Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6619715, 9 pages
https://doi.org/10.1155/2021/6619715

Research Article

Anti-Attack Scheme for Edge Devices Based on Deep Reinforcement Learning

Rui Zhang[1], **Hui Xia**[1], **Chao Liu**[1], **Ruo-bing Jiang**[1] and **Xiang-guo Cheng**[2]

[1]*The College of Information Science and Engineering, Ocean University of China, Qingdao /266100, China*
[2]*The College of Computer Science and Technology, Qingdao University, Qingdao /266100, China*

Correspondence should be addressed to Hui Xia; xiahui@ouc.edu.cn and Xiang-guo Cheng; chengxg@qdu.edu.cn

Received 9 December 2020; Revised 9 March 2021; Accepted 29 March 2021; Published 15 April 2021

Academic Editor: Yaguang Lin

Internet of Things realizes the leap from traditional industry to intelligent industry. However, it makes edge devices more vulnerable to attackers during processing perceptual data in real time. To solve the above problem, we use the zero-sum game to build the interactions between attackers and edge devices and propose an antiattack scheme based on deep reinforcement learning. Firstly, we make the *k NN-DTW* algorithm to find a sample that is similar to the current sample and use the weighted moving mean method to calculate the mean and the variance of the samples. Secondly, to solve the overestimation problem, we develop an optimal strategy algorithm to find the optimal strategy of the edge devices. Experimental results prove that the new scheme improves the payoff of attacked edge devices and decreases the payoff of attackers, thus forcing the attackers to give up the attack.

1. Introduction

Internet of Things (IoT) [1, 2] integrates various sensors or controllers with sensing and monitoring capabilities as well as advanced technologies (e.g., mobile communication technology and intelligent analysis technology) into all aspects of industrial production, realizing the leap from traditional industry to intellect industry. It has been widely used in logistics [3, 4], transportation [5], energy [6], and so on. During the application process of IoT, mass perception data is produced in end devices, which requires the edge devices to have higher real time, security [7, 8], and privacy [9, 10]. However, edge devices are usually located in a nearby user or on a routing path to the cloud, making them more vulnerable to attackers. For example, machine learning models on edge devices during the training period are vulnerable to well-designed adversarial examples [11, 12]. UPGUARD, an American cybersecurity firm, found that hundreds of millions of Facebook user records stored on Amazon's cloud computing servers could be easily accessed by anyone. Tens of thousands of private Zoom videos are uploaded to the public web page that anyone can watch online. The above threats can cause network penetration, personal data theft, and the epidemic spread of intelligent computer viruses. Therefore, preventing attacks and ensuring data security are the key to improve the efficient application of this system.

Currently, resisting malicious attackers mostly adopts traditional skills in IoT, such as encryption method and identity management technology. The encryption method is the most common traditional skill [13, 14]. However, due to the limited resources of edge devices in IoT, making the lightweight encryption program becomes one of the biggest challenges. Identity management technologies are the first line of resisting malicious attackers. However, the existing identity management technology cannot achieve identity authentication between multi-layer architectures. In recent years, a few emerging safety precaution technologies are widely used in IoT [15–17], such as trusted execution environments and machine learning technologies. However, most machine learning technologies, which are based on the assumption that training data remains constant during training, are incompatible with the environment where the data changes dynamically in real-time in IoT [18, 19].

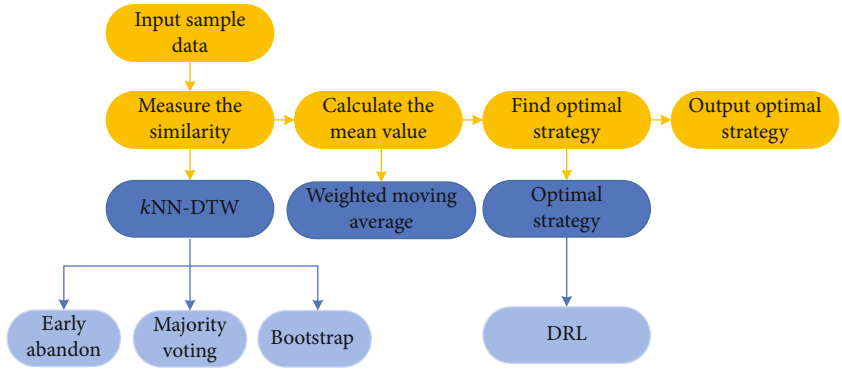


Figure 1: The structure diagram of the proposed scheme.

Inspired by the above schemes, from the point of view of attacker payoffs, we build the interactions between edge devices and attackers as the zero-sum game and propose an antiattack scheme for edge devices based on deep reinforcement learning. The structure diagram of the proposed scheme is shown in Figure 1. The major contributions are as follows:

(1) To find the optimal strategy of edge devices, we propose the *k NN-DTW* algorithm to find a similar sample to the current sample and then use the weighted moving mean method to calculate the mean and the variance of the samples;

(2) To weaken the influence of time series' irregularity, we emphasize the influence of the latest data on forecast value and then set weight for samples by the law that the object is big when near and small when far;

(3) To overcome the overestimation problem of the optimal strategy, we design an optimal strategy algorithm to find the edge device's optimal strategy by maximizing their accumulated payoff and then achieve the purpose of defending against attackers.

