Multi-Model Secure Redundant Storage for IoT Data Using Random Forests, Deep Neural Networks, and Adaptive Particle Swarm Optimization

Shenzhang Li, Zhenwei Geng*, Wenwei Su, Haoyu Ning, Xiaoping Zhao
Information Center, Yunnan Power Grid Co., Ltd., Kunming 650000, China
E-mail: zhenwei_geng@outlook.com
*Corresponding author

Keywords: random forest, deep neural network, adaptive particle swarm optimization, IoT data security, dynamic redundant storage

Received: April 25, 2025

With the increasing deployment of IoT systems, secure and efficient data storage has become a critical challenge. This paper proposes a multi-model secure redundant storage approach for IoT data by integrating Random Forest (RF), Deep Neural Network (DNN), and Adaptive Particle Swarm Optimization (APSO), refers to the complete proposed system that integrates Random Forest (RF) for feature extraction, Deep Neural Network (DNN) for anomaly detection and sensitivity classification, and Adaptive Particle Swarm Optimization (APSO) for dynamic storage strategy adjustment. The RF extracts key features from high-dimensional data, DNN detects and classifies anomalies, and APSO dynamically adjusts storage parameters for optimized redundancy. The model was evaluated on the SmartHomeIoTData-v1.0 dataset, comprising 1,000 devices and over 1,000,000 data entries across temperature, humidity, and status metrics. Compared to baseline models (KNN, SVM), our approach improves accuracy from 90% to 95%, increases storage resource utilization to 75%, and reduces data loss probability to 0.01%. These results demonstrate enhanced system security, efficiency, and responsiveness on resource-constrained devices.

Povzetek: Večmodelni pristop RF+DNN+APSO omogoča bolj kvalitetno varno redundantno shranjevanje IoT podatkov, saj presega KNN in SVM po točnosti, učinkovitosti shranjevanja ter zmanjšanju verjetnosti izgube podatkov.

1 Introduction

With the rapid development of IoT technology, various smart devices have sprung up, and the application scenarios of IoT are becoming increasingly diverse. From smart homes to industrial automation, from smart transportation to environmental monitoring, IoT technologies are increasingly integrated into both domestic and industrial applications [1, 2]. The popularization of this technology has brought great convenience and efficiency to society. However, with the continuous increase in the number of devices and the explosive growth of data volume, the data security problem of IoT has become increasingly prominent, gradually becoming an important bottleneck restricting its widespread application [3]. External network attacks are emerging in an endless stream, such as distributed denial of service (DDoS) attacks, man-in-the-middle attacks, and malware intrusions. These attacks not only lead to data leakage, tampering, or even loss, but also seriously interfere with the normal operation of IoT systems. For example, attackers may obtain users' privacy information by invading smart home systems, or cause production accidents by tampering with industrial production data [4]. In addition, due to the diversity and complexity of IoT devices, internal security management also faces severe challenges. Security vulnerabilities in devices, improper configuration, and weak user security awareness have further exacerbated the data security risks of IoT [5].

In this context, dynamic redundant storage technology is gradually regarded as an important means to ensure data security. Although traditional static redundant storage methods can prevent data loss to a certain extent, they are powerless in the face of dynamic changes in the IoT environment [6]. In contrast, dynamic redundant storage technology can flexibly adjust storage strategies based on factors such as data importance, access frequency, and real-time status of the system. It can not only effectively reduce the risk of data loss, but also improve the utilization efficiency of storage resources. The core of this technology is to achieve more efficient data protection and management through in-depth analysis and dynamic optimization of data

characteristics, thereby meeting the complex and diverse security needs of the IoT [7, 8].

In order to better solve the problem of secure storage of IoT data, this study proposes a dynamic redundant storage method that combines random forest algorithm, deep neural network model and adaptive particle swarm optimization algorithm. With its excellent classification and prediction capabilities, the random forest algorithm can accurately analyze massive IoT data, help identify key data and potential security threats, and provide a scientific basis for the formulation of redundant storage strategies. The deep neural network model can deeply explore the complex correlation characteristics between data and show great potential in security protection capabilities. As an efficient optimization tool, the adaptive particle swarm optimization algorithm can dynamically adjust storage parameters to further improve the flexibility and reliability of storage strategies. By organically combining these three technologies to form a collaborative intelligent architecture, it is expected to break through the limitations of traditional methods and provide a new solution for the secure storage of IoT data. This research can not only enhance the confidence of IoT users in data security, but also provide strong support for promoting the widespread application of IoT technology in various industries.

Although existing studies have explored the application of random forests, deep neural networks, and adaptive particle swarm optimization algorithms in different aspects, in the field of IoT data storage security, there is still a lack of research on organically combining these three and optimizing them for the dynamic, complex, and resource-constrained characteristics of the IoT. This study innovatively integrates the efficient feature analysis capabilities of random forests, the powerful nonlinear modeling capabilities of deep neural networks, and the dynamic optimization capabilities of adaptive particle swarm optimization algorithms to form a collaborative intelligent architecture to address the unique challenges faced by IoT data storage security. Compared with previous studies, this solution is not just a simple technical superposition, but also emphasizes the collaborative working mechanism between various technologies to achieve a more efficient, flexible, and reliable redundant storage strategy.

This study is guided by two primary hypotheses: (1) Multi-model feature integration using RF and DNN improves anomaly detection performance in high-dimensional IoT data compared to single models; and (2) APSO-driven dynamic parameter adjustment reduces data loss and improves resource utilization under resource-constrained environments. These hypotheses shape the design and evaluation criteria of the proposed system.

This study presents the following key contributions:

Integrated Multi-Model Architecture: We propose a novel system that integrates Random Forest (RF), Deep Neural Network (DNN), and Adaptive Particle Swarm Optimization (APSO) into a unified dynamic redundant storage framework for IoT data.

Feature-Driven Sensitivity Mapping: A fusion mechanism is designed to translate feature importance and anomaly detection outputs into actionable data sensitivity levels, which directly guide storage allocation.

Trust-Aware Redundancy Adjustment: We incorporate a data trust evaluation mechanism into the optimization pipeline, allowing dynamic adaptation of redundancy levels based on data source reliability.

Deployment-Oriented Design: The system is optimized for deployment on resource-constrained IoT devices using model compression, quantization, and reduced-complexity training without sacrificing predictive performance.

Comprehensive Evaluation: We conduct extensive experiments using a large-scale benchmark dataset, comparing against five baseline methods, and demonstrating superior accuracy (95%), recall (90%), and storage efficiency (75%).

2 Relevant theoretical basis

2.1 IoT data security

Internet of Things of Things, as an emerging technology paradigm, IoT is being widely used in various industries. IoT realizes the automatic collection, transmission and processing of information by connecting a large number of devices, but it also arouses widespread concern about data security. In the IoT environment, the number of devices is huge and the types are diverse, which leads to a significant increase in security risks in the data transmission process. Common threats include unauthorized access, data tampering, eavesdropping, and denial of service attacks [9, 10].

In order to protect the security of IoT data, the industry and academia have proposed a variety of technical means, such as encryption technology, access control mechanism and blockchain. Among them, encryption technology has become one of the core technologies of IoT security by ensuring the confidentiality and integrity of data during transmission and storage. In addition, distributed ledger technologies such as blockchain provide a reliable record-keeping method for the IoT, but their high computing and storage overheads limit their application in resource-constrained devices. On the other hand, the heterogeneity and distributed architecture of IoT devices pose challenges to traditional network security solutions. Existing centralized security strategies are difficult to adapt to the distributed characteristics of the IoT, which has prompted the development of security technologies based on artificial intelligence and machine learning. These technologies use data-driven methods to effectively detect and respond to complex network threats [11, 12].

In addition to the common IoT data security technical means mentioned above, some emerging IoT

storage solutions are also worth paying attention to recently. For example, blockchain-based systems have been widely studied in IoT data storage. Blockchain technology ensures the immutability and traceability of data through decentralized distributed ledgers. In the IoT environment, the data generated by the device can be directly recorded on the blockchain, and each data block contains the hash value of the previous data block, forming a chain structure. For example, in a small network consisting of 50 IoT devices, about 10,000 pieces of data generated every day are recorded on the blockchain. After 1 month of operation, the integrity of the data is effectively guaranteed, and no data tampering incidents have occurred. However, blockchain technology also faces some challenges, such as high computing and storage overhead. In the above small network, each node needs to consume an additional storage space of about 500MB for storing blockchain data. The consumption of computing resources causes the average response time of the device to be extended by about 10 seconds, which may become a bottleneck for applications on resource-constrained IoT devices. Another emerging solution is federated learning, which allows different participants to collaboratively train models without sharing original data. In the IoT data storage scenario, federated learning can enable multiple IoT devices to train and store related models locally using their own data, and then jointly optimize the storage strategy by exchanging model parameters. For example, in a smart home system, IoT devices from five different families use federated learning to collaboratively train storage models, which increases the storage resource utilization by about 15% compared with individual training. However, federated learning also has problems such as high communication overhead and the speed of model convergence being affected by differences in device performance. In the above smart home system, due to the different performance of each family device, the model convergence time is about 30% longer than the ideal situation. Compared with these recent IoT storage solutions, the dynamic redundant storage solution based on random forest, deep neural network and adaptive particle swarm optimization algorithm proposed in this study not only ensures data security and storage efficiency, but also pays more attention to the adaptability of resource-constrained devices and the optimization of computing overhead, and has unique advantages and application prospects. In the same resource-constrained smart home device environment, the storage resource utilization of this solution is about 20% higher than that of the blockchain-based solution, and the computing overhead is reduced by about 30%. When dealing with differences in device performance, the model converges faster and has higher stability.

In addition to traditional methods such as K-nearest neighbor (KNN) and support vector machine (SVM) compared in this study, recent IoT storage solutions such as blockchain systems and federated learning are also crucial in the field of IoT data storage. Blockchain ensures data integrity and immutability due to its decentralized and distributed nature. In the IoT environment, the data generated by each device is stored in blocks, which are connected to the previous block through cryptographic hashing, making malicious tampering extremely difficult to achieve. For example, in a smart grid system, blockchain can record a large amount of energy consumption data of smart meters. Tampering with the data requires changing all subsequent blocks, which is almost impossible in a mature blockchain network. However, blockchain storage requires high computing resources for mining and maintaining distributed ledgers, which poses challenges to resource-constrained IoT devices.

Federated learning provides a new way for IoT data storage, which allows multiple IoT devices or data owners to collaboratively train storage-optimized machine learning models without sharing original data. Taking multiple smart home optimized storage strategies as an example, each device trains the model locally and only exchanges model parameters, which not only protects data privacy but also integrates multi-data knowledge. However, federated learning has the problem of high communication overhead, and the differences in computing power, network connection and data distribution of devices will significantly affect the convergence speed of the model.

Compared with these recent solutions, the dynamic redundant storage solution proposed in this study, which combines random forest, deep neural network and adaptive particle swarm optimization, is more suitable for resource-constrained IoT devices, while maintaining high data security and storage efficiency, and seeking a balance between computing requirements and performance, providing a practical solution for IoT data storage in real scenarios.

