q_t - Q| - \frac{1}{2}\delta^2 & otherwize. \end{cases} \quad (9)$$

Where δ is a hyperparameter, and the value is 1. When $|Q_t - Q| \le \delta$, Huber loss tends to MSE, and vice versa tends to MAE.

TABLE V shows the algorithm of SSDDQN. The initial value of the greedy strategy $\varepsilon = 1$, but as the agent's learning gradually decreases, it will eventually converge to the lower limit of the classifier $\varepsilon = 0.01$. The discount factor $\gamma = 0.01$, because the agent values the current reward. Experience replay memory $M = 1000$, sampling batch $k = 100$, neural network training batch $T = 200$.

The current network in the SSDDQN model first uses an autoencoder (AE) to reconstruct the traffic features and learn the traffic data's hidden features. Then uses a deep neural network (DNN) to learn the traffic features and classify them, as shown in Figure 6. The target neural network uses only DNN, as shown on the right side of Fig 6. In the encoder part in the AE on the left side of Fig 6, the number of neurons in the first input layer is 122, the number of neurons in the second layer is 64, and the number of neurons in the third layer is 16, the decoder part is the opposite of the encoder part. It was demonstrated in the literature [33] that DNN with four hidden layers achieves optimal results. Therefore, four hidden layers use in the DNN on the right side of Fig 6, the number of hidden layer neurons is 100, and the last layer is the $softmax$ classification output layer. The final result is divided into two or five classes, corresponding to two classes of traffic: normal and abnormal; or five classes of traffic: normal, DoS, Probe, R2L, and U2R.

IV. Experiments and Results

The Section presents the paper's work in the following aspects: (1) Subsection *A* introduces our experimental environment. (2) Subsection *B* introduces the evaluation metric of the experiments. (3) Subsection *C* shows the results of comparing the SSDDQN model with other ML models.

A. Experiment Environment

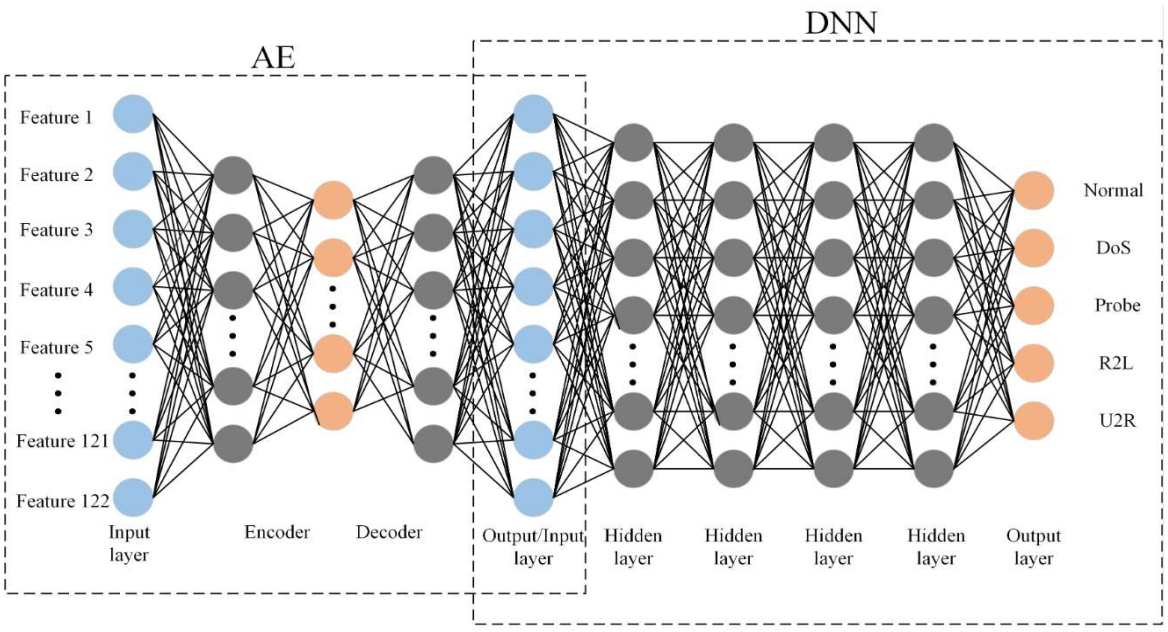

Fig. 6. Current neural network structure

The experiments were deployed on a PC platform with Windows 10, Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz processor, 16GBRAM, and all languages used python. The scikit-learn library was used in the experiment, and the Tensorflow library and the backend Keras were also used.

B. Evaluation Metric

A general ML-based method uses four metrics to evaluate its performance: accuracy, precision, detection rate (DR, DR also calls recall), and F1-score. The accuracy is used to evaluate the method's overall effectiveness. The precision and DR are used to evaluate each class's recognition efficiency. F1-Score considers both precision and recall and is the weighted average of precision and recall rate as a comprehensive overall metric. Therefore, the F1-Score metric is more representative when evaluating the model. FPR is used to evaluate the metric that is predicted to be wrong in the positive class. In addition, the paper also introduces an abnormal detection model efficiency metric for model efficiency. The calculation equations for each evaluation metric are as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

$$DR = Recall = \frac{TP}{TP + FN} \tag{12}$$

$$FPR = \frac{FP}{FP + TN} \tag{13}$$

$$Efficiency = \frac{TP + TN - FP - FN}{TP + TN + FP + FN} \tag{14}$$

True Positive (TP) refers to the number of correctly labeled samples in the target samples. False Positive examples (FP) refer to the number of wrongly labeled samples in non-target samples. Real Negative (TN) refers to the number of samples in the target sample that are mislabeled as other classes. False Negative (FN) refers to the number of non-samples labeled in non-target samples.

