Prospects for the Application of Reinforcement Learning to Network Traffic Classification Tasks

G D Asyaev

Lecturer, Department of system programming, FSAEIHE SUSU (NRU), 76, Lenin prospect 454080, Chelyabinsk, Russia

E-mail: asiaevgd@susu.ru

Abstract. The basic principles and methods of reinforcement learning are reviewed. The problems and approaches for applying a model based on reinforcement learning in the framework of attack prevention are described. The model is built and the hyperparameters of machine learning for the task of classifying network traffic are selected, and its performance on the test data set is evaluated by such quality metrics as accuracy and completeness. The dataset used to implement an agent for selecting the optimal defense strategy for a particular attack has been finalized. Developed an algorithm for using a reinforcement learning neural network for the traffic classification task. A table of rules and rewards for the problem is generated. An agent has been developed and trained to interact with the system. We describe the application of reinforcement learning to the traffic classification task.

1. Introduction

Modern machine learning models that solve the problem of classifying network traffic [21] use a set of marked data representing a set of features (system state) and a target function for training [13]:
- the presence/absence of an attack within a binary classification;
- a type of attack within the multiclass classification.

This approach has some problematic factors: a strong exposure to the training sample (the model in most cases can only recognize attacks that were specified in the training sample), rather poor recognition of new types of attacks. However, the choice of defense strategy is often determined by humans. The main goal of this research is to use an add-on to an existing machine learning model to automatically select the most optimal strategy based on the type of threats.

Article I. Reinforced learning

1.1. General principles of reinforcement learning

Learning with reinforcement is an algorithm that initially has no information about the system and its state [2]. Its general principle can be described cyclically as follows (fig. 1):
1. The model gets the S1 state from the environment.
2. Based on this state, some actions take place in the system (A1)
3. The system switches to a new state (S1)
4. The model receives some reward (R1) from the system based on the action performed.
The main task of the model is to maximize the value of the reward [3]. Thus, the model will perform actions until it reaches the threshold label for the reward. The entire learning process can be represented as a time series, at each stage of which the reward value changes.

Using a formula, the reinforcement learning process can be written as [5]:

\[ G(t) = \sum_{k=0}^{M} R_{t+k+1} \]  

where \( R \) is the reward,  
\( t \) - time interval,  
\( k \) is the number of iteration.

### 1.2. Monte Carlo and Time Difference Methods

Initially, two types of tasks can be distinguished within reinforcement learning [25]:

- Episodic, where there is a beginning and an end point. This creates an episode: a list of states, actions, rewards, and future states [6].
- A constant task whose actions go on forever [6]. In this case, the system must learn to choose optimal actions and simultaneously interact with the environment.

Based on the selected type, reinforcement learning methods such as the Monte Carlo method and the time-difference method are applied [8].

1. The basic principle of the Monte Carlo method (fig. 2) is to evaluate the model performance after the end of the episode [9] using the reward value (it is accumulated over the whole training interval and must be maximal). After that the episode starts all over again, but with new knowledge.

\[
V(S_t) \leftarrow V(S_t) + \alpha \left[ G_t - V(S_t) \right]
\]  

**Figure 1.** Learning algorithm with reinforcement.

**Figure 2.** The Monte Carlo method.
2. The temporal method is used in a continuous learning task [13]. It updates the reward value after each time interval (fig. 3).

\[
V(S_t) \leftarrow V(S_t) + \alpha [G_t - V(S_t)]
\]

\[
V(S_t) \leftarrow V(S_t) + \alpha [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]
\]

**Figure 3.** Temporal method.

At time t+1 all values are updated, namely the reward value changes to Rt+1 and the current score to V(St+1) [14].

It is worth noting that since for the task of traffic classification [24] it is important to be able to detect attacks as early as possible and given the constant change in the state of the system over time, the use of the temporal method is optimal [22].

A separate task for the study is to determine the right strategy between searching for additional information about the system (this can be various characteristics of the system, statistics of its loads, ports used, protocols, vulnerabilities already found) and using what is already known to maximize the value of the reward.