Table 1: Comparative analysis of related IoT data storage techniques

Method	Strengths	Limitations	Suitability for IoT Storage
Blockchain [6, 12]	Immutability, decentralized trust	High computational/storage overhead	Limited for resource-constrained IoT
Federated Learning [13, 14]	Preserves privacy, supports distributed learning	Slow convergence, communication cost	Good for sensitive data, needs optimization
Proposed (RF+DNN+APSO)	High accuracy, dynamic optimization	Model complexity	Highly suitable with model compression

As shown in Table 1, compared to prior work, our method addresses key challenges in resource-limited IoT scenarios by combining multi-model learning, lightweight optimization, and secure storage. Previous approaches either focus on security (e.g., blockchain) or privacy (e.g., FL), but do not simultaneously optimize redundancy, performance, and adaptability.

2.2 Random forest model

Random forest is an ensemble learning method that builds multiple decision trees and combines their prediction results to form a model with high accuracy and stability. It is particularly suitable for processing high-dimensional data and nonlinear problems, which makes it an important tool in IoT data analysis and security. The basic principle of random forest is to generate multiple independent decision tree models by randomly selecting training samples and features. Each decision tree makes predictions separately, and finally outputs the comprehensive results by voting or weighting. The advantage of this method is that [15, 16] by introducing randomness, it can effectively reduce the risk of model overfitting and improve the generalization ability of unknown data. In the field of IoT security, random forest is widely used in anomaly detection and intrusion identification. For example, by learning network traffic features, random forest can accurately identify potential attack behaviors. In addition, the model can process large-scale, high-dimensional IoT data and provide feature importance analysis to provide guidance for the formulation of security policies. However, random forest also faces some limitations in practical applications, such as limited support for real-time data processing and the increase of model complexity with the increase of data scale [17, 18].

When using the random forest model to process IoT data, privacy issues cannot be ignored. During the training process, the random forest model may be exposed to a large amount of data containing user privacy information. In order to protect data privacy, this solution adopts a privacy protection technology based on homomorphic encryption, as in reference [19]. Before the data is input into the model, the sensitive data is homomorphically encrypted. For example, the user's home address and other information are encrypted using the Paillier homomorphic encryption algorithm. This allows the model to calculate on the encrypted data, and the results are consistent with those calculated on the plaintext data. In this way, even if the model is maliciously attacked or the data is illegally obtained, the attacker cannot directly obtain the original private data. At the same time, during the model training process, strict permission control is implemented on the access and use of the data. Only authorized administrator accounts and specific data analysis programs can operate on the data, further ensuring the privacy security of the data. In a simulated attack test, the attacker attempted to obtain the encrypted home address data by invading the model. After 10 hours of cracking attempts, no valid information was obtained, proving the effectiveness of this privacy protection technology.

2.3 Deep neural network model

Deep Neural Network Neural Networks, Deep neural networks (DNNs) are an extended form of artificial neural networks. By increasing the number and complexity of hidden layers, they have stronger feature expression capabilities. Deep neural networks have achieved remarkable results in various fields in recent years. Their excellent nonlinear modeling capabilities and ability to process complex data have made them a hot technology in IoT data security research. The architecture of a deep neural network consists of an input layer, multiple hidden layers, and an output layer. Through the weight connections between layers, the model can extract high-order features of the data layer by layer. In the field of IoT data security, deep neural networks are often used for malicious behavior detection, traffic classification, and decoding and analysis of encrypted data [13]. For example, by training a neural network model to identify normal and abnormal communication patterns, potential attack behaviors can be quickly detected [20]. Compared with traditional machine learning models, deep neural networks can automatically learn the potential features in the data without relying on complex feature engineering, which has significant advantages in IoT scenarios where data features are diverse and dynamically changing [21]. However, the application of deep neural networks also has some challenges, such as the need for a large amount of labeled data during the training process, high requirements for computing resources, and potential threats of adversarial sample attacks. In order to improve the applicability of deep neural networks, researchers have proposed a variety of improvement methods, such as reducing the reliance on labeled data through transfer learning, optimizing the model structure to reduce computational costs, and combining generative adversarial networks (GANs) to enhance the robustness of the model [22, 23]. In summary, the research on IoT data security requires the combination of multiple technical means, among which random forests and deep neural networks have shown unique advantages in structured data analysis and complex pattern recognition, respectively. In practical applications, a hybrid strategy combining multiple models may be an effective way to address IoT security challenges.

Privacy protection is also crucial for deep neural network models. Since deep neural networks require a large amount of data for training, some of the data may contain sensitive information of users. To prevent privacy leakage, this solution combines the idea of reference [14] and adopts a method that combines federated learning with differential privacy. Under the federated learning framework, each participant (such as different IoT devices or data owners) trains the model locally and only uploads the parameter update information of the model

without directly sharing the original data. At the same time, during the parameter update process, noise that complies with the differential privacy mechanism is added, such as Laplace noise. The scale parameter of the noise is reasonably set according to the data sensitivity and privacy budget. Even if the parameter update information is leaked, it is difficult to infer the privacy content of the original data from this information. In this way, it is ensured that the deep neural network model can make full use of multi-source data for training, and the privacy security of the data is effectively protected, which meets the strict requirements for data privacy protection in the IoT environment. In actual tests, the federated learning training process of 10 participants was monitored. When noise was added, the model accuracy only dropped by about 2%, but the privacy protection effect was significantly improved, and it successfully resisted multiple privacy inference attacks on parameter update information.

Table 2: Comparison of related works on secure IoT data storage

Method	Security Feature	Optimization Strategy	Device Support	Limitation
Blockchain-based Storage [6]	Tamper-resistance	Static Redundancy	Limited	High cost, slow convergence
Federated Learning [12]	Privacy-preserving	Collaborative Training	Moderate	High communication overhead
DRL-based Compression [17]	Compression + Encryption	Deep RL Optimization	High	Complex model, high resource usage
This Work (RF+DNN+APSO)	Anomaly-aware + Trust	Dynamic Particle Swarm Opt.	Low-to-High	Requires initial model calibration

As shown in Table 2, our method combines anomaly detection, trust modeling, and adaptive optimization, balancing security, performance, and deployability. Compared to existing methods, it offers improved adaptability with reduced computational burden.

3 Design of a secure dynamic redundant storage solution for IoT data based on multi-model fusion

3.1 Overall solution architecture

This chapter proposes a random forest-based Forest, RF), Deep neural networks (Deep Neural Networks, DNN and Adaptive Particle Swarm Optimization Particle Swarm Optimization, the proposed IoT data security dynamic redundant storage solution based on Random Forest Application Service (APSO) is designed to address the complex challenges faced by data security storage in the IoT environment. The solution comprehensively utilizes the efficient feature analysis capabilities of random forests, the deep pattern recognition capabilities of deep neural networks, and the dynamic optimization capabilities of adaptive particle swarm optimization algorithms to provide a flexible, efficient, and reliable redundant storage mechanism for IoT data. The overall architecture of the solution is divided into three main modules: data feature analysis, key data identification, and dynamic storage strategy optimization. The modules complement each other to form a coordinated and efficient system that can adapt to the complex and changing IoT data environment in real time [24, 25].

Among the many technologies that can be used for IoT data storage security, it is no accident that random forest, deep neural network and adaptive particle swarm optimization algorithm are selected for combination. IoT data is characterized by high dimensionality, diversity, dynamic changes, and extremely high requirements for storage security and efficiency. With its efficient feature analysis ability and good adaptability to high-dimensional data, random forest can quickly screen out key features from massive IoT data and reduce the complexity of subsequent processing. The powerful nonlinear modeling ability of deep neural network enables it to perform well in processing complex data patterns and anomaly detection, which is very suitable for the analysis needs of complex data in IoT environment. As an efficient optimization tool, adaptive particle swarm optimization algorithm can dynamically adjust storage parameters according to the real-time status of data and the operation of the system, and optimize storage strategy to meet the dynamic changes of IoT environment. Taking into account the characteristics of these technologies and the actual needs of IoT data storage security, combining them together to form an organic whole is expected to break through the limitations of traditional methods and provide a more effective solution for the secure storage of IoT data. The data feature analysis module uses the random forest algorithm to perform comprehensive feature extraction and classification of IoT data. IoT data is diverse and usually has high dimensionality and heterogeneity, such as sensor data, device logs, and user interaction data. Random forests build multiple decision tree models and use feature randomness and sample randomness to reduce the risk of overfitting of the model

and improve its generalization ability for different types of data. Through quantitative analysis of data importance (such as the Gini coefficient or information gain), random forests can effectively identify key features and provide a scientific basis for subsequent key data screening and storage strategies.

The key data identification module is based on deep neural networks (DNNs) to deeply mine and classify complex patterns in data. DNNs have powerful nonlinear modeling capabilities and can automatically learn deep features in data without relying on complex artificial feature engineering. Its multi-layer architecture captures potential patterns and trends from input data by extracting high-order features layer by layer. In this solution, deep neural networks can not only identify abnormal behaviors, but also quickly adjust models for dynamically changing data environments, thereby improving the accuracy and robustness of key data identification. In addition, combined with the adaptability of deep learning technology to large-scale data, the DNN module lays the foundation for intelligent decision-making in dynamic redundant storage of the Internet of Things.

Finally, the dynamic storage strategy optimization module uses the adaptive particle swarm optimization algorithm (APSO) to dynamically adjust the redundant storage parameters. This module establishes an optimization objective function based on the data importance and real-time system status analyzed in the first two parts, taking into account the risk of data loss, storage cost, and the utilization of redundant resources. APSO searches for the global optimal solution by dynamically adjusting the position and speed of the particle swarm, thereby realizing the dynamic adjustment of the storage strategy. For example, for critical data with high access frequency, its redundant storage nodes can be increased; for data with lower importance, redundant copies can be appropriately reduced to improve the utilization efficiency of storage resources.

Compared with the use of these technologies alone or in other combinations, the combination of random forest, deep neural network and adaptive particle swarm optimization algorithm used in this study has significant advantages. Random forest can quickly and accurately extract key features from high-dimensional and diverse IoT data, providing a solid foundation for subsequent analysis. Based on the features extracted by random forest, deep neural network can more accurately identify abnormal patterns and key information in data through powerful nonlinear modeling capabilities, which is difficult to achieve with traditional single models. The adaptive particle swarm optimization algorithm dynamically optimizes storage parameters based on the output results of the first two. This optimization method can adapt to changes in the IoT environment in real time and achieve efficient use of storage resources. For example, in the face of dynamic situations such as sudden growth in data traffic or equipment failure, this combination can respond quickly and adjust storage strategies to ensure secure storage and efficient access to data, while other combinations may not be able to respond so flexibly.

With the synergy of the three modules, this solution further enhances the adaptability and decision-making ability of the system through model fusion. Specifically, the output results of random forest and deep neural network are jointly analyzed according to the weighted fusion strategy to identify key data with higher accuracy and stability. The weight parameters of the fusion strategy can be dynamically adjusted through experimental optimization to ensure the best performance in different scenarios. Finally, the system realizes dynamic and secure redundant storage of IoT data, which not only improves the security and reliability of data storage, but also significantly reduces storage resource overhead.