C. Results

The Subsection uses the proposed SSDDQN model for a comprehensive comparison with different machine learning models, including current representative traditional ML, DL, and DRL. Among them, traditional ML selects decision tree (DT), support vector machine (SVM), and random forest (RF). DL selects multi-layer perceptron (MLP), one-dimensional convolutional neural network (1D-CNN), and gate recurrent unit (GRU). DRL selects DQN, DDQN, and Dueling DQN. DT, SVM, and RF adopt default parameters, and SVM defaults to RBF kernel. MLP uses one hidden layer, and the number of nodes is 100. 1D-CNN uses three convolutional layers (the number of convolution kernels are 32, 64, and 64, respectively), two pooling layers (step size is 3), one fully connected layer (the number of nodes is 1024), and one layer $softmax$ output layer. Three GRU layers (the number of hidden nodes are 32, 32, and 64, respectively) are used in the GRU, one layer softmax output layer. In 1D-CNN and GRU, the optimizer uses $adam$, the loss function uses $categorical\_crossentropy$, and the activation function uses Relu. Both DQN and DDQN adopt the DNN structure in Figure 6, and Dueling DQN adopts a three-layer convolutional layer, and the learning rate and discount factor are both 0.01.

The paper first uses the entire NSL-KDD original dataset for experiments. The unbalanced number of attack types in the training and test set of the NSL-KDD dataset brings certain challenges to model training and testing. We perform binary classification and five classification experiments on the dataset. The binary classification includes: normal and abnormal, and the five classification includes: normal, DoS, Probe, R2L, U2R.

The binary classification results show in TABLE VI. The experimental results show that SSDDQN outperforms the comparison model in all metrics except for FPR, which is slightly lower than GRU. SSDDQN has about 5% improvement in accuracy, 6% improvement in F1-Score, and 5% improvement in efficiency compared to DDQN. Due to the particularity of the NSL-KDD dataset, in the target network of SSDDQN, the next stage of feature labels is no longer used, but the unsupervised learning algorithm K-Means is used to cluster traffic features, which are divided into normal and abnormal, and output predicted label. Finally, the maximum Q-value function is obtained by predicting the label, the target Q-value function is obtained by obtaining the maximum Q-value function, and the loss function is calculated. When using the backpropagation algorithm to update the neural network's parameters, the neural network has a stronger ability to learn unknown attacks, thereby improving the prediction performance of unknown attacks.

TABLE VII compares the SSDDQN model's comparison results and the current representative references using the NSL-KDD dataset for binary classification experiments, analyzing and comparing the experimental results of the literature, the more the number of features used, the lower the accuracy rate. Reference [42] used three feature selection algorithms, selects 29 features from the dataset, and used recurrent neural network (RNN) for training and testing, so each metric is the best. Followed by the literature [43], the author used feature dimensionality reduction technology, using about 16% of the features, the experimental results reached 99%. In other comparative literature, different feature selection techniques are used to keep the attack types in the training set and test set as consistent as possible to obtain higher detection performance. The paper uses the entire NSL-KDD dataset for experiments without feature selection, and the experimental accuracy reaches 83% and outperforms the [39], [40], and [56], thus proving that the SSDDQN model can more effectively deal with the network environment with a variety of attack types.

The five classification experiments are shown in TABLE VIII. In the evaluation metric in Section B, the precision metric is also vital. This metric considers as many abnormal intrusions as possible, and it is vital to identify them accurately. Analyzing the experimental results, the SSDDQN model outperforms other ML models in all metrics, except for the precision metric, which is slightly lower than MLP. DR metric is also crucial to ensure a minimum number of false negatives in the case of as many invasions as possible. The DR in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

TABLE VI
BINARY CLASSIFICATION PERFORMANCE EVALUATION ON NSL-KDD DATASET

Metric \ Model	DT	SVM	RF	MLP	1D-CNN	GRU	DQN	DDQN	Dueling DQN	SSDDQN
Accuracy	0.7877	0.7627	0.7687	0.7728	0.7968	0.7918	0.7490	0.7802	0.8076	0.8310
Precision	0.9193	0.9202	0.9678	0.9203	0.9175	0.9765	0.9037	0.9236	0.9687	0.9762
DR	0.6873	0.6385	0.6140	0.6578	0.7065	0.6499	0.6257	0.6692	0.6847	0.7231
F1-Score	0.7866	0.7539	0.7514	0.7672	0.7983	0.7804	0.7394	0.7761	0.8020	0.8308
FPR	0.0797	0.0732	0.0270	0.0753	0.0839	0.0207	0.0881	0.0731	0.0301	0.0266
Efficiency	0.5753	0.5254	0.5373	0.5455	0.5936	0.5836	0.4980	0.5604	0.6151	0.6619

TABLE VII
BINARY CLASSIFICATION PERFORMANCE COMPARISON ON NSL-KDD DATASET
(------ MEANS THAT THE VALUE IS NOT PROVIDED IN THE LITERATURE)

Approach	Year	Feature selection	Feature number	Accuracy	FPR	Precision	DR
Proposed method	2021	No	41	0.8301	0.0266	0.9762	0.7231
GWOSVM [39]	2021	Modified grey wolf optimization	27%	0.7900	0.2400	------	0.8300
Optimal multimodal fusion framework [44]	2021	CBOA-FS	21	0.9921	------	0.9893	0.9959
V-ELM [43]	2020	Feature reduction	16%	0.9918	------	------	0.9950
ANN+RNN [42]	2020	IG+RG+CA	29	0.9960	0.0020	0.9900	0.9900
J48 [40]	2020	InfoGainAttributeEval	14	0.8267	0.1570	0.8370	0.8070
SAAE-DNN [41]	2020	Statistical Filtering	21	0.8774	------	0.8647	0.8418
RL-based context-aware robust [56]	2020	Chi-square	36	0.8180	0.0026	------	------

TABLE VIII
FIVE CLASSIFICATION PERFORMANCE EVALUATION ON NSL-KDD DATASET

Metric \ Model	DT	SVM	RF	MLP	1D-CNN	GRU	DQN	DDQN	Dueling DQN	SSDDQN
Accuracy	0.7490	0.7707	0.7473	0.7844	0.7744	0.7539	0.6855	0.7343	0.7771	0.7943
Precision	0.7566	0.8245	0.8053	0.7773	0.7675	0.7747	0.6633	0.6661	0.8165	0.8281
DR	0.7490	0.7707	0.7473	0.7544	0.7744	0.7739	0.6855	0.7343	0.7771	0.7943
F1-Score	0.7190	0.7366	0.7040	0.7232	0.7312	0.7165	0.6536	0.6902	0.7429	0.7622
FPR	0.1473	0.1620	0.1810	0.1597	0.1561	0.1591	0.1887	0.1720	0.1563	0.1402
Efficiency	0.7231	0.7256	0.6964	0.7107	0.7347	0.7130	0.6392	0.6914	0.7382	0.7571