The structure of this paper is as follows: in Section 2, we define problems that we seek to solve in this article. In Section 3, we discuss the antiattack scheme for edge devices. In Section 4, we verify the effectiveness of the antiattack scheme for edge devices. Section 5 contains conclusions and future research.

2. Related Work

This section introduces the latest development and research of antiattack schemes from two aspects: traditional security protection schemes and emerging security protection schemes in IoT.

2.1. Traditional Safety Precaution Technologies. Homomorphic encryption, differential privacy, and identity authentication are three traditional protection technologies. Homomorphic encryption can process sensitive data without decryption to protect data privacy. Lu et al. [20] encrypted structured data by using homomorphic Paillier cryptographic system technology. Tan et al. [21] used the technique of finite field theory and proposed a private comparison algorithm based on full homomorphic encryption for encrypted integers. Differential privacy technology is used to ensure the privacy of any single item in the data set under the statistical query. Wang [22] proposed a data-driven spectrum trading solution that could maximize the income of PUS and retain SU's privacy differences. However, the computing resources on the edge devices are quite limited and cannot support the huge computing power consumed by using encryption schemes. Identity management technologies can set access authority by identity management and access control to prevent illegal user intrusion. Alizadeh et al. [23] summarized authentication technology in mobile cloud computing. Malik et al. [24] proposed an identity authentication and expeditious revocation framework based on the blockchain, which can quickly update the status of revoked vehicles in the shared blockchain. Zhang et al. [25] proposed a smart contract framework comprised of several access control contracts, a judge contract, and a registered contract, which gave a trusted access control strategy. However, using an access control strategy for precautions makes it hard to clear different users' roles and their rights in IoT.

2.2. Emerging Safety Precaution Technologies. The improvement of traditional security schemes can be used to enhance the security of edge devices in the IoT [26]. With the rise of artificial intelligence, some emerging security prevention technologies, such as trusted execution environment and machine learning technology, are gradually used to improve the security of edge devices in the IoT. The trusted execution environment [27] can be used to ensure the security of the running environment of the software. Running an application in a trusted execution environment can guarantee the security of data even if edge devices are compromised. Trusted execution environment, such as trustzone, intel management engine, and ARM trustzone, are quite popular. Han et al. [28] built a complete framework that supports visibility into encrypted traffic and can be used in secure and functional networks. However, trusted execution environment usually has its loophole, such as Qualcomm loopholes, and trustonic loopholes. Ghaffarian and Shahriari [29] proposed a neural vulnerability analysis method based on custom intermediate graph representation of the program for

software vulnerability analysis. Scandariatio et al. [30] explored machine learning-based text mining to predict security loopholes in a software source code. However, most machine learning methods assume that statistical data remain unchanged during the training process, but the data in the IoT changes dynamically in real time.

3. Problem Definition

In this article, we build the interactions between edge devices and attackers as a zero-sum game [31]. Namely, the payoff of attackers equals the loss of edge devices. In each round of interaction, the attacker attacks the sample from edge devices to gain illegal payoff, and the edge device plays his strategy to defend against attackers. Previous studies usually determine the optimal strategy of edge devices by calculating the Nash equilibrium in the handcrafted abstraction of the domain [32]. Currently, some researchers introduce the recursion technique to the neural network to determine players' optimal strategy by predicting human action in a strategic environment. From [33], the payoff function of edge devices can be defined as

$$U(a, \mu, C) = \log\left(a^T \mu\right) - \frac{1}{2 * \left(a^T \mu\right)^2} a^T C a, \tag{1}$$

where a is the strategy vector of edge devices, that is, $a = \{a_1, a_2, \cdots, a_n\}^T$; μ, $\mu = \{\mu_1, \mu_2, \cdots, \mu_n\}^T$ is sample mean payoff; and C is the covariance matrix of sample payoff. As can be seen from Equation (1), if we know the sample mean payoff and the covariance matrix, we can determine the optimal strategy of edge devices by maximizing the payoff function. That is,

$$a^{opt} \in \arg \max_a U(a, \mu, C). \tag{2}$$

However, the sample mean payoff and the covariance matrix are unknown. But, we can take advantage of the similarity between the historical sample and current sample to predict the sample mean payoff and the covariance matrix and then determine the optimal strategy of the edge device.

When we determine the optimal strategy of the edge device, we should try to resolve the following three problems: (1) during the process of measuring similarity between samples, we need to avoid using improper measuring methods which might cause the disappearance of optimal solution; (2) during the process of calculating the sample mean payoff and covariance matrix, we need to weaken the influence of time series' irregularity; (3) during the process of finding the optimal strategy of edge devices, we need to break the correlation between training samples and solve the problem of overestimation.