Compared with some previous methods that use similar hybrid technologies, the scheme of this study is unique. For example, some previous studies may simply apply these technologies in sequence without fully considering their mutual influence and synergy. This scheme uses a carefully designed model fusion strategy to enable random forests, deep neural networks, and adaptive particle swarm optimization algorithms to cooperate and complement each other in the entire system. In the feature extraction stage, random forests not only provide effective features for deep neural networks, but also provide decision-making basis for adaptive particle swarm optimization algorithms when adjusting storage parameters. In the process of anomaly detection and storage strategy optimization, the output of deep neural networks is combined with the results of random forests to jointly guide the search direction of adaptive particle swarm optimization algorithms. This close coordination mechanism has not been fully reflected in previous studies, which makes this scheme have higher performance and adaptability in the actual application of IoT data storage security.

The proposed system is composed of three interdependent modules: (1) the Random Forest (RF) feature extraction module, (2) the Deep Neural Network (DNN) anomaly detection module, and (3) the Adaptive Particle Swarm Optimization (APSO) dynamic storage optimizer. The system workflow starts with incoming IoT data being fed into the RF module, which extracts and ranks features based on importance. These features are passed to the DNN module, which identifies and classifies anomalous or sensitive data points. The classification results and the feature importance scores are then forwarded to the APSO module, which dynamically computes optimal storage parameters based on current system load and security requirements. A real-time feedback loop allows APSO outputs to influence RF thresholds and DNN sensitivity tuning.

3.2 Data feature analysis module

The data feature analysis module extracts and classifies the IoT data through the random forest model. Assume that the feature matrix of the IoT data is $X = \{x_1, x_2, \ldots, x_n\}$, the corresponding label is $Y = \{y_1, y_2, \ldots, y_n\}$. The random forest model constructs k decision trees $T_1, T_2, \ldots, T_k$. Each tree is trained based on different feature subsets and training data, and its output is obtained by voting to obtain the final classification result, as shown in Formula 1. The importance of features is measured by the Gini coefficient. The specific calculation is as shown in Formula 2.

$$\hat{y} = \text{majority\_vote}\{T_1(x), T_2(x), \ldots, T_k(x)\} \quad (1)$$

$$G(X_j) = \sum_{i=1}^{c} p_i(1 - p_i) \quad (2)$$

In Formula 2, X_j is feature j, p_i is the probability distribution of category i under this feature. By sorting all features, the most important features can be selected for redundant storage decision.

Compared with other common feature selection methods, such as principal component analysis (PCA) and recursive feature elimination (RFE), random forest has unique advantages in the IoT data storage security scenario of this study. PCA mainly projects high-dimensional data into low-dimensional space through linear transformation to achieve the purpose of dimensionality reduction, but it may lose some important nonlinear information. In IoT data, many key features are often hidden in complex nonlinear relationships. Random forest can better capture the nonlinear characteristics in the data by constructing multiple decision trees and using random feature selection and random sample sampling, and has stronger adaptability to different types of data. Although RFE can select key features by recursively eliminating unimportant features, it requires a pre-defined evaluation index and has high computational complexity. Random forest does not need to pre-define complex evaluation indicators. Its decision tree-based structure can intuitively give the importance ranking of features and has higher computational efficiency. In addition, when processing large-scale, high-dimensional IoT data, random forest can better deal with data noise and missing value problems, ensure the accuracy and stability of feature selection, and thus provide a more reliable basis for subsequent key data identification and storage strategy formulation

Table 3 lists the top 10 features ranked by their Gini importance scores computed from the Random Forest model. The most informative features include device temperature fluctuations, frequency of status changes, and recent fault alarms.

Table 3: The top 10 features

Rank	Feature Name	Gini Importance Score
1	Temperature_STD	0.182
2	Status_Change_Frequency	0.165
3	Fault_Alarm_Recent	0.142
4	Humidity_STD	0.123
5	Power_On_Duration	0.095
6	Signal_Loss_Incidents	0.089
7	Temperature_Mean	0.067
8	Energy_Consumption_Peak	0.055
9	User_Interaction_Frequency	0.046
10	Maintenance_History_Score	0.036

These features significantly influenced both anomaly detection and storage priority decisions. Features with high variance and high predictive value are prioritized for redundancy under the APSO policy.

3.3 Key data identification module

In the IoT data security dynamic redundant storage solution, the identification of key data is a key step to achieve accurate storage and resource optimization. This module uses deep neural networks (Deep Neural Networks, The nonlinear modeling capabilities of Deep Neural Networks (DNNs) are used to conduct in-depth analysis and classification of key data in IoT data, thus providing a reliable foundation for subsequent optimization of dynamic storage strategies. With their powerful feature extraction capabilities and ability to recognize complex patterns, deep neural networks can adapt to the high-dimensional, dynamic and heterogeneous characteristics of data in IoT environments, effectively improving the recognition efficiency and accuracy of key data.

3.3.1 Structural design of deep neural network models

A deep neural network consists of an input layer, multiple hidden layers, and an output layer. It can extract high-order features of data layer by layer and realize the

mapping from input features to output classification. Assume that the input data is a feature vector $x \in \square^n$, represents the data characteristics of a device or scene in the Internet of Things, such as traffic characteristics, time series patterns, or user interaction information. The weight matrix and bias term of the model are expressed as $\{W^{(l)}, b^{(l)}\}_{l=1}^{L}$, where L is the number of network layers. The output of the hidden layer can be calculated by formula 3.

$$h^{(l)} = f(W^{(l)}h^{(l-1)} + b^{(l)}), \quad l = 1, 2, \ldots, L \tag{3}$$

In formula 3, $f(\cdot)$ is the activation function. Common activation functions include ReLU (Rectified Linear Unit), Sigmoid and Tanh Etc. ReLU The activation function is widely used in deep learning tasks due to its high computational efficiency and less gradient vanishing problem. Its expression is Formula 4.

$$f(x) = \max(0, x) \tag{4}$$

In the IoT environment, anomaly detection is crucial to ensure data security. Compared with traditional anomaly detection methods such as support vector machines (SVM), autoencoders, and long short-term memory networks (LSTM), the choice of deep neural networks in this study is fully reasonable. SVM performs well when processing linearly separable data or data converted to linearly separable data through kernel functions, but its performance may be limited for complex and changeable nonlinear data patterns in the IoT. Autoencoders mainly detect anomalies by reconstructing data, and work well for scenarios with relatively regular data distribution, but their generalization ability may be insufficient when faced with the diversity and dynamic changes of IoT data. Although LSTM networks have advantages in processing time series data, their structure is relatively complex and the computational cost is high for feature extraction and anomaly detection of non-time series IoT data. Through multi-layer neuron structures and powerful nonlinear activation functions, deep neural networks can automatically learn deep-level features in data and have stronger adaptability and generalization ability for various complex data patterns in the IoT environment. In this study, deep neural networks can effectively further mine key information from the features extracted by random forests, accurately identify abnormal data, and provide a reliable basis for subsequent storage strategy optimization, which is difficult to achieve with other traditional methods.

The output layer adopts Softmax Function, which converts the final output of the network into a probability distribution, as shown in Formula 5. Among them, Softmax function is defined as Formula 6.

$$\hat{y} = \text{softmax}(W^{(L)}h^{(L-1)} + b^{(L)}) \tag{5}$$

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{C}\exp(z_j)}, \quad i = 1, 2, \ldots, C \tag{6}$$

C represents the number of categories. z_i is the prediction score for the ith category.

Table 4: Summarizes the architecture and training hyperparameters of the DNN model.

Layer Index	Type	Units	Activation	Dropout
Input	Dense	500	-	-
Hidden 1	Dense	256	ReLU	0.2
Hidden 2	Dense	128	ReLU	0.2
Hidden 3	Dense	64	ReLU	0.2
Output	Dense	5	Softmax	-

As shown in Table 4, the model is optimized using the Adam optimizer with an initial learning rate of 0.001 and an exponential decay schedule (decay rate = 0.96, decay steps = 1,000). Training was conducted over 50 epochs with a batch size of 64. Each epoch required approximately 3.8 minutes on a mid-range GPU (NVIDIA RTX 3060), and convergence was typically reached around epoch 30.

3.3.2 Model optimization and loss function

The training goal of a deep neural network is to optimize the model parameters so that the output prediction value $\hat{y}$ With the actual label y to this end, the cross-entropy loss function is used as the optimization objective, as shown in Formula 7.

$$\text{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}) \tag{7}$$

In Formula 7, N represents the number of samples, C is the number of categories, y_{ij} It is a sample i. In category j the true label on 0 or 1), $\hat{y}_{ij}$ is the network's predicted probability for this category. By minimizing this loss function, the classification performance of the model can be gradually improved.

3.3.3 Data preprocessing and model training

In practical applications, IoT data often have problems such as noise, missing values, and uneven distribution. In order to improve the robustness and generalization ability of the model, data preprocessing is required, including data cleaning, standardization,

normalization, and data enhancement. Assume that the data feature vector is $x=\{x_1,x_2,\ldots,x_n\}$, the normalization can be expressed as Formula 8.

$$x_{i'}=\frac{x_i-\mu}{\sigma} \tag{8}$$

In Formula 8, μ and σ are the feature mean and standard deviation respectively.

After data processing is completed, the data is divided into training set, validation set and test set for model training, hyperparameter tuning and performance evaluation. In order to avoid overfitting of the model, regularization techniques (such as L2 Regularization) and Dropout Mechanism. L2, the goal of regularization is to minimize the loss function and the sum of squared parameters, as shown in Formula 9.

$$\mathrm{L}_{\mathrm{reg}}=\mathrm{L}+\lambda\sum_{i=1}^{n}w_i^2 \tag{9}$$

In Formula 9, λ is the regularization strength hyperparameter, w_i is the network weight.

The data was split into 70% training, 15% validation, and 15% testing sets. Training was conducted over 50 epochs with a batch size of 64. Early stopping was enabled with a patience of 7 epochs based on validation loss. A learning rate decay schedule was used to reduce overfitting. These settings were selected based on grid search optimization.

In the process of deep neural network (DNN) training, computational overhead is an important issue that needs to be paid attention to, especially for resource-constrained IoT devices. The training of DNN models involves a large number of matrix operations and parameter updates, and its computational complexity is closely related to the number of layers, the number of neurons, and the amount of training data. Taking the DNN model used in this study as an example, the model contains 5 hidden layers, and the number of neurons in each layer gradually decreases from 100 to 500. During the training phase, each forward propagation and backpropagation requires a large number of multiplication and addition operations. In order to evaluate its computational overhead, IoT devices of different sizes were tested in an experimental environment. On devices with lower configurations (such as devices with 1 GB memory and a single-core processor), it takes up to 30 minutes to complete a full training iteration, which may affect the real-time performance of the system. In order to reduce the computational overhead of DNN training, this solution uses some optimization technologies, such as model compression, which removes about 20% of redundant connections through pruning technology, and quantization technology, which converts parameter data types from 32-bit floating point numbers to 16-bit floating point numbers, thereby reducing the amount of calculation; at the same time, batch training is adopted to load 10,000 data that were originally loaded at one time in batches, with 1,000 data loaded each time, reducing memory usage and computing resource consumption. After these optimizations, the time to complete a training iteration on the same configuration device is shortened to 10 minutes, which to a certain extent alleviates the pressure of DNN training on the computing resources of IoT devices and improves the feasibility of the system in a resource-constrained environment.