TABLE IX
MULTIPLE ATTACK TYPES CLASSIFICATION PERFORMANCE EVALUATION ON NSL-KDD DATASET

Attack type	Sample size	Number of correct classifications	Accuracy	Total number
saint	4	4	0.0125	319
mscan	1036	985	0.9890	996
apache2	791	735	0.9973	737
snmpgetattack	58	4	0.0220	178
processtable	701	652	0.9518	685
httptunnel	33	0	0	133
snmpguess	10	6	0.0181	331
mailbomb	233	214	0.7303	293

SSDDQN reaches 79.43%, which is higher than DDQN, Dueling DQN, and 1D-CNN by about 6%, 2%, and 2%, respectively. The F1-Score metric is often considered a better predictive performance metric for multi-classification and highly class-unbalanced datasets. In terms of the F1-Score metric, the SSDDQN model achieves better results, improving about 7% in the F1-Score metric than the DDQN model. Efficiency metric better reflects the model's efficiency, and SSDDQN improves about 6% compared to DDQN, thus showing higher efficiency. In the five classifications, due to the low number of Probe, R2L, and U2R, when the K-Means algorithm is used to predict its label, it is easy to predict as normal and DoS. In the model training process, the prediction error of the target Q-value function is increased, thereby

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

increasing the training error of the current network. Therefore, the performance of the five classification is lower than that of the binary classification, and the FPR of the five classification is higher than that of the binary classification.

In addition, in order to verify the ability of the SSDDQN model to detect unknown attacks, we implement classification experiments for the multiple attack types. Since there are 9 unknown attack types with a sample size of less than 20, it is difficult to sample their samples when using small-batch random sampling methods, and it is almost hard to detect them. Therefore, we only show the detection results of the other 8 unknown attack types with a large number of samples in Table IX. Analyzing the experimental results, the number of samples of the mscan, apache2, processtable, and mailbomb attack types is relatively large, so the number of samples drawn is relatively large, and the detection accuracy rate is relatively high. The number of samples of Httptunnel attack types is 33, but it cannot be detected. Although the sample number of the remaining 3 attack types is low, they all have a certain detection ability. In the references compared in TABLE VII, the authors used different feature selection algorithms to remove high-relevance features and reduce the number of unknown attack types in the test set to improve the detection accuracy of abnormal attack traffic. 36 features were selected in the literature [56], which is slightly lower than the total number of features. Combining accuracy and FPR evaluation metric, the model in literature [56] is superior to our method in detecting normal traffic but slightly lower than our method in detecting abnormal traffic. Therefore, the SSDDQN model proposed in the paper can more effectively deal with the network environment with unknown attacks.

Fig. 7 and Fig. 8 provide the training and prediction time of all models on the NSL-KDD dataset. For traditional ML, the internal structure of DT and RF is simple, so shorter training and prediction time is obtained. The SVM uses an RBF nonlinear kernel function and has the highest overall prediction time. Both DQN and DDQN use deep neural networks with four hidden layers and have relatively low training and prediction times. In contrast, 1D-CNN and GRU are relatively complex in network structure, so they have relatively high training and prediction times. The clustering algorithm is added to SSDDQN, and the training time is improved compared to DQN and DDQN, but slightly lower than 1D-CNN and Dueling DQN. The SSDDQN prediction time is 0.49s, which is relatively low overall. Therefore, the SSDDQN model can meet the requirement of real-time detection and can be deployed in harsh network environments.

Fig. 9 evaluates the SSDDQN model's performance from three aspects: Correct estimated, False negative, and False positive. It can see from the figure that the SSDDQN model performs well in DoS and Probe abnormal attack traffic detection. Since the R2L and U2R abnormal attack traffic in the NSL-KDD original dataset accounted for 0.85% and 0.04% of the entire sample, respectively, the number is minimal, so the feature learning of this part of the abnormal attack sample is low, and it is easy to produce false negatives.

Fig. 10 and 11 show the confusion matrix of the DDQN model and proposed SSDDQN model, where the y-axis is the real label, and the x-axis is the predicted label. The figure can observe that 97% of the normal traffic in SSDDQN is correctly classified, which is better than DDQN. In DDQN, all R2L abnormal attacks were misclassified, 86% of which were predicted as normal and 13% predicted as Probe. In SSDDQN, 14% of R2L are correctly classified, 83% predicted as normal, and 5% predicted as U2R. For the lower number of U2R abnormal attacks, they are all misclassified in DDQN and SSDDQN. Also, 75% of DoS abnormal attacks in DDQN are correctly classified, and 64% of Probe abnormal attacks are correctly classified. In SSDDQN, 82% of DoS and Probe abnormal attacks were correctly classified. SSDDQN model's performance is better than DDQN in detecting normal traffic