4. Antiattack Scheme for Edge Devices

This section introduces the following three problems that we seek to solve: how to find the sample that are similar to the current sample, how to calculate the mean value and the covariance matrix, and how to calculate the optimal strategy of edge devices to resist attackers.

4.1. Measuring Similarity of Sample. To find a sample similar to the current sample, we propose the *k NN-DTW* algorithm to determine the category of the current sample and then find the similar sample with the current sample. The *k NN-DTW* algorithm is a combination of the *k*-nearest neighbor algorithm and the dynamic time warping method (*DTW*); the *k NN* algorithm classifies the current sample and the *DTW* method finds the similar sample in the same category samples with the current sample.

In the *k NN* algorithm, the choice of k has a significant impact on the classification results. We use the bootstrap method to find the optimal value of k. Assuming the value of k and the probability of time series being correctly classified ρ satisfy the following regression model,:

$$\rho_i = h(k_i, \beta) + \varepsilon, i = 1, 2, \cdots, n, \tag{3}$$

where $h(\cdot)$ is the mapping from k to ρ, β is a coefficient vector, and $\{\varepsilon_i\}$ is a numerical vector, i.e., $F(x)$. We use the least square method to estimate β, i.e., $\widehat{\beta} = g(\rho_1, \cdots, \rho_n)$, make the regression residual empirical distribution function to estimate $F(x)$, and apply the bootstrap method to estimate the covariance matrix $\mathrm{Var}(\widehat{\beta})$ of β. If the estimation error of each coefficient (the square root of the diagonal element in $\mathrm{Var}(\widehat{\beta})$) meets the threshold ε_0, the value of k can be determined by maximizing ρ.

After the category of the current sample is determined by *k NN*, we use *DTW* to measure the distance between the current sample and the historical sample. *DTW* method locally scales two samples on the time axis to make the morphology of the two sets, so that the *DTW* method can measure the distance between time samples that have different lengths. Comparing with *Euclidean distance*, the *DTW* method is more elastic and supports local time shifts and in the length of time series, but the time and special complexity of this method is $O(nm)$, where n and m are the lengths of two time series, respectively. To decrease the space and time complexity of the *DTW* method, we apply early abandoning method to optimize the computations of the *DTW* method. The detailed process is as follows:

Step 1. Given two time series X and Y,

$$\begin{aligned} X &= (x_1, x_2, \cdots, x_n), \\ Y &= (y_1, y_2, \cdots, y_m), \end{aligned} \tag{4}$$

where n is the length of time series X, and m is the length of time series X

Step 2. Define the warping path as $P = p_1, p_2, \cdots, p_K$, where $\max(n, m) < K < m + n + 1$, $p_k = (i, j)$ is the kth element in warping path P, i is the ith cell of time series X, and j is the jth cell of time series Y, the i and j of $p_k = (i, j)$ are monotonically increasing,

$$p_k = (i, j), p_{k+1} = \left(i', j'\right), i \le i' \le i + 1, j \le j' \le j + 1. \tag{5}$$

Specially, when calculating warping path, it must ensure that every coordinate in the time series X and Y is involved. That is, the calculation starts from $p_1 = (1, 1)$ and ends at $p_k = (n, m)$

Step 3. Find the warping path between two time series with the shortest cumulative distance,

$$P = \arg\min_{p_k \in P} \sum_{k=1}^{K} p_k. \tag{6}$$

To obtain the warping path with the shortest cumulative distance, Eq. (6) can be solved iteratively by using the dynamic programming method

$$D(i, j) = \text{Dist}(i, j) + \min\{D(i-1, j), D(i, j-1), D(i-1, j-1)\} \tag{7}$$

Step 4. Set the distance threshold $\varepsilon, \varepsilon > 0$, if the distance $D(i, j) > \varepsilon$ in cell (i, j), the calculation of the distance between two time series on the path will be terminated

Step 5. Determine the distance between two time series $D(n, m)$

4.2. Calculating the Mean Value and the Covariance Matrix. To weaken the influence of time series' irregularity, we emphasize the influence of the latest data on forecast value and set weight for samples by the law that the object is big when near and small when far. Namely, the sample elements that are close to the prediction period will be given a relatively big weight. We use the weighted moving average method to calculate sample's mean value. That is,

$$\mu_t = \frac{y_t w_t + y_{t-1} w_{t-1} + \cdots + y_1 w_1}{w_t + w_{t-1} + \cdots + w_1}, \tag{8}$$

where w_t refers to the weight of sample data y_t; it follows the rule that weight decreases as the distance increases, i.e., $w_t > w_{t-1} > \cdots > w_1$. Accordingly, the covariance matrix C can be calculated as

$$C = E[(X - \mu_X)(Y - \mu_Y)]. \tag{9}$$

4.3. Preventing Malicious Attacks. After finding the similar sample, we first take the mean payoff and covariance matrix of the similar sample as the mean payoff and covariance matrix of the current sample, respectively. And then, to weaken the influence of time series irregularity, we emphasize the influence of the latest data on forecast value and set weight for samples by the law that the object is big when near and small when far. Finally, we find the solution to the optimal strategy of edge devices by maximizing the payoff function. The detailed process is shown in Algorithm 1.