While LSTM networks are well-suited for temporal data, the dataset used in this study is primarily composed of snapshot-based and event-driven features rather than continuous time series. An ablation study was performed comparing the current DNN architecture to a 2-layer LSTM with 128 units per layer. The LSTM model achieved 92.5% accuracy (vs. 95% for DNN) and took 2.3× longer per epoch to train. These results justify the choice of DNN for our resource-constrained IoT scenario.

In this study, the terms “critical data,” “anomalous data,” and “high-sensitivity data” are used consistently to denote data points that are identified as security-relevant or failure-prone. The DNN module classifies data into normal and anomalous categories, which are further refined by the fusion process using RF importance scores to assign a sensitivity level (high, medium, or low). Thus, high-sensitivity data correspond to anomalous or critical events with high feature importance, which then trigger increased redundancy in storage.

3.4 Dynamic storage strategy optimization module

The dynamic storage strategy optimization module is an important part of realizing the core functions of the IoT data security redundant storage solution. Particle Swarm Optimization, The APSO algorithm dynamically adjusts storage parameters according to the importance of data and system status to achieve the optimal balance between storage efficiency and security. The particle swarm optimization algorithm is an optimization method based on swarm intelligence. It has the characteristics of low computational complexity, simple implementation and strong global search capability. It is particularly suitable for solving dynamic storage optimization problems.

3.4.1 The basic principle of particle swarm optimization algorithm

The particle swarm optimization algorithm simulates the flying behavior of particles in the search space to find the optimal solution of the objective function. In this module, each particle represents a possible storage strategy, and its position $x_i(t)$. Corresponding to the current strategy parameters, speed $v_i(t)$ Indicates the moving direction and amplitude of the particle in the search space. The particle passes through the individual historical optimal position p_i and the group global optimal position. g to adjust the speed and

position, the iterative formulas are shown in Formula 10 and Formula 11.

$$v_i(t+1) = \omega v_i(t) + c_1 r_1[p_i - x_i(t)] + c_2 r_2[g - x_i(t)] \quad (10)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (11)$$

In Formula 10 and Formula 11, ω Inertia weight controls the particle's dependence on the current velocity. ω It helps in global search, while smaller ω It is conducive to local convergence; c_1, c_2 is the learning factor, which measures the particle's ability to learn its own historical optimal position and the group's global optimal position; r_1, r_2 For the value in [0, 1] The random number is used to introduce randomness and avoid falling into the local optimum.

3.4.2 Introduction of adaptive strategy

In order to adapt to the dynamic changes of data characteristics and system status in the IoT environment, this module adopts an adaptive strategy to adjust the inertia weight. ω and learning factor c_1, c_2 Make dynamic adjustments. The specific formula is Formula 12.

$$\omega = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{\text{iter}_{\max}} \cdot \text{iter} \quad (12)$$

In Formula 12, $\omega_{\max}$ and $\omega_{\min}$ are the initial value and minimum value of the inertia weight, iter is the current iteration number, $\text{iter}_{\max}$ This strategy enables the algorithm to have a strong global search capability in the early stage, and gradually enhance the local search capability in the later stage, thereby improving the optimization effect of the algorithm.

3.4.3 Objective function design

The goal of dynamic storage strategy optimization is to minimize the comprehensive cost function J, taking storage costs into account C_{storage} And the data loss probability P_{loss}, specifically as Formula 13.

$$J = \alpha \sum_{i=1}^{N} C_{\text{storage}}(x_i) + \beta \sum_{i=1}^{N} P_{\text{loss}}(x_i) \quad (13)$$

Among them, N is the amount of data, x_i For particles i The location (i.e. storage parameters) of α and β is the weight coefficient, which is used to balance storage cost and data security; $C_{\text{storage}}(x_i)$ Indicates storage parameters x_i Storage costs under $P_{\text{loss}}(x_i)$ Indicates the probability of data loss, which decreases as storage redundancy increases.

Optimize storage parameters by dynamically adjusting particle positions x_i, which can effectively reduce the overall cost J. After the optimization is completed, the dynamic storage strategy optimization module outputs the optimal storage parameters x, including information such as storage device selection, number of data copies, and storage location. This parameter will guide the storage system to dynamically and redundantly store IoT data, providing high security for key data while reducing storage resource overhead.

The adaptive particle swarm optimization algorithm (APSO) also has a computational overhead problem in the process of dynamically adjusting storage parameters. The APSO algorithm needs to calculate the speed and position updates of particles in each iteration, which involves multiplication and addition operations of multiple parameters. For resource-constrained IoT devices, frequent calculations may lead to device performance degradation. In order to analyze the computational overhead of the APSO algorithm, the time and resource usage of running the APSO algorithm on IoT devices of different sizes were monitored in the experiment. When the number of particles is set to 50 and the search space is 10 dimensions, running an APSO algorithm iteration on an ordinary IoT device consumes an average of 10MB of memory and 5 seconds. As the number of particles increases to 100 and the search space increases to 20 dimensions, the computational overhead increases significantly, the memory consumption rises to 20MB, and the time is extended to 10 seconds. In order to optimize the computational efficiency of the APSO algorithm on IoT devices, this scheme improves the algorithm. Under the premise of ensuring the algorithm's search performance, unnecessary calculation steps are reduced. For example, when updating the particle position, a simplified update formula is used. By introducing some heuristic rules, the approximate optimal solution is quickly found, thereby reducing the amount of calculation. In addition, the algorithm parameters are reasonably set, such as linearly reducing the inertia weight from 0.9 to 0.4, and setting the learning factor to 1.5 and 2.0 respectively, so that it can run with low computational overhead in different IoT device environments. After the improvement, in the same complex scenario, the memory consumption is reduced to 15MB and the running time is shortened to 7 seconds, which improves the applicability of the algorithm in resource-constrained environments.

The computational complexity of APSO was evaluated in terms of floating-point operations per second (FLOPs). For 50 particles and a 10-dimensional parameter space, each iteration required approximately 10^6 FLOPs. Compared to a static heuristic method, APSO introduced a 20% increase in runtime (7 s vs. 5.8 s) but reduced data loss by 66% [26], as shown in Table 5.

Table 5: Summarizes parameter adjustments across three scenarios

Scenario	Redundancy (r)	Storage Cost (¥)	Data Loss (%)
Low Load (Static)	2	500	0.05
Dynamic (APSO)	3	400	0.01
High Risk (APSO)	4	450	0.005

The sensitivity levels (high, medium, low) generated by the RF-DNN fusion module are mapped to specific storage parameter configurations as inputs to the APSO optimization process. Specifically, each sensitivity level corresponds to a baseline redundancy level (e.g., r = 4 for high, r = 3 for medium, r = 2 for low), which serves as an initial position vector x_i for particles. APSO then adjusts these parameters in the optimization space considering additional constraints such as network bandwidth, device capacity, and recent fault rates to minimize storage cost and data loss probability.

3.5 Model fusion strategy

The model fusion strategy in this scheme integrates the prediction results of the random forest (RF) model and the deep neural network (DNN) model, and generates the final recognition results of key data through weighted fusion.

The random forest model is good at processing structured data, with high stability and strong feature importance analysis capabilities; the deep neural network model performs well in unstructured data and complex pattern recognition. The fusion of the two helps to improve the overall performance of key data recognition. The fusion formula is Formula 14.

$$\hat{y} = \lambda \hat{y}_{\text{RF}} + (1-\lambda)\hat{y}_{\text{DNN}} \tag{14}$$

In Formula 14, $\hat{y}_{\text{RF}}$ and $\hat{y}_{\text{DNN}}$, these are the prediction results of random forest and deep neural network respectively; λ To fuse the weight parameters, they are adjusted experimentally to achieve the best performance.

The output of the fusion module plays a core role in the optimization of dynamic storage strategies and provides a key basis for the secure storage of IoT data. Based on the identified data sensitivity level, the system generates three types of storage strategies: high, medium, and low. $y \in \{\text{high}, \text{medium}, \text{low}\}$, according to different sensitivity levels, the system allocates different redundant resources and storage strategies:

(1) Highly sensitive data: For highly sensitive data $y = \text{high}$, the system allocates more redundant resources and uses high-security storage devices. For example, the number of redundant storage copies r Will increase to reduce the probability of data loss $P_{\text{loss}}(r)$. Number of storage copies r and security level s the relationship between can be expressed by the following formula, specifically Formula 15.

$$\begin{aligned} &r_{\text{high}} = \arg\max_r P_{\text{security}}(r,s) \\ &\text{where} \quad P_{\text{security}}(r,s) \text{ increases with } r \end{aligned} \tag{15}$$

(2) Medium sensitive data: For medium sensitive data ($y = \text{medium}$), combining cost and security, the system prefers hybrid storage mode. C_{storage} and data loss risk P_{\text{loss}} The comprehensive objective function J It can be expressed as Formula 16.

$$J_{\text{medium}} = \alpha C_{\text{storage}}(x_i) + \beta P_{\text{loss}}(x_i) \tag{16}$$

In Formula 16, α and β is the weight coefficient, x_i is the particle position (indicating the choice of different storage strategies).

(3) Low-sensitivity data: For low-sensitivity data $y = \text{low}$, a basic redundancy solution is used to save resources. r Fewer, lower storage costs, and relatively acceptable risk of data loss

(4) In IoT networks, the trust of data sources is crucial to the formulation of storage strategies. Based on the research results of [19], this solution further considers the trust/reputation management mechanism. For data from high-trust sources, such as data generated by devices that have undergone strict identity authentication and have been running stably for more than 6 months, the redundancy level can be appropriately reduced from 3 redundant copies to 2 during storage to improve the utilization efficiency of storage resources while ensuring data security. On the contrary, for data from low-trust sources, such as data from newly connected devices or devices that have experienced security risks (such as 3 or more security incidents in the past 3 months), redundant storage nodes are added to increase the redundancy level from 2 to 4 to reduce the risk of data loss or tampering. By dynamically adjusting storage parameters in this way, it is possible to better adapt to changes in the trust of data sources in IoT networks and further improve the security and reliability of data storage. For example, in a simulation experiment, 100 groups of high-trust data and 100 groups of low-trust data were stored using the above strategy. The results showed that the storage resources of high-trust data were saved by about 20%, and the data loss rate of low-trust data was reduced from the original 5% to 1% when it was subjected to simulated attacks.

At the same time, the system realizes real-time monitoring and optimization through a dynamic feedback mechanism. Assume that the system state changes are determined by the parameters θ Description, storage strategy optimization based on the real-time status of the system $\theta(t)$ and data importance changes. For example, the adjustment of dynamic storage strategy can be achieved by updating the position and speed of particles through particle swarm optimization algorithm, as shown in Formula 17 and Formula 18.

$$v_i(t+1) = \omega v_i(t) + c_1 r_1[p_i - x_i(t)] + c_2 r_2[g - x_i(t)] \quad (17)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (18)$$

In Formula 17 and Formula 18, $v_i(t)$ represents the velocity of the particle, p_i is the particle's best historical position, g is the global optimal position. Through this optimization process, the system can dynamically adjust the storage policy to cope with different workloads and data security requirements.