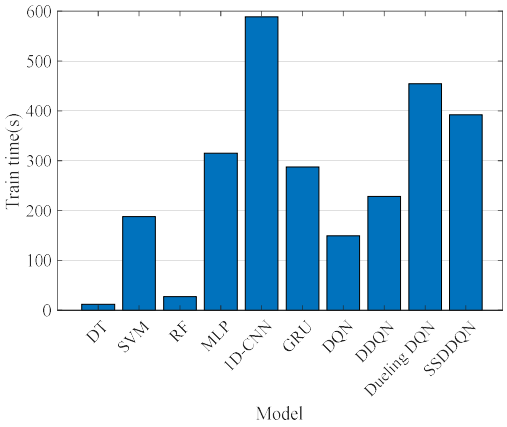

Fig. 7. Training time of all models

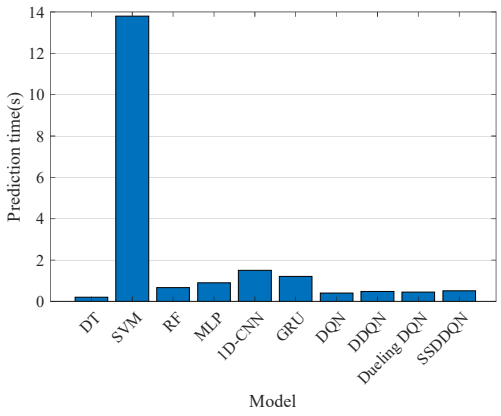

Fig. 8. Prediction time of all models

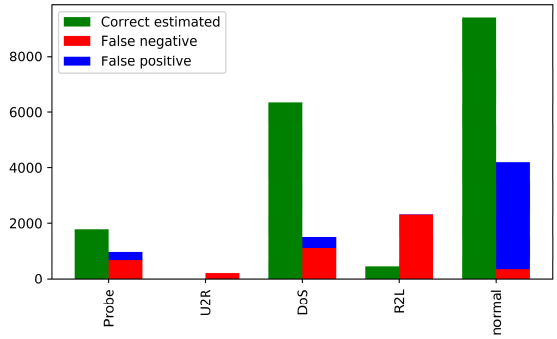

Fig. 9. SSDDQN model performance

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

from the above data. It shows excellent performance in the detection of DoS and Probe abnormal traffic. Due to the relatively low number of R2L and U2R traffic samples, especially U2R abnormal traffic, which accounts for 0.04% of the entire traffic sample, the unsupervised learning algorithm can easily cluster into the largest number of normal traffic, and errors occur when predicting the target Q-value function. Moreover, this part of the traffic has fewer features, and the model cannot thoroughly learn its features, so the detection effect is low.

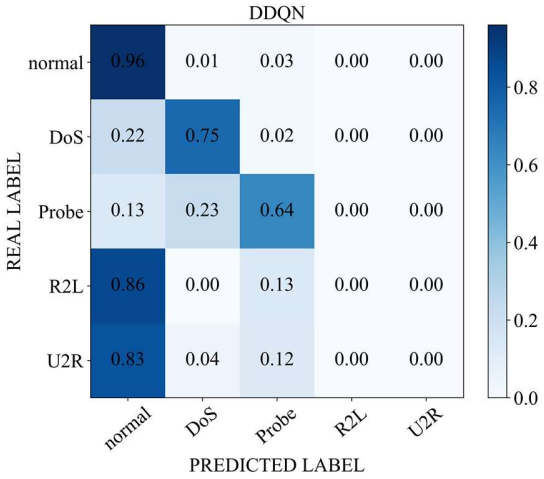

Fig. 10. Confusion matrix of the DDQN model

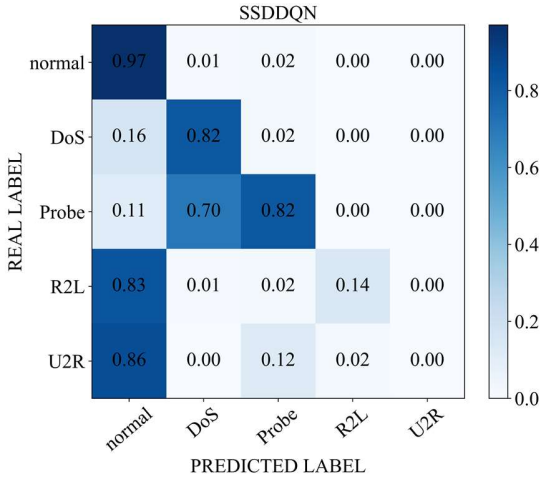

Fig. 11. Confusion matrix of the SSDDQN model

In addition, we further validate the performance of the SSDDQN model using the newer AWID dataset, and TABLE X provides the results of the AWID dataset binary classification experiments. Analyzing the experimental results, SSDDQN has the highest accuracy, F1-Score, and efficiency. RF has the highest precision, which metrics that RF can detect the normal behavior more effectively, and therefore the FPR of RF reaches lower values of 0.02%. SVM has the highest DR value, indicating that SVM can guarantee fewer false negatives. Compared with the NSL-KDD dataset, the AWID dataset has no unknown attacks, and the dataset is extremely unbalanced, with abnormal traffic accounting for only 9% of the entire dataset. Experimental results show that SSDDQN can perform excellently in an extremely unbalanced dataset without unknown attacks.

The experimental results of four classification for the AWID dataset are provided in TABLE XI. It can be seen that SSDDQN has also achieved good results on four classification. DDQN can also be effectively detected on the AWID dataset and is higher than the other comparison models, except for being lower than SSDDQN. The F1-Score of SSDDQN reaches 98.22%, and the model efficiency reaches 96.76%, both higher than other comparable models, indicating that SSDDQN is more effective in dealing with multi-classification traffic data and an imbalanced number of classes. SSDDQN has the lowest FPR, so it has a strong ability to identify each class accurately. Since the current neural network in SSDDQN uses AE to reconstruct the current traffic features, the DNN can learn the traffic features more effectively. Therefore, SSSDDQN also has certain advantages in four classification experiments on the newer AWID dataset.

The four classification confusion matrices for DDQN and SSDDQN on the AWID dataset are provided in Fig. 12 and 13. It can be seen that both DDQN and SSDDQN perform well in abnormal attack traffic detection. In SSDDQN, 99% of the normal traffic was correctly classified, higher than 97% of DDQN. 99% of the abnormal impersonation traffic was correctly classified, higher than the 93% of DDQN. Overall,