However, the above method is prone to overestimation. To solve the above problem, we design Algorithm 2 to find the optimal strategy of the edge devices by maximizing their accumulated payoff.

Reinforcement learning is aimed at maximizing the reward for the long term to find the payoff maximum of the agent. Thus, players of the game are transformed into separate agents. We use *Deep Q Network* to find the agent's optimal strategy. In this algorithm, the state set S of agent is defined as $S = \{s_1, s_2\}$, where s_1 means that the current data is normal (not attacked) and s_2 means that the current data is abnormal (already attacked); the above states can be described by the *Markov decision process*. The action set A is defined as $A = \{a_1, a_2\}$, where a_1 means that the *agent* accepts the current data set, a_2 means that the *agent* rejects the current data set, and action reward R is defined as

Input: Similarity sample set W;
Output: Optimal strategy $a^{\max}$;
1: for $n = 2$: N do
2: Calculate *similarity* with *DTW*;
3: if *similarity* < 1 then
4: Continue;
5: else:
6: $U_P = \max_a U(a, \mu, C)$;
7: end for
8: return $a^{\max}$

Algorithm 1: Optimal strategy.

$$R = \begin{cases} 1, a = a_1, s = s_1 \text{ or } a = a_2, s = s_2, \\ -1, \text{others}. \end{cases} \tag{10}$$

The agent interacts with its fellow agents and stores its experience of strategy transitions (s_j, a_j, r_j, s_{j+1}) in replay memory R. To break the correlation between training samples, during the process of training, we select samples randomly from replay memory R to train model for finding the optimal strategy of the agent. The detailed process is shown Algorithm 2, where $\widehat{Q}$ is the payoff when the agent adopts the optimal strategy.

5. Stimulation Results

We use the Anaconda-integrated development tool to validate the proposal. First, we analyze the feasibility of weakening the influence of time series irregularity to prove the reasonableness of setting sample data's weight according to the rule of the object being big when near and small when far. Second, we compare the *DTW* method with seven classical distance methods like *correlation distance*, *Jaccard distance*, and cosine distance to verify the reasonableness of *k NN-DTW*. Finally, we apply optimal strategy to the rock-paper-scissors game [34] to verify the practicability of optimal strategy by comparing the winner (choose winner's strategy) and opponent (choose opponent's strategy) strategies.

5.1. Feasibility Analysis of Weakening the Influence of Time Series Irregularity. Tables 1–3 analyze the influence of weighting weights on each parameter in the target payoff function according to the law that the object is big when near and small when far. To better describe the complete process,

01: Initialize replay memory R;
02: Initialize anticipatory parameters η;
03: Initialize target function Q with weight ϑ;
04: **for** $episode = 1$, M **do**
05: Set policy $\sigma \leftarrow \begin{cases} a^{opt}, \eta \\ \beta, 1-\eta \end{cases}$;
06: Receive initial observation state s_1 and reward r_1;
07: **for** $t = 1$, T **do**
08: Select action at from policy σ;
09: Execute action at and observe reward r_{t+1} and observe new state s_{t+1};
10: Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in R;
11: Sample random minibatch of transition (s_j, a_j, r_j, s_{j+1}) from R
12: **if** t terminates at step $t+1$ **then**
13: $Q_t = r_t$;
14: **else**
15: $Q_t = r_t + r \max_{a_{t+1}} Q(\eta a_t + (1-\eta)\beta \mid s_t)$;
16: **end if**
17: Perform a gradient descent step on $(\widehat{Q} - Q_t)^2$ with respect to network parameters;
19: Periodically update the target networks Q;
20: **end for**
21: **end for**

Algorithm 2: Optimal strategy.

Table 1: Sample dataset.

Number	Data 1	Data 2	Data 3	Data 4	Weight
1	22.44	21.95	21.96	15.82	0.099066
2	23.01	21.94	22.67	16.69	0.08585
3	23.1	22.4	22.4	16.43	0.082904
4	23.2	21.7	21.88	15.84	0.07981
5	21.51	20.85	21.44	15.66	0.078681
6	22.08	21.3	21.45	16.2	0.073956
7	21.86	21.15	21.79	15.97	0.07361
8	21.39	20.77	21.11	16.65	0.07244
9	21.43	20.85	21.23	16.45	0.067382
10	21.48	20.96	21.25	15.89	0.051061
11	21.19	20.8	21.1	16.56	0.046455
12	22	21.24	22	16.54	0.045742
13	22.47	21.5	21.64	16.28	0.03978
14	21.6	20.74	20.81	16.44	0.03306
15	20.48	19.51	19.88	16.11	0.028871
16	20.19	19.75	19.8	15.9	0.019483
17	20.06	18.16	18.59	15.1	0.010127
18	18.77	18.06	18.31	14.51	0.004814
19	18.35	17.6	18.2	14.06	0.003591
20	18.41	17.73	17.99	13.88	0.003319