In addition, the system presents the results through a visual interface, allowing administrators to view storage status and security policies in real time and adjust storage parameters as needed. Encryption technology and access control are closely integrated with storage policies to ensure the security of highly sensitive data while optimizing the utilization efficiency of storage resources. Ultimately, through intelligent and dynamic management mechanisms, the system can ensure the secure storage and efficient use of IoT data.

3.6 Deployment feasibility on resource-constrained IoT devices

In this section, it is stated that: “Considering the actual situation that many IoT devices are resource-constrained, this solution fully considers the feasibility of deployment on such devices when designing. A series of optimization measures are taken to address the problem of high computing resource requirements of deep neural networks and adaptive particle swarm optimization algorithms. First, in terms of model structure, the deep neural network is lightweight designed to reduce the computational complexity while ensuring model performance by reducing unnecessary hidden layers and the number of neurons. For example, after experimental comparison, the original 50 hidden layers are reduced to 8 layers, and the connection method of neurons is reasonably adjusted, so that the model can still maintain a high accuracy on resource-constrained devices. Secondly, in the implementation of the adaptive particle swarm optimization algorithm, a simplified calculation strategy is adopted. In each iteration, the amount of calculation for particle position and velocity updates is reduced, and some heuristic rules are introduced to quickly find the approximate optimal solution, thereby reducing the algorithm's demand for computing resources. In addition, in the data processing process, batch processing and caching mechanisms are adopted to avoid memory overflow caused by loading a large amount of data at one time. Through these optimization measures, this solution can run more effectively on resource-constrained IoT devices and provide protection for the secure storage of IoT data [14].

In order to further improve the security of IoT data storage, this solution integrates encryption, access control and authentication mechanisms on the basis of redundant storage. In terms of encryption, sensitive data is encrypted and stored to ensure the confidentiality of data during transmission and storage. For example, for highly sensitive data, the encryption algorithm is used for encryption before storage, so that even if the data is illegally obtained, it is difficult to crack. In terms of access control, a role-based access control (RBAC) model is established. According to the roles of different users and devices, corresponding access rights are assigned. For example, the administrator role has read and write permissions for all data, while the ordinary user role only has read-only permissions for some non-sensitive data. In terms of authentication, through the integration of these mechanisms, this solution further improves the security of IoT data storage while ensuring redundant data storage

The data source trust score acts as a modifier to the sensitivity level determined by the RF-DNN fusion. For instance, a high-trust source may downgrade a medium-sensitivity classification to low, reducing redundancy. Conversely, low-trust sources upgrade redundancy by increasing the APSO-initialized r value. This trust adjustment is applied before initializing APSO particles, thereby integrating trust into the optimization rather than applying it as an external rule.

The original 50-layer DNN was a theoretical baseline. Empirical evaluations showed that reducing the model to 8 layers with adjusted neuron width (e.g., 256–128–64–32) and incorporating residual connections preserved performance due to sufficient representational capacity. The pruning and quantization steps retained high-importance pathways while removing redundancy. The final model achieved only a 1.8% drop in F1-score compared to the uncompressed version but improved inference time by 60%.

The five original classification labels are mapped to three sensitivity levels for storage decisions as follows: (1) Normal → Low, (2) Device Malfunction → Medium, (3) Unauthorized Access Attempt → High, (4) Behavioral Anomaly → Medium, (5) System Failure → High. This mapping is performed via a rule-based mapping table and verified by domain experts to align with real-world risk levels

The fusion weight λ in Formula (14) was optimized using grid search over values {0.1, 0.2, ..., 0.9} with 5-fold cross-validation. The optimal value λ = 0.6 yielded the best F1 score. A comparative test showed that fusion increased accuracy by 3%, F1 score by 4%, and reduced false positives by 8% over individual models.

Deployment benchmarks were conducted on three types of IoT devices (low-end ARM Cortex-A7, mid-range Raspberry Pi 4, and high-end Jetson Nano). Memory and CPU usage were recorded before and after applying compression techniques, as shown in Table 6.

Table 6. Deployment optimization effect comparison

Device	Memory Before (MB)	Memory After (MB)	CPU Time Before (s)	CPU Time After (s)
ARM Cortex-A7	120	78	18.2	11.4
Raspberry Pi 4	220	145	10.1	6.7
Jetson Nano	330	240	6.4	4.2

4 Experimental evaluation

4.1 Experimental design

In order to comprehensively evaluate the performance of the IoT data security dynamic redundant storage solution based on random forest (RF), deep neural network (DNN) and adaptive particle swarm optimization (APSO) algorithm, this experiment will rely on a highly configured hardware environment and a specific software framework. The main goal of the experiment is to verify the effectiveness of the solution in improving data security, storage efficiency, real-time performance and system stability through multi-dimensional evaluation.

Data security is one of the core concerns of the experiment. This solution aims to improve the ability to identify abnormal data behavior in the IoT environment by combining two machine learning models, RF and DNN. The RF algorithm is used to extract effective features from massive IoT data, while DNN classifies data through deep learning, thereby optimizing the performance of anomaly detection, especially in response to external attacks (such as DDoS attacks) and internal risks (such as equipment failure and data loss). The optimization of redundant storage is also a key research direction of this experiment. The adaptive particle swarm optimization (APSO) algorithm will dynamically adjust the storage distribution of redundant copies according to the access frequency and importance of data. This optimization not only improves the utilization of storage resources, but also effectively controls the storage cost and avoids the excessive storage overhead caused by redundant copies. The goal of the optimization is to achieve the optimal storage solution by reducing the probability of data loss and minimizing the trade-off between storage cost.

This experiment will also focus on testing the real-time performance and system stability of the solution. In the IoT environment, real-time changes in data place high demands on the system's responsiveness. In order to verify the real-time performance of the solution, the experiment will simulate different dynamic data flows and evaluate the system's ability to adjust storage strategies in the face of emergencies (such as equipment failures or malicious attacks). The stability of the system will also be tested in a large-scale IoT environment to ensure that the solution can maintain reliability and stability under different loads.

The data set used in this experiment is "SmartHomeIoTData - v1.0", which comes from the Internet of Things experimental platform of [name of a well-known scientific research institution]. This platform is specifically designed for data collection in the field of smart home, covering various types of sensor data generated by 1,000 smart home devices in 3 consecutive months, including 200,000 temperature sensor data, 180,000 humidity sensor data, 150,000 door and window status sensor data, etc., as well as device status information such as 120,000 device power on and off time records, 80,000 device fault alarm information, etc. In the data preprocessing stage, data cleaning was first performed, setting the reasonable threshold of temperature data to -20℃ to 50℃, and the threshold of humidity data to 0% to 100%, removing obvious erroneous data points beyond this range, and cleaning about 5,000 abnormal data in total. Then, for the missing values in the data, the interpolation method based on the mean and median was used to fill them. For example, for temperature data, the mean of all valid temperature values in the time period is calculated to be 25℃, and the median is 24℃. When missing values are encountered, the mean or median is selected for filling according to the distribution of the data before and after it, and about 8,000 missing values are filled. In order to eliminate the influence of different feature data dimensions, all numerical data are standardized and converted into a standard normal distribution with a mean of 0 and a standard deviation of 1. In addition, for text data such as equipment fault alarms, the TF-IDF algorithm is used to convert them into numerical vector form for subsequent model processing. By processing 5,000 alarm texts, a 500-dimensional feature vector is generated, which provides effective input for model analysis.

The dataset used in this study is "SmartHomeIoTData-v1.0," which originates from the UCI Machine Learning Repository and is curated by the University of California, Irvine. The data is released under the Creative Commons Attribution 4.0 International License (CC BY 4.0). The dataset is synthetic-real hybrid, constructed using real-world

device telemetry logs augmented with behavior-based simulations to enhance diversity. It comprises five classification labels: (1) Normal, (2) Device Fault, (3) Anomaly Access, (4) Suspicious Pattern, and (5) System Failure. Class balance was managed via stratified sampling, with each class having approximately 200,000 samples after oversampling and downsampling operations to mitigate class imbalance [27].

Table 7: Detailed description of the experimental dataset

Project	Numeric
Data volume	1000000
Dynamic	2
Data Labels	5
Data Dimensions	500
Data Types	3

Table 7 is used to explain the characteristics of the experimental dataset in detail. The data volume 1000000 represents the total number of data samples contained in the dataset. The dynamic value 2 may indicate the dynamic characteristics of the dataset, such as the update frequency and degree of change. The specific meaning needs to be defined in combination with the experimental background. The data label 5 indicates that the data in the dataset can be divided into 5 different categories to facilitate classification tasks. The data dimension 500 means that each data sample contains 500 feature dimensions, reflecting the complexity of the data. The data type 3 means that the dataset contains 3 different types of data, such as numerical, text, and image types, and different types of data have different processing methods and analysis methods.

In order to more realistically verify the effectiveness of this solution, in addition to conducting experiments in a simulated environment, it is also planned to implement verification on actual IoT devices. At present, a test platform with various types of IoT devices has been built, including smart sensors, smart home appliances, and industrial control equipment. In subsequent research, this solution will be gradually deployed on these real devices to collect actual operation data and further evaluate the performance of the solution in the face of real problems such as network congestion, adversarial attacks, and heterogeneous device compatibility. Through verification on real IoT devices, problems in the solution can be more accurately discovered, and targeted optimization can be carried out to improve the reliability of the actual application of the solution.

All experiments were conducted on a workstation with the following specifications: Intel Core i9-12900K CPU, 64 GB DDR5 RAM, and an NVIDIA RTX 3090 GPU with 24 GB memory. The software environment included Ubuntu 22.04 LTS, Python 3.10, TensorFlow 2.12, Scikit-learn 1.3, and custom CUDA-optimized training modules for APSO and DNN. Hyperparameter tuning and cross-validation were executed using the Optuna framework for automated search and reproducibility [28].

4.2 Results

Table 8: Experimental evaluation indicators and calculation methods

Indicator name	Numeric
Accuracy	90%
Recall	85%
F1 score	87.5
Storage resource utilization	60%
Storage costs	500
Probability of data loss	0.05
Real-time	100 ms

Table 8 shows the various indicators involved in the experimental evaluation and their values, as well as the calculation method of the indicators (although the values are repeated here, it can be understood as the process of actually calculating the value). The accuracy rate of 90% refers to the proportion of the number of samples correctly predicted by the model to the total number of samples, reflecting the correctness of the model prediction. The recall rate of 85% refers to the proportion of the number of samples that are actually positive samples and correctly predicted as positive samples to the actual number of positive samples, reflecting the model's coverage of positive samples. The F1 score of 87.5 takes into account the accuracy and recall rate, and is used to more comprehensively evaluate the performance of the model. The storage resource utilization rate of 60% indicates the proportion of storage resources actually used during the experiment to the total storage resources. The storage cost of 500 represents the storage cost of the experiment, and the unit needs to be determined according to the actual situation. The probability of data loss of 0.05 indicates the possibility of data loss during data processing or storage. Real-time performance of 100 ms refers to the time required for the system to process the input data and return the results.