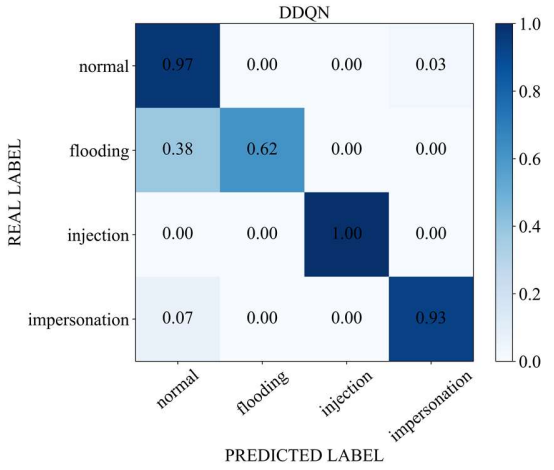

Fig. 12. Confusion matrix of the DDQN model

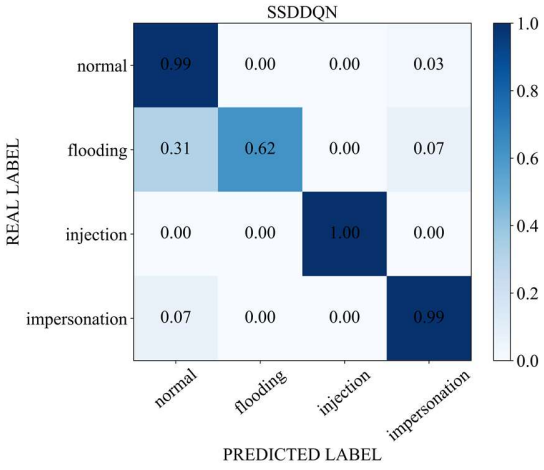

Fig. 13. Confusion matrix of the SSDDQN model

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

SSDDQN still maintains good performance on the AWID dataset without unknown attacks and with extreme imbalance. In addition, SSDDQN can effectively detect a relatively low number of abnormal attack traffic of impersonation and flooding.

Fig. 14 shows that the best F1-Score value is obtained for a discount factor of 0.01, after which it slowly decreases. The F1-Score value picks up at a discount factor of 0.7 and finally decreases gradually again, with the lowest F1-Score value at a discount factor of 0.99. It suggests that the agents can get better detection results if they are more focused on the current reward.

In addition, the ablation experiments of SSDDQN were conducted in the paper, including DNN, AE, AE-DNN, SSDDQN with AE, and SSDDQN with DNN, as shown in TABLE XII. Among them, the network structure of AE and DNN is consistent with Fig 6. The DL model training batches are both 5. The DRL model training batches are 200, the learning rates are 0.01, and the discount factors are 0.01. It can be seen from Table XII that the difference between the results of DNN and AE-DNN is tiny, and SSDDQN with DNN achieves a better value, which is slightly higher than SSDDQN with AE. Overall, SSDDQN can achieve good results using a simple DNN, and the model robustness is further improved by using AE to reconstruct the traffic features. Compared with DDQN, SSDDQN with DNN only adds a K-Means algorithm, which is better than DDQN in all evaluation metrics. Incorporating the K-Means algorithm in DDQN copes with unknown attacks and shows excellent performance in extremely unbalanced traffic data. In general, SSDDQN can be applied to the current complex and dynamic network environment.

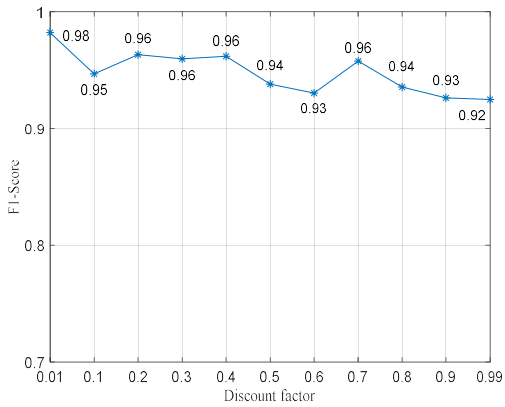

Fig. 14. F1-Score metric changes with the line graph of the discount

Fig. 15, 16, and 17 provide the confusion matrixes for AE-DNN, SSDDQN with AE, and SSDDQN with DDQN, respectively. In AE-DNN, abnormal impersonation traffic has no detection capability. In SSDDQN with AE, 97% of the normal traffic is correctly classified, and 99% of the abnormal impersonation traffic is correctly classified. In SSDDQN with DNN, 97% of the normal traffic is correctly classified, and 93% of the abnormal impersonation traffic is correctly classified. It can be seen that adding AE to SSDDQN with DNN can improve the detection of normal traffic and abnormal impersonation traffic.

V. Conclusion

The complexity and dynamics of the current network bring some challenges to network traffic monitoring, and abnormal traffic detection is a vital detection technology that can effectively monitor network performance and improve network service quality. Although the current ML methods have improved accuracy and reduced false positives, they are all based on supervise learning algorithms and cost much manual labeling. Currently, DRL has achieved good results in games, robotics, manless driving, computer vision, and natural

TABLE X
Binary Classification Performance Evaluation on AWID Dataset

Metric \ Model	DT	SVM	RF	MLP	1D-CNN	GRU	DDQN	SSDDQN
Accuracy	0.9483	0.9469	0.9506	0.9343	0.9525	0.9470	0.9651	0.9899
Precision	0.8083	0.6259	0.9950	0.5902	0.9420	0.8108	0.9213	0.9699
DR	0.4412	0.7939	0.3688	0.5130	0.4161	0.4173	0.7140	0.8978
F1-Score	0.6708	0.6999	0.5381	0.5489	0.5772	0.5510	0.8045	0.9325
FPR	0.0088	0.0401	0.0002	0.0301	0.0022	0.0082	0.0312	0.0023
Efficiency	0.8966	0.8939	0.9013	0.8686	0.9050	0.8940	0.9302	0.9797

TABLE XI
Four Classification Performance Evaluation on AWID Dataset

Metric \ Model	DT	SVM	RF	MLP	1D-CNN	GRU	DDQN	SSDDQN
Accuracy	0.9253	0.9564	0.9436	0.9344	0.9537	0.9527	0.9647	0.9819
Precision	0.9422	0.9593	0.9466	0.9298	0.9424	0.9365	0.9740	0.9840
DR	0.9253	0.9564	0.9436	0.9344	0.9537	0.9527	0.9647	0.9819
F1-Score	0.9284	0.9431	0.9210	0.9316	0.9357	0.9367	0.9673	0.9822
FPR	0.2183	0.3307	0.6668	0.4477	0.5305	0.5141	0.0948	0.0557
Efficiency	0.8837	0.9426	0.8928	0.8748	0.9119	0.9105	0.9327	0.9676

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3120804, IEEE Transactions on Network and Service Management