we set each sample dataset only has 20 data and make the weight for each sample, as shown in Table 1. Table 2 shows the prediction error between the weighted mean and mean under the same strategy profile $(1, 1, 1, 0)$, where 1 means that the edge device accepts the data, and 0 means that the edge device rejects the data.

Table 2: Player's mean payoff comparison.

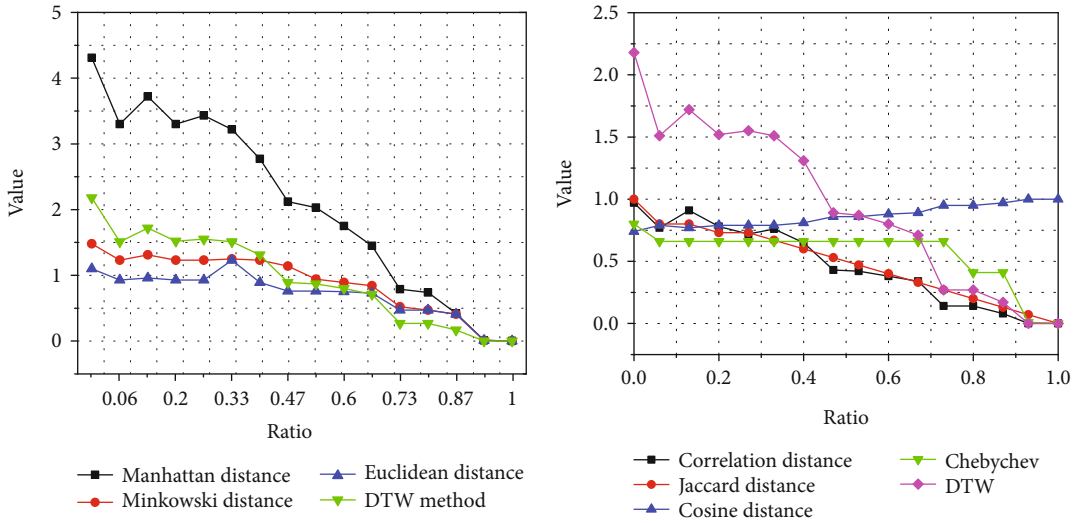
Value	Data 1	Data 2	Data 3	Data 5
Mean payoff	21.251	20.448	20.775	15.849
Weight mean payoff	21.9937	21.219	21.5435	16.1693
Error	0.742695	0.771	0.7685	0.3203
Action	1	1	1	0

Table 3 shows the effects of weakening the influence of time series irregularity on the parameters of the objective function (e.g., C). According to the table, setting the weight of the data according to the rule of the object is big when near and small when far has a greater influence on C and a weaker influence on $\log(a^T \mu)$. Therefore, the proposed method can weaken the irregularity of the time series.

5.2. Verification of the Reasonableness of k NN-DTW. To verify the reasonableness of combining *DTW* method and *k NN* algorithm, Figure 2 shows the *DTW* method with seven classical distance methods like *correlation distance*, *Jaccard distance*, and *cosine distance*. From Figure 2, we can see that the *cosine distance* and *Chebyshev distance* are the worst. For example, when the ratio of the same elements in the range from 20% to 33%, the results of the *Cosine distance* are all 0.79 while the ratio of the same elements in the range from 6% to 67%; the results of the *Cosine distance* are all 0.66. Therefore, *cosine distance* and *Chebyshev distance* are not suitable for measuring the distance between the samples in this paper. Although other methods also produce the same results, the number of the same results is less than that of *cosine distance* and *Chebyshev distance*. For example, the results of the *DTW* method are the same if and only if the ratio of the same elements is 80% or 87%. And the results

Table 3: The influence of time series irregularity.

Parameter	$a^T\mu$	C	a^TCa	$\log(a^T\mu)$
No-weight	62.47	$\begin{bmatrix} 2.15 & 2.10 & 2.07 & 0.99 \\ 2.10 & 2.18 & 2.14 & 1.03 \\ 2.07 & 2.14 & 2.15 & 1.03 \\ 0.99 & 1.03 & 1.03 & 0.69 \end{bmatrix}$	19.09	1.80
Weight	64.76	$\begin{bmatrix} 0.49 & 0.48 & 0.48 & 0.35 \\ 0.48 & 0.46 & 0.47 & 0.34 \\ 0.48 & 0.47 & 0.48 & 0.35 \\ 0.35 & 0.34 & 0.35 & 0.25 \end{bmatrix}$	4.29	1.81



(a) Similarity comparison (I)

(b) Similarity comparison (II)

Figure 2: Similarity comparison.

