International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 8426-8442

https://doi.org/10.70517/ijhsa464717

Network Security Protection Model Based on Fuzzy Reasoning

Xuewei Liu[1], Bingfu Hu[2,*] and Ruiwei Duan[2]
[1] Science and Technology Division, Weifang Engineering Vocational College, Weifang, Shandong, 262500, China
[2] Department of Information Engineering, Weifang Engineering Vocational College, Weifang, Shandong, 262500, China
Corresponding authors: (e-mail: lidandan891@126.com).

Abstract To predict cybersecurity incidents in an IoT environment, transform passive defense into “active” defense, and minimize potential damage to network systems, cybersecurity situational awareness technology has been rapidly developing. This paper integrates common methods for multi-source data fusion, applies an adaptive neural fuzzy inference system to assess cybersecurity situational awareness, and proposes an improved LSTM cybersecurity situational awareness prediction model based on the Sparrow Search Algorithm, achieving cybersecurity protection under multimedia fusion technology. Experimental results show that by quantitatively calculating cybersecurity posture values at different levels—service level, host level, and network system level—this assessment model provides more comprehensive assessment information compared to traditional methods, with more accurate results. Additionally, the average relative error of the improved SSA-LSTM neural network cybersecurity posture prediction algorithm stabilizes around 5.29%, enabling effective prediction of cybersecurity posture over the coming period.

Index Terms Adaptive Neural Fuzzy Inference; Sparrow Search Algorithm; LSTM Network; Cybersecurity Situation

I. Introduction

With the development of network information technology, information dissemination has undergone a landmark transformation, with internet-based information dissemination growing rapidly. Network dissemination now features multimedia capabilities to achieve better dissemination effects [1]. The Internet of Things (IoT) is based on the internet and extends and expands it, enabling a wide range of sensor devices and multimedia devices to connect to the network. The addition of new devices has led to an exponential increase in the types and quantities of multimedia information generated and transmitted [2]-[4]. New application scenarios such as social media, self-media, live video streaming, high-definition television, and online meetings have emerged, enabling the media industry to achieve a broader service scope and higher service quality [5]. However, while the technological revolution has brought about revolutionary changes, it has also imposed higher demands on network bandwidth security, user data privacy, and service latency. Additionally, traditional networks use a centralized topology, and the central server is increasingly burdened by network security, computing pressure, and storage pressure, especially security and privacy issues, which are receiving increasing attention [6], [7].

In terms of network security, the volume of multimedia data generated by IoT devices has increased by more than 40%, and the development of audio fingerprint forgery technology has created security risks for biometric authentication. The lack of exploration into the interconnectivity of multimodal data means that while single-modal data may not pose risks, analyzing multimodal data reveals potential issues. Malicious data or command injection attacks on video streaming platforms persist, enabling attackers to steal platform administrator information or delete/modify data [8]-[11]. Additionally, IoT user nodes typically have limited functionality and are easily accessible. Due to hardware and software limitations, they are highly susceptible to attacks. Furthermore, the centralized topology of IoT networks means that when the central server is attacked, the entire network may be affected or even paralyzed [12]-[14]. Therefore, exploring multimedia cybersecurity protection technologies in IoT environments is urgent.

Lv et al. [15] proposed an anonymous batch authentication scheme based on hash message authentication codes for multimedia IoT privacy protection, which avoids communication overhead and privacy leakage issues caused by revocation lists. Hurrah et al. [16] developed a watermarking scheme for multimedia image copyright protection, data security, and content authentication. In terms of copyright protection, it employs robust watermarks embedded using the block-wise coefficient difference algorithm, data encryption via a novel Arnold transform-based encryption algorithm, and fragile watermarks embedded in the spatial domain for tamper detection and localization to achieve content authentication. Mishra et al. [17] proposed a reliable authentication protocol based on smart cards for

secure multimedia communication in IoT-supported wireless sensor networks, which is effective against attacks. Dhar et al. [18] studied the security issues of multimedia data sharing in wireless IoT networks through distributed processing using blockchain and the InterPlanetary File System (IPFS), and constructed an advanced security model.

Zhu et al. [19] developed a multimedia fusion privacy protection algorithm for IoT data security, leveraging blockchain technology, smart device encryption, and distributed hash table storage. Park and Lee [20] designed an enhanced eID card personal information protection scheme based on secret sharing technology to safeguard electronic identity information and privacy in online sensor networks. This scheme maintains protection functionality even if the eID card is lost and resists attacks such as spoofing and eavesdropping. Singh and Lee [21] combined machine learning and deep learning to study network intrusion and privacy protection in IoT. They used various classification models to detect and classify network attacks, conducted in-depth analysis of network attack features using random forests and feedforward neural networks, and utilized federated learning to construct a privacy protection technology. Currently, there is still a lack of multi-modal adaptive protection technologies, and some methods have high false positive rates and processing delays, leading to insufficient protection effectiveness.

Fuzzy inference algorithms are a type of reasoning method based on fuzzy logic, serving as a mathematical tool for handling fuzzy characteristics, and have important applications in the field of security [22]. Liu et al. [23] employed fuzzy inference mechanisms to locate target positions under fuzzy reasoning, combining alternative selection strategies from thought sets to achieve target visual monitoring in IoT environments. This provides a reference for the design of security protection models for multimedia fusion in IoT.

This paper proposes a cybersecurity protection framework for multimedia fusion technology in IoT environments, focusing on cybersecurity situation assessment and prediction. In terms of NSSE, by combining the advantages of fuzzy reasoning in describing cybersecurity metrics and the strong learning capabilities of neural networks, expert experience is better utilized in the situation assessment process, leading to the proposal of an adaptive neural fuzzy reasoning system for application. In terms of NSSP, based on the LSTM model for situation prediction, the improved SSA algorithm is used for hyperparameter optimization, and the SA algorithm and Sobol sequence are used to optimize the SSA algorithm. Finally, evaluation simulation experiments are set up to calculate the comprehensive alert situation values at the service level, host level, and system level, respectively. Through experiments and comparisons with other prediction algorithms, the accuracy and convergence of the prediction algorithm proposed in this paper are verified.

II. Cybersecurity Situation Awareness and Security Protection Framework System

Cybersecurity situational awareness technology can comprehensively collect situational element data from the network environment. By leveraging computational power and corresponding algorithms, it efficiently processes data to analyze and extract actual useful information hidden within big data, thereby deriving corresponding situational inferences. This provides cybersecurity managers with data and speed advantages, enabling them to more reasonably allocate cybersecurity defense measures and minimize the damage caused by network attacks. Figure 1 shows the cybersecurity situational awareness model architecture.

