Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8832697, 20 pages
https://doi.org/10.1155/2020/8832697

Research Article

Novel Defense Schemes for Artificial Intelligence Deployed in Edge Computing Environment

Chengcheng Zhou,[1] **Qian Liu**,[2] **and Ruolei Zeng**[3]

[1]*School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China*
[2]*Institute of Informationization and Software Industry, China Center for Information Industry of Development, Beijing 100036, China*
[3]*Computer Sciences, School of Computer, Data & Information Sciences, University of Wisconsin-Madison, Madison WI 53706, USA*

Correspondence should be addressed to Chengcheng Zhou; czho9311@163.com and Ruolei Zeng; zrlphilkar@hotmail.com

Received 7 April 2020; Revised 26 May 2020; Accepted 30 June 2020; Published 3 August 2020

Academic Editor: Ke Xiong

The last few years have seen the great potential of artificial intelligence (AI) technology to efficiently and effectively deal with an incredible deluge of data generated by the Internet of Things (IoT) devices. If all the massive data is transferred to the cloud for intelligent processing, it not only brings considerable challenges to the network bandwidth but also cannot meet the needs of AI applications that require fast and real-time response. Therefore, to achieve this requirement, mobile or multiaccess edge computing (MEC) is receiving a substantial amount of interest, and its importance is gradually becoming more prominent. However, with the emerging of edge intelligence, AI also suffers from several tremendous security threats in AI model training, AI model inference, and private data. This paper provides three novel defense strategies to tackle malicious attacks in three aspects. First of all, we introduce a cloud-edge collaborative antiattack scheme to realize a reliable incremental updating of AI by ensuring the data security generated in the training phase. Furthermore, we propose an edge-enhanced defense strategy based on adaptive traceability and punishment mechanism to effectively and radically solve the security problem in the inference stage of the AI model. Finally, we establish a system model based on chaotic encryption with the three-layer architecture of MEC to effectively guarantee the security and privacy of the data during the construction of AI models. The experimental results of these three countermeasures verify the correctness of the conclusion and the feasibility of the methods.

1. Introduction

In recent years, tens of billions of physical devices have been connected to the Internet by the IoT technology generating zillion-byte deluge of data [1]. To realize the purpose of humans to perceive data from the physical world and make decisions, AI, as an enabling technology, is introduced to intelligently process these large-scale data and identify complex patterns in the data [2]. Meanwhile, the continuously accumulated IoT data and greatly improved computing power have also played an important role in the further innovation of AI technology. Driving by this trend, AI has made substantial breakthroughs in a wide range of industries, such as computer vision [3–5], natural language processing [6, 7], autonomous vehicles [8], and robotics [9].

In traditional AI systems, the data bulks from the IoT devices are collected and transferred to the cloud data center for processing and analyzing [10]. Although the reliability of the cloud-centric manner has been proven, large-scale data and long-distance transmission across the network can cause network node congestion and latency issues [11], which results in an inability to cope with time-sensitive AI applications. One promising paradigm of AI computing that mitigates these problems is MEC [12, 13], which pushes cloud services from the network core to the network edges closer to IoT devices (data sources) and end-users. Therefore, integrating MEC and AI has become a great trend, which leads to the birth of edge intelligence or edge-enabled AI [14, 15].

Edge intelligence allows to offload some lightweight and time-sensitive AI computing tasks from the cloud to edge

servers near IoT devices or end-users, while retaining tedious and data-intensive computing tasks in cloud data center [14]. Specifically, Figure 1 illustrates the training and inference/test (In this paper, the terms inference and test are used interchangeably, and they refer to the same thing) process of the AI model in the edge environment. The original training datasets generated by IoT devices are stored in a cloud database for AI model training in the cloud. To achieve the high performance of the AI model, it is essential to retrain and update the AI model based on newly generated incremental data to adapt to the changes in data distribution [16]. Thus, the incremental data needs to be continuously transmitted from the IoT devices to the cloud and then be aggregated into an incremental dataset for model training. Afterwards, the cloud sends the well-trained AI model to the edge closer to the end-user where the inference is carried out. In this case, end-users can achieve real-time and timely feedback from the edge. This edge-enabled AI system greatly reduces the communication delays in the network and provides better response for end-users compared to the classic cloud paradigm. Avoiding sending all data to the cloud can reduce the expensive bandwidth required for the connection thus creating reliable connectivity due to a lower risk of network congestion [12, 17].

The advent of MEC gives impeccable illusions to AI. In fact, recent studies show that AI itself would encounter several security issues, which has proved to be fragile and vulnerable. In particular, in the physical world, it is fatal to conduct evil attacks on safety-critical AI systems. For instance, AI video surveillance systems can ignore those being monitored who carry simple printed patterns and attempt to hide themselves intentionally [18]. Even more, autonomous vehicles can be maliciously manipulated against patches with a visually natural characteristic [19]. In an automatic speech recognition model, by injecting a small noise, any speech waveform can be tampered with and becomes a completely different target phrase [20]. Therefore, it turns out that most AI systems are inherited with many security risks. Coincidentally, these AI systems are strongly associated with and empowered by MEC due to its time-sensitive requirement.

Specifically, the development of AI is driven by massive amounts of high-quality data. The quality and security of the collected data directly affect the performance of the AI model and then threaten the AI applications mentioned above in terms of security. Recent research has shown that attackers can incorporate specially processed fake data into the incremental dataset to undermine the integrity of the training data [21]. This situation is vividly described as a "poisoning attack." Note that the original data of model training is less likely to be tampered with due to its confidentiality. For example, Shafahi et al. [22] carried out a clean-label poisoning attack by designing harmful images in the feature space around the target image to the aim of changing the decision boundary of the AI classifier. In addition, attackers can implant hidden backdoor samples into the training data, and the triggered backdoor will induce the model to make mistaken decisions, which is "backdoor attack." Gu et al. [23] first found that the AI model can be embedded in the highly concealed backdoor which is difficult to perceive, except for the backdoor generator. It can be seen that the continuous accumulation of large amounts of data provides more opportunities for the injection of malicious data and also makes it the most direct and effective way to launch attacks during model training, as shown in Figure 1. In addition, studies have demonstrated that adding some imperceptible disturbances to original test samples can deceive well-trained AI models creating a false inference result with a high-confidence [24]. This is known as the adversarial example which takes into account the defect that the causality of the data is not obtained in the design of the AI model. Particularly, the transferable nature of adversarial examples has garnered continued attention from both the industry and academia [25]. Performing adversarial attacks by maliciously requesting inference (see Figure 1) will bring a fatal blow to the AI system. Regarding the issue of privacy disclosure of AI models (see Figure 1), a study has clarified that it is possible to infer and filch data used for model training even if little is known about the parameters and structure of the model [26]. The excessive collection of personal data caused by AI applications undoubtedly increases the risk of privacy disclosure.

To mitigate the corresponding security threats towards the AI system, in this paper, we propose three defense countermeasures against the threats of AI model training, inference, and private data. First of all, to solve the security challenges that AI faces in the training phase, we implement a cloud-edge collaborative antiattack scheme with a three-step design. This scheme fully takes the near-user advantages of MEC to realize a reliable incremental updating of AI by ensuring the data security generated in the training phase. Moreover, to effectively address the security problem in the inference stage of the AI model, we propose a defense strategy that includes two mechanisms, namely traceability and adaptive punishment mechanisms. With the help of the edge, the purpose of the strategy is to more effectively and radically eliminate attack intent in the inference phase. Most importantly, to protect the security and privacy of a large amount of training data required for the construction of AI models, in this paper, we propose a data transmission model based on chaotic encryption technology. This model aims at cutting off the transmission of raw data (unencrypted data) and protecting the security and privacy of the data from the propagation path. At present, most existing works are to defend against one of the aforementioned attacks, which will be elaborated separately in Section 2. In our work, a complete AI security challenge solution that includes three schemes was explored to deal with various potential attacks. Additionally, our motivation for using MEC to empower defense strategies stems from the various advantages that MEC brings to AI itself. The value of edge layer that can improve defense response speed has not been considered in terms of AI security defense strategy in most existing works. In particular, the existing cloud-based defense and data protection strategies are often time-consuming and high-complexity. At last, it is worth noting that the three schemes we proposed demonstrate their originality and effectiveness in

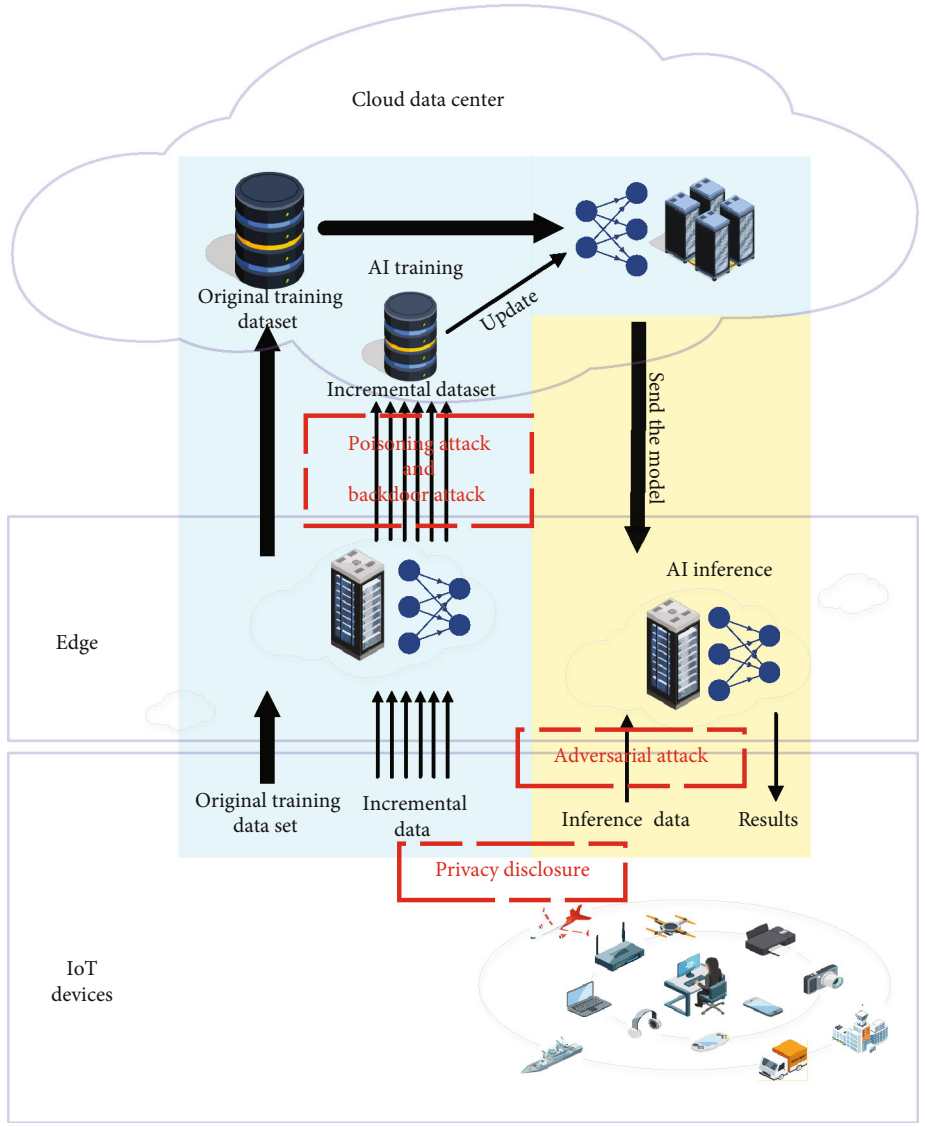

Figure 1: An architecture on edge-enabled AI security threats.

system model construction, such as the chaotic encryption technology and the adaptive penalty mechanism-based game model.