TABLE XII
FOUR CLASSIFICATION ABLATION EXPERIMENTS ON AWID DATASET

Metric \ Model	DNN	AE	AE-DNN	DDQN	SSDDQN with AE	SSDDQN with DNN	SSDDQN
Accuracy	0.9349	0.9296	0.9341	0.9647	0.9537	0.9734	0.9819
Precision	0.9293	0.9251	0.9409	0.9740	0.9424	0.9775	0.9840
DR	0.9349	0.9296	0.9341	0.9344	0.9647	0.9734	0.9819
F1-Score	0.9315	0.9250	0.9341	0.9316	0.9673	0.9742	0.9822
FPR	0.4536	0.4962	0.0960	0.4477	0.0948	0.0944	0.0557
Efficiency	0.8756	0.8673	0.9317	0.8748	0.9327	0.9493	0.9676

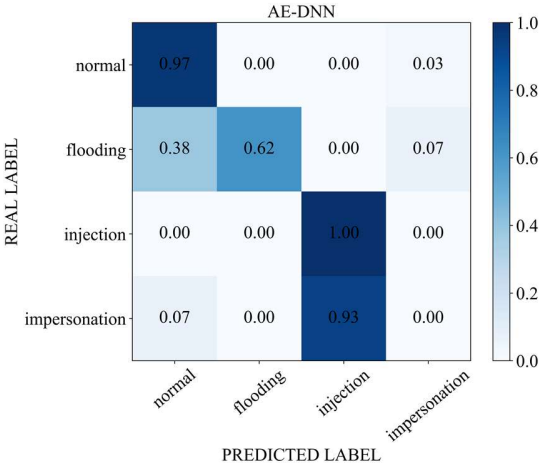

Fig. 15. Confusion matrix of the AE-DNN model

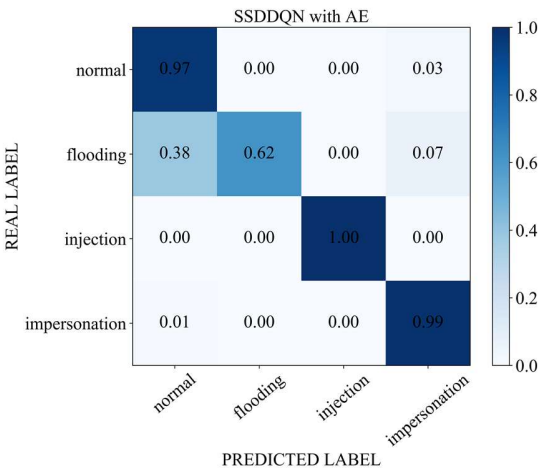

Fig. 16. Confusion matrix of the SSDDQN with AE model

language processing, but relatively few abnormal traffic detection works.

The paper proposes a semi-supervised learning model SSDDQN based on DDQN. In the SSDDQN model's current network, the unsupervised learning algorithm AE is used to reconstruct the traffic features, and then the DNN is used for predictive classification. In the target network, we only need to input the next stage's traffic features without inputting feature labels, which significantly reduces the cost of manual labeling. The unsupervised learning algorithm uses K-Means to cluster traffic features, output cluster labels, and calculate the Q-value function through the target DNN. The paper compares

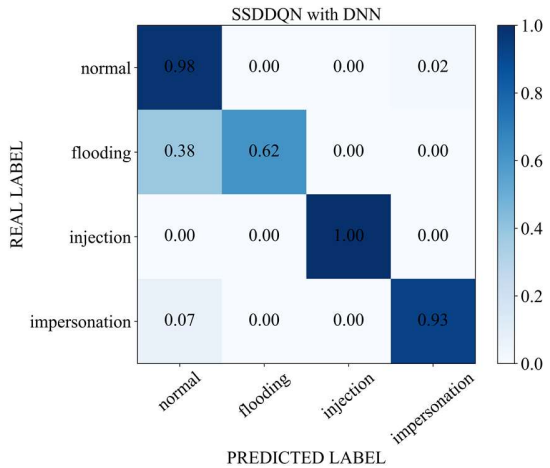

Fig. 17. Confusion matrix of the SSDDQN with DNN model

traditional ML, representative DL, and DRL models. Firstly, we implement binary classification and five classification experiments on the NSL-KDD dataset and compare all models' training time and prediction time. The experimental results show that the SSDDQN method proposed in this paper has achieved good results in abnormal traffic detection, especially in the binary classification problem. SSDDQN achieves good accuracy and F1-Score metrics in the five classification problems, and the training time and prediction time are relatively low. However, the optimization effect of the SSDDQN model is limited, and it has almost no detection ability to detect the lowest number of U2R abnormal attack traffic. Secondly, we compare the binary classification experiment results in the newer literature and comprehensively analyze the experimental results from the feature selection algorithm, the number of feature selection, accuracy, and FPR, etc. In addition, we implement a multi-classification experiment for classifying attack types to verify the ability of SSDDQN to detect unknown attacks. Thirdly, this paper also uses the AWID dataset to verify the performance of SSDDQN further. We perform binary classification and four classification experiments on the AWID dataset and test the impact of the discount factor on the model's performance. Finally, the paper conducts ablation experiments on SSDDQN to describe the contribution of each part to the overall performance.

In future work, our research will carry out from two aspects: (1) Application of multi-agent DRL to abnormal traffic detection. (2) Improve the detection ability of a small amount of sample data through adversarial learning between agents.

CONFLICT OF INTEREST

All authors of the paper immediately below certify that they have no affiliations with or involvement in any organization or entity with any financial interest.

REFERENCES

[1] M. Price‐Williams, N. Heard and P. Rubin‐Delanchy, "Detecting weak dependence in computer network traffic patterns by using higher criticism", *J. R. Star. Soc. C-appl*, vol. 68, no. 3, pp. 641-655, 2019.

[2] S. Latha and S. J. Prakash, "A survey on network attacks and Intrusion detection systems", *IEEE 4th ICACCS*, pp. 1-7, 2017.

[3] J. Wan, B. Chen, M. Imran, et al, "Toward dynamic resources management for IoT-based manufacturing", *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 52-59, 2018.

[4] B. K. J. Al-Shammari, N. Al-Aboody and H. S. Al-Raweshidy, "IoT traffic management and integration in the QoS supported network", *IEEE Internet Things*, vol. 5, no. 1, pp. 352-370, 2017.