The model is divided into four interconnected phases: cybersecurity situational element acquisition, situational assessment, situational prediction, and visualization. Its basic functions align with the definition of cybersecurity situational awareness. Below is a brief overview of the data processing processes and corresponding functions of each module.

(1) Cybersecurity Situation Element Acquisition Module: This module is responsible for collecting and extracting various cybersecurity audit data from the network environment, such as network traffic and security events, and performing data preprocessing. After obtaining standardized data, it extracts effective information from the data to identify potential abnormal activities in the network environment and extract various situation element data that affect the cybersecurity situation, including attack behaviors in the network and statistics on network vulnerabilities.

(2) Cybersecurity Situation Assessment Module: This module performs fusion analysis on the situation element data after the preprocessing steps, assigns different weights to different element data, calculates the overall cybersecurity situation value, and describes the overall cybersecurity status.

(3) Cybersecurity Situation Prediction Module: This module, based on the previous two steps, uses historical situation element data and situation value sequence information, along with relevant time series processing techniques, to predict future cybersecurity trends.

(4) Visualization Module: The final visualization module dynamically and intuitively displays the results of situation awareness, including all intelligence information obtained throughout the cybersecurity situation awareness process, such as identified attack activities, the overall cybersecurity status of the network environment, and predictions of future situation developments.

The above constitutes the main content of the cybersecurity situation awareness framework. By following these steps, cybersecurity managers can utilize the results of cybersecurity situation awareness to make effective decisions and complete the closed-loop process of cybersecurity defense. At the same time, we can see that throughout the entire situation awareness process, the results of situation assessment and situation prediction ultimately need to be presented to cybersecurity managers as a basis for decision-making. Therefore, these two stages are the core processes of cybersecurity situation awareness and the primary research content of the subsequent chapters of this paper.

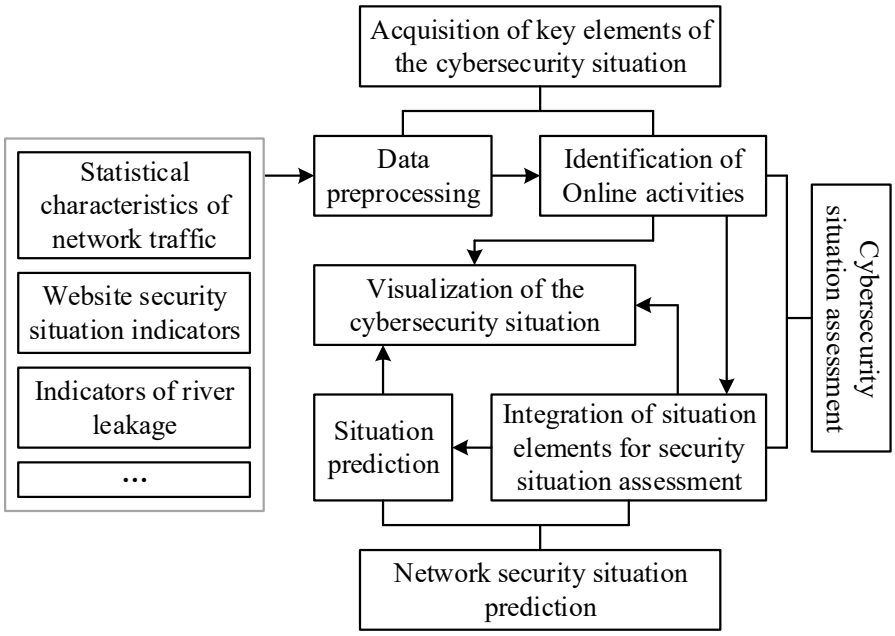

Figure 1: Network security situational perception model

III. Network security situation evaluation model based on fuzzy reasoning algorithms

III. A. Fundamentals of Neural Networks

Artificial neurons are machine simulations of human brain neurons. Similar to the working mechanism of human brain neurons, artificial neurons perform mathematical operations on input data, then apply an activation function to the results to produce activation values, which serve as the output of the neuron and are passed on to the next node in the neural network. The output calculation process is shown in Figure 2. In this figure, x_i represents the input of the current neuron, w_i denotes the trainable connection weights, where the input is multiplied by the corresponding weights, b denotes the bias, and Σ denotes the weighted sum of the bias and each input. The resulting value is then mapped to an output value y through the nonlinear activation function denoted by f, with the calculation formula shown in Equation (1):

$$y = f(\sum_{i=1}^{n} x_i \cdot \omega_i + b) \tag{1}$$

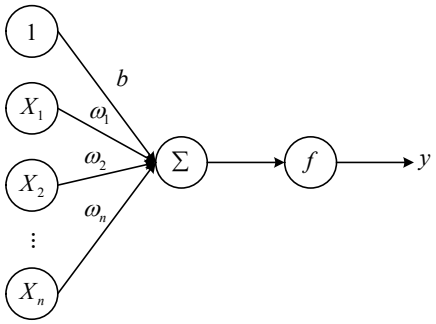

Figure 2: Neuron structure

The activation functions of neurons are generally nonlinear functions, which effectively enhance the network's representation capabilities as the number of layers increases. In practice, the sigmoid function, tanh function, and ReLU function are the most commonly used. The sigmoid function is given by Equation (2):

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2}$$

The tanh function is given by equation (3):

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3}$$

The ReLU function is given by equation (4):

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

As can be seen, the above activation functions are all nonlinear functions and share the characteristics of being differentiable and computationally simple. These properties are crucial for the normal learning and training of neural networks. Additionally, activation functions have the characteristic of activating only within specific intervals, which enables neurons to better simulate the characteristics of human brain neurons and enhances the efficiency and stability of training.

In addition to the basic structure that approximates human brain neurons, another important feature of neural networks is that they have a large number of parameters that can be learned and trained. These parameters are primarily the connection weights ω_i in the diagram. It is precisely because neural networks can efficiently train a large number of connection parameters that they have such powerful representation capabilities, enabling them to handle various artificial intelligence tasks with ease.

III. B. Adaptive Neural-Fuzzy Inference System

ANFIS combines fuzzy inference systems (FIS) and artificial neural networks (ANN), incorporating the advantages of both ANN and FIS. In the ANFIS architecture, the ANN extracts fuzzy rules from input information and adaptively utilizes the parameters of fuzzy membership functions during the hybrid learning process. ANFIS can utilize input-output data pairs and employ hybrid learning algorithms to establish associations between inputs and outputs that rely on human knowledge. ANFIS is a multi-layer feedforward network consisting of five layers [24]. The layers in the ANFIS architecture contain numerous nodes defined by node functions.