The main contributions of this paper are summarized as follows:

(1) We innovatively develop a cloud-edge collaborative antiattack scheme that includes an edge-filtering unit (EFU) running on the edge gateway, a cloud data security center (DSC), and a cloud model training center (MTC) running on the cloud server. The experimental results prove that with the help of MEC, reliable incremental updates of the AI training phase can be achieved

(2) We explore an edge-enhanced defense based on adaptive traceability and punishment against the security threat in the inference stage of the AI model. Its innovation lies in the establishment of an adaptive punishment mechanism based on evolutionary game theory, which can completely suppress the attack behavior and fundamentally eliminate the attack intention by setting reasonable penalty factors. Compared with traditional evolutionary games, the introduction of the adaptive penalty factor can ignore the effects of various initial proportions

(3) We propose a three-tier architecture that marginalizes AI and delivers tasks that must be completed by the cloud layer in the traditional AI to the edge layer to assist in the completion. Such an architecture reduces the burden of transmitting data from the cloud layer and broadband and reduces transmission and response delays, making smart devices perform better. To effectively protect the privacy problem of data, a scheme for encrypting the transmitter based on chaotic synchronization is proposed. The scheme has strong applicability and high confidentiality performance and is an effective means of data privacy protection.

The rest of this paper is structured as follows. Section 2 goes over diverse defensive strategies against security threats to AI. Section 3, Section 4, and Section 5 describe the three defense systems, respectively, in detail. Section 6 reveals the experimental results corresponding to the three proposed defense systems. In Section 7, our conclusions are presented.

2. Related Work

In response to the above security issues, many researchers have proposed different solutions. The existing security strategy for poisoning attacks in the model training phase mainly focuses on data detecting and sanitization techniques. Laishram et al. [27] controlled the training dataset by detecting and filtering malicious training samples during the retraining to realize the defense against poisoning attacks. Paudice et al. [28] introduced an anomaly detection model based on similar-distance filtering mechanism. Steinhardt et al. [29] proposed a defense framework that is allowed to detect and remove outliers outside the feasible training dataset. As for backdoor attacks, the countermeasures based on the detector construction are still applicable. Wang et al. [30] demonstrated that their proposed detection and reconstruction system can be well promoted against backdoor attacks faced by neural networks. Liu et al. [31] inhibited backdoor neurons by cutting out some neurons in the model. Chen et al. [32] detected and removed neurons without the need for any training dataset to ensure that deep neural networks are protected from backdoor attacks. Although the above defense technologies have proved effective to some extent, there are still some limitations. Due to the high concealment of security threats caused by poisoning and backdoor attacks during the training phase and the long delay of detection, current defense methods are facing challenges. Furthermore, traditional cloud detector-based defense methods cannot resolve the limitations of the cloud-computing paradigm, which results in its centralized execution in the cloud. The lack of up-to-date threat data of the cloud has caused detectors to respond to new types of attacks too slowly.

The defense method of adversarial attack in the inference phase mainly focuses on adversarial training, gradient masking, and input transformation. Szegedy et al. [24] mixed adversarial samples with training data to enhance the robustness of the model while reducing the sensitivity of the model to adversarial samples. But this method is difficult to deal with large-scale training data. Kurakin et al. [33] and Tramer et al. [34] proposed more efficient adversarial training methods to deal with large-scale data. Ross et al. [35] also improved model robustness by smoothing and regularizing the gradient of the model. Meng et al. [36] detected and reformed the adversarial examples towards the manifold of normal examples before the samples entered the model. Existing methods cannot guarantee the removal of all adversarial disturbances in low-complexity tasks. In addition, due to the diversity of adversarial samples, multiple defense models need to be trained to detect or resist more types of adversarial sample attacks, so that the defense model has the ability to generalize all adversarial samples as much as possible. But such an iterative training process greatly increases the time consumption of defensive strategy generation. To avoid the mentioned defects, we regard the adversarial attack as a competitive game between end-users who implement model inference. Game models are widely used to solve security-related problems. For example, Sun et al. [37] illustrated an evolutionary game model to protect user cooperation security of fog computing. We believe that the game model can also play an essential role in resisting the adversarial attacks suffered during the AI inference.

The privacy protection method of AI data is mainly based on differential privacy and homomorphic encryption technology to implement the encryption of training data and output information [38, 39]. Huang et al. [40] proposed a distributed learning algorithm based on alternating direction method of multipliers, which can be applied to a wider range of distributed learning problems and guarantees a balance between practicability and privacy protection. Differential privacy technology is less robust and inefficient, and it is difficult to resist sophisticated attacks. Aono et al. [41] proposed a system that utilizes homomorphic encryption technology to protect data privacy and avoid leaking local data information to curious servers. Homomorphic encryption technology still has certain limitations, for example, it does not support noninteger data [42]. The chaotic model is highly nonlinear and extremely sensitive to initial values [43]. Therefore, chaotic encryption technology is widely used in many fields such as secure communication. To the best of our knowledge, chaotic encryption technology is not used in data privacy protection in machine learning. Recently, chaotic encryption technology has also been applied to the data transmission field of the Internet. Particularly, Hui et al. [44] proposed a new data transmission scheme by using the synchronization of fractional-order chaotic models, which effectively guarantees the security of data transmission in the industrial IoT. Inspired by this, we believe that chaotic encryption technology can be an effective method for data privacy protection in artificial intelligence.

3. The Cloud-Edge Collaborative Antiattack Scheme

3.1. Security Issues in the Training Phase. Training is the core process in the generation of AI models, the performance of which is consequently highly associated with the quality of training data. The security threats that AI faces during the training phase are mainly attacks on training data. The original data used for model training is extremely confidential and hard to be attacked. However, the AI is constantly updated incrementally as the application scenario changes. The incremental data used for model updates is vulnerable to attacks, which mainly consists of the poisoning and backdoor attacks.

Attackers make malicious samples through malicious tampering and label inversion in poisoning attacks. As shown in the left half of Figure 2, the label of a normal "cat" sample is inverted to "dog." If these toxic samples are eventually applied to incremental updates, the AI will suffer from poisoning attacks. As a result, it is extremely easy for

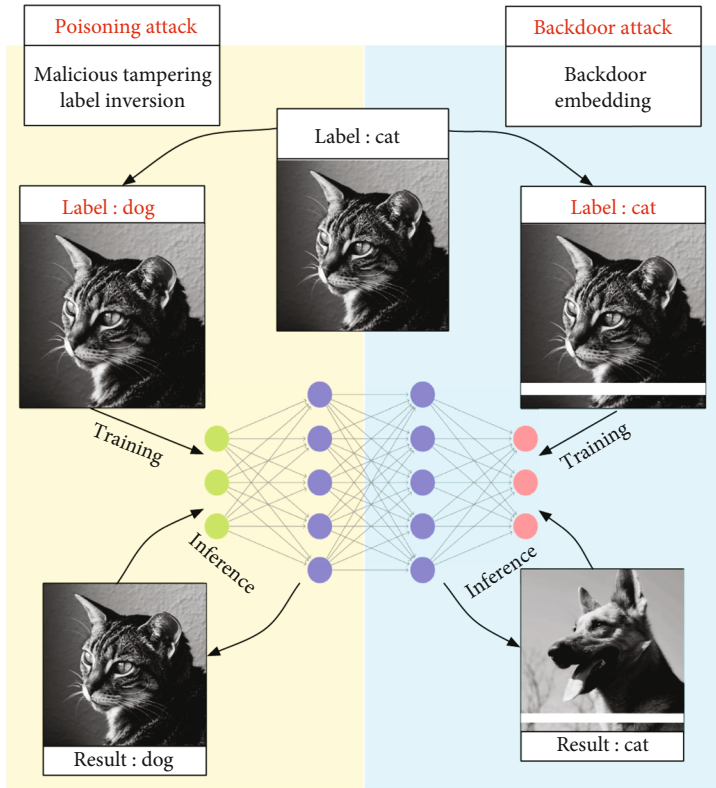

Figure 2: Poisoning attack and backdoor attack in AI training.

the model to mistakenly recognize a "cat" picture as a "dog" during the inference phase. Attackers can maliciously interfere with the recognition results of "cat" photos, and the security of the AI model has been breached.

Attackers implement hidden damage to the AI model through backdoor implantation in backdoor attacks. As shown in the right half of Figure 2, a row of white pixels is implanted as a backdoor at the bottom of a normal "cat" sample with the label unchanged. The result is that any photos containing the backdoor pixels, such as the "dog" in Figure 2, will be easily recognized as "cat" while the recognition results of normal "cat" and "dog" pictures are not affected. The attacker has implemented a targeted attack on all photos through the backdoor attack.

To cope with the security threats that AI faces during the training phase, this paper proposes a cloud-edge collaborative antiattack scheme that implements the detection of training attacks by setting traps. The security scheme mainly includes an EFU, a DSC, and a MTC. Unknown samples will be added to the incremental data by default and a pending model will be incrementally updated. The pending model is deployed to the edge to identify and defend against malicious attacks by comparing its results with that of the old model.

3.2. System Model. We propose a cloud-edge collaborative antiattack scheme in this section as shown in Figure 3. The cloud and the edge cooperate with each other to defend against backdoor and poisoning attacks that AI faces in the training phase. The cloud consists of two modules, a data security center and a model training center. Each edge server is deployed with an edge-filtering unit. First, the edge-filtering unit detects known attacks and screens for suspicious unknown threats, which are uploaded to the data security center to be identified. Then, newly identified attacks are used for the retraining of the cloud detector to update the edge-filtering unit. Finally, the cloud model training center will be used to train the security AI model and the pending AI model.

3.2.1. The Edge-Filtering Unit. The EFU is set at the edge of IoT as the first line of defense against AI training attacks. It is used for preliminary filtering of poisoning and backdoor attacks. The EFU near the attack site can filter out known threats at the initial data source to reduce the unnecessary bandwidth consumption and relieve the defense pressure of the cloud security center. More importantly, it can also realize the preliminary screening of unknown threats to further improve the antiattack ability of the AI defense system. The EFU includes two modules, the known hostile data detection (KHDD) and the unknown hostile data filtering (UHDD). The sufficiently powerful MEC power enables the KHDD to implement filtering of known threats in the edge area based on detectors deployed from the cloud DSC. The core components of the UHDD are composed of the active and the pending AI models running simultaneously. The principle is to implement the screening of suspicious threats based on the difference between the two models' judgment of the same unknown data. The pending model may have been attacked; for that, it is trained from all the data uploaded from the edge during the pending period with unknown hostile data

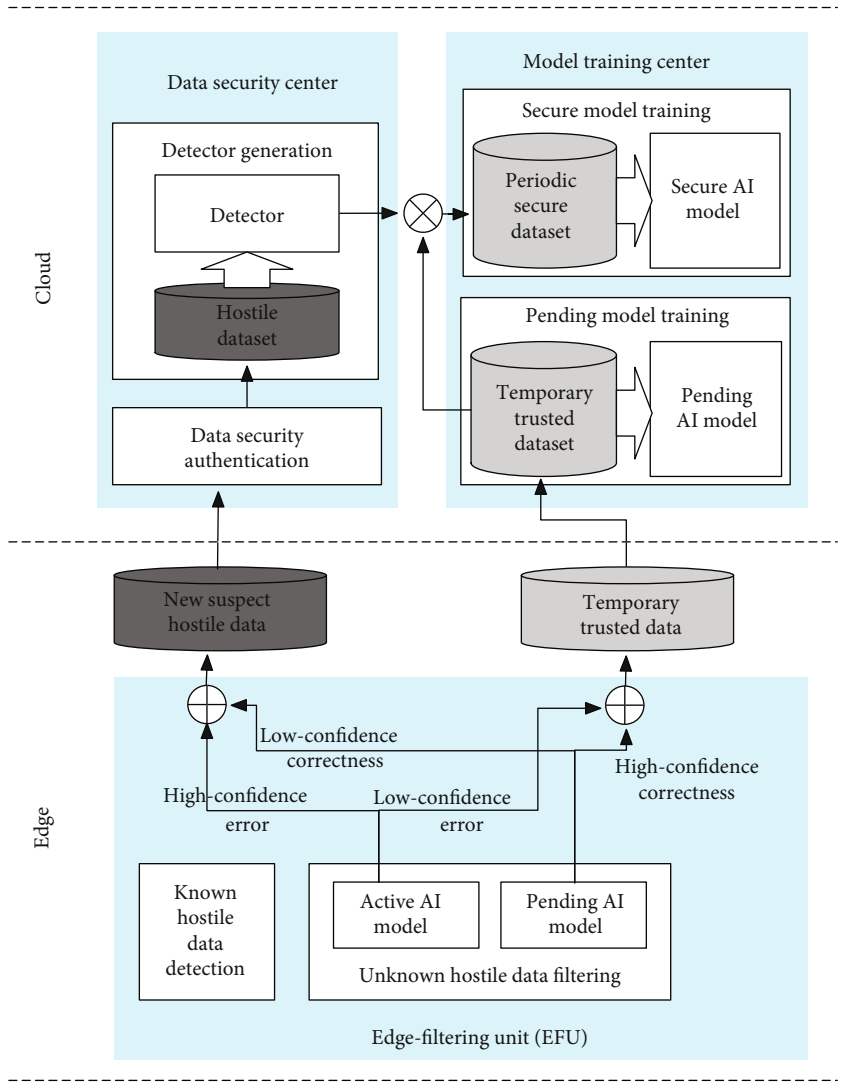

Figure 3: The cloud-edge collaborative antiattack scheme in the training phase of AI.

included. The active model is not fresh but safe because it is trained based on trusted data before the pending period and officially released by the cloud. The recognition results of AI models are divided into high-confidence and low-confidence according to the given confidence threshold. The unknown sample will be judged as the new suspicious threat if its output result is a high-confidence error in the active model and a low-confidence right in the pending model. New suspicious threat samples are then uploaded to the cloud DSC for the final threat identification. The unknown sample will be determined as the temporary trusted data if its output result is low-confidence error in the active model and high-confidence right in the pending model. Temporary trusted samples are then uploaded to the cloud MTC for the training of pending models.