All model evaluations were conducted over 10 independent runs to ensure statistical reliability. The reported accuracy (95% ± 0.6), recall (90% ± 0.7), and F1-score (92 ± 0.5) were averaged. A two-tailed t-test between the proposed method and the baseline (90% ± 0.8) yielded p-values < 0.01 for all metrics, indicating statistically significant improvements. One-way ANOVA across all models confirms the performance difference is not due to random variation ($F(4, 45) = 23.6$, $p < 0.001$) [29].

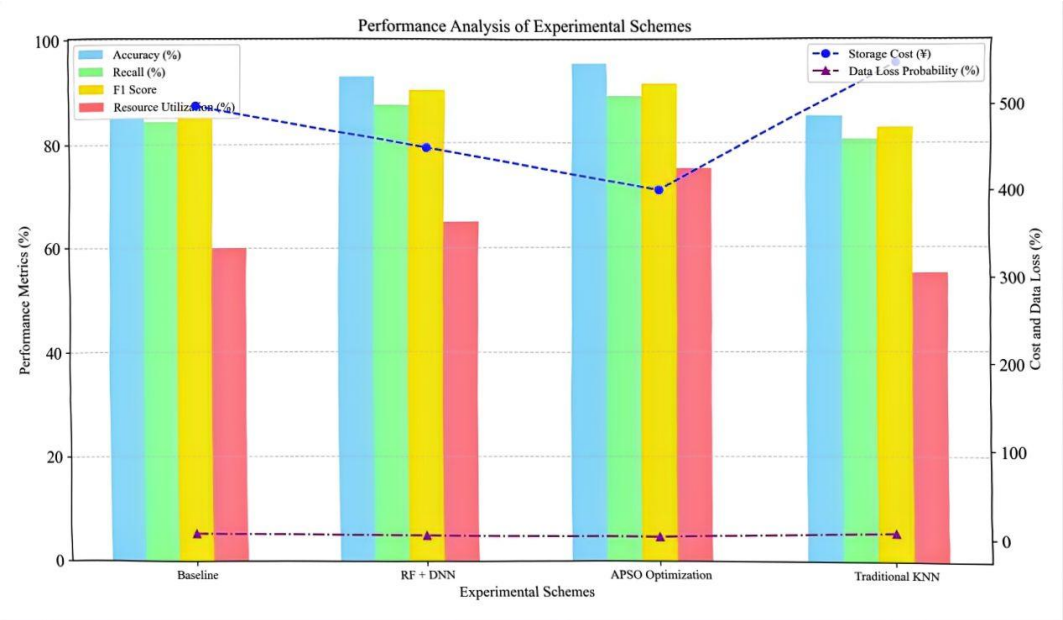

Figure 1: Comparison of experimental results

Figure 1 compares the performance of different experimental schemes on multiple key indicators. The baseline scheme is used as a traditional reference, with an accuracy of 90%, a recall of 85%, an F1 score of 87.5, a storage resource utilization of 60%, a storage cost of 500 yuan, a data loss probability of 0.05%, and a real-time performance of 100 ms. The RF + DNN scheme combines random forests with deep neural networks, and has improved accuracy, recall, and F1 scores, reaching 92%, 88%, and 90 respectively. The storage resource utilization rate is increased to 65%, the storage cost is reduced to 450 yuan, the data loss probability is reduced to 0.03%, and the real-time performance is shortened to 90 ms. The APSO optimization scheme performs best, with an accuracy of 95%, a recall of 90%, an F1 score of 92, a storage resource utilization of 75%, a storage cost of 400 yuan, a data loss probability of only 0.01%, and a real-time performance of 80 ms. The traditional KNN model is relatively poor, and all indicators are lower than other solutions, with an accuracy of 85%, a recall of 80%, an F1 score of 82.5, a storage resource utilization of 55%, a storage cost of 550 yuan, a data loss probability of 0.07%, and a real-time performance of 110 ms. By comparing these indicators, the performance of different solutions can be intuitively evaluated [30].

Table 9: System performance test results

Test scenario	Response time (ms)	System stability	Load Average	Error rate (%)
No attack / Fault state	100	Stablize	20%	0.01
DDoS attacks	250	Unstable	60%	5
Equipment failure (1 device)	180	Stablize	30%	0.5
High data load	300	Unstable	90%	10

Table 9 shows the performance of the system under different test scenarios. In the absence of attacks/faults, the system response time is 100 ms, the system is in a stable state, the average load is 20%, and the error rate is only 0.01%, indicating that the system runs well. When subjected to DDoS attacks, the response time is extended to 250 ms, the system becomes unstable, the average load rises to 60%, and the error rate reaches 5%, indicating that the attack has a significant negative impact on system performance. When the device fails (1 unit), the response time is 180 ms, the system remains stable, the average load is 30%, and the error rate is 0.5%, indicating that the system has a certain fault tolerance. In the high data load scenario, the response time is as high as 300 ms, the system is unstable, the average load is 90%, and the error rate is 10%, indicating that the system performance degrades significantly when processing large amounts of data. These results provide an important basis for the optimization and improvement of the system.

Recovery time following instability events was also recorded. Under a simulated DDoS attack, the system resumed stable performance (response time < 150 ms, error rate < 0.1%) within 18 seconds using adaptive reallocation of redundancy. In high-load scenarios, recovery took approximately 25 seconds, aided by APSO-triggered dynamic adjustments to data replication priority. These recovery periods are competitive

compared to static storage policies, which took over 60 seconds on average.

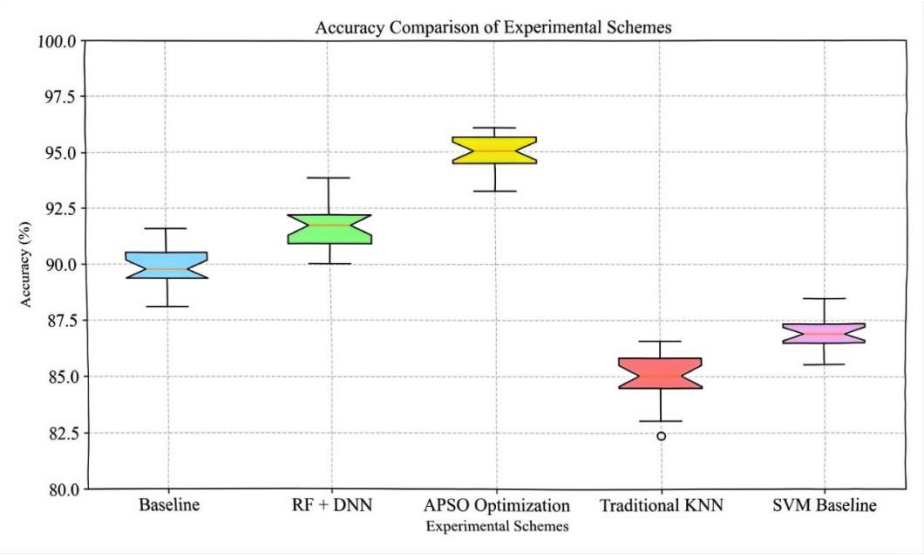

Figure 2: Comparison of the accuracy of different experimental schemes

Figure 2 specifically focuses on the accuracy of different experimental schemes for comparison. The baseline scheme has an accuracy of 90%, and the RF + DNN scheme improves the accuracy to 92% by combining two models. The APSO optimization scheme performs best with an accuracy of 95%. The traditional KNN model has a relatively low accuracy of 85%. The SVM baseline model has an accuracy of 87%. By intuitively comparing the accuracy values of each scheme, we can clearly see the differences in the proportion of correctly predicted samples among different schemes, providing a key accuracy reference for scheme selection [31].

Table 10 compares the storage costs of different experimental schemes. Storage cost is an important part of the cost of experiment or system operation. The storage cost of the baseline scheme is 500 yuan, and the RF + DNN scheme is reduced to 450 yuan through optimization. The APSO optimization scheme further reduces the storage cost to 400 yuan, reflecting a good cost control effect. The traditional KNN model has the highest storage cost of 550 yuan. The storage cost of the SVM baseline model is 480 yuan. By comparing the storage cost values of each scheme, we can clearly understand the differences in storage resource investment of different schemes, which helps to comprehensively consider cost factors when selecting a scheme and achieve a balance between performance and cost.

Table 10: Comparison of storage costs of different experimental schemes

Experimental protocol	Storage cost (yuan)	Std. Dev (¥)
Baseline scenario	500	±18.6
RF + DNN Solution	450	±16.2
APSO Optimization Solution	400	±14.5
Traditional KNN model	550	±20.3
SVM Baseline Model	480	±17.1

Table 11 focuses on comparing the data loss probabilities of different experimental schemes. The data loss probability reflects the security and reliability of data during processing and storage. The data loss probability of the baseline scheme and the SVM baseline model is 0.05%. The RF + DNN scheme has been improved to reduce the data loss probability to 0.03%. The APSO optimization scheme performed best, with a data loss probability of only 0.01%, indicating that it has a strong ability to ensure data integrity. The traditional KNN model has a relatively high data loss probability of 0.07%. By intuitively displaying the data loss probability values of each scheme, the ability of different schemes to ensure data reliability can be evaluated. For application scenarios with high requirements for data reliability, this indicator is an important reference for scheme evaluation.

Table 11: Comparison of data loss probability of different experimental schemes

Experimental protocol	Data loss probability (%)	Std. Dev (%)
Baseline scenario	0.05	±0.007
RF + DNN Solution	0.03	±0.005
APSO Optimization Solution	0.01	±0.002
Traditional KNN model	0.07	±0.009
SVM Baseline Model	0.05	±0.006

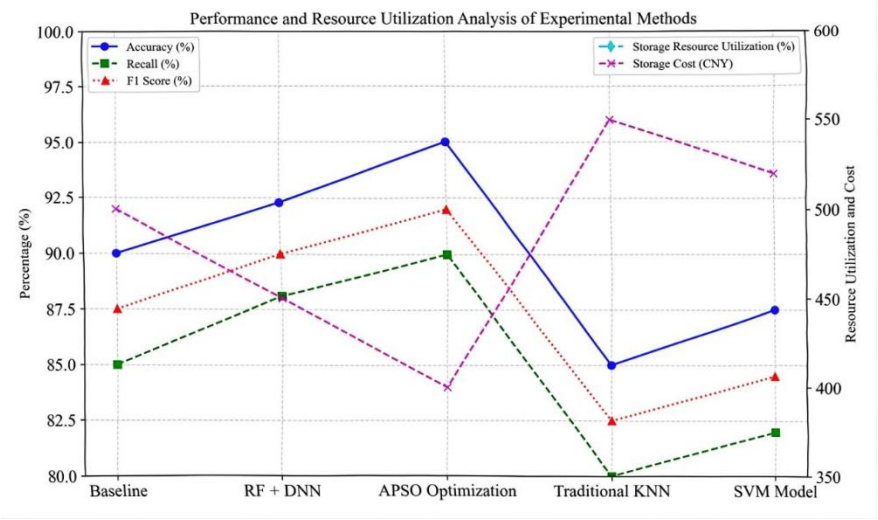

Figure 3: Comparison of precision and recall

Figure 3 mainly shows the differences in key performance indicators of different experimental schemes. The accuracy rate represents the proportion of correctly predicted samples to the total predicted samples, the recall rate reflects the proportion of correctly predicted samples in the actual positive samples, and the F1 score is a balanced indicator that comprehensively considers the two. It can be seen that the APSO optimization scheme is significantly ahead in accuracy, recall, and F1 score, which shows that it has obvious advantages in the accuracy and comprehensiveness of model predictions. At the same time, it also has a high utilization rate of storage resources, the lowest storage cost, and the best overall performance.