ANFIS is a hybrid technology that combines fuzzy inference systems with high reasoning capabilities. This system leverages the low-order computational capabilities of neural networks to gradually enhance performance. Fuzzy logic is a multi-valued logic that determines intermediate values between false and true. Therefore, the degrees considered may range from extremely low, low, high to extremely high. Using IF-THEN rules, a knowledge base is constructed within the fuzzy inference framework. Inputs from the database, $v(x_1), v(x_2), \cdots, v(x_n)$, are forwarded through a classifier named ANFIS to identify errors in a five-node layer structure. Based on the five layers, the first and fourth layers have adaptive nodes, while the remaining layers have fixed nodes. The ANFIS architecture is illustrated in Figure 3.

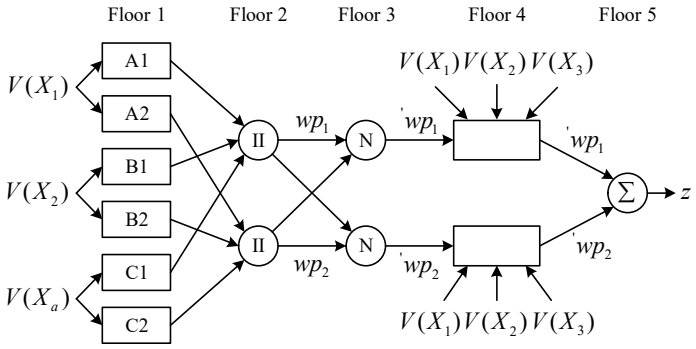

Figure 3: Structure of the ANFIS model

The rules of ANFIS are based on the following:

If $v(x_1)$ is A_j and $v(x_2)$ is B_j, then C_j is $S_j = a_j v(x_1) + b_j v(x_2) + c_j v(x_n) + g_j$, where $v(x_1), v(x_2) \cdots v(x_n)$ are the inputs, A_j, B_j, C_j are fuzzy sets, and S_j is the output within the fuzzy region defined by the fuzzy rule. a_j, b_j, c_j and g_j are parameters specified by the training process.

Layer 1 - In this layer, each j node is a square node with a node function:

$$L_{1,j} = \mu_{A1} v(x_1), L_{1,j} = \mu_{B1} v(x_2), L_{1,j} = \mu_{C1} v(x_n) \tag{5}$$

$\mu_{A1} v(x_1), \mu_{B1} v(x_2), \mu_{C1} v(x_n)$ are generally selected as bell-shaped, with values $\max = 1, \min = 0$ taken as:

$$\mu_{A1} v(x_1) = \mu_{B1} v(x_2) = \mu_{C1} v(x_n) = 1 / (1 + ((x - u_j) / v_j)^{2yj}) \tag{6}$$

where u_j, v_j, y_j are the parameters of the dataset. In this layer, these parameters are described as basic parameters.

Layer 2 - In this layer, each node is a circular node, defined as the product of the output result and the input signal:

$$L_{2,j} = wp_j = \mu_{Aj} v(x_1) \times \mu_{Bj} v(x_2) \times \mu_{Cj} v(x_n), j = 1, 2 \tag{7}$$

Each output node demonstrates the trigger strength of the rule.

Layer 3: Each node is a circular node called N. The j th node calculates the ratio of the trigger strength of the j th rule to the total trigger strength of all rules as follows:

$$L_{3,j} = wp_j = wp_j / (wp_1 + wp_2), j = 1, 2 \tag{8}$$

Layer 4: In this layer, each node j is a square node with node functionality. That is:

$$L_{4,j} = wp_j S_j, j = 1, 2 \tag{9}$$

where wp_j is the third-layer output, and a_j, b_j, c_j and g_j are the set parameters, these layer parameters are described as result parameters.

The single node in the fifth layer is a circular node labeled in this layer, and its total output is calculated as the sum of all input signals:

$$L_{s,j} = \frac{\sum_j wp_j S_j}{\sum_j wp_j} \tag{10}$$

Therefore, the predefined threshold φ is compared with the neural network output Z, which is expressed as follows:

$$output = \begin{cases} error, Z < \varphi \\ noerror, Z \geq \varphi \end{cases} \tag{11}$$

III. C. Network Security Status Assessment

III. C. 1) Cybersecurity Index System

In the process of assessing the cybersecurity landscape, the primary reason for adopting data fusion methods is their ability to extract information from multiple sensor sources. Data from multiple sources provides a more comprehensive, objective, and accurate reflection of the actual network state. In real-world networks, data sources capable of extracting information include: Intrusion Detection Systems (IDS), vulnerability scanning systems, firewalls, antivirus software, host logs, Netflow, and others. To validate the model in simulation experiments, this paper only extracts indicators reflecting security status from two major data sources: the intrusion detection system Snort and the vulnerability scanning system Nessue.

For intrusion detection systems, Snort defines three levels of attack intensity, represented by the numbers 1, 2, and 3, indicating strong, medium, and weak attacks, respectively. In this paper, metrics are selected based on a daily basis, counting the number of strong, medium, and weak attacks each host experiences within a day, while also considering the number of source IP addresses, resulting in a total of four metrics.

In terms of vulnerabilities, this paper primarily considers the threats posed by various vulnerabilities to the confidentiality, integrity, and availability of hosts. The definitions are as follows:

Confidentiality: Refers to the protection of data, programs, and other information in the network from unauthorized users or entities. That is, information can only be used by authorized users, which is the general understanding of security. Network systems must prevent unauthorized access or leakage of information.

Integrity: This refers to the original content, form, and flow of network information not being altered by human factors, i.e., it cannot be modified by unauthorized third parties.

Availability: This refers to the characteristic that information in the network can be accessed by authorized users or entities and can be used as needed. That is, network information services allow authorized users or entities to use them when needed, or when part of the network is damaged and needs to be downgraded for use, they can still provide effective services to authorized users or entities.

In terms of vulnerabilities, this paper selects four indicators: the cumulative sum of vulnerabilities in confidentiality, integrity, and availability for each host, denoted as C, I, and A, respectively, and the total number of vulnerabilities in the host, denoted as N. The calculation method can be expressed by the following formula:

$$C = e_1 c_1 + e_2 c_2 + \cdots + e_N c_N \tag{12}$$

$$I = e_1 i_1 + e_2 i_2 + \cdots + e_N i_N \tag{13}$$

$$A = e_1 a_1 + e_2 a_2 + \cdots + e_N a_N \tag{14}$$

In the above equation, c_m, i_m, a_m represent the severity values of the m th vulnerability of the host in terms of confidentiality, integrity, and availability, respectively, while e_m represents the attack complexity of the m th vulnerability. At this point, all the metrics used in this paper have been determined, as shown in Figure 4.