Another problem of reinforcement learning is choosing the optimal strategy (π) [15]. There are approaches related to the trying of all possible variants of actions, but their number can be infinitely large and resource-consuming. It is more reasonable to choose the strategy with the highest expected payoff. In addition, it is possible to superstructure the algorithm by adding a regression model and try to predict the obtained value of the gain from the chosen action of the system.

The utility function approach uses multiple estimates of the expected payoff for only one strategy (neither current or optimal) [16]. In this case we try to estimate either the expected payoff, starting from the state Si, when further following the strategy (formula 2), or the expected payoff, when making a decision Ai in the state Si and further following the strategy (formula 3):

\[
V(S) = \mathbb{E}[R|S, \pi] \quad (2)
\]

\[
Q(A,S) = \mathbb{E}[R|S, \pi, A]Q(S,A) \quad (3)
\]

### 1.3. Reinforcement learning approaches

As a basis for the network traffic classification problem [23], a reinforcement learning model is chosen based on the optimization of the utility function V(S), namely the optimization of the expected reward when a particular state occurs. The model will use this value to determine which state is worth choosing [17] at a particular time interval, where the value of each state shows the accumulated reward value from the next step (fig. 4).

\[
u_\pi(s) = \mathbb{E}_\pi [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s]
\]

**Figure 4.** Utility estimation formula.

The detection and protection mechanism based on the Fuzzy Q-learning (FQL) algorithm can adapt to choose the best attack detection strategy from performing response actions. Regardless of the regularity of attacks, the IDS can adjust its learning parameters using Fuzzy Q-learning to identify future attacks [18]. Fig. 5 shows the proposed FQL architecture.
Article II. Experimental data

The main goal of the research is to make an add-on to an existing machine learning model for traffic classification, so that depending on the state of the system, we can propose one or another protection strategy. To build the model, we used a dataset from the Kaggle website, "UNSW_NB15 by the IXIA PerfectStorm tool. Australian Centre for Cyber Security (ACCS)." Unprocessed network packets from the UNSW-NB 15 dataset were created using the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) to create a hybrid of real contemporary normal activity and synthetic contemporary attack behavior. The training sample size is 82332 records (fig. 6).

The target function represented different types of attacks. Additionally, system modules were artificially generated on which these types of attacks interacted.

The Random Forest model was chosen as the machine learning model for classification, which is an ensemble of decision tree classifiers for different subsamples of the dataset and uses averaging to improve prediction accuracy and control over fitting. It was decided to use a large enough number of trees, but with a shallow depth to reduce the likelihood of overfitting, as there could be a risk of the model adjusting heavily to the training data.

The figure below (fig. 7) shows a comparative table of the results of the Random Forest model for different finetuning parameters and data preprocessing.
Figure 7. Comparative table of model parameters and its quality.

In order to implement different scenarios (policies), the dataset has been supplemented with a countermeasure for each type of attack, depending on the module under attack.

Article III. General work algorithm

The initial idea is to remember when the model predicts the correct strategy (reward) and ignore/zero the weights when the model is wrong. Then convert the predicted values into the target function and train the neural network on the marked-up data using back propagation of error if neural network is used, or standard machine learning methods.

It is worth noting that with this approach the neural network was never able to learn because the rewards received (which symbolize that the model is moving in the right direction) were quite rare and the model was mostly wrong. Because of the unbalanced sample, the neural network was unable to remember the rather rare instances of guessing the strategy and, as a consequence, to detect the relationship.

Based on the above, it is proposed to use a set of policies within the superstructure to classify network traffic [17]. Work algorithm:

1. There is a matrix of states and their probability of occurrence.
2. N sessions are sampled.
3. The best choice takes place.
4. For the best ones, the likelihood of state-action increases.
5. Update and go back to step 1.

Updating the policy in the case of a finite number of actions and a finite number of states is a probability table, where for each state there is a vector of dimensionality of the number of actions, in each cell of which there is some probability. One should thus try to maximize the probability of good actions.