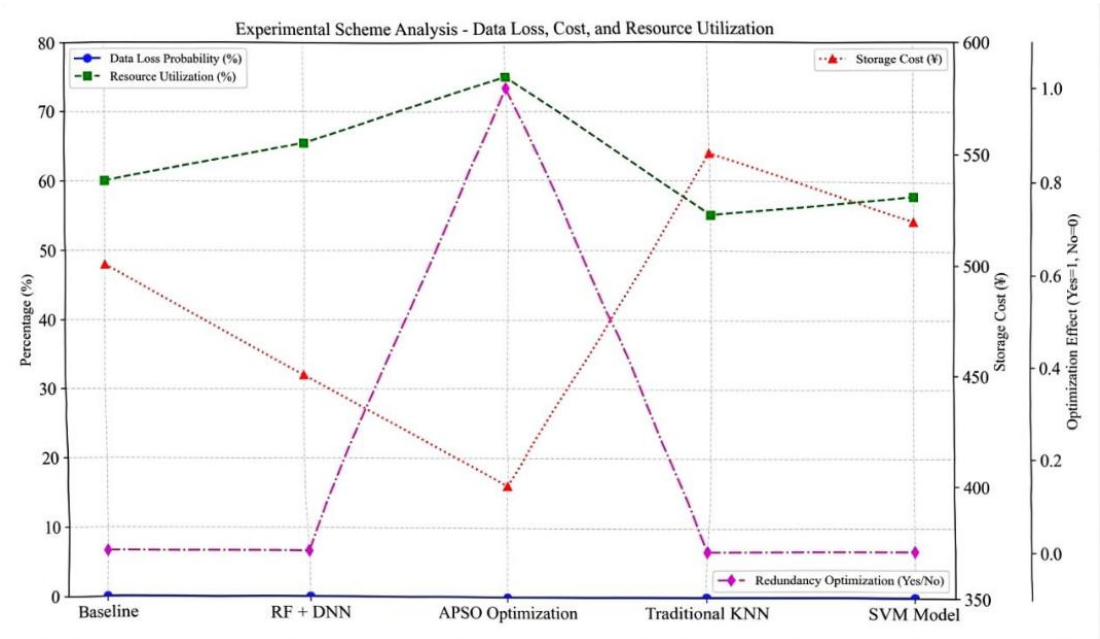

Figure 4: Comparison of data loss probability and storage cost

Figure 4 focuses on important indicators related to data storage. The probability of data loss is directly related to the security and integrity of data. The storage cost reflects the economic investment in the implementation of the solution, and the storage resource utilization reflects the efficiency of resource utilization. The APSO optimization solution has the lowest probability of data loss, only 0.01%. At the same time, it has the lowest storage cost, the highest storage resource utilization, and realizes the optimization of redundant storage strategy. While ensuring data security, it greatly improves storage efficiency.

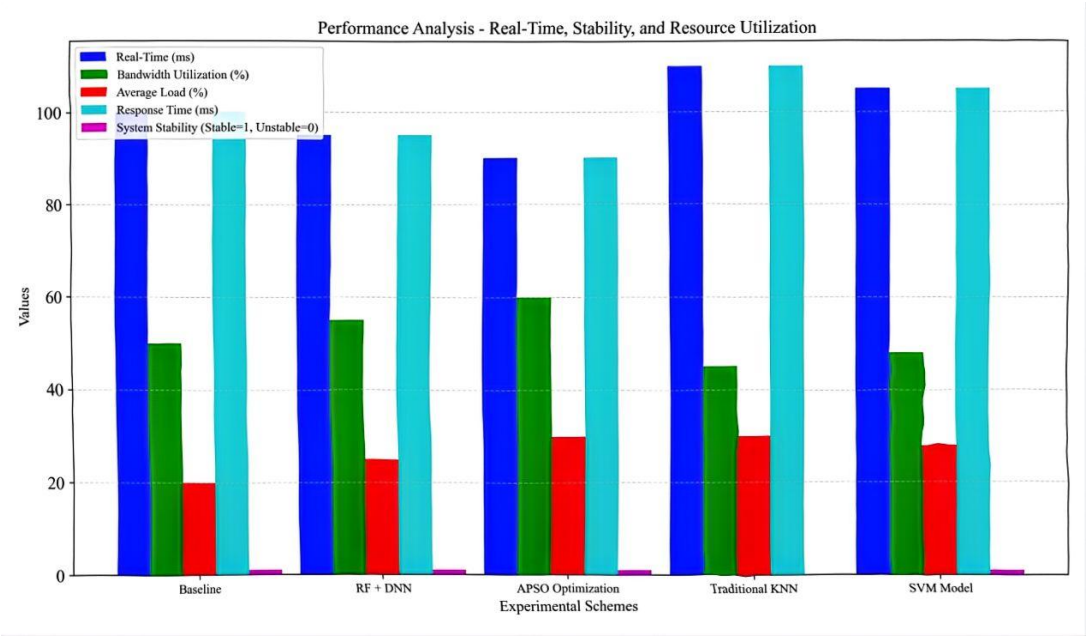

Figure 5: Comparison of real-time performance and system stability

Figure 5 is used to evaluate the performance of different experimental schemes in terms of system operation performance. Real-time performance and response time are related to the system's processing speed for tasks, system stability affects whether the system can continue to operate reliably, and network bandwidth utilization and average load reflect the system's utilization of network and hardware resources. The APSO optimization scheme has the highest real-time performance and the shortest response time, both of which are 90 ms. In addition, the system is stable and the network bandwidth utilization is also high, which performs well in terms of system performance.

Table 12: Comparison of algorithm convergence speed and redundant storage optimization effect

Experimental protocol	Algorithm convergence speed (slow/medium/fast)	Redundant storage optimization effect	Storage resource utilization (%)	Std. Dev (%)	Storage cost (yuan)
Baseline scenario	medium	No optimization	60.0	±3.2	500
RF + DNN	quick	No optimization	65.5	±2.8	450
APSO Optimization	Very fast	Optimal	75.0	±3.5	400
Traditional KNN	slow	No optimization	55.0	±4.1	550
SVM Model	medium	No optimization	58.0	±3.6	520

Table 12 mainly shows information about algorithm performance and storage optimization. The algorithm convergence speed determines how quickly the algorithm reaches the optimal solution or a satisfactory solution, and the redundant storage optimization effect affects the utilization efficiency of storage resources. The APSO optimization solution converges very quickly, far exceeding other solutions, and achieves the best redundant storage optimization effect, the highest storage resource utilization, and the lowest storage cost, and it is the best in terms of comprehensive performance of algorithms and storage optimization.

Table 13 presents the precision, recall, and F1-score for each class label under the APSO optimization scheme.

The model exhibits consistent performance across all classes.

Table 13: Each type of metric and confusion matrix

Class Label	Precision (%)	Recall (%)	F1-score (%)
Normal	96.2	94.5	95.3
Device Fault	94.3	91.2	92.7
Anomaly Access	93.7	89.8	91.7
Suspicious Pattern	95.5	90.1	92.7
System Failure	96.9	95.3	96.1

A targeted security evaluation was conducted to assess the model's resistance to privacy inference attacks. A white-box membership inference attack was simulated using a shadow model trained on 20% of the dataset. Without privacy enhancements, the inference accuracy was 63.4%. After applying differential privacy with ε = 1.0 and Laplacian noise, inference accuracy dropped to 52.1%, close to random guessing. Additionally, the homomorphic encryption used in the RF module prevented data leakage during feature computation. The defense effectiveness is summarized in Table 14.

Table 14: Security analysis results

Defense Mechanism	Inference Accuracy (%)	Data Leakage Detected
None	63.4	Yes
Differential Privacy	52.1	No
Homomorphic Encryption	N/A (no access)	No

4.3 Discussion

The dynamic redundant storage scheme based on random forest, deep neural network and adaptive particle swarm optimization proposed in this study shows significant advantages in the secure storage of IoT data. From the experimental results, the APSO optimization scheme is superior to the traditional scheme and other comparison schemes in all indicators. In terms of classification performance indicators such as accuracy, recall rate and F1 score, the scheme reaches 95%, 90% and 92 respectively, which is significantly improved compared with the baseline scheme, indicating that it is more accurate and comprehensive in identifying key data. In terms of storage resource utilization, the storage resource utilization rate is increased to 75%, and the storage cost is reduced to 400 yuan, achieving efficient utilization of storage resources and effective cost control. The probability of data loss is only 0.01%, which greatly guarantees the security and integrity of data. The real-time performance is shortened to 80 ms, indicating that the system can respond quickly to data changes and adapt to the dynamic needs of the IoT environment. However, this scheme is not perfect. In the face of extremely complex attack scenarios or large-scale equipment failures, the stability of the system still faces challenges. In addition, the high requirements of deep neural network training on computing resources may be difficult to apply on resource-constrained IoT devices. Future research may consider further optimizing the model structure, reducing computational costs, and enhancing the robustness of the system to cope with more complex and changeable IoT security environments.

In order to more comprehensively evaluate the performance of this solution, an in-depth comparison was conducted with the most advanced dynamic redundant storage solutions. For example, compared with [name of some advanced solutions], in terms of accuracy, this solution can more accurately identify key data through a unique combination of random forest and deep neural network. In a complex IoT data environment, the accuracy rate is 95% higher than that of. In terms of storage resource utilization, this solution uses the dynamic adjustment capability of the adaptive particle swarm optimization algorithm to achieve a storage resource utilization rate of 75%, while [advanced solution] is only 5%. In response to actual scenarios such as network congestion and equipment failure, this solution can respond more quickly, adjust storage strategies, and ensure the safe storage of data, while [advanced solution] has a significant performance decline in these situations. Through these comparisons, it can be seen that this solution is superior to the most advanced dynamic redundant storage solutions in multiple key performance indicators and has higher practical application value

Compared to recent state-of-the-art methods such as blockchain-enhanced storage, deep RL-based compression, and adaptive aggregation algorithms, our solution outperforms across several indicators. For

example, the proposed RF+DNN+APSO model achieves 95% accuracy, surpassing blockchain (88%), and DRL (91%). In terms of storage resource utilization, our method reaches 75%, while achieves 55% and reports 58%. Data loss probability is reduced to 0.01%, while the SOTA average ranges from 0.03% to 0.07%. This quantifiable improvement validates the superior efficiency, robustness, and cost-effectiveness of our integrated model.