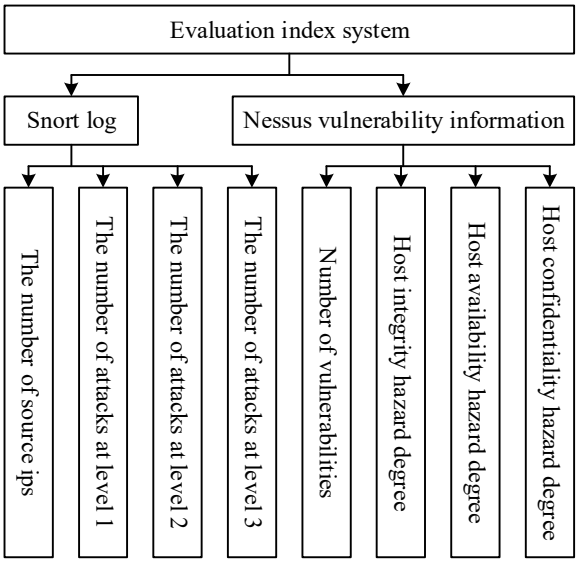

Figure 4: Index System of the evaluation model

Additionally, considering that many attacks are based on vulnerabilities, we need to verify the validity of attack events when compiling attack statistics, i.e., determine whether the attack was successful. The specific method for determining validity is as follows: when extracting security metrics from a host, for attack incidents based on specific vulnerabilities, we compare them with the vulnerability scan results to check whether the vulnerability exploited by the attack exists. If it exists, the attack is considered valid; if it does not exist, the attack is deemed invalid. For invalid attack incidents, when extracting metrics, we downgrade the attack severity, adjusting it to 3 (i.e., weak).

When associating attack events with corresponding vulnerabilities, we rely on the Bugtraq ID of the vulnerability. Bugtraq is a vulnerability database collected by Symantec Corporation's SecurityFocus organization, which describes vulnerability attributes including name, BID number, category (cause), CVE, attack source, release date, affected systems or software, and more. Additionally, it provides detailed attack methods or scripts based on the vulnerability.

III. C. 2) Network Security Status Assessment Based on ANFIS

To assess the overall cybersecurity landscape, it is necessary to first evaluate each host. By multiplying the security status value of each host by its respective weight and summing the results, the overall security status value of the

network can be obtained. The security status values of the hosts are calculated using MATLAB, as its integrated fuzzy logic toolbox includes an ANFIS editor. The ANFIS editor in MATLAB has three limitations: it only supports first-order or zero-order Takagi-Sugeno systems; it only supports single-output systems and uses weighted averaging for defuzzification; and the weight of each rule is l. These limitations do not affect the modeling in this paper.

First, establish a Takagi-Sugeno-type fuzzy system. In MATLAB, enter the “fuzzy” command to use the pop-up graphical interface for modeling. Construct eight inputs, each containing three linguistic variables {few, mid, many}, representing {low, medium, high} for each indicator. Select the “bell-shaped” membership function and establish 435 fuzzy rules based on expert experience. The “bell-shaped” membership function can be expressed by the following equation:

$$bell(x;a,b,c)=\frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}} \tag{15}$$

Examples of fuzzy rules are as follows:

If (source is many) and (attack 3 is many) and (vulnerability is mid) and (integrity is many) and (confident is few) and (available is many) Then (situation is mf405)

Among these, source, attack_3, vulnerability, etc., are all indicators, while mf405 in the latter part is a polynomial representing the linear combination of these indicators.

After constructing the Takagi-Sugeno fuzzy inference system, we installed an intrusion detection system and a vulnerability scanning system in a real network environment, extracted security metrics from them as input to the fuzzy inference system, obtained corresponding outputs, and made minor adjustments to the output results based on expert experience, thereby forming a set of samples, which were deduplicated to yield nearly 5,000 entries.

By entering the command “anfisedit” in MATLAB, the graphical interface of the ANFIS editor is displayed. The previously constructed Takagi-Sugeno system and the sample set are loaded, and training is performed using a hybrid algorithm combining backpropagation (BP) and least squares. Since the error in the original sample set was already very small, training was conducted for 50 iterations, resulting in a post-training error of less than 0.01.

It should be noted that the previously constructed Takagi-Sugeno system can already fuse the extracted security metrics to complete the situation assessment process. The subsequent training can be viewed as a form of feedback aimed at making the model more robust and better suited to real-world cybersecurity situation assessments. In terms of expert intervention, fuzzy reasoning is more suitable for cybersecurity situation assessments than neural networks.

IV. LSTM-based network security situation prediction method

IV. A. LSTM Model and Optimization Algorithm

IV. A. 1) LSTM Model

LSTM is an improved and efficient RNN. Traditional RNNs encounter gradient vanishing and explosion issues during training, but LSTM avoids these issues during training [25[].

The LSTM model update method is as follows (16):

$$\begin{cases} f_t=\sigma(W_f\times[h_{t-1},x_t]+b_f) \\ i_t=\sigma(W_i\times[h_{t-1},x_t]+b_i) \\ \tilde{C}_t=\tanh(W_c\times[h_{t-1},x_t]+b_c) \\ C_t=f_t\times C_{t-1}+i_t\times\tilde{C}_t \\ o_t=\sigma(W_a\times[h_{t-1},x_t]+b_a) \\ h_t=o_t\times\tanh(C_t) \end{cases} \tag{16}$$

x_i and h_i represent the input and output of the LSTM at time t; f,i,o represents the forget gate, input gate, and output gate, respectively; C_t represents the cell state of the LSTM; W and b represent the model's weights and biases; σ is the sigmoid function; and σ is a function that maps values in the range from negative infinity to positive infinity to the range of 0 to 1. After multiple experiments, the following hyperparameters were identified as having a significant impact on model performance:

1) Units: During training, this parameter represents the number of hidden layer neurons in the LSTM. This hyperparameter determines the dimension of the LSTM hidden layer output. A larger value for Units results in a more complex LSTM neural network, higher computational accuracy, but also greater computational complexity.

2) BatchSize: The batch size refers to the number of samples taken in a single learning iteration.