[5] G. Fernandes, J. J. P. C. Rodrigues, L. F. Carvalho, et al, "A comprehensive survey on network anomaly detection", *Telecommun. Systems*, vol. 70, no. 3, pp. 447-489, 2019.

[6] K. Wang and S. J. Stolfo, "Anomalous payload-based network intrusion detection", *Proc. Recent Advances in Intrusion Detection*, pp. 203-222, 2004.

[7] M. S. Kim, H. J. Kong, S. C. Hong, et al, "A flow-based method for abnormal network traffic detection", *IEEE/IFIP Network Operations Management Symposium*, pp. 599-612, 2004.

[8] Y. Qiao, X. W. Xin, Y. Bin, et al, "Anomaly intrusion detection method based on HMM", *Electron. Lett.*, vol. 38, no. 13, pp. 663-664, 2002.

[9] E. C. Bayazit, O. K. Sahingoz, B. Dogan, "Malware Detection in Android Systems with Traditional Machine Learning Models: A Survey", *IEEE HORA*, pp. 1-8, 2020.

[10] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey", *Comput. Electron. Agr*, vol. 147, pp. 70-90, 2018.

[11] S. Pouyanfar, S. Sadiq, Y. Yan, et al, "A survey on deep learning: Algorithms, techniques, and applications", *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1-36, 2018.

[12] K. Arulkumaran, M. P. Deisenroth, M, Brundage, et al, "Deep reinforcement learning: A brief survey", *IEEE Signal Proc. Mag.,* vol. 34 no. 6, pp. 26-38, 2017.

[13] N. C. Luong, D. T. Hoang, S. Gong, et al, "Applications of deep reinforcement learning in communications and networking: A survey", *IEEE Commun. Surv. Tut.*, vol. 21 no. 4, pp, 3133-3174, 2019.

[14] H. Van Hasselt, A. Guez and D. Silver, "Deep reinforcement learning with double q-learning", *Proc. AAAI Conf. artificial Intell.*, vol. 30, no. 1, 2016.

[15] H. Choi, M. Kim, G. Lee, et al, "Unsupervised learning approach for network intrusion detection system using autoencoders", *J. Supercomput.*, vol. 75, no. 9, pp. 5597-5621, 2019.

[16] M. Ahmed, R. Seraj and S. M. S. Islam, "The k-means algorithm: A comprehensive survey and performance evaluation", *Electron*., vol. 9, no. 8, pp. 1295, 2020.

[17] M. Tavallaee, E. Bagheri, W. Lu, et al, "A detailed analysis of the KDD CUP 99 data set", *IEEE symposium on computational intelligence for security and defense applications*, pp. 1-6, 2009.

[18] C. Kolias, S. Gritzalis, A. Stavrou, and G. Kambourakis, "Intrusion detection in 802.1 1 networks: Empirical evaluation of threats and a public dataset", *IEEE Commun. Surv. Tut.*, vol. 18, no. 1, pp. 184–208, 2015.

[19] B. Ingre, A. Yadav and A. K. Soni, "Decision tree based intrusion detection system for NSL-KDD dataset", *IEEE ISNCC*, pp. 207-218, 2017.

[20] W. Zhongsheng, W. Jianguo, Y. Sen, et al, "Traffic identification and traffic analysis based on support vector machine", *Concurr. Comp- Pract E*, vol. 32, no. 2, pp. e5292, 2020.

[21] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems", *ACM Comput. Surv.*, vol. 51, no. 3, pp. 1-36, 2018.

[22] L. Van Efferen and A. M. T. Ali-Eldin, "A multi-layer perceptron approach for flow-based anomaly detection", *IEEE ISNCC*, pp. 1-6, 2017.

[23] H Wang, Z Cao and B Hong. "A network intrusion detection system based on convolutional neural network", *J. Intell. Fuzzy Sys.*, vol. 38, no. 6, pp: 7623-7637, 2020.

[24] A. F. M. Agarap. "A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data", *Proc. 10th Int. Conf. Machine Learning and Computing*, pp. 26-30, 2018.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, et al, "Playing atari with deep reinforcement learning", 2013.

[26] Z. Wang, T. Schaul, M. Hessel, et al. "Dueling network architectures for deep reinforcement learning", *PMLR Int. Conf. machine learning*, pp. 1995-2003, 2016.

[27] Y. Lei, "Network anomaly traffic detection algorithm based on SVM", *IEEE Int. Conf. Robots Intelligent System (ICRIS)*, pp. 217-220, 2017.

[28] L. Kong, G. Huang and K. Wu. "Identification of abnormal network traffic using support vector machine", *IEEE 18th Int. Conf. PDCAT*, pp. 288-292, 2017.

[29] D. H. Hoang and H. D. Nguyen, "A PCA-based method for IoT network traffic anomaly detection", *IEEE 20th ICACT*, pp. 381-386, 2018.

[30] Y. Zhang, X. Chen, D. Guo, et al, "PCCN: Parallel Cross Convolutional Neural Network for Abnormal Network Traffic Flows Detection in Multi-Class Imbalanced Network Traffic Flows", *IEEE Access*, vol.7, pp. 119904-119916, 2019.

[31] O. Salman, I. H. Elhajj, A. Chehab, et al, "A machine learning based framework for IoT device identification and abnormal traffic detection", *T. Emerg. Telecommun. T.*, 2019.

[32] T. Y. Kim and S. B. Cho, "Web traffic anomaly detection using C-LSTM neural networks", *Expert Syst. Appl.*, vol. 106, pp. 66-76, 2018.

[33] M. Gao, L. Ma, H. Liu, et al, "Malicious Network Traffic Detection Based on Deep Neural Networks and Association Analysis", *Sensors*, vol. 20, no. 5, pp. 1452, 2020.

[34] Z. Shi, J. Li, C. Wu, et al, "DeepWindow: An efficient method for online network traffic anomaly detection", *IEEE 21st Int. Conf. HPCC; IEEE 17th Int. Conf. SmartCity; IEEE 5th Int. Conf. DSS.*, pp. 2403-2408, 2019.

[35] P. Sun, P. Liu, Q. Li, et al, "DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system", *Secur. Commun. Netw.*, vol. 2020, pp. 1-11, 2020.