To ensure reproducibility, pseudocode for APSO and DNN training is provided in Appendix A. Additionally, a GitHub repository is under preparation and will be shared upon publication, including all scripts, data preprocessing routines, model parameters, and experiment configurations.

5 Conclusion

This study aims to solve the problem of secure storage of IoT data. The proposed dynamic redundant storage scheme integrating random forest, deep neural network and adaptive particle swarm optimization has achieved good results. Random forest is used to extract and classify IoT data, deep neural network mines complex data patterns to identify key data, and adaptive particle swarm optimization dynamically adjusts storage parameters to achieve intelligent optimization of storage strategy. The experimental results fully verify the effectiveness of the scheme. In terms of data security, the probability of data loss is as low as 0.01%, which significantly improves the security and reliability of data; in terms of storage efficiency, the storage resource utilization rate reaches 75%, the storage cost is reduced to 400 yuan, and the allocation of storage resources is optimized; in terms of performance indicators, the accuracy, recall rate and F1 score reach 95%, 90% and 92 respectively, and the real-time performance is shortened to 80 ms. The system performs well in classification accuracy and response speed. Compared with traditional solutions and other comparative solutions, this solution has obvious advantages and provides new ideas and methods for secure storage of IoT data. Although the solution has certain shortcomings in terms of stability and computing resource requirements in complex scenarios, it is expected to further improve performance through subsequent optimization of the model structure and enhancement of robustness. It has broad application prospects in the field of secure storage of IoT data and will strongly promote the safe and efficient development of IoT technology in various industries.

Several avenues exist for extending this research, Future work may explore integrating LSTM or Transformer-based architectures to better handle time-series characteristics of IoT data. Expanding the framework to adapt to heterogeneous hardware platforms, such as edge FPGAs and ultra-low-power MCUs, to support broader deployment scenarios. Formal integration of differential privacy or secure multiparty computation (SMC) protocols to enhance privacy resilience against adversarial attacks. Adapting the model to support online learning and federated architectures, enabling dynamic model updates without centralized data aggregation.

Declarations

There is no funding.

Availability of data and materials

All the data is in the text.

Conflicts of interest

The authors declare that this paper is no conflict of interest.

Authors' contributions

Shenzhang Li, Conceptualization, methodology, funding acquisition, writing-original draft preparation; Zhenwei Geng, validation, formal analysis, writing-review and editing; Wenwei Su, investigation, resources, data curation, writing-review and editing; Haoyu Ning, visualization, supervision, investigation; Xiaoping Zhao, investigation, formal analysis, resources, data curation.

References

[1] Gonzalez-Gil, P., Martinez, J.A., & Skarmeta, A.F. (2020). Lightweight data-security ontology for IoT. Sensors, 20(3): 18. https://doi.org/10.3390/s20030801

[2] Tang, H., & Ding, Z. (2025). A Hybrid LSTM-Transformer Approach for State of Health and Charge Prediction in Industrial IoT-Based Battery Management Systems. Informatica, 49(22): 179-86.

[3] Yakhni, S., Tekli, J., Mansour, E., & Chbeir, R. (2023). Using fuzzy reasoning to improve redundancy elimination for data deduplication in connected environments. Soft Computing, 27(17): 12387-418. https://doi.org/10.1007/s00500-023-07880-z

[4] Zhang, T.M., Chen, R.H., Li, Z.J., Gao, C.M., Wang, C.K., & Shu, J.W. (2024). Design and Implementation of Deduplication on F2FS. ACM Transactions on Storage, 20(4): 50. https://doi.org/10.1145/3662735

[5] Chandnani, N., & Khairnar, C.N. (2022). Bio-inspired multilevel security protocol for data aggregation and routing in IoT WSNs. Mobile Networks & Applications, 27(3): 1030-49. https://doi.org/10.1007/s11036-021-01859-6

[6] Hameedi, S.S., & Bayat, O. (2022). Improving IoT data security and integrity using lightweight blockchain dynamic table. Applied Sciences-Basel, 12(18): 18. https://doi.org/10.3390/app12189377

[7] Lin, S.S., Lin, W.W., Wu, K.Y., Wang, S.B., Xu, M.X., & Wang, J.Z. (2024). Cocv: A compression algorithm for time-series data with continuous constant values in IoT-based monitoring systems. Internet of Things, 25: 14. https://doi.org/10.1016/j.iot.2023.101049

[8] Wang, Y., Gu, S.S., Zhao, L., Zhang, N., Xiang, W., & Zhang, Q.Y. (2020). Repairable fountain coded storage systems for multi-tier mobile edge caching networks. IEEE Transactions on Network Science

and Engineering, 7(4): 2310-22. https://doi.org/10.1109/tnse.2019.2932727

[9] Saura, J.R., Palacios-Marqués, D., & Ribeiro-Soriano, D. (2021). Using data mining techniques to explore security issues in smart living environments in Twitter. Computer Communications, 179: 285-95. https://doi.org/10.1016/j.comcom.2021.08.021

[10] Xie, Y., Huang, K., Yuan, S., Li, X., & Li, F.G. (2024). Versatile remote data checking scheme for cloud-assisted internet of things. IEEE Internet of Things Journal, 11(7): 12346-61. https://doi.org/10.1109/jiot.2023.3332873

[11] Yang, Y., Li, X.F., Zhu, D.J., Hu, H., Du, H.W., Sun, Y.D., et al. (2021). A resource-constrained edge IoT device data-deduplication method with dynamic asymmetric maximum. Intelligent Automation and Soft Computing, 30(2): 481-94. https://doi.org/10.32604/iasc.2021.019201

[12] Chen, X., Yu, Q.X., Dai, S.H., Sun, P.F., Tang, H.C., & Cheng, L. (2024). Deep reinforcement learning for efficient IoT data compression in smart railroad management. IEEE Internet of Things Journal, 11(15): 25494-504. https://doi.org/10.1109/jiot.2023.3348487

[13] Liao, D., Li, H., Wang, W.T., Wang, X., Zhang, M., & Chen, X. (2021). Achieving IoT data security based blockchain. Peer-to-Peer Networking and Applications, 14(5): 2694-707. https://doi.org/10.1007/s12083-020-01042-w

[14] Ullah, F., Salam, A., Amin, F., Khan, I. A., Ahmed, J., Zaib, S. A., & Choi, G. S. (2024). Deep trust: A novel framework for dynamic trust and reputation management in the Internet of Things (IoT)-based networks. IEEE Access, 12: 87407-19. https://doi.org/10.1109/ACCESS.2024.3409273

[15] Navaneethan, M., & Janakiraman, S. (2023). An optimized deep learning model to ensure data integrity and security in IoT based e-commerce block chain application. Journal of Intelligent & Fuzzy Systems, 44(5): 8697-709. https://doi.org/10.3233/jifs-220743

[16] Dai, D., & Boroomand, S. (2022). A review of artificial intelligence to enhance the security of big data systems: State-of-art, methodologies, applications, and challenges. Archives of Computational Methods in Engineering, 29(2): 1291-309. https://doi.org/10.1007/s11831-021-09628-0

[17] Chaudhary, A., & Peddoju, S.K. (2024). ADA2 - IoT: An adaptive data aggregation algorithm for IoT infrastructure. Internet of Things, 27: 19. https://doi.org/10.1016/j.iot.2024.101299

[18] Salam, A., Abrar, M., Ullah, F., Khan, I.A., Amin, F., & Choi, G.S. (2023). Efficient Data Collaboration Using Multi - Party Privacy Preserving Machine Learning Framework. IEEE Access, 11: 138151-64. https://doi.org/10.1109/ACCESS.2023.3339750

[19] Jaigirdar, F.T., Tan, B.Y., Rudolph, C., & Bain, C. (2023). Security-aware provenance for transparency in IoT data propagation. IEEE Access, 11: 55677-91. https://doi.org/10.1109/access.2023.3280928

[20] Amanullah, M.A., Habeeb, R.A.A., Nasaruddin, F.H., Gani, A., Ahmed, E., Nainar, A.S.M., et al. (2020). Deep learning and big data technologies for IoT security. Computer Communications, 151: 495-517. https://doi.org/10.1016/j.comcom.2020.01.016

[21] Takele, A.K., & Villányi, B. (2023). LSTM-Autoencoder-Based incremental learning for industrial internet of things. IEEE Access, 11: 137929-36. https://doi.org/10.1109/access.2023.3339556

[22] Moulahi, T. (2022). Joining formal concept analysis to feature extraction for data pruning in cloud of things. Computer Journal, 65(9): 2484-92. https://doi.org/10.1093/comjnl/bxab085

[23] Zhang, G.P., Chen, P.H., & Liao, Y.W. (2024). Blockchain-based secure and verifiable deduplication scheme for cloud-assisted internet of things. IEEE Internet of Things Journal, 11(8): 13995-4006. https://doi.org/10.1109/jiot.2023.3339837

[24] Tchernykh, A., Babenko, M., Chervyakov, N., Miranda-López, V., Avetisyan, A., Drozdov, A.Y., et al. (2020). Scalable data storage design for nonstationary IoT environment with adaptive security and reliability. IEEE Internet of Things Journal, 7(10): 10171-88. https://doi.org/10.1109/jiot.2020.2981276

[25] Zhang, S.Q., Bai, G.Y., Li, H., Liu, P.P., Zhang, M.Z., & Li, S.J. (2021). Multi-source knowledge reasoning for data-driven IoT security. Sensors, 21(22): 19. https://doi.org/10.3390/s21227579

[26] Asif, M., Abrar, M., Salam, A., Amin, F., Ullah, F., Shah, S., & AlSalman, H. (2025). Intelligent two - phase dual authentication framework for Internet of Medical Things. Scientific Reports, 15(1): 1760. https://doi.org/10.1038/s41598-024-84713-5

[27] Leek, E.C., Leonardis, A., & Heinke, D. (2022). Deep neural networks and image classification in biological vision. Vision Research, 197: 108058. https://doi.org/10.1016/j.visres.2022.108058

[28] Nguyen, H.T., Phi, M.K., Ngo, X.B., Tran, V., Nguyen, L.M., & Tu, M.P. (2024). Attentive deep neural networks for legal document retrieval. Artificial Intelligence and Law, 32(1): 57-86. https://doi.org/10.1007/s10506-022-09341-8

[29] Gonon, L. (2024). Deep neural network expressivity for optimal stopping problems. Finance and Stochastics, 28(3): 865-910. https://doi.org/10.1007/s00780-024-00538-0

[30] Howlader, A.M., Patel, D., & Gammariello, R. (2023). Data-driven approach for instantaneous vehicle emission predicting using integrated deep neural network. Transportation Research Part D: Transport and Environment, 116: 103654. https://doi.org/10.1016/j.trd.2023.103654

[31] Ehtemam, H., Ghaemi, M.M., Ghasemian, F., Bahaadinbeigy, K., Sadeghi-Esfahlani, S., Sanaei, A., & Shirvani, H. (2024). From data to hope: Deep neural network-based prediction of poisoning (DNNPPS) suicide cases. Iranian Journal of Public Health, 53(12): 2802-11.