Since for each state there is its distribution over all actions [18],

\[ \pi(S) = A_{S,A} \]  

All actions are equal when the session is passed at initial initialization. Sampling. Finish the session and look at the top 5% of the best state-action pairs.

\[ \text{Elite} = [(S_0, A_0), (S_1, A_1), ..., (S_n, A_n)] \]  

If such pairs were found, we can say that they carried useful information. Thus, for these pairs we can increase the probability value in order to increase the number of repetitions of these actions.

\[ \pi(A|S) = \frac{\sum_{S,A\in \text{Elite}}[S_t = S][A_t = S]}{\sum_{S,A\in \text{Elite}}[S_t = S]} \]
We end up with a ready teaching sample: good condition - good action. You can use training with a teacher. And by state from this base learn to predict action by state.

The figure shows the implementation of the basic structure of agents, which is the basis of learning agents (fig. 8):

- New $Q(S,A)$ is the new value that will be defined for the current state and action.
- $\alpha$ - the variable responsible for the learning rate;
- $R(S,A)$ is a function of the reward for an action performed;
- $Q(S,A)$ function of current values
- $\gamma$ - a coefficient that determines how much agents care about reward in the near future compared to what is in the distant future.
- $\text{Max } Q(S',A')$ represents the maximum expected reward for the current step and the number of possible actions for the new state [19].

![Figure 8. Function determining Q-tables weights updates.](image)

![Figure 9. Environment-agent interaction.](image)

Fig. 9 shows the interaction of the environment with the learning agent. The input to the environment is a set of attributes, on the basis of which the environment responds to the actions of the agent [20]. The agent, in turn, receives a reward from the environment for the corresponding actions, trying to maximize it.

The agent then receives a state from the environment, on the basis of which it can perform further action. The agent's actions directly affect the state of the environment, which is why the agent needs to know the actual state of the environment.

The figure below shows the graph of the loss function during training (the red line is the training data, and the green line is the test data). We can clearly see that after the 80th epoch the error stopped decreasing and the error function does not increase on the test data set. This indicates a low probability of model retraining.
Figure 10. The graph of the loss function.

Figure 11 shows a graph of the model quality metric (the red line is the training data, and the green line is the test data). The accuracy value on the validation dataset was 87%, which indicates successful training of the model.

Figure 11. The graph of quality metric change with increasing epochs.

In this paper, the basic principles and methods of reinforcement learning were discussed in detail. A machine learning model for network traffic classification tasks was built and its performance on a test dataset was evaluated using quality metrics such as accuracy and completeness. The dataset used to implement an agent for selecting the optimal defense strategy for a particular attack has been finalized. A reinforcement learning model was built and trained to solve the problem. It is worth noting that the developed model, although it allows you to identify the optimal protection strategy, but in the real problem should be the final decision to leave the information security specialist.

2. References
[1] Huang Y, Li S, Li C, Hou Y T and Lou W 2020 A deep reinforcement learning-based approach to dynamic eMBB/URLLC multiple
[2] Apruzzese G, Colajanni M, Ferretti L, Guido A and Marchetti M 2018 On the effectiveness of machine and deep learning for cyber security In 2018 International Conference on Cyber Conflict (CyCon) pp 371-390

[3] Berman D S, Buczak A L, Chavis J S and Corbett C L 2019 A survey of deep learning methods for cyber security Information 10(4) 122

[4] Milosevic N, Dehghan-Tanha A and Choo K K R 2017 Machine learning aided Android malware classification Computers and Electrical Engineering 61 266-274

[5] Wang Y, Ye Z, Wan P and Zhao J 2019 A survey of dynamic spectrum allocation based on reinforcement learning algorithms in cognitive radio networks Artificial Intelligence Review 51(3) 493- 506

[6] Nguyen N D, Nguyen T and Nahavandi S 2017 System design perspective for human-level agents using deep reinforcement learning: A survey IEEE Access 5 27091-27102