3.2.2. The Cloud Data Security Center. The main purpose of the cloud data security center includes the final determination of suspicious threats and the generation of threat detectors. The cloud achieves authoritative identification of suspicious threats by comprehensively using multiple identification methods on the basis of powerful computing resources. If a new suspected threat sample is determined to be an attack, it will be saved to the hostile database for updating the detector. The updated detector is first deployed to the EFU for detection and prevention of known threats. In addition, it will be used for refiltering of temporary trusted data to form a periodic secure dataset in the cloud MTC.

3.2.3. The Cloud Model Training Center. The cloud model training center is used for the training of all AI models, including the secure model and the pending model. The model to be determined is based on a temporary trusted database, which is tested and filtered by the data security center for a period of time to form a periodic secure database for the generation of the secure model.

For the poisoning attack shown in Figure 2, the actual content of the malicious image, a cat and the old active model will recognize it as a "cat" with a high degree of confidence in its recognition results. But it does not match the sample label,

and it will be classified as a high-confidence error. At the same time, the pending model which has been trained on the same toxic sample will output the correct result "dog." This pending phase only occurs during the initial attack phase with the short-term and limited attack samples and the confidence of the correct results of the pending model will not be very high. Therefore, the poisoned sample will be identified as suspicious attack data in the edge-filtering unit and cannot be used to update the final secure model.

For the backdoor attack shown in Figure 2, the backdoor attack "cat" sample with the white-pixel stripe will be considered as temporary trusted data for the training of the pending model in the early stage since the backdoor attack occurs. After a period of backdoor implantation, the attacker entered the "dog" image (labeled as "cat") with the same white-pixel stripe to activate the backdoor. The old active model will recognize it as a "dog" with a high-confidence error. The pending model will recognize it as a "cat" with low-confidence correctness. The backdoor sample will be judged as a suspicious threat and added to the hostile database. As the number of attacks increases, the detector will gradually have the ability to identify the back door (the white-pixel stripe). Attack data with this backdoor characteristic in the temporary trusted database will be filtered out in the incremental update of the periodic secure database.

4. An Edge-Enhanced Defense Based on Adaptive Traceability and Punishment

4.1. Security Issues in the Inference Phase. Inference refers to the stage in which a well-trained AI model is used to classify/-predict real-world samples, which occurs after training. Despite the state-of-art AI inference model is available, it suffers from the security threat of adversarial attacks at the same time. Malicious attackers use adversarially tampered test samples to deceive the AI model into producing an erroneous output with a higher probability, causing the model to produce wrong behavior, thereby achieving the purpose of attacking the AI system [24]. This adversarial example is generated by carefully interfering with the normal sample (e.g., by adding adversarial perturbation or imperceptibly little noise) so that the human eye cannot discern the difference between the adversarial sample and the original sample but can deceive the model. As shown in Figure 4, the AI model correctly classified the cat image as "cat" with lower confidence, but with the carefully constructed noise, the AI model actually identified it as "dog" with high-confidence. However, there is no doubt that both pictures clearly show the "cat." Except that tampering is not easily perceived, most existing AI models are vulnerable to adversarial examples [45, 46]. There exist "universal perturbations," so that samples transformed by this perturbation enable transferring across models with different architectures or trained on different training sets [25, 47]. This "transferability" allows attackers to trick the AI system in a "black box attack" [48].

To cope with the security threats that AI faces during the inference phase, this paper proposes an edge-enhanced adversarial defense based on adaptive traceability and punishment (EADATP).

4.2. System Model. In this section, taking full advantage of the MEC, the EADATP can trace back to the test adversaries in a timely and accurate manner and impose severe punishments, to achieve comprehensive, high-efficiency, low-energy-consuming defense against inference attacks via adversarial examples. Due to its adaptive adjustment ability, the defense system is more practical in the real world. In the following, the implementation of this defense strategy will be explained in detail around the construction of traceability technology and punishment mechanism.

After a user sends a test request to the edge server, according to the testing process, the EADATP defense framework will respond in two stages, as shown in Figure 5.

Stage 1. Before the formal testing process, the user first needs to perform identity authentication, and a unique user ID is generated at the same time. After that, the user is required to sign a contract with legal benefits and accept or reject the terms of the contract. If the contract is accepted, the test sample can enter the testing process; otherwise, the edge server will refuse to provide testing services for the user. The terms in the contract indicate that users must comply with the test security provisions and that if a security incident occurs, users will be penalized. The punishment mechanism is formulated and published by the test platform, which will be explained in detail later.

Stage 2. When the sample enters the testing phase, the database deployed at the MEC layer will store the user's ID and authentication information which is used to trace the source of the malicious user after the accident. After the sample passes the test model, the user will receive the prediction result if TRUE is returned; otherwise, the traceability mechanism is required to take effect because the returned FALSE result indicates that an adversarial attack has occurred. After the traceability mechanism takes effect, the edge server sends a tracing instruction to the database, and the user can be accurately traced according to the authentication information. After that, the penalty clause in the contract is executed against the user, and the user must accept the penalty.

In the following, we consider two enabling technologies mentioned above with respect to implementing a defense scheme.

4.2.1. Authentication-Enhanced Traceability Technology. For users who forge or hide IP addresses, such as using a virtual private network (VPN), traditional traceability techniques are difficult to trace. Therefore, we introduce an authentication-enhanced traceability technology, that is, users accessing the edge platform for the first time need to bind their mailbox, mobile phone number, and bank account as authentication information. Even if a malicious user logs in to the test platform with a hidden IP address, the technology can still find the attacker based on the path provided by the authentication information.

4.2.2. Adaptive Evolutionary-Based Punishment Mechanism. To complete the penalty clause in the contract, an adaptive punishment mechanism based on evolutionary game theory [49] is established, which is formulated by the edge-based inference platform, so that the benefits of the user with attack

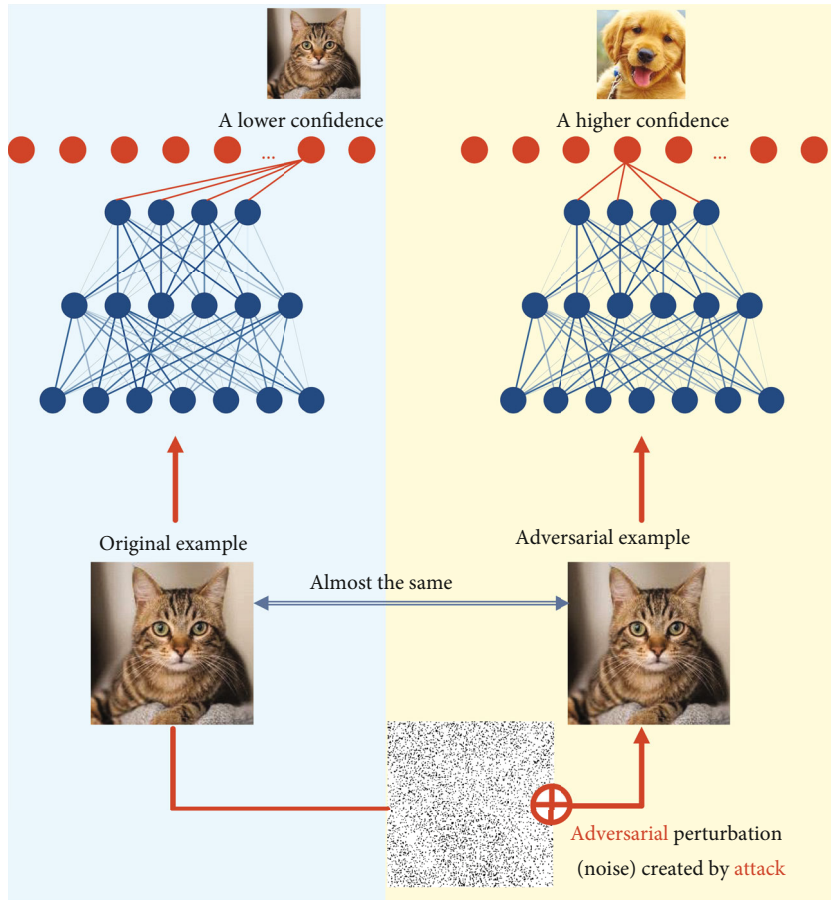

Figure 4: Adversarial attacks occur during the inference phase.

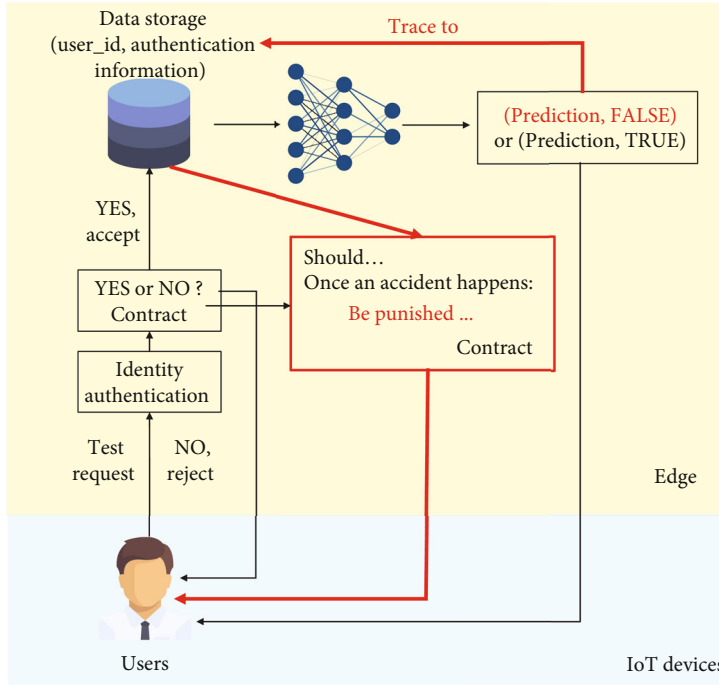

Figure 5: An edge-enhanced defense based on adaptive traceability and punishment.

behavior after being punished are less than that of the user with normal behavior. This is a dynamic game process such that users constantly adjust their strategies in accordance with their existing interests in order to pursue their own interests in the direction of high returns. Furthermore, the test platform adaptively adjusts the punishment intensity to achieve a more reasonable punishment mechanism design. Ultimately, this punishment strategy is to reduce the motivation of the attack and fundamentally force the user to stop the attack.

As the premise of evolutionary game theory is that participants have limited rationality rather than complete rationality, this makes the mathematically based theory of practical significance, because in the actual test case, users' choices and decisions on attack or nonattack are affected by irrational impulses, emotions, and other limited rationality, rather than being restrained by several feasible strategy options that are formulated in advance. Based on the assumption of limited rationality, taking the Evolutionary Stable Strategy (ESS) [50] as the basic equilibrium theorem and the Replicator Dynamics (RD) [51] as the core, the effects of different punishment conditions on the user's behavior choice are of practical significance.

(1) Game Model Establishment. Based on the principle of evolutionary games, the basic framework of the model is given below (see Figure 6), which is based on the following assumptions.