3) TimeSteps: The time step represents the number of time points each sample has when input into the LSTM prediction model. This is an internal concept of the sample and differs from BatchSize, which represents the amount of sample data input into the LSTM model in a single iteration and falls under the category of sample quantity.

IV. A. 2) Sparrow Search Algorithm

The sparrow search algorithm [26] is defined as follows:

Assume that there are N sparrows in a D-dimensional space, where d represents the dimension of the variable to be optimized. The positions of the sparrows can be described by the following matrix (17):

$$X=\begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & \cdots & x_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & \cdots & x_{n,d} \end{bmatrix} \tag{17}$$

The formula for updating the discoverer's position is as follows (18):

$$x_{ij}^{t+1}=\begin{cases} x_{ij}^{t}\cdot\exp(\dfrac{-i}{\alpha\cdot S}), R_2<ST \\ x_{ij}^{t}+Q\cdot M, R_2\geq ST \end{cases} \tag{18}$$

t is the current training count, S represents the maximum training count value, x_{ij}^{t+1} represents the value of the j th dimension of the i th sparrow in the $t+1$ th training; $\alpha\in(0,1]$; Q is randomly generated; M is a matrix of dimension $1\times d$, where all elements are 1; $R2(R2\in[0,1])$ and $ST(ST\in[0.5,1])$ represent the warning value and safety value, respectively.

When $R2<ST$, this indicates that the foraging environment is safe, and the foraging range is larger. If $R2\geq ST$, it suggests that intruders or predators may have appeared in the environment, and all sparrows need to fly to other safer areas.

Except for the discoverer, the remaining joiners update their positions as follows in equation (19):

$$x_{i,j}^{t+1}=\begin{cases} Q\cdot\exp(\dfrac{x_{worst}^{t}-x_{ij}^{t}}{i^2}) & if\ i>n/2 \\ x_{pbest}^{t+1}+|x_{i,j}^{t}-x_{pbest}^{t+1}|\cdot A^{+}\cdot L & otherwise \end{cases} \tag{19}$$

Slightly different from the original formula, the modification is as follows (20):

$$x_{i,j}^{t+1}=\begin{cases} Q\cdot\exp(\dfrac{x_{worst}^{t}-x_{i,j}^{t}}{i^2}), i>\dfrac{n}{2} \\ x_{pbest}^{t+1}+\dfrac{1}{D}\sum_{d=1}^{D}(rand\{-1,1\}\cdot|x_{i,j}^{t}-x_{pbest}^{t+1}|), i\leq\dfrac{n}{2} \end{cases} \tag{20}$$

X_{pbest}^{t+1} represents the best position occupied by the discoverer at time $t+1$, and x_{wost}^{t} represents the worst position of the entire population at time t. When $i>n/2$, this indicates that the sparrows in the worst position have low fitness and are likely to starve. At this point, it is necessary to change positions to forage. When $i<n/2$, the newcomer finds a position around the discoverer to forage.

In addition to the above two roles, in practice, 10%-20% of the sparrows in the population are able to recognize danger. The position updates for this group of sparrows are as follows (21):

$$x_{i,j}^{t+1}=\begin{cases} x_{gbest}^{t}+\beta\cdot|x_{i,j}^{t}-x_{gbest}^{t}| & if\ f_i>f_g \\ x_{i,j}^{t}+K\cdot(\dfrac{|x_{i,j}^{t}-x_{worst}^{t}|}{(f_i-f_w)+\varepsilon}) & if\ f_i=f_g \end{cases} \tag{21}$$

X_{gbest}^{t} denotes the optimal position of the sparrow at time t; β is a random number used to control the magnitude of the sparrow's position update.

$K \in [-1,1]$ is also a random number, ε ensures the rationality of the denominator, f_i represents the value of the fitness function, and f_g and f_w represent the current optimal and worst fitness values, respectively. When $f_i > f_g$, the population is likely to be attacked by outsiders. When $f_i = f_g$, the sparrows in the central position need to move to avoid being invaded by outsiders and reduce the risk of being preyed upon.

IV. A. 3) SA Algorithm

This paper reduces the overhead of the Sparrow Search algorithm based on the excellent mathematical properties and powerful search capabilities of SA. It also introduces the Sobol sequence to optimize the initial Sparrow population. The Sobol sequence is a generator that can produce low-diversity sequences, generating more uniform sequences to initialize the Sparrow population positions and reducing the differences between individuals in the Sparrow population.

Our goal is to generate a sequence of values with small deviations within a unit interval $x^1, x^2, \ldots, 0 < x^i < 1$. First, we need a set of direction numbers $v_1, v_2, \ldots$. Each v_i is a binary fraction that can be expressed as follows (22):

$$v_i = m_i / 2^i \tag{22}$$

m_i is a positive odd number, $0 < m_i < 2^i$, and v_i in the above equation is generated using the following formula, which is shown in equation (23):

$$f(x) = x^d + a_1 x^{d-1} + \ldots + a_{d-1} x + P \tag{23}$$

In the above equation, P is a constant. For $i > d$, there is formula (24) to calculate v_i:

$$v_i = a_1 v_{i-1} \oplus a_2 v_{i-2} \oplus \ldots \oplus a_d v_{i-d} \oplus [v_{i-d} / 2^d] \tag{24}$$

In the above equation, $\oplus$ represents the exclusive OR operation. Similarly, for m_i, there is a recursive formula (25):

$$m_i = 2a_1 m_{i-1} \oplus 2^2 a_2 m_{i-2} \oplus \ldots \oplus 2^d a_d m_{i-d} \oplus m_{i-d} \tag{25}$$

The formula for generating the final sequence $x^1, x^2, \ldots, x^n$ is shown in equation (26) below:

$$x^n = b_1 v_1 \oplus b_2 v_2 \oplus \ldots \oplus b_n v_n \tag{26}$$

The initial sparrow population parameters are ultimately generated through the above method. Compared with pseudo-random number sequences, the values generated by the Sobol sequence are more uniformly distributed, which further ensures the diversity of the initial sparrow population.

The SSA algorithm optimization model consists of two steps.

Step 1: Use the Sobol sequence to generate the initial sparrow population parameters.

Step 2: Select the optimal training parameters for the SSA algorithm based on the fixed point set of the SA algorithm.