[7] Hasselt H V, Guez A and Silver D 2016 Deep reinforcement learning with double Q-learning In The Thirtieth AAAI Conference on Artificial Intelligence AAAI Press pp 2094-2100

[8] Nguyen T T, Nguyen N D, Bello F and Nahavandi S 2019 A new tensioning method using deep reinforcement learning for surgical pattern cutting In 2019 IEEE International Conference on Industrial Technology (ICIT) doi: 10.1109/ICIT.2019.8755235

[9] Papamartzivanos D, Mrmol F G and Kambourakis G 2019 Introducing deep learning self-adaptive misuse network intrusion detection systems IEEE Access 7 13546-13560

[10] Zhang Y, Qiu M, Tsai C W, Hassan M M and Alamri A 2016 A survey of game theoretic methods for cyber security for next generation wireless networks Expert Systems with Applications 493 61-66

[11] Feng M and Xu H 2017 Deep reinforcement learning based optimal defense for cyber-physical system in the presence of unknown cyber attack In Computational Intelligence (SSCI) IEEE Symposium Series on pp 1-8

[12] Lopez-Martin M, Carro B and Sanchez-Esguevillas A 2020 Application of deep reinforcement learning to intrusion detection for supervised problems Expert Systems with Applications 141 112963

[13] Dey S, Ye Q and Sampalli S 2019 A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks Information Fusion 49 205-215

[14] Muoz P, Barco R and de la Bandera I 2013 Optimization of load balancing using fuzzy Q-learning for next generation wireless networks Expert Systems with Applications 40(4) 984-994

[15] Wang Y, Wang Y, Liu J, Huang Z and Xie P 2016 A survey of game theoretic methods for cyber security In 2016 IEEE First International Conference on Data Science in Cyberspace (DSC) pp 631-636

[16] Boche H and Deppe C 2019 Secure identification under passive eavesdroppers and active jamming attacks IEEE Transactions on Information Forensics and Security 14(2) 472-485

[17] Elderman R, Pater L J, Thie A S, Drugan M M and Wiering M 2017 Adversarial reinforcement learning in a cyber security simulation In International Conference on Agents and Artificial Intelligence (ICAART) 2 pp 559-566

[18] Chen T, Liu J, Xiang Y, Niu W, Tong E and Han Z 2019 Adversarial attack and defense in reinforcement learning-from AI security perspective Cybersecurity 2(1) 11

[19] Liu Y, Dong M, Ota K, Li J and Wu J 2018 Deep reinforcement learning based smart mitigation of DDoS flooding in software-defined networks In 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD) pp 1-6

[20] Wu M, Song Z and Moon Y B 2019 Detecting cyber-physical attacks in CyberManufacturing systems with machine learning methods Journal of Intelligent Manufacturing 30(3) 1111-1123

[21] Nikolskaya K Y, Ivanov S A, Golodov V A, Minbaleev A V and Asyaev G D 2017 Review of modern DDoS-attacks, methods and means of counteraction 2017 International Conference "Quality Management, Transport and Information Security, Information Technologies" (IT&QM&IS) pp 87-89
[22] Nikolskaia K and Minbaleev A 2020 Legal Regulation of Incidents Related to DDoS Attacks 2020 International Conference Quality Management, Transport and Information Security, Information Technologies (IT&QM&IS) pp 53-55

[23] Nikolskaia K and Naumov V 2020 Ethical and Legal Principles of Publishing Open Source Dual-Purpose Machine Learning Algorithms 2020 International Conference Quality Management, Transport and Information Security, Information Technologies (IT&QM&IS) pp 56-58

[24] Nikolskaia K and Minbaleev A 2020 New Perspectives on Ethics and the Laws of Artificial Intelligence in the Investigation of Incidents Related to DDoS Attacks 2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon) pp 1-5

[25] Cheskidov P, Nikolskaia K and Minbaleev A 2019 Choosing the Reinforcement Learning Method for Modeling DDos Attacks 2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon) pp 1-4