(1) *Game Players*. The two parties participating in the game are denoted as user 1 and user 2 who will request the test service, as shown in Figure 6. We suppose that both parties have a relationship of interest, and individual users are equally and independently. In addition, users are affected by limited rationality such that they have the ability to perform statistical analysis and to determine the benefits of different strategies. The strategies mentioned here will be elaborated in Assumption 2

(2) *Strategy Selection*. Since the behavior of each population can be regarded as a strategy, both user 1 and user 2 are supposed to have two strategies, namely normal behavior and attack behavior, respectively, recorded as Action 1 and Action 2, as illustrated in Figure 6. In the initial stage of the game between the two parties, the probability of the user choosing Action 1 is $p(0 < p < 1)$, and the probability of choosing Action 2 is $1 - p(0 < p < 1)$. We assume that the proportion of users who choose a particular behavior is equal to the probability that a single user chooses this particular behavior such that the probability that each individual of the entire user group chooses Action 1 and Action 2 is p and $1 - p$, respectively

(3) *Consumption Cost of Users*. Users requesting a test service from the edge server need to pay a certain test fee M, that is, the consumption cost of the user, which is irrelevant to the behavior strategy selected by the user. In addition, the cost is determined by the size of the test sample. Because the test process at the MEC requires a large amount of data communication, that is, a large amount of bandwidth is consumed. The larger the test sample size, the more bandwidth is consumed, resulting in higher data communication overhead. As a result, the more expensive it is for users to use the platform for testing

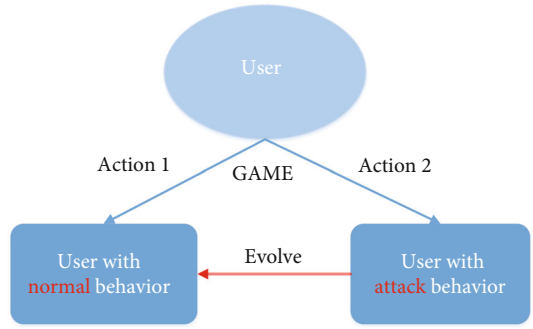

Figure 6: User behavior game model.

(4) *Revenue Generated by Users*. If the user adopts a proper test behavior, the edge server will return the correct test result to the user, which means that the AI system is successfully applied, and the user will indirectly obtain the revenue A from the outside. This external refers to the test task assignment subject. The user and the subject have a contractual relationship. If the user takes an offensive action, he will obtain improper benefits E from the attack

(5) *Penalties and Rewards for Users*. If the user takes an attack, the platform punishes the user by limiting the user's test service and bandwidth, which indirectly results in economic losses [37]. We suppose that the penalty term has a linear relationship with the user's test consumption cost M, that is, $D(M) = \alpha M$, where α is set as a penalty factor and greater than 0, which is more realistic. The severity of the penalty is determined by the cost of the test. If the user chooses a proper behavior, more bandwidth will be given, which will cause financial rewards R.

Based on the proposed model framework and related assumptions, the payoff matrix of two user populations is shown in Table 1. The connotation of each parameter in the payoff matrix is illustrated in Table 2.

(2) Equilibrium Analysis of the Evolutionary. Based on the above assumptions and the user's payoff matrix, referring to the ESS (see [50]), the total (average) expected returns $\bar{\mu}$ of the population user 1 that adopts the two strategies can be mathematically represented as

$$\bar{\mu} = p\mu_1 + (1 - p)\mu_2, \tag{1}$$

Table 1: Payoff matrix of users.

User 1 / User 2	Action 1: normal p	Action 2: attack $1-p$
Action 1: normal p	$A-M, A-M$	$A-M+R,$ $E-M-kD$
Action 2: attack $1-p$	$E-M-kD,$ $A-M+R$	$E-M-D,$ $E-M-D$

where μ_1 is the expected profits when user 1 selects normal behavior and μ_2 indicates the expected profits when user 1 performs an attack. The formulae for μ_1 and μ_2 are shown as

$$\begin{aligned}\mu_1 &= p(A-M)+(1-p)(A-M+R) = A-M+(1-p)R,\\ \mu_2 &= p(E-M-kD(M))+(1-p)(E-M-D(M))\\ &= E-M-pkD(M)-(1-p)D(M).\end{aligned} \tag{2}$$

Therefore, by bringing μ_1 and μ_2 into Eq. (1), the total expected returns can be obtained as

$$\begin{aligned}\bar{\mu} = {} & pA-M+p(1-p)R+(1-p)E-p(1-p)kD(M)\\ & -(1-p)^2D(M).\end{aligned} \tag{3}$$

Referring to the RD theorem (see [51]), the Replicator Dynamics equation of the proportion p for user 1 is

$$\begin{aligned}\frac{dp}{dt} &= F(p) = p(\mu_1-\bar{\mu})\\ &= p(1-p)[A+R-E+D(M)-p(R-kD(M)+D(M))].\end{aligned} \tag{4}$$

According to the conditions satisfied by the ESS, by setting Eq. (4) equal to zero, the possible equilibrium points of the RD system are $p_1^*=0$, $p_2^*=1$, and $p_3^*=(A+R-E+\alpha M)/(R-k\alpha M+\alpha M)$.

Since p_1^* and p_2^* are fixed and p_3^* is uncertain, to determine the final evolutionary stability strategy, p_3^* needs to be discussed on a case-by-case basis. More importantly, the punishment strategy is the target that needs to be discussed, because a reasonable punishment value is tried to be formulated so that all users can reach the state of taking nonattack behaviors as quickly as possible. It is worth noting that the profits from action are the norm for behavioral evolution. The following three cases need to be discussed.

Case 1. When $p_3^*<0$, $0<A+R<E-\alpha M$ can be deduced, and $0<\alpha<(E-A-R)/M$ can be further obtained. It can be known from these formulae that the user's rewards from being punished for adopting an attack behavior are greater than those for normal behavior, which means the punishment is lighter. Since $F'(p=0)<0$ and $F'(p=1)>0$, p_1^* is the user's evolutionary stable point.

Case 2. When $0\le p_3^*\le 1$, $A+R\ge E-\alpha M$ and $E-k\alpha M\ge A$ can be deduced, and $(E-A-R)/M\le\alpha\le(E-A)/kM$ can be further obtained. It can be known from these formulae that due to the moderate penalties, the user's rewards from an attack are smaller than that from normal behavior. Because $F'(p=0)\ge 0$ and $F'(p=1)\ge 0$, the evolution finally reaches a stable point at p_3^*.

Case 3. When $p_3^*>1$, $A>E-k\alpha M$ can be deduced, and $\alpha>(E-A)/kM$ can be further obtained. It can be known from this formula that the punishment imposed on the user who takes the attack behavior is heavier than that in Case 2. Users not only have small gains but also have a loss of costs. Because $F'(p=0)>0$ and $F'(p=1)<0$, the evolution finally reaches a stable point at p_2^*.

According to the above analysis, it can be seen that the punishment mechanism adopted by the test platform for the user behavior evolution game system to reach a stable point at p_2^* can achieve the desired effect, that is, all users will adopt normal test behavior. For the penalty factor α, it can more intuitively measure the penalties that the platform should apply. The penalty factor α has three ranges of values, corresponding to three situations. As α keeps increasing, the user's revenue from attacks continues to decrease. Due to the existence of this punishment mechanism, users with malicious behaviors will stop attacking through the process of evolutionary learning.

Due to the lack of prior knowledge of the value of the penalty factor, it is difficult to determine a reasonable and appropriate value for the penalty factor. Although it can be known from the above analysis that choosing a larger penalty factor is more conducive to the evolution of the entire game system to a stable point p_2^*, an excessive penalty factor causes users' expected returns to be too low. Therefore, in order to choose a more reasonable penalty factor and avoid blindly adopting an excessively large penalty factor, the self-adaptive penalty factor α_s is introduced instead of α in Eq. (4). The self-adaptive penalty factor can be formulated as

$$\alpha_s = [\ln(\beta(e-1)+1)]^{1/\beta}e^{1/c\beta}, \tag{5}$$

where $c(0<c<1)$ is a constant and $\beta=p(t_i)$, $i=1,2,3,\cdots,n$. Let $p(t)$ be the implicit function (solution) of Eq. (4). And $p(t_i)$ is the function value of the implicit function at $t=t_i$ $(i=1,2,3,\cdots,n)$. Therefore, α_s can be adaptively adjusted as the function value of the implicit function changes in the time domain T. Additionally, in Eq. (5), let $L=\ln(\beta(e-1)+1)$, referring to Taylor's theorem [52], L can also be expanded as

$$L=\sum_{x=1}^{n}\frac{(-1)^{x+1}}{x}[(e-1)\beta]^x+R_{n+1}[(e-1)\beta], \tag{6}$$

where $R_{n+1}[(e-1)\beta]$ is a Taylor's remainder which can be extremely close to zero when n is approaching to infinite. Therefore, Eq. (5) can be transformed as

Table 2: Parameters and connotations.

Parameters	Connotations
A	Indirect benefits from successful testing (AI system is successfully applied)
M	Consumption costs for requesting test services
$D(M)$	Economic loss caused by limiting test services or bandwidth, e.g., penalty cost and time cost due to incomplete test task contract period
R	Bandwidth rewards for nonattack action
E	Improper benefits from attack behavior
k	Probability of an accident caused by an attack

$$\alpha_s = \left[\sum_{x=1}^{n} \frac{(-1)^{x+1}}{x}[(e-1)\beta]^x\right]^{1/\beta} e^{1/c\beta} + o[R] = A^{1/\beta}e^{1/c\beta} + o[R], \tag{7}$$

where the harmonic series A based on Taylor's expansion is used to limit the value of α_s within a reasonable range. Therefore, by setting different n, the constraint ability of A can be adjusted to meet the needs of various evolutionary game models.

5. Data Privacy Protection

5.1. Description of Problem. While the era of big data brings great convenience to users, it also faces many challenges. For example, users face the risk of data privacy disclosure while enjoying the service [53]. The application of AI technology depends on big data, so the leakage of data privacy is also one of the significant challenges facing AI security. Homomorphic encryption technology is an effective way to protect the private data of users. In federal learning, users encrypt the key parameters of the data by using homomorphic encryption technology and send them to the cloud server to protect their private data. The cloud server uses only some algorithms to aggregate encrypted data packets to update the model parameters and then downloads the new model to each edge node [54]. Homomorphic encryption and deep learning both consume a lot of computing resources. Therefore, combining deep learning with homomorphic encryption technology will greatly increase the time for network training and inference [55]. Weighing the robustness of privacy protection and the efficiency of deep learning, the chaotic encryption technology is the best choice.

5.2. System Model. In this subsection, we propose a data privacy protection scheme based on chaotic encryption technology in IoT, as shown in Figure 7. The user layer is the source of the data that is collected in real-time by a large number of devices. To protect the data privacy of users, the user equipment is equipped with a chaotic encryption device, which encrypts the original data and sends it to the edge device layer. The MEC is equipped with corresponding chaotic decryption devices and has the right to decrypt user data. Cloud accepts encrypted data packets forwarded by MEC without authority to decrypt. The cloud can only aggregate the large number of packets used by some algorithm to update the parameters of the model and then drop the new model to the MEC. The data trained in the cloud is encrypted data, which effectively prevents the leakage of private data. The cloud can get a lot of data from different edge nodes, which can fully optimize the parameters of the training model, making its model efficient and accurate.