IV. B. Cybersecurity Situation Prediction Model

To comprehensively assess the development trends of the network environment, this section proposes a situation prediction model based on an LSTM neural network, as shown in Figure 5. First, after inputting the situational sequence data, it is necessary to perform preprocessing such as normalization and standardization on the situational sequence data according to the relevant data preprocessing methods described in Chapter 2. Then, the processed data is input into the LSTM neural network prediction model. Combining the concept of hyperparameter optimization, the improved SSA algorithm is used to select a set of the best-performing hyperparameters for training the prediction model. Finally, the prediction results are output to facilitate adjustments to the current network environment.

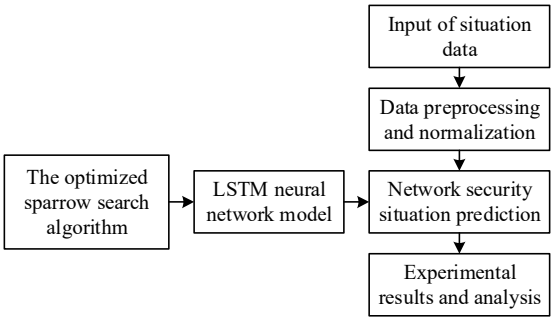

Figure 5: Prediction Steps of LSTM model

During the training process of neural network models, this paper utilizes the improved SSA algorithm from the previous subsection to optimize the neural network model in addressing the issue of hyperparameter selection. In the entire network security situation prediction process, the hyperparameters of the LSTM model largely determine the performance of the prediction model. During LSTM iterative training, fine-tuning a single hyperparameter can cause the model performance to fluctuate. To make the prediction model performance more stable, the loss function of the LSTM model training is used as the fitness function of the SSA algorithm during the LSTM model hyperparameter optimization process. The fitness function in the SSA algorithm iteration is shown in Equation (27) below:

$$fit = (\frac{1}{P}\sum_{p=1}^{p}\frac{y_t - y_p}{y_p}) \tag{27}$$

The sparrow population dimension is set to 3, representing three important hyperparameters in LSTM model training: Units, BatchSize, and TimeSteps. Through continuous iteration of the SSA algorithm, the fitness values calculated by the fitness function gradually converge to the global optimum. The parameters of the sparrow individuals that reach the global optimum are the final hyperparameters determined for training the LSTM security situation prediction model.

V. Situation assessment and prediction simulation and analysis

The simulation environment for the experiment is the Windows 10 operating system, and the evaluation process is based on the fuzzy reasoning toolbox of the Matlab 2016a simulation software. The prediction process is based on the PyCharm compilation environment, using the deep learning framework Keras written in Python 3.7 to train the model and test it. The hardware configuration is: 64-bit operating system, with an Intel Core i7-10510U processor.

V. A. Situation Assessment Simulation and Analysis

V. A. 1) Experimental Procedure

To validate the rationality and effectiveness of the evaluation methods in Chapter 3, we selected intrusion information collected from hacker groups in the Honeynet Project as the experimental dataset. The Honeynet honeypot network, comprising over 30 cybersecurity professionals, is dedicated to understanding the attack tools, network information, and behavioral patterns used by hackers during attacks, thereby effectively reflecting hackers' attack patterns.

In the Honeynet Project, a vulnerable honeypot network was specifically designed to intentionally attract attackers, enabling cybersecurity professionals to better understand hackers' attack patterns and respond effectively. The project connects to a single integrated services digital network (ISDN) via services provided by a local internet service provider, with an interconnection method similar to that of small and medium-sized enterprises and residential users.

By analyzing the Honeynet dataset, the network topology can be roughly inferred based on the IP addresses and service port information of the host devices in the dataset. The Honeynet honeypot network includes nine network devices, each corresponding to a unique IP address, with IP addresses ranging from 172.16.1.101 to 172.16.1.109. Each host provides corresponding services, including File Transfer Protocol (FTP), Remote Login (Telnet), Remote Procedure Call (RPC), Domain Name System (DNS), Socks Proxy Service, and Hypertext Transfer Protocol (HTTP). For example, Host 104 offers FTP, RPC, DNS, and Socks services. The data collection period is set from April 2000 to June 2000, and the data collected includes alert attack records within the network during this period. Combining

the environmental and asset information of critical devices mentioned in the alerts, some information is selected for display, as shown in Table 1.

Table 1: Asset Information Table in the Network

IP address	OS	Role	Port	Loophole
172.16.1.101	SMS	Main engine	21,23,53,111	CVE-1999-0789 CVE-2000-0508
172.16.1.102	Windows98	Main engine	21,23,111	CVE-1999-1544
172.16.1.103	SMS	Main engine	21,23,53,111	CVE-1999-0219 CVE-2000-1042
172.16.1.104	RedHat 6.2 Linux	Main engine	21,23,111	CVE-2000-0040 CVE-2000-0666
172.16.1.105	Windows98	Main engine	21,53,111	CVE-1999-1058
172.16.1.106	Windows NT SP4	Main engine	21,53,111	CVE-1999-1544 CVE-1999-0977
172.16.1.107	RedHat 6.2 Linux	Server	21,80,111	CVE-1999-0607 CVE-1999-0666
172.16.1.108	SMS	Main engine	21,23,111	CVE-1999-0696 CVE-1999-0704
172.16.1.109	Windows98	Main engine	21,111	CVE-1999-0704

Based on fuzzy reasoning involving three factors—alarm threat level, alarm success rate, and alarm cycle—this experiment employs the Fuzzy Logic toolbox in MATLAB. This tool provides a convenient means of implementing fuzzy logic control, eliminating the need for related fuzzy operations. Instead, it only requires setting the necessary parameters to obtain a fuzzy controller. Based on the design outlined earlier, a fuzzy inference system with three inputs and one output was constructed in Fuzzy Logic. The membership functions for the three inputs were set, and inference rules were established using the IF-THEN principle.

V. A. 2) Comparison and analysis of experimental results

To evaluate the effectiveness of the assessment model proposed in this paper in cybersecurity posture assessment, data from November was selected for analysis. The fuzzy inference toolbox in Matlab was used for simulation, and the security posture was assessed from the following three levels.

(1) Service Level

At the service level, the cybersecurity posture was assessed using the cybersecurity asset configuration information and service information provided by hosts in the Honeynet Project. The security posture was analyzed from three aspects: FTP service, RPC service, and HTTP service. Figure 6 shows the service-level cybersecurity posture assessment values for Host 107. First, as shown in the figure, the FTP service assessment value for Host 107 is relatively low, while the RPC service posture assessment value is relatively high. This indicates that the RPC service is highly susceptible to attacks by hacker groups and may also suggest the presence of vulnerabilities or multiple security flaws in the service. Therefore, cybersecurity personnel should inspect the devices providing this service or update the relevant programs. Second, the most frequent attack periods are typically weekends, indicating that systems are more susceptible to hacker attacks during these times. This serves as a reminder for network security personnel to strengthen defenses during weekends. Third, for different hosts, specific services can be prioritized for monitoring during periods with higher threat assessment values. Finally, based on attack frequency and timing, it can be inferred that the attackers may be part-time hackers, which is effective for narrowing down the scope of potential attackers.