Table 4: Payoff matrix.

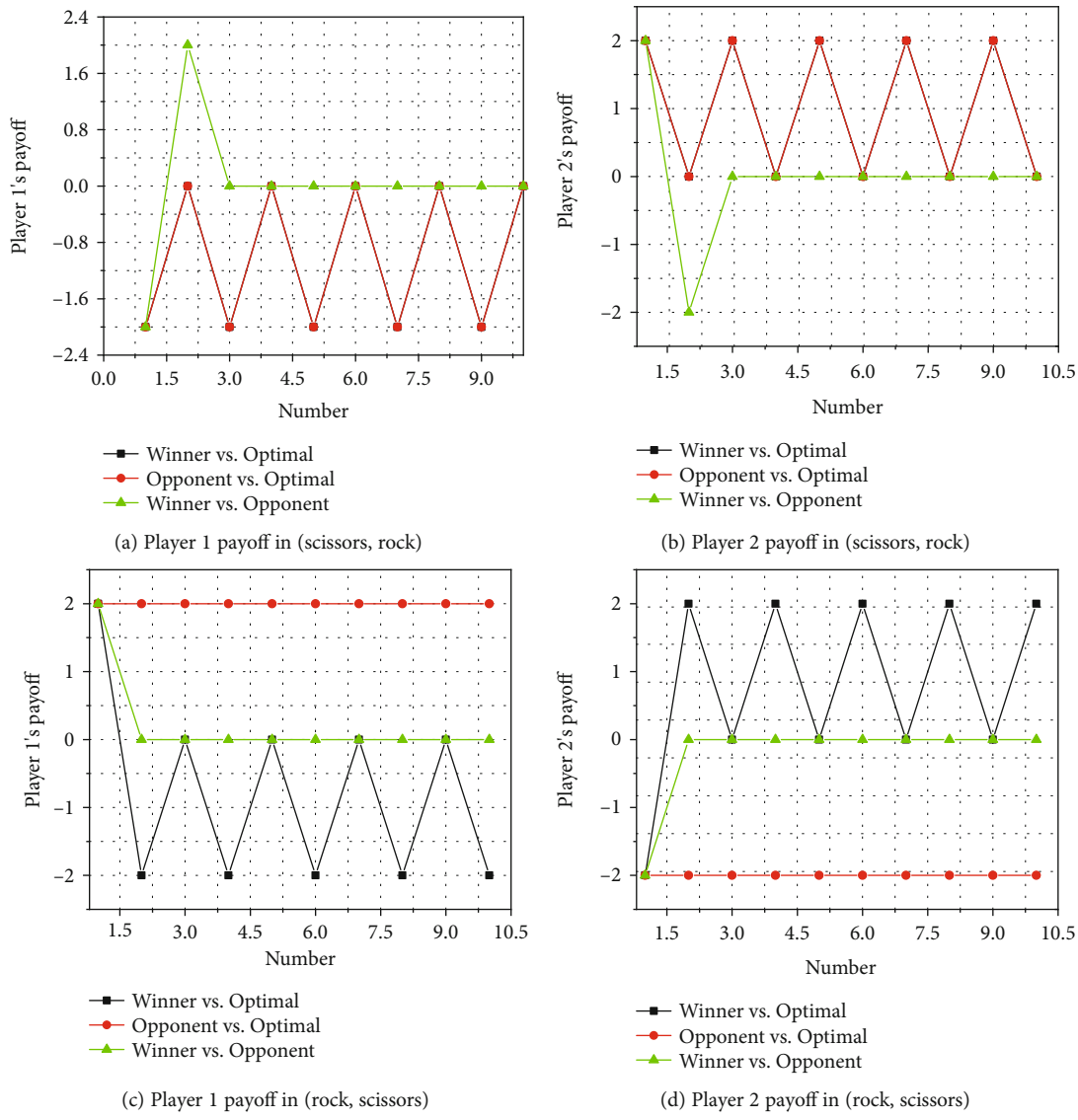
Player1/2	Rock	Scissors	Paper
Rock	0, 0	2, -2	-2, 2
Scissors	-2, 2	0, 0	2, -2
Paper	2, -2	-2, 2	0, 0

of *Euclidean distance* are the same if and only if the ratio of the same elements is 47% or 53%.