5.2.1. Data Transmission Scheme Based on Chaotic Encryption. We introduce two classical fractional-order chaotic models as chaotic transmitters of the user layer and chaotic receiver of the MEC layer, respectively. The fraction-order Liu's model is described as (see [56] and the references therein)

$$\begin{cases} \mathrm{D}_t^{\alpha} x_1(t) = -x_1(t) - x_2^2(t), \\ \mathrm{D}_t^{\alpha} x_2(t) = 2.5x_2(t) - 4x_1(t)x_3(t), \\ \mathrm{D}_t^{\alpha} x_3(t) = -5x_3(t) + 4x_1(t)x_2(t), \end{cases} \tag{8}$$

where α is derivative order. It has been shown that model (8) exhibits chaotic behavior when $\alpha > 0.916$. We suppose that the uncertainty Δf_i and the external disturbance d_i are bounded, that is, $|\Delta f_i| < \eta_i$, $|d_i| < \kappa_i$, where η_i and κ_i are positive constants $(i = 1, 2, 3)$. In this paper, the fraction-order Liu's model with the uncertainty Δf_i and the external disturbance d_i are described as

$$\begin{cases} \mathrm{D}_t^{\alpha} x_1(t) = -x_1(t) - x_2^2(t) + \Delta f_1(X, t) + d_1(t), \\ \mathrm{D}_t^{\alpha} x_2(t) = 2.5x_2(t) - 4x_1(t)x_3(t) + \Delta f_2(X, t) + d_2(t), \\ \mathrm{D}_t^{\alpha} x_3(t) = -5x_3(t) + 4x_1(t)x_2(t) + \Delta f_3(X, t) + d_3(t). \end{cases} \tag{9}$$

The fraction-order Newton-Leipnik's model is represented as follows (see [56]):

$$\begin{cases} \mathrm{D}_t^{\beta} y_1(t) = -0.4y_1(t) + y_2(t) - 10y_2(t)y_3(t), \\ \mathrm{D}_t^{\beta} y_2(t) = -y_1(t) - 0.4y_2(t) + 5y_1(t)y_3(t), \\ \mathrm{D}_t^{\beta} y_3(t) = 0.175y_3(t) - 5y_1(t)y_2(t), \end{cases} \tag{10}$$

where β is derivative order, a and b are model parameters, and $u_{ij}(i = 1, 2, 3, j = 1, 2.)$ are the controllers. The uncontrolled Newton-Leipnik's model (10) displays chaotic

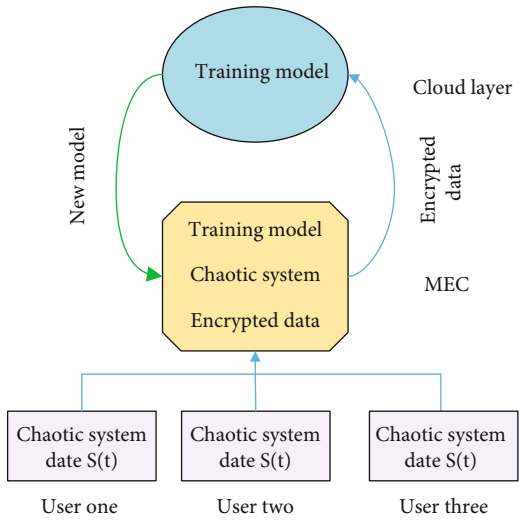

Figure 7: The constituent elements of green IoT.

behavior when $\beta > 0.9365$. We also suppose that the uncertainty Δg_i and the external disturbance $\bar{d}_i$ are bounded, that is, $|\Delta g_i| < \xi_i$, $|\bar{d}_i| < \mu_i$, where ξ_i and μ_i are positive constants $(i = 1, 2, 3)$. The fraction-order Newton-Leipnik's model with the uncertainty Δf_i and the external disturbance d_i are described as

$$\begin{cases} D_t^{\beta} y_1(t) = -0.4y_1(t) + y_2(t) - 10y_2(t)y_3(t) \\ +\Delta g_1(Y,t) + \bar{d}_1(t) + u_{11} + u_{12}, \\ D_t^{\beta} y_2(t) = -y_1(t) - 0.4y_2(t) + 5y_1(t)y_3(t) \\ +\Delta g_2(Y,t) + \bar{d}_2(t) + u_{21} + u_{22}, \\ D_t^{\beta} y_3(t) = 0.175y_3(t) - 5y_1(t)y_2(t) \\ +\Delta g_3(Y,t) + \bar{d}_3(t) + u_{31} + u_{32}, \end{cases} \tag{11}$$

Liu's model is used in the sending node to generate the chaotic signals $x_1(t)$, $x_2(t)$, and $x_3(t)$, and Newton-Leipnik's model is used in the sending node to generate the chaotic signals $y_1(t)$, $y_2(t)$, and $y_3(t)$.

5.2.2. Synchronization between the Drive Model and the Response Model. The errors model of drive model (9) and response model (11) are defined as

$$\begin{cases} e_1(t) = y_1(t) - x_1(t), \\ e_2(t) = y_2(t) - x_2(t), \\ e_3(t) = y_3(t) - x_3(t), \end{cases} \tag{12}$$

First, we rewrite drive model (9) and response model (11) into a vector form, and then

$$\begin{aligned} D_t^{\alpha} X(t) &= A_1 X(t) + B_1 F(X) + \Delta f(X) + M(t), \\ D_t^{\beta} Y(t) &= A_2 Y(t) + B_2 G(Y) + \Delta g(Y) + \bar{M}(t) + u(X,Y), \end{aligned} \tag{13}$$

where

$$A_1 = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 2.5 & 0 \\ 0 & 0 & -5 \end{pmatrix}, \quad A_2 = \begin{pmatrix} -0.4 & 1 & 0 \\ -1 & 0.4 & 0 \\ 0 & 0 & 0.175 \end{pmatrix},$$

$$B_1 = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & 4 \end{pmatrix}, \quad B_2 = \begin{pmatrix} -10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & -5 \end{pmatrix},$$

$$\begin{aligned} F(X) &= \left(x_2^2, x_1x_3, x_1x_2\right)^T, \\ G(Y) &= \left(y_2y_3, y_1y_3, y_1y_2\right)^T, \\ M(t) &= \left(d_1(t), d_2(t), d_3(t)\right)^T, \end{aligned} \tag{14}$$

and

$$\bar{M}(t) = \left(\bar{d}_1(t), \bar{d}_2(t), \bar{d}_3(t)\right)^T. \tag{15}$$

Assumption 1. It is assumed that the master model and slave model uncertainties $\Delta f(X)$ and $\Delta g(Y)$ and external disturbances $d_{mi}(t)$ and $d_{si}(t)$ are bounded by

$$\begin{aligned} &|\Delta f_i(X,t)| < \eta_i, \Delta g_i(Y,t)| < \xi_i, \\ &|d_i(t)| < \kappa_i, |\bar{d}_i(t)| < \mu_i, i = 1, 2, 3. \end{aligned} \tag{16}$$

Therefore, we have

$$\begin{aligned} &|\Delta f_i(X,t) - \Delta g_i(Y,t)| < q_i, \\ &|d_i(t) - \bar{d}_i(t)| < \delta_i, i = 1, 2, 3. \end{aligned} \tag{17}$$

Based on the above Assumption, we can get the following theorem.

Theorem 1. *Under Assumption 1 and the feedback controller $u(X,Y)$, the drive model (9) and the response model (11) are synchronization, where $u(X,Y) = u_1(X,Y) + u_2(X,Y)$, and*

$$\begin{aligned} u_1(X,Y) &= (u_{11}, u_{21}, u_{31})^T \\ &= -A_2X - B_2G(X) + D_t^{\beta}X - \Delta f(X) - M(t) \\ &\quad + B_2[X + G(X) - Y - G(Y)], \end{aligned} \tag{18}$$

$$\begin{aligned} u_2(X,Y) &= (u_{12}, u_{22}, u_{32})^T \\ &= \begin{pmatrix} e_1 + 4.6e_2 - \chi_2 e_2 - \left(\hat{q}_2 + \hat{\delta}_2 + k_2\right) \operatorname{sgn}(s_2) \\ (b+5)e_3 - \chi_3 e_3 - \left(\hat{q}_3 + \hat{\delta}_3 + k_3\right) \operatorname{sgn}(s_3) \\ -(10-a)e_1 - e_2 - \chi_1 e_1 - \left(\hat{q}_1 + \hat{\delta}_1 + k_1\right) \operatorname{sgn}(s_1) \end{pmatrix}. \end{aligned} \tag{19}$$

Proof. Taking the feedback controller $u_1(X, Y)$ of the response model (11) as (18), then the error model of model (9) and model (11) can be obtained in the following form:

$$\mathrm{D}_{\mathrm{t}}^{\beta} e = (A_2 - B_2)e + \Delta g(Y) + \bar{M}(t) - \Delta f(X) - M(t) + u_2(X, Y), \tag{20}$$

where $u_2(X, Y) = (u_{12}, u_{22}, u_{32})^T$. That is

$$\begin{cases} \mathrm{D}_{\mathrm{t}}^{\beta} e_1 = (10 - a)e_1 + e_2 + \Delta g_1(Y, t) + d_{s1}(t) - \Delta f_1(X, t) - d_{m1}(t) + u_{12}, \\ \mathrm{D}_{\mathrm{t}}^{\beta} e_2 = -e_1 - 4.6e_2 + \Delta g_2(Y, t) + d_{s2}(t) - \Delta f_2(X, t) - d_{m2}(t) + u_{22}, \\ \mathrm{D}_{\mathrm{t}}^{\beta} e_3 = (b + 5)e_3 + \Delta g_2(Y, t) + d_{s3}(t) - \Delta f_3(X, t) - d_{m3}(t) + u_{32}. \end{cases} \tag{21}$$

The sliding surface for the error model is presented by

$$s_i(t) = \left(\mathrm{D}_{\mathrm{t}}^{\beta} + \chi_i\right) \int_0^t e_i(\tau)\mathrm{d}\tau, \tag{22}$$

where χ_i are positive constants. Taking the first-order derivative of (22) yields

$$\dot{s}_i(t) = \mathrm{D}_{\mathrm{t}}^{\beta} e_i(t) + \chi_i e_i(t). \tag{23}$$

To estimate the unknown controller parameters, appropriate update laws are derived as follows:

$$\dot{\widehat{q}}_i = \gamma_i |s_i(t)|, \quad \widehat{\delta}_i = \omega_i |s_i(t)|, \tag{24}$$

where $\widehat{q}_i$ and $\widehat{\delta}_i$ are the actual values of q_i and δ_i, and γ_i and ω_i are positive constants. Consider the positive definite Lyapunov function $V(t) = V_1(t) + V_2(t) + V_3(t)$, where $V_i(t)$ is presented as follows:

$$V_i(t) = \frac{1}{2}\left[s_i^2 + \gamma_i^{-1}(q\wedge_i - q_i)^2 - \omega_i^{-1}(\delta\wedge_i - \delta_i)^2\right]. \tag{25}$$

Taking the derivative of both sides of (25) with respect to time, we have

$$\dot{V}_1(t) = s_1\dot{s}_1 - \gamma_1^{-1}(\widehat{q}_i - q_i)\dot{\widehat{q}}_1 - \omega_1^{-1}\left(\widehat{\delta}_i - \delta_i\right)\dot{\widehat{\delta}}_1. \tag{26}$$

Substituting (23) and (24) into (26), we get

$$\dot{V}_1(t) = s_1\left[\mathrm{D}_{\mathrm{t}}^{\beta} e_1(t) + \chi_i e_1(t)\right] + (\widehat{q}_1 - q_1)|s_1| + \left(\widehat{\delta}_1 - \delta_1\right)|s_1|. \tag{27}$$

Applying the control inputs $u_{12}(t)$ as

$$u_{12} = -(10 - a)e_1 - e_2 - \chi_1 e_1 - \left(\widehat{q}_1 + \widehat{\delta}_1 + k_1\right)\ \mathrm{sgn}\ (s_1). \tag{28}$$

Then, we can obtain that

$$\begin{aligned} \dot{V}_1(t) = s_1\big[&\Delta g_1(Y, t) + d_1(t) - \Delta f_1(X, t) - \bar{d}_1(t) \\ &- \left(\widehat{q}_1 + \widehat{\delta}_1 + k_1\right)\ \mathrm{sgn}\ (s_1)\big] + (\widehat{q}_1 - q_1)|s_1| + \left(\widehat{\delta}_1 - \delta_1\right)|s_1|. \end{aligned} \tag{29}$$

In view of Assumption 1, we yield

$$\begin{aligned} \dot{V}_1(t) &\le s_1\Big[\left|\Delta g_1(Y, t) - \Delta f_1(X, t)\right| + \left|\bar{d}_1(t) - d_1(t)\right| \\ &\quad - \left(\widehat{q} + \widehat{\delta} + k_1\right)\ \mathrm{sgn}\ (s_1)\Big] + (\widehat{q}_i - q_i)|s_1| \\ &\quad + \left(\widehat{\delta}_i - \delta_i\right)|s_1| \le -k_1|s_1| + (q_i - \widehat{q}_i)|s_1| \\ &\quad + \left(\delta_i - \widehat{\delta}_i\right)|s_1| + (\widehat{q}_i - q_i)|s_1| + \left(\widehat{\delta}_i - \delta_i\right)|s_1| \\ &\le -k_1|s_1|. \end{aligned} \tag{30}$$

Similarly, using the other controllers as follows:

$$\begin{cases} u_{22} = e_1 + 4.6e_2 - \chi_2 e_2 - \left(\widehat{q}_2 + \widehat{\delta}_2 + k_2\right)\ \mathrm{sgn}\ (s_2), \\ u_{32} = (b + 5)e_3 - \chi_3 e_3 - \left(\widehat{q}_3 + \widehat{\delta}_3 + k_3\right)\ \mathrm{sgn}\ (s_3). \end{cases} \tag{31}$$

Implementing the same calculation steps as above, we therefore obtain that

$$\dot{V}_2(t) \le -k_2|s_2|, \dot{V}_3(t) \le -k_3|s_3|. \tag{32}$$

By using the Lyapunov stability theory, we obtain the trajectories of the synchronization error (21) that will converge to the proposed sliding surface $s_i(t) = 0$.