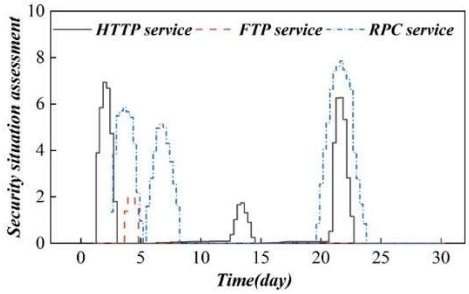

Figure 6: Service-level security posture assessment value of 107 hosts

(2) Host Level

Assess the network security posture at the host level. Select host 109 and host 102 to draw a host-level security posture diagram. Figure 7 shows the security posture assessment values for host 102 and host 109. Similar to the analysis process for service-level posture assessment results, the figure indicates that, first, hosts are more susceptible to threat incidents around weekends, so administrators should prioritize security during this period and strengthen security management of hosts within network devices. Second, for a specific host within a network device, if it is attacked on a particular day, hackers may continue to target it, so security for that host should remain a priority in the short term. Furthermore, when comparing the threat levels of the two hosts, Host 109 is more susceptible to external attacks and requires special attention from network administrators. Finally, during periods with high threat levels, certain ports and services can be disabled to minimize losses caused by network attacks.

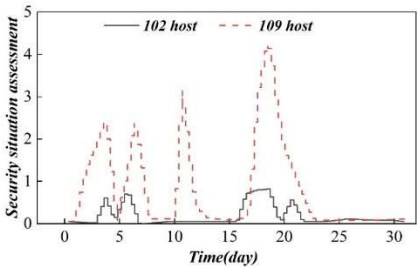

Figure 7: The security posture assessment value of the 102 host and the 109 host

(3) System-level

Assessing the cybersecurity posture at the system level involves a comprehensive evaluation of the cybersecurity posture based on three aspects: service level, host level, and alert level. Figure 8 shows the system-level cybersecurity posture assessment values for the Honeynet system. From the figure, it can be concluded that, on one hand, the cybersecurity posture values during weekends are relatively high, indicating that cyberattacks are more frequent on weekends, and the system faces more severe threats. Therefore, enhanced protection measures should be implemented during this period. On the other hand, after experiencing significant and severe attacks, the likelihood of further attacks remains high in the short term. Consequently, cybersecurity personnel need to implement protective measures for network devices and promptly monitor and inspect network status in the coming days.

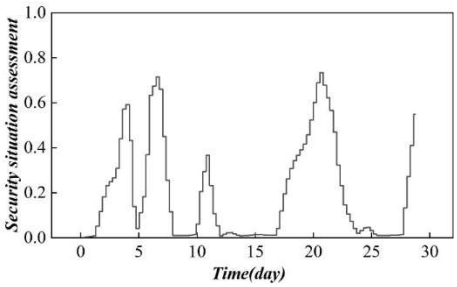

Figure 8: Honeynet system-level network security situation assessment value

V. B. Situation Prediction Simulation and Analysis

V. B. 1) Experimental Procedure

This paper will analyze and validate the proposed network security situation prediction algorithm based on the SSA-LSTM neural network. The analysis will use network security protection data obtained from a certain company, and the network security situation threat value will be obtained based on the situation assessment method proposed in Chapter 3. Using 400 sample data points for experimental analysis and comparison, the first 350 sample data points are used as the training set to train and optimize the neural network in real time, while the remaining 50 sample data points are used as the test set to evaluate the algorithm's predictive performance.

In the experiment, a neural network training model was first established, with the security posture vector dimensions of the past 7 days as input and the vector values of the following day as output, i.e., for predictive analysis. The number of hidden layer neurons was adjusted in real-time during the experiment. Table 2 shows the input and output information of the samples.

In this experiment, RNN, LSTM, and SSA-LSTM models are selected for analysis and comparison. When setting the model hyperparameters, the improved SSA is used to optimize and select the hyperparameters. The sparrow population size is 50, the iteration count is 400, and the sparrow individual dimension is 3, representing the hyperparameters that need to be optimized in LSTM training, namely Units, Batch Size, and Time Steps.

Table 2: Sample input output information

Sample number	Sample input	Sample output
1	$X_1, X_2, X_3, X_4, X_5, X_6, X_7$	X_8
2	$X_2, X_3, X_4, X_5, X_6, X_7, X_8$	X_9
3	$X_3, X_4, X_5, X_6, X_7, X_8, X_9$	X_{10}
…	…	…
N	$X_n, X_{n+1}, X_{n+2}, \ldots, X_{n+6}$	X_{n+7}

V. B. 2) Experimental Results and Analysis

After training the above three models, their performance was validated using two metrics: prediction accuracy and convergence efficiency. Experimental results were obtained on both the training set and the test set.

(1) Convergence Efficiency

To validate the convergence efficiency of the SSA-LSTM prediction algorithm, it was trained alongside the RNN prediction algorithm and the LSTM prediction algorithm on the same test set, and their error convergence performance was examined. Figure 9 illustrates the differences among the three algorithms. It can be observed that the RNN exhibits a large initial error, significant fluctuations in error convergence, and a longer time to converge to the optimal error value, requiring over 170 iterations. The LSTM algorithm with more hidden layers has a smaller initial MSE compared to RNN and converges faster, requiring 130–170 iterations. However, the larger number of parameters makes it prone to getting stuck in local optima, resulting in error performance that is only marginally better than RNN overall. The SSA-LSTM algorithm, however, starts with a smaller error and exhibits significantly faster convergence speed than the previous two algorithms, with shorter convergence time and higher stability. Therefore, it can be concluded that incorporating the improved Sparrow Search Algorithm into the LSTM neural network indeed improves the network's convergence performance.