By comparing Figures 2(a) and 2(b), it can be seen that *DTW* method works best, followed by *Jaccard distance*. For example, when the number of the same elements is 80%, 87%, and 93%, the results of *Jaccard distance* are 0.2, 0.13, and 0.07; the results of *Euclidean distance* are 0.52, 0.41, and 0.01; the results of *Manhattan distance* are 0.74, 0.42, and 0.01; and the results of *DTW* method are 0.27, 0.17, and 0.0004, respectively. From the above results, we can draw a conclusion that *Euclidean distance* and *Manhattan distance* have a similar impact on measuring the distance between samples, while the results measured by these two methods varied greatly when the data in the two samples varied from 87% to 93%. While the *Jaccard distance* and *DTW* method have a similar impact measuring the distance between samples, the results measured by these two methods varied slightly when the data in the two samples varied from 87% to 93%. The *DTW* method is more suitable for measuring the distance between samples; this is because the *DTW* method can measure the distance between samples of different lengths. Therefore, we combine the *DTW* method with the *k NN* algorithm to measure the distance between samples.

5.3. Application of Antiattack Scheme. First, we need to define the rock-paper-scissors game's payoff matrix, as shown in Table 4. The rock-paper-scissors game is a typical example of zero-sum game. In the game, two players have the same strategy set, which is (rock, paper, scissors). If two players play the same strategy, then both of them get 0 for a draw; otherwise, the winner gets 2 and the loser gets -2.

Figure 3 shows the changing trend of payoff that player 1 and player 2 play optimal strategy, winner strategy, and opponent strategy in the initial states $S_0 = \{\text{scissors}, \text{rock}\}$



(a) Player 1 payoff in (scissors, rock)

(b) Player 2 payoff in (scissors, rock)

(c) Player 1 payoff in (rock, scissors)

(d) Player 2 payoff in (rock, scissors)

Figure 3: Payoff comparison.

Table 5: Total payoff comparison.

Players	Winner	Opponent	Optimal
Total	-16	8	8

and $S_1 = \{\text{rock}, \text{scissors}\}$. In the figure, winner vs. optimal means that player 1 plays winner strategy and player 2 players optimal strategy in the game. Similarly, we can know the meaning of opponent vs. optimal and winner vs. opponent. Figures 3(a)–3(d) show the payoff of player 1 and player 2 in S_0 and S_1 states, respectively. From Figure 3(a), we can see that due to player 1 adjusts scissors strategy to rock strategy when starting the second round of the game, the payoff of player 1 is -2 in the first round, and the payoff of player 1 is 2 in the second round. It is worth noting that the payoff trend of player 1 and player 2 is the same in winner vs. optimal and opponent vs. optimal because the strategies adjusted by winner strategy and opponent are the same in the initial state S_0. According to Figures 3(a)–3(d), we can draw a conclusion that the optimal strategy is optimal in state S_0, while in state S_1, optimal strategy is superior to winner strategy and inferior to opponent. As can be seen Table 5, the overall payoff of players is same in opponent strategy and optimal strategy. To sum up, optimal strategy scheme can help to determine the player's strategy and maximize the player's payoff.

6. Conclusion

In *IoT*, defending against attacks by determining the optimal strategy of the edge device for ensuring data security is the key to improve its effectiveness. In this article, we propose an antiattack scheme for edge devices based on deep reinforcement learning to solve this issue. And the core of this scheme is he optimal strategy algorithm. Detailed simulation experiment verified the effectiveness of this new scheme.

In future studies, we will focus on creating a new methodology to determine the similarity between data samples and use machine learning approaches to solve more data security problems.

Data Availability

All data generated or analyzed during this study are included in this article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61872205, the Shandong Provincial Natural Science Foundation under Grant No. ZR2019MF018, and the Source Innovation Program of Qingdao under Grant No. 18-2-2-56-jch.

References

[1] N. Lin, X. P. Wang, Y. H. Zhang, X. Hu, and J. Ruan, "Fertigation management for sustainable precision agriculture based on Internet of Things," *Journal of Cleaner Production*, vol. 277, article 124119, 2020.

[2] S. Chen, Y. Tao, D. Yu, F. Li, and B. Gong, "Privacy-preserving collaborative learning for multiarmed bandits in IoT," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3276–3286, 2021.

[3] W. Liu, S. Wang, D. Dong, and J. Wang, "Evaluation of the intelligent logistics eco-index: evidence from China," *Journal of Cleaner Production*, vol. 274, article 123127, 2020.

[4] F. Basso, J. Leonardo, and M. Ronnqvist, "Coalition formation in collaborative production and transportation with competing firms," *European Journal of Operational Research*, vol. 289, no. 2, pp. 569–581, 2021.

[5] Y. Yan, Q. Li, W. Huang, and W. Chen, "Operation optimization and control method based on optimal energy and hydrogen consumption for the fuel cell/supercapacitor hybrid tram," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 2, pp. 1342–1352, 2021.

[6] S. R. Pokhrel, H. L. Vu, and A. L. Cricenti, "Adaptive admission control for IoT applications in home WiFi networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 12, pp. 2731–2742, 2019.

[7] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li, "Deep learning based inference of private information using embedded sensors in smart devices," *IEEE Network*, vol. 32, no. 4, pp. 8–14, 2018.

[8] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 968–979, 2020.