Theorem 1 proves that the drive model (9) and the response model (11) are synchronized, which is a necessary condition for chaotic encryption. Note that model (9) and model (11) have different orders α and β, which is designed to enhance confidentiality. In particular, initial value conditions of model (9) and model (11), order α and order β, can be used as keys for chaotic encryption schemes. In addition, the effects of model (9) and model (11) on synchronization under uncertainties and external disturbances are considered. It means that when the model is subject to some internal or external random interference with uncertain factors, the synchronization between the two models can still be achieved stably. Our scheme not only improves the security performance of the chaotic encrypted transmission scheme but also enhances the stability and anti-interference as well as expands its applicability.

6. Experimental Results

In this section, we will perform simulation verification on the theoretical results. There are three main scenarios in our experimental part. First of all, we will verify the effectiveness of the antiattack scheme in the training phase against poisoning attacks and backdoor attacks, respectively. Secondly,

considering the adjustment of different penalty factors and the introduction of adaptive penalty factors, we will simulate how the penalty mechanism affects the choice of user behavior based on ESS to verify the achievement of defense in the inference against adversarial attacks. At last, we will show the chaotic behavior of the chaotic model (8), the synchronization between the driving model (10) and the response model (11), and a continuous data signal as an example to verify the validity of the system model encryption and decryption.

Next, we will analyze these defense issues in detail.

Scenario 1. *Experiments on proposed defense strategies against threats in AI training.*

In our proposed scheme, there are three core components, the EFU running on the edge gateway, the DSC and MTC running on the cloud server. We implement the edge gateway using a Raspberry Pi 3 Model B+ (1.4 GHz CPU and 1 GB RAM) running Raspbian Stretch with desktop. IoT devices can be connected to the edge gateway via a wired serial peripheral interface (SPI) with Modbus Protocol or wireless interface, including Bluetooth Low Energy (BLE) and WiFi. The edge storage is extended by a 1 TB network attached storage (NAS) also based on a Raspberry Pi 3 Model B+. The cloud server is equipped with NVIDIA TITAN X GPU (NVIDIA Digits Dev Box) and Intel i7-9700K CPU (3.8 GHz). Both the edge gateway and the cloud server are connected to the Internet. Edge gateway and edge storage are locally connected over a wireless LAN.

To verify the effectiveness of the antiattack scheme in the training phase against poisoning attacks, we adopted and segmented the Dogs vs. Cats Dataset [57]. The basic secure training set was composed of 10000 images to generate an active basic model. The remaining 15000 images form 30 incremental datasets for incremental training of the model. Then, poisoning samples were randomly added into the incremental datasets to generate toxic incremental datasets. The same method is used to make toxic test dataset, and the original test dataset is retained for comparison.

We compare the experimental results of the antiattack scheme proposed in this work with the nondefense scheme as shown in Figure 8. It can be seen that the model continuously learns from toxic samples and starts to adapt to poisoning attacks as the number of incremental iteration increases, making the test error on the toxic test set decrease from 67% to 4.1% under the nondefense situation. At the same time, the ability of the attacked model to recognize the original data decreased, and the test error on the original test set increased from 1.8% to 13.94%. This shows that the AI model was successfully attacked by poisoning attacks. In the experiments that adopted the antiattack scheme, the test error gradually decreased from 67%, which is for the learning of normal incremental data. But the final test error was only 18.4%. The test results on the original test set were not affected. This shows that our antiattack scheme can effectively prevent poisoning attacks. Figure 9 shows that the poisoning attack success rate on the nondefense model is 85.45%, while it is only 2.6% on the antiattack model, which also

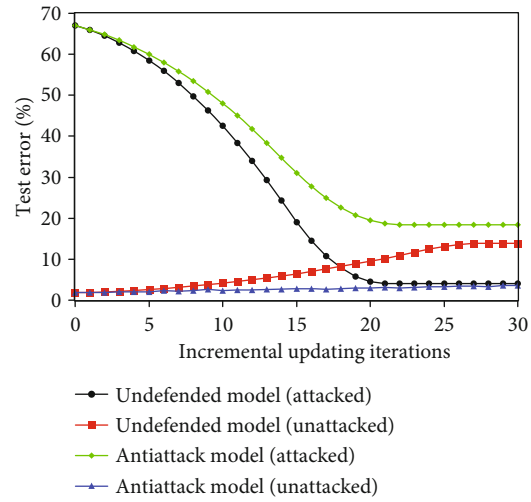

Figure 8: Test error of the undefended and antiattack models on the poisoning and original test datasets.

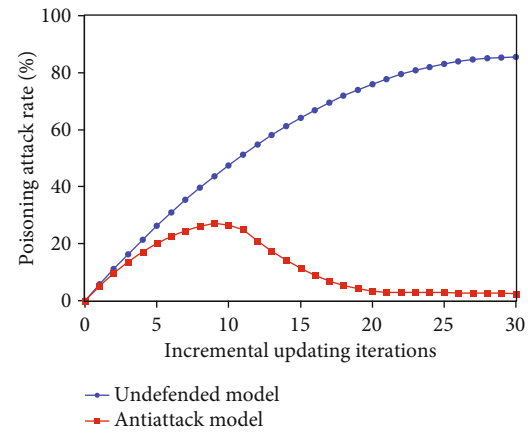

Figure 9: Poisoning attack rate of the undefended and antiattack models.

proves the conclusion. In the antiattack process, the attack success rate has increased slightly to 17.4% due to the limited cognition of the poison with fewer attack samples in the initial stage. However, the scheme's ability to capture the attack increases with the attack success rate decline to 2.5%.

To verify the effectiveness of the antiattack scheme against backdoor attacks, we randomly added backdoor samples to the segmented incremental datasets and the test dataset. The experimental results of the antiattack and nondefense schemes against backdoor attacks are shown in Figure 10. It can be seen that the test error of the nondefense model on the backdoor test set decreases from 56% to 5.2% as the number of incremental updating iteration increases. And the result of the antiattack model decreases from 56% to 26.5%. But the test errors of the two models on the original test set are not affected because the backdoor attack is more concealed. This shows that the undefended AI model was successfully attacked by the backdoor attack, and the scheme proposed in this work effectively defends the backdoor

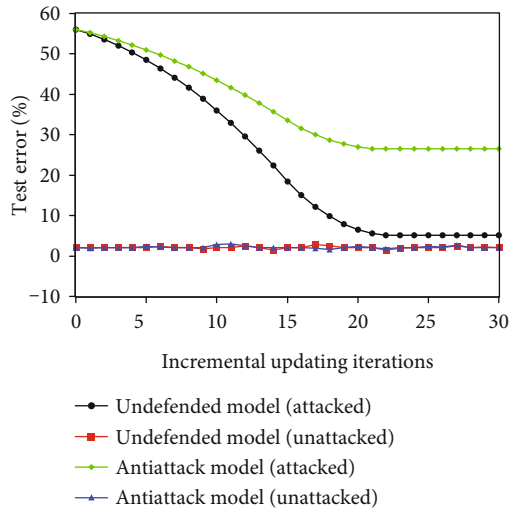

Figure 10: Test error of the undefended and antiattack models on the backdoor and original test datasets.

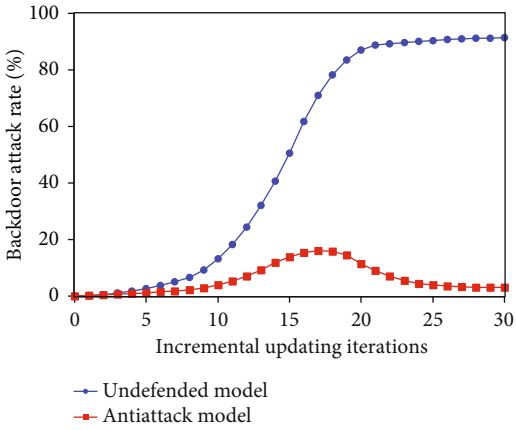

Figure 11: Backdoor attack rate of the undefended and antiattack models.

attack. The Figure 11 shows that the success rate of backdoor attacks on the nondefense model is 91.4%, while that of the antiattack model is only 3.8%, which also proves the conclusion. Compared with the changing of the poisoning attack success rate, it can be seen that the attack time of poisoning attacks is shorter. The attacked model begins to show symptoms of being attacked soon. The backdoor attack has a significant latency period (no obvious symptoms of being attacked was seen during the first 10 iterations), but the attack intensity after awakening is stronger.

Scenario 2. *Experiments on proposed defense strategies against threats in AI inference.*

In this scenario, we will carry out a more in-depth simulation analysis to verify the effectiveness and correctness of an adaptive evolutionary-based punishment mechanism against adversarial attacks in AI inference via ESS Analysis with diverse penalty factor α and adaptive penalty factor α_s.

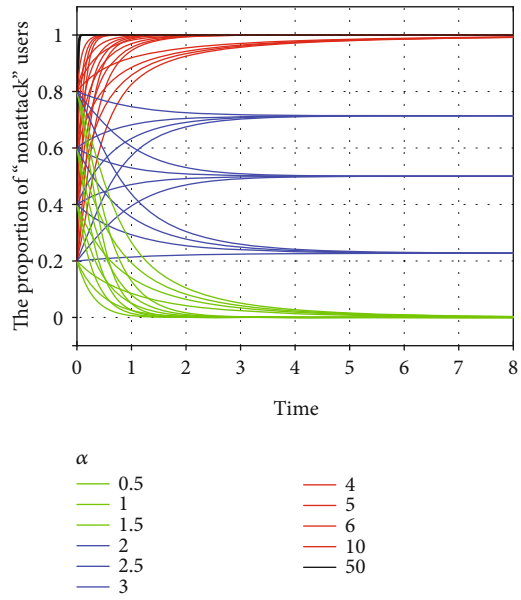

Figure 12: ESS Analysis with diverse values of penalty factor α.

In Eq. (4), we set $A = 3, R = 2, E = 10$ and $M = 3$ as fixed values and adjust the value of penalty factor α. The initial proportion of users who adopt a nonattack strategy is 0.2, 0.4, 0.6, and 0.8, respectively. Figure 12 simulates the effects of different penalty factors on the stability of user behavior evolution. α has different value ranges, that is, $0 < \alpha < 1.67$ in Case 1 (light green curves), $0 \le \alpha \le 3.89$ in Case 2 (dark blue curves), and $\alpha > 3.89$ in Case 3 (red curves and a black curve), corresponding to lighter, moderate, and heavier punishment, respectively. When the punishment is light ($\alpha = 0.5$), after a period of evolution, the light green curves will eventually stabilize at p_1^* such that all test users will eventually choose the attack behavior. With the increase of the penalty factor ($\alpha = 1, 1.5$), the evolution speed of the game is suppressed, and adversarial behavior can be combated to a certain extent. When moderate punishment is applied to users ($\alpha = 2, 2.5, 3.5$), as the penalty factor increases, although the evolution speed cannot be greatly improved, more users are ultimately willing to choose normal test behavior, and the proportion of such users has increased significantly. When greater penalties are imposed on users ($\alpha = 4, 5, 6, 10, 50$), nearly a hundred percent users choose proper conduct, and as α increases, the speed of approaching the stable point p_3^* becomes faster and faster.