[36] A. S. Khatouni, N. Seddigh, B. Nandy, et al, "Machine learning based classification accuracy of encrypted service channels: analysis of various factors", *J. Netw. Syst. Manag.*, vol. 29, no. 1, pp. 1-27, 2021.

[37] F. Pacheco, E. Exposito, M. Gineste, et al, "Towards the deployment of machine learning solutions in network traffic classification: A systematic survey", *IEEE Commun. Surv. Tut.*, vol. 21, no. 2, pp. 1988-2014, 2018.

[38] W. M. Shbair, T. Cholez, J. François, et al, "A Survey of HTTPS Traffic and Services Identification Approaches", *arXiv preprint,* 2020.

[39] M. safaldin, M. Otair and L. Abualigah, "Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks", *J. Amb. Intel. Hum. Comp.*, vol. 12, no. 2, pp. 1559-1576, 2021.

[40] A. M. Mahfouz, D. Venugopal, S. G. Shiva, "Comparative analysis of ML classifiers for network intrusion detection", *Proc. Int. Congr. Inf. Commun. Technol.*, pp. 193-207, 2020.

[41] C. Tang, N. Luktarhan and Y. Zhao, "SAAE-DNN: Deep Learning Method on Intrusion Detection", *Symmetry*, vol. 12, no. 10, pp. 1695, 2020.

[42] N. Chaibi, B. Atmani and M. Mokaddem, "Deep Learning Approaches to Intrusion Detection: A new Performance of ANN and RNN on NSL-KDD", *Proc. 1st Int. Conf. Intell. Syst. Pattern Recog.*, pp. 45-49, 2020.

[43] G. S. Kushwah and V. Ranga, "Voting extreme learning machine based distributed denial of service attack detection in cloud computing", *J. Inf. Secur. Appl.*, vol. 53, 2020.

[44] P. T. Nguyen, V. D. B. Huynh, K. D. Vo, et al, "Deep Learning Based Optimal Multimodal Fusion Framework for Intrusion Detection Systems for Healthcare Data", *CMC-Comput. Mater. Con.*, vol. 66, no. 3, pp. 2555-2571, 2021.

[45] K. Jiang, W. Wang, A. Wang, et al, "Network intrusion detection combined hybrid sampling with deep hierarchical network", *IEEE Access*, vol. 8, pp. 32464-32476, 2020.

[46] S. M. Kasongo and Y. A. Sun, "Deep learning method with wrapper based feature extraction for wireless intrusion detection system", *Comput. Secur.*, vol. 92, 2020.

[47] S. Bhandari, A. K. Kukreja, A. Lazar, et al, "Feature selection improves tree-based classification for wireless intrusion detection", *Proc. 3rd Int. Workshop Syst. Netw. Telemetry and Analytics*, pp. 19-26, 2020.

[48] M. E. Aminanto, H. C. Tanuwidjaja, P. D. Yoo, et al, "WiFi intrusion detection using weighted-feature selection for neural networks classifier", *IEEE IWBIS.*, pp. 99-104, 2017.

[49] Q. Duan, X. Wei, J. Fan, et al, "CNN-based Intrusion Classification for IEEE 802.11 Wireless Networks", *IEEE 6th ICCC.*, pp. 830-833, 2020.

[50] S. Shamshirband, B. Daghighi, N. B. Anuar, et al, "Co-FQL: Anomaly detection using cooperative fuzzy Q-learning in network", *J. Intell. Fuzzy Syst.*, vol. 28, no. 3, pp. 1345-1357, 2015
[51] A. Servin and D. Kudenko, "Multi-agent reinforcement learning for intrusion detection: A case study and evaluation", *Proc. 6th German Conf. Multiagent Syst. Technol.*, pp. 159-170, 2008.
[52] H. Benaddi, K. Ibrahimi, A. Benslimane, et al, "A Deep Reinforcement Learning Based Intrusion Detection System (DRL-IDS) for Securing Wireless Sensor Networks and Internet of Things", *6th Int. Wireless Inter. Conf*, pp. 73-87, 2019.
[53] E. Suwannalai and C. Polprasert, "Network Intrusion Detection Systems Using Adversarial Reinforcement Learning with Deep Q-network", *IEEE 18th Int. Conf. ICT&KE.*, pp. 1-7, 2020.
[54] G. Caminero, M. Lopez-Martin and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection", *Comput. Netw.*, vol. 159, pp. 96-109, 2019
[55] M. Lopez-Martin, B. Carro and A Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems", *Expert Syst Appl*, vol. 141, 2020.
[56] K. Sethi, E. S. Rupesh, R. Kumar, et al, "A context-aware robust intrusion detection system: a reinforcement learning-based approach", *Int. J. Inf. Secur.*, vol. 19, no. 6, pp. 657-678, 2020.
[57] *KDD Cup1999*, Oct 1999, [Online] Available: http://Kdd.Ics.Uci.Edu/Databases/Kddcup99.html accessed on 8/18/2014.
[58] M. A. Wiering, M. Van Otterlo, "Reinforcement learning", *Adaptation, learning, and optimization*, vol.12, no. 3, 2012.
[59] C. J. Watkins and P. Dayan, "Q-learning", *Machine learning*, vol. 8, no. 3-4, pp. 279-292, 1992.

Shi Dong received M.S in Computer science from University of Electronic and technology of China in 2009. In 2013, Ph.D in computer science from southeast university. He worked as post doctor researcher in Huazhong University of Science and Technology. He is a visiting scholar in Washington University in St. Louis. He is currently a Distinguished Professor with Zhoukou Normal University and master supervisor of Wuhan Textile University. His current research interests include network management and network security.

Yuanjun Xia is now pursuing a master's degree in Computer Science and Technology at Wuhan Textile University in Wuhan, Hubei, China. He is a student member of CCF, and his research field is abnormal traffic detection and network traffic identification.

Tao Peng received the M.Sc degree and PH.D. Degree in Computer Science from the Huazhong University of Science and Technology in 2006 and 2011 separately. He is currently an associate professor at the School of Computer and Artificial Intelligence, Wuhan Textile University. His research interests include computer vision, pattern recognition, network security.