[9] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: a survey," in *Wireless Algorithms, Systems, and Applications. WASA 2015*, K. Xu and H. Zhu, Eds., vol. 9204 of Lecture Notes in Computer Science, pp. 685–695, Springer, Cham, 2015.

[10] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[11] T. Tsiligkaridis, "Information Aware max-norm Dirichlet networks for predictive uncertainty estimation," *Neural Networks*, vol. 135, pp. 105–114, 2021.

[12] B. Henz, E. Gastal, and M. Oliveira, "Synthesizing camera noise using generative adversarial networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 3, pp. 2123–2135, 2021.

[13] H. Wang, J. Ning, X. Huang, G. Wei, G. S. Poh, and X. Liu, "Secure fine-grained encrypted keyword search for e-healthcare cloud," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[14] H. Cheng, H. Wang, X. Liu, Y. Fang, M. Wang, and X. Zhang, "Person re-identification over encrypted outsourced surveillance videos," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[15] H. Xia, L. Li, X. Cheng, X. Cheng, and T. Qiu, "Modeling and analysis botnet propagation in social internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7470–7481, 2020.

[16] Z. Cai and T. Shi, "Distributed query processing in the edge assisted IoT data monitoring system," *IEEE Internet of Things Journal*, 2020.

[17] S. Chen, Y. Tao, D. Yu, F. Li, and B. Gong, "Distributed learning dynamics of multi-armed bandits for edge intelligence," *Journal of Systems Architecture*, vol. 114, article 101919, 2021.

[18] F. Li, D. Yu, H. Yang, J. Yu, and K. Holger, "Multi-armed-bandit-based spectrum scheduling algorithms in wireless networks: a survey," *IEEE Wireless Communications Magazine*, vol. 27, no. 1, pp. 24–30, 2020.

[19] T. Zhu, T. Shi, J. Li, Z. Cai, and X. Zhou, "Task scheduling in deadline-aware mobile edge computing systems," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4854–4866, 2019.

[20] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: an efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.

[21] B. Tan, H. Lee, H. Wang, S. Q. Ren, and A. M. M. Khin, "Efficient private comparison queries over encrypted databases using fully homomorphic encryption with finite fields," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2020.

[22] J. Wang, "Data-driven spectrum trading with secondary users' differential privacy preservation," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 438–447, 2019.

[23] M. Alizadeh, S. Abolfazli, M. Zamani, S. Baharun, and K. Sakurai, "Authentication in mobile cloud computing: A survey," *Journal of Network and Computer Applications*, vol. 61, pp. 59–80, 2016.

[24] N. Malik, P. Nanda, and A. Arora, "Blockchain based secured identity authentication and expeditious revocation framework for vehicular networks," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp. 674–679, New York, NY, USA, 2019.

[25] Y. Zhang, S. Kasahara, and Y. Shen, "Smart contract-based access control for the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1594–1605, 2019.

[26] Q.-S. Hua, Y. Shi, Z. Cai, X. Cheng, and H. Chen, "Faster parallel core maintenance algorithms in dynamic graphs," *IEEE*

Transactions on Parallel and Distributed Systems, vol. 31, pp. 1287–1300, 2020.

[27] X. Ruan, "Platform Embedded Security Technology Revealed," in *Safeguarding the Future of Computing with Intel Embedded Security and Management Engine*, p. 272, Apress, 2014.

[28] J. Han, S. Kim, D. Cho, B. Choi, J. Ha, and D. A. Han, "A secure middlebox framework for enabling visibility over multiple encryption protocols," *IEEE/ACM Transactions on Networking*, vol. 28, no. 6, pp. 2727–2740, 2020.

[29] S. Ghaffarian and H. Shahriari, "Neural software vulnerability analysis using rich intermediate graph representations of programs," *Information Sciences*, vol. 553, pp. 189–207, 2021.

[30] R. Scandariatio, J. Walden, and A. Hovsesepyan, "Predicting vulnerable software components via text mining," *IEEE Transactions on Software Engineering*, vol. 40, no. 10, pp. 993–1006, 2014.

[31] H. Xia, R. Zhang, X. Cheng, T. Qiu, and D. O. Wu, "Two-stage game design of payoff decision-making scheme for crowdsourcing dilemmas," *IEEE/ACM Transactions on Networking*, vol. 28, no. 6, pp. 2741–2754, 2020.

[32] H. Xia, F. Xiao, S. Zhang, C.-q. Hu, and X.-z. Cheng, "Trustworthiness inference framework in the social internet of things: a context-aware approach," *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pp. 838–846, 2019.

[33] Y. Guo, X. Fu, Y. Shi, and M. Liu, "Robust log-optimal strategy with reinforcement learning," 2018, https://arxiv.org/abs/1805.00205.

[34] Z. Cai, X. Zheng, and J. Yu, "A differential-private framework for urban traffic flows estimation via taxi companies," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6492–6499, 2019.