When p_0 is small, a larger penalty factor is usually selected to bring the evolution curve closer to the stable point p_2^*, but excessive penalties will extremely reduce the user's overall expected return. Taking $p_0 = 0.2, 0.4$ as an example, Figure 13 simulates the effect of the introduction of the adaptive penalty factor on the evolutionary trend of the game model. In Eq. (4), we set the original penalty factor $\alpha_{s0} = 4$. When p_0 is small, according to Eq. (7), the adaptive penalty factor α_s keeps the model evolution oscillating in the early stage, so that the evolution curve reaches a reasonable and stable p value, that is, the model has a high

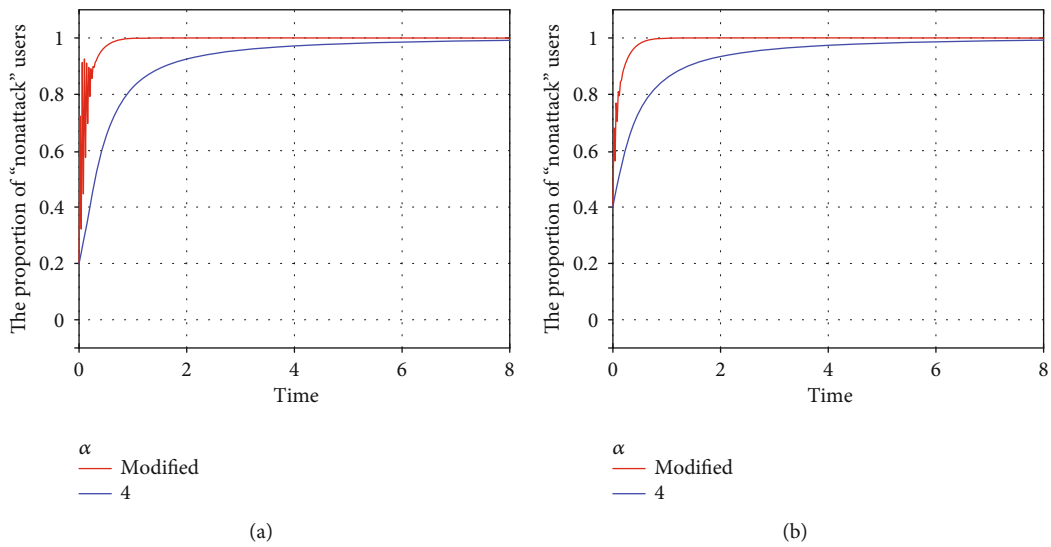

Figure 13: ESS Analysis with adaptive penalty factor α_s.

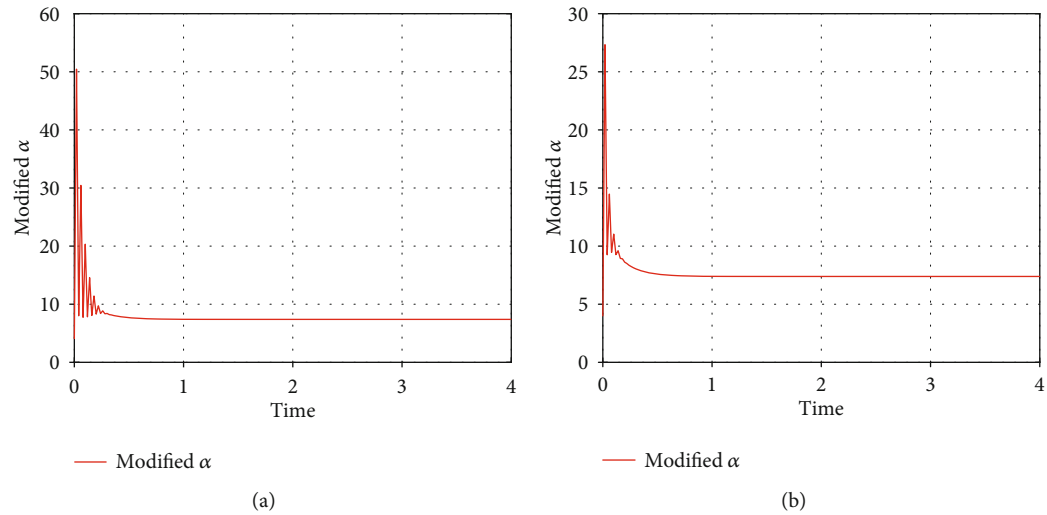

Figure 14: Adaptive penalty factor α_s changes over time.

percentage of "nonattack" users in an earlier stage. In addition, if the original penalty factor is set too high, although the adaptive penalty factor reduces the model's evolution speed, it greatly improves the user's overall expected return. The evolutionary process of α_s is shown in Figure 14. α_s hits a higher value instantly and then rapidly oscillates down to a reasonable and stable value. Therefore, the introduction of an adaptive penalty factor can automatically and more effectively guide the setting of penalty intensity.

Scenario 3. *Experiments on data privacy protection.*

In this scenario, we will introduce numerical simulations to verify the correctness and applicability of our conclusions. First, we need to show the chaotic behavior of model (8). The core principle of chaotic encryption technology is to rely on the nonlinear nature of chaotic models that are extremely sensitive to initial conditions. This will facilitate the encryption of the main signal and has a high level of confidentiality.

It has been shown that model (8) exhibits chaotic behavior when $\alpha = 0.95$, which is depicted in (a) and (b) of Figure 15. In fact, we can start from three planes xOy, xOy, and xOy, and space O-xyz shows the chaotic characteristics of model (8) in different dimensions. We only drew two pictures, and the chaotic behavior of model (10) can be seen [56].

In view of Theorem 1, we can get the essential conclusions that the drive model (9) and the response model (11) are synchronization. As shown in Figure 7, the user needs to encrypt the signal $S(t)$ and send it to the MEC. However, the chaotic model used for user encryption is different from the chaotic model used for MEC decryption. Therefore, the conclusions of Theorem 1 ensure that two different chaotic models can complete the encryption and decryption of the signal separately. To verify the effectiveness of the chaotic

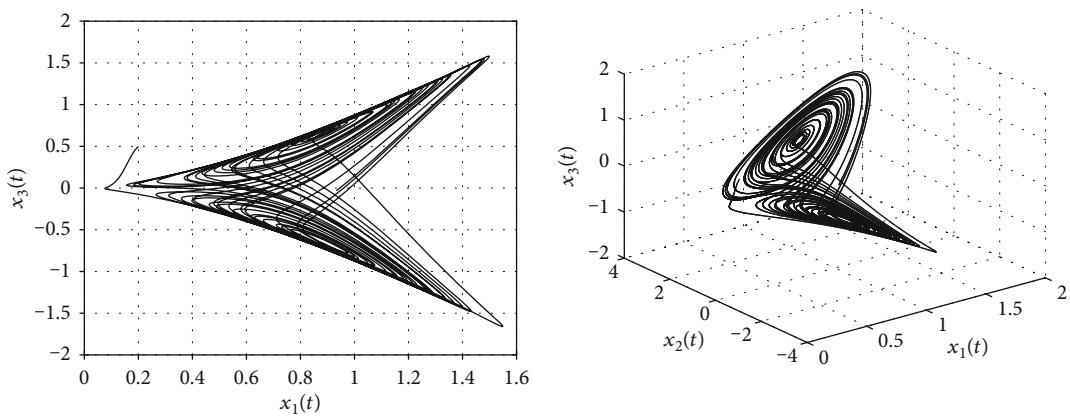

FIGURE 15: Chaotic behavior of model (8) with initial value conditions $(x_1(0), x_2(0), x_3(0) = (0.2, 0, 0.5))$ and order $\alpha = 0.95$.

encryption scheme proposed in this paper, as an example, we take the continuous function $S(t) = 3\cos(0.5\pi t)$ as the main signal. We use the classic n-shift encryption method to encrypt the main signal. The basic principle of the n-shift encryption is to use a nonlinear function $f[x(t), k(t)]$ (see [58, 59])

$$f(x,k) = \begin{cases} (x+k) + 2h & 2h \le (x+k) \le -h, \\ (x+k), & -h < (x+h) < h, \\ (x+k) - 2h & h \le (x+k) \le 2h, \end{cases} \tag{33}$$

operating on the signal $S(t)$ to get the encrypted signal $c(t)$, where the parameter h is chosen such that $S(t)$ and $k(t)$ lie within $(-h, h)$ and $k(t)$ as an essential encryption key is one of the three variables generated by the drive model (9). The decryption process also uses the function $f[c(t), -\widehat{k}(t)]$, which has the same analytical formula as (33), and $\widehat{k}(t)$ as an essential decryption key is one of the three v4ariables generated by the response model (9).

The main signal $S(t)$, encrypted signal $C(t)$, decryption signal $\widehat{S}(t)$, and the synchronization between the main signal $S(t)$ and decryption signal $\widehat{S}(t)$ are illustrated in Figures 16–19, respectively. The complete synchronization of the main signal and the decrypted signal effectively verify the correctness of the proposed encryption scheme. Data security and privacy protection have become one of the core issues in AI. We firmly believe that chaotic encryption technology will play an essential role in data security transmission and privacy protection.

7. Conclusions

The integration of MEC and AI is getting closer due to the agile response brought by MEC and the high performance of AI technology. But this edge-enabled AI poses some huge security threats. In this paper, we proposed corresponding defense solutions against three security threats in AI model training, AI model inference, and private data. More specifically, we mainly completed the following work:

(1) We developed a cloud-edge collaborative antiattack scheme, which fully takes the near-user advantages of MEC to defend against the security threats that AI faces in the training phase, including poisoning attacks and backdoor attacks. The scheme mainly includes the EFU, the DSC, and the MTC. The EFU leverages the computing power of IoT devices and deploys both a secure and a pending model on the edge to filter out suspicious threats based on their output differences. The DSC is used to identify suspicious threats for generating and updating detectors based on the hostile dataset. The MTC trains security models and pending models for deployment to the edge. The scheme designs a three-step model updating pipeline including the filtering stage, the pending stage, and the correction stage. It realizes a reliable incremental updating of AI by ensuring the data security generated in the training phase.

(2) We studied an edge-enhanced defense strategy based on adaptive traceability and punishment mechanism to effectively solve the security problem in the inference stage of the AI model. This strategy makes full use of the advantages of MEC, so that the traceability and punishment mechanism can get a quick response to efficiently perform adversarial defense. In addition, an adaptive punishment mechanism based on evolutionary game theory has been established with the aim of completely suppressing the attack behavior and radically eliminating attack intention by setting reasonable penalty factors. Compared with traditional evolutionary games, the introduction of the adaptive penalty factor can ignore the effects of various initial proportions. Since multiple training processes are avoided, compared with existing defense methods, the establishment of this strategy has a lower time consumption. Therefore, our strategy is built on the fast response edge layer with an ultrafast adjustment approach to achieve real-time autonomous defense.

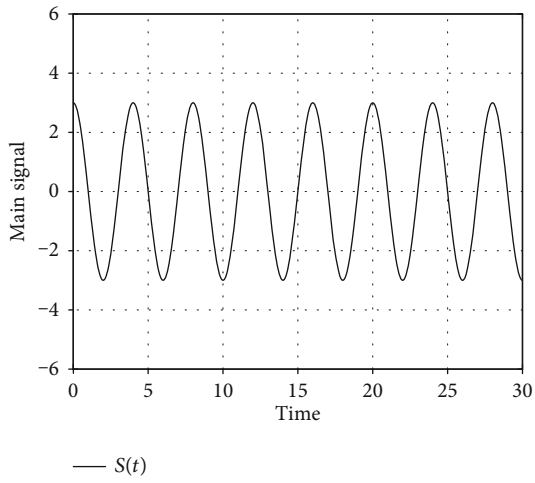

Figure 16: Main signal $S(t)$.

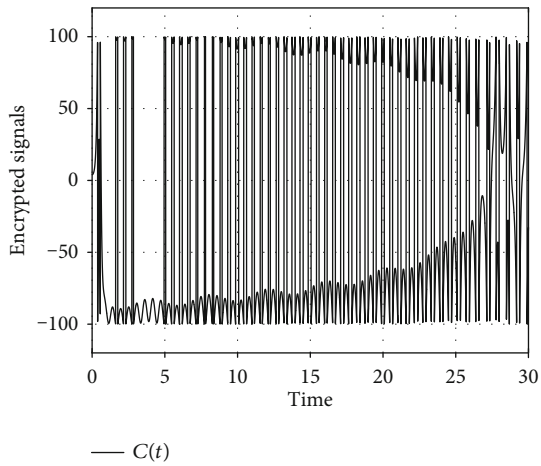

Figure 17: Encryption signal $C(t)$.

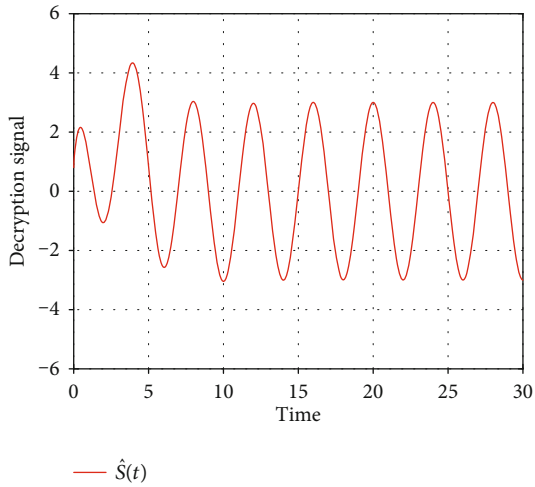

Figure 18: Decryption signal $\widehat{S}(t)$.

(3) We proposed a system model based on chaos encryption with the three-layer architecture of MEC to effectively guarantee the security and privacy of the data during the construction of AI models. In this model, users (IoT smart devices) deliver the encrypted data to the MEC node, which has the right to decrypt the data and perform training, so that AI can process data and respond quickly at the edge to realize edge intelligence. The MEC forwards the encrypted data to the cloud. The cloud has no right to decrypt which can only perform aggregation training on encrypted data packets and send new models to the edge nodes to continuously update the model of the edge nodes, making it more intelligent. Different from the traditional homomorphic encryption and the method based on differential privacy, in the chaotic encryption scheme, the synchronization between the startup system and the response system is the key to achieve the secure transmission of data. In this paper, we chose two fractional-order chaotic models with uncertainty and external disturbance as the driving system and response system to increase the safety performance. Theorem 1 proved that the two models can be completely synchronized. The experiment took a continuous data signal as an example to verify the correctness of the theorem conclusion and the feasibility of the data encryption scheme. Based on the advantages of low cost, high confidentiality, and wide availability of chaotic encryption, we believed that chaotic encryption will become the most effective technical solution to solve the data security and privacy problems of AI.

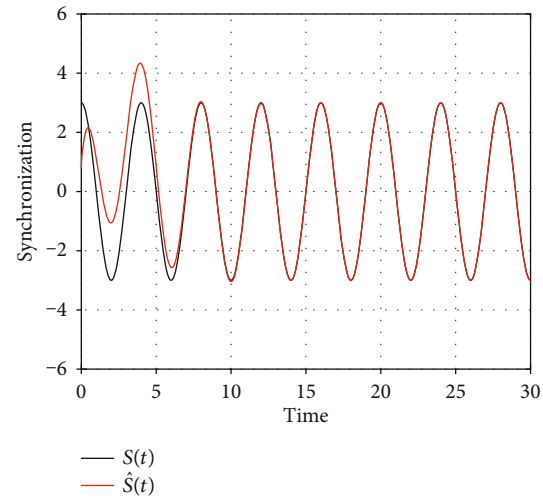

Figure 19: Synchronization between the main signal $S(t)$ and decryption signal $\widehat{S}(t)$.

Data Availability

The Dogs vs. Cats data used to support the findings of this study are available on Kaggle (see the hyperlink below). Hyperlink: Microsoft Research. (2017, February). Dogs vs. Cats, Version 1.0. Retrieved March 10, 2020 from https://www.kaggle.com/c/dogs-vs-cats.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning," in *Thirty-first AAAI conference on artificial intelligence*, San Francisco, CA, USA, 2017.

[4] R. Alp Güler, N. Neverova, and I. Kokkinos, "Densepose: dense human pose estimation in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7297–7306, Salt Lake City, UT, USA, 2018.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Las Vegas, NV, USA, 2016.

[6] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Advances in neural information processing systems*, pp. 5998–6008, Long Beach, CA, USA, 2017.

[7] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirectional transformers for language understanding," 2018, http://arxiv.org/abs/810.04805.

[8] V. Rausch, A. Hansen, E. Solowjow, C. Liu, E. Kreuzer, and J. K. Hedrick, "Learning a deep neural net policy for end-to-end control of autonomous vehicles," in *2017 American Control Conference (ACC)*, pp. 4914–4919, IEEE, Seattle, WA, USA, 2017.

[9] R. R. Murphy, *Introduction to AI Robotics*, MIT press, 2019.

[10] B. Heintz, A. Chandra, and R. K. Sitaraman, "Optimizing grouped aggregation in geo- distributed streaming analytics," in *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 133–144, Portland Oregon USA, 2015.

[11] Q. Pu, G. Ananthanarayanan, P. Bodik et al., "Low latency Geo-distributed data analytics," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 421–434, 2015.

[12] C. Gong, F. Lin, X. Gong, and Y. Lu, "Intelligent cooperative edge computing in the Internet of Things," *IEEE Internet of Things Journal*, 2020.

[13] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

[14] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: paving the last mile of artificial intelligence with edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

[15] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge Ai: intelligentizing MEC, caching and communication by federated learning," 2018, http://arxiv.org/abs/1809.07857.

[16] M. Song, K. Zhong, J. Zhang et al., "In-situ AI: towards autonomous and incremental deep learning for IoT systems," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 92–103, Vienna, Austria, 2018.

[17] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: the communication perspective," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[18] S. Thys, W. Van Ranst, and T. Goedem, "Fooling Automated Surveillance Cameras: Adversarial Patches to Attack Person Detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Long Beach, CA, USA, 2019.

[19] A. Liu, X. Liu, J. Fan et al., "Perceptual-sensitive GAN for generating adversarial patches," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1028–1035, 2019.

[20] N. Carlini and D. Wagner, "Audio adversarial examples: targeted attacks on speech-to-text," in *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 1–7, IEEE, San Francisco, CA, USA, 2018.

[21] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: poisoning attacks and countermeasures for regression learning," in *IEEE Symposium on Security and Privacy (S&P)*, San Francisco, CA, USA, 2018.

[22] A. Shafahi, W. R. Huang, M. Najibi et al., "Poison frogs! Targeted clean-label poisoning attacks on neural networks," in *Advances in Neural Information Processing Systems 2018*, vol. 31, pp. 6103–6113, Montréal, Canada, 2018.

[23] T. Gu, B. Dolan-Gavitt, and S. Garg, *Badnets: identifying vulnerabilities in the machine learning model supply chain*, NIPS MLSec Workshop, 2017.

[24] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," in *International Conference on Learning Representations(ICLR)*, Banff, AB, Canada, 2014.

[25] W. Zhou, X. Hou, Y. Chen et al., "Transferable adversarial perturbations," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 452–467, Munich, Germany, 2018.

[26] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, San Jose, CA, USA, 2017.

[27] R. Laishram and V. Phoha, "Curie: a method for protecting SVM classifier from poisoning attack," 2016, http://arxiv.org/abs/1606.01584.

[28] A. Paudice, L. Muoz-Gonzlez, A. Gyorgy, and E. C. Lupu, "Detection of adversarial training examples in poisoning attacks through anomaly detection," 2018, http://arxiv.org/abs/1802.03041.

[29] J. Steinhardt, P. W. W. Koh, and P. S. Liang, "Certified defenses for data poisoning attacks," *Advances in Neural Information Processing Systems*, vol. 30, pp. 3517–3529, 2017.

[30] B. Wang, Y. Yao, S. Shan et al., "Neural cleanse: identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, San Francisco, CA, USA, 2019.

[31] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: defending against backdooring attacks on deep neural networks," in *International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 273–294, Springer, Cham, 2018.

[32] B. Chen, W. Carvalho, N. Baracaldo et al., "Detecting backdoor attacks on deep neural networks by activation clustering," 2018, http://arxiv.org/abs/1811.03728.

[33] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," in *International Conference on Learning Representations(ICLR) 17*, Toulon, France, 2016.

[34] F. Tramr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: attacks and defenses," in *International Conference onLearning Representations(ICLR) 18*, Vancouver, Canada, 2017.

[35] A. S. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *Thirty-second AAAI conference on artificial intelligence*, New Orleans, LA, USA, 2018.

[36] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147, Dallas, TX, USA, 2017.

[37] Y. Sun, F. Lin, and N. Zhang, "A security mechanism based on evolutionary game in fog computing," *Saudi journal of biological sciences*, vol. 25, no. 2, pp. 237–241, 2018.

[38] C. Yin, J. Xi, R. Sun, and J. Wang, "Location privacy protection based on differential privacy strategy for big data in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3628–3636, 2017.

[39] A. Kim, Y. Song, M. Kim, K. Lee, and J. Cheon, "Logistic regression model training based on the approximate homomorphic encryption," *BMC Medical Genomics*, vol. 11, no. 4, 83 pages, 2018.

[40] Z. Huang, R. Hu, Y. Guo, E. Chan-Tin, and Y. Gong, "DP-ADMM: ADMM-based distributed learning with differential privacy," *EEE Transactions on Information Forensics and Security*, vol. 15, pp. 1002–1012, 2019.

[41] Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2017.

[42] T. Plantard, W. Susilo, and Z. Zhang, "Fully homomorphic encryption using hidden ideal lattice," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 12, pp. 2127–2137, 2013.

[43] W. Chen, "Nonlinear dynamics and chaos in a fractional-order financial system," *Chaos, Solitons and Fractals*, vol. 36, no. 5, pp. 1305–1314, 2008.

[44] H. Hui, C. Zhou, S. Xu, and F. Lin, "A novel secure data transmission scheme in industrial Internet of Things," *China Communications*, vol. 17, no. 1, 2020.

[45] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, http://arxiv.org/abs/1412.6572.

[46] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*, Saarbrücken, Germany, 2016.

[47] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 86–94, Honolulu, HI, USA, 2017.

[48] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," 2016, http://arxiv.org/abs/1605.07277.

[49] J. Weibull, *Evolutionary Game Theory*, MIT press, 1997.

[50] J. M. Smith, "The theory of games and the evolution of animal conflicts," *Journal of Theoretical Biology*, vol. 47, no. 1, pp. 209–221, 1974.

[51] P. Schuster and K. Sigmund, "Replicator dynamics," *Journal of Theoretical Biology*, vol. 100, no. 3, pp. 533–538, 1983.

[52] R. P. Kanwal and K. C. Liu, "A Taylor expansion approach for solving integral equations," *International Journal of Mathematical Education in Science and Technology*, vol. 20, no. 3, pp. 411–414, 1989.

[53] P. Asuquo, H. Cruickshank, J. Morley et al., "Security and privacy in location-based services for vehicular and mobile communications: an overview, challenges, and countermeasures," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4778–4802, 2018.

[54] B. McMahan and D. Ramage, "Federated learning: collaborative machine learning without centralized training data," *Google Research Blog*, vol. 3, 2017.

[55] T. Graepel, K. Lauter, and M. Naehrig, "ML Confidential: Machine Learning on Encrypted Data," in *International Conference on Information Security and Cryptology*, Springer, Berlin, Heidelberg, 2013.

[56] I. Petráš, *Fractional-Order Nonlinear Systems: Modeling, Analysis and Simulation*, Springer Science and Business Media, 2011.

[57] M. Research, *Dogs vs. Cats, Version 1.0*, 2017, October 2020, https://www.kaggle.com/c/dogs-vs-cats.

[58] B. Vaseghi, M. Pourmina, and S. Mobayen, "Secure communication in wireless sensor networks based on chaos synchronization using adaptive sliding mode control," *Nonlinear Dynamics*, vol. 89, no. 3, pp. 1689–1704, 2017.

[59] K. Fallahi, R. Raoufi, and H. Khoshbin, "An application of Chen system for secure chaotic communication based on extended Kalman filter and multi-shift cipher algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 13, no. 4, pp. 763–781, 2008.