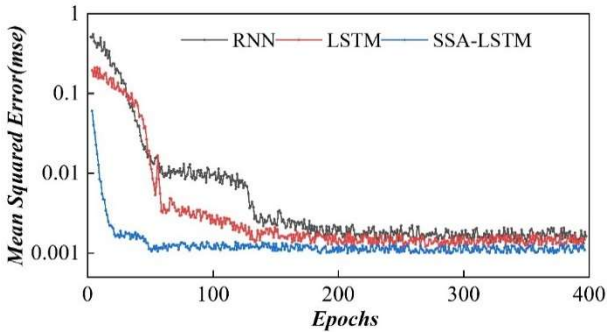

Figure 9: Comparison of MSE convergence rate of algorithms

(2) Prediction Accuracy

To verify the prediction accuracy of the improved SSA-LSTM prediction algorithm, the three prediction methods are analyzed and compared here. Figure 10 shows the normalized situation prediction curves for the last 50 days for the three methods. Through comparative analysis, it was found that all three prediction algorithms exhibit good predictive performance for the security situation over the next period. However, the improved SSA-LSTM algorithm demonstrates superior ability to track and analyze security trend dynamics, overcoming the original algorithm's tendency to converge on local optima. This improvement is primarily attributed to the enhanced SSA algorithm's rapid global search capability during parameter optimization, enabling it to swiftly identify the optimal parameter set.

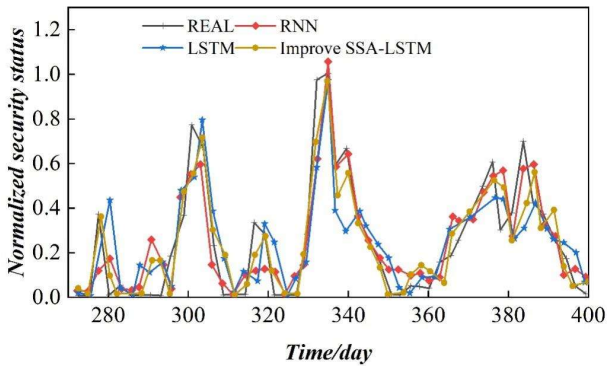

Figure 10: Normalized situation prediction curve

Here, we test the actual prediction performance of three prediction algorithms in detail. For the RNN network, we select test points from the training set. Figure 11 shows the residual distribution of the RNN. Analysis reveals that the first 230 training samples effectively exhibit a horizontal band pattern in residuals, but the corresponding error tolerance range fluctuates between ±1. Subsequent samples exhibit significant fluctuations, leading to a noticeable decline in prediction accuracy. This is attributed to the “forgetting” issue associated with RNN's long-term memory. Overall, the error under RNN training remains relatively high.

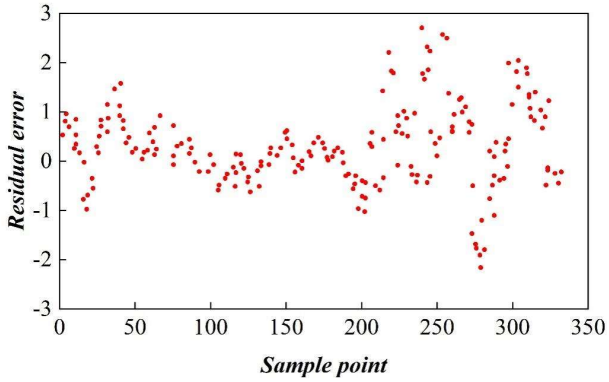

Figure 11: RNN residual difference cloth

For the LSTM network, test points were selected from the training set. Figure 12 shows the residual distribution of the LSTM. Analysis reveals that the first 200 or so training samples can effectively illustrate the horizontal band pattern of the residuals, with the corresponding error tolerance range fluctuating between -0.5 and 0.5. Subsequent test points exhibit a noticeable improvement in horizontal band diffusion compared to RNN networks, indicating that LSTM indeed has a significant impact on memory, leading to relatively favorable values in subsequent prediction results.

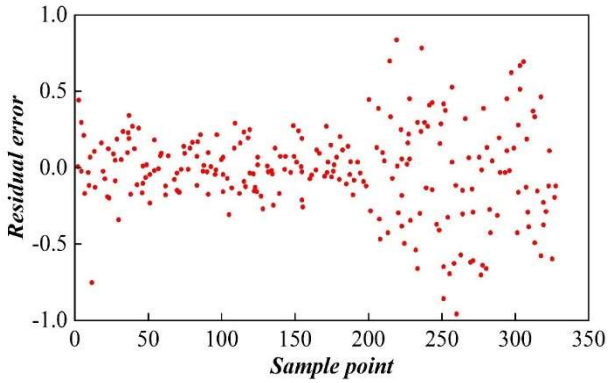

Figure 12: LSTM residual difference cloth

For the improved SSA-LSTM network, test points were selected from the training set. Figure 13 shows the residual distribution of the improved SSA-LSTM. Analysis reveals that the improved method exhibits a good residual level band pattern across the entire training sample, and the residuals remain within a relatively small range, indicating that the improved SSA-optimized LSTM does indeed perform well.

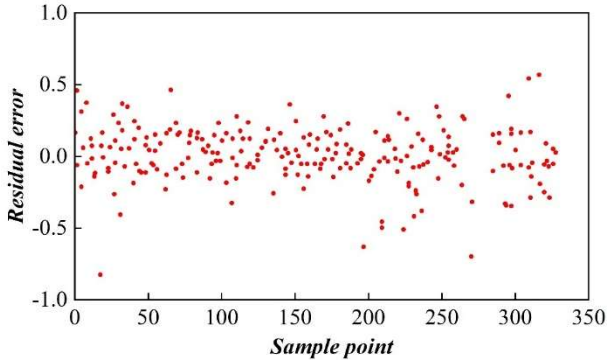

Figure 13: SSA-LSTM residual difference cloth

To further compare the actual prediction performance of the three algorithms, Table 3 shows the mean square error and average relative error for different methods. It can be seen that among the three, the improved SSA-LSTM prediction algorithm performs better, with the smallest values for both metrics (0.1055 and 2.18%), indicating its strong predictive capability. This also reflects that the accuracy of the prediction algorithm has indeed improved after parameter optimization.

Table 3: Prediction algorithm error comparison

Prediction model	RNN	LSTM	Improve SSA-LSTM
MSE	0.3358	0.2267	0.1055
MRE	4.35%	3.66%	2.18%

After training the improved SSA-LSTM neural network prediction algorithm proposed in this paper, it was tested separately on the test set, and the relative error for each sample was calculated. It can be seen that although the accuracy of the prediction algorithm proposed in this paper on the test set is slightly lower than that on the training set, the overall prediction accuracy is still acceptable, with the average relative error stabilizing at around 5.29%. The specific error analysis for the test set is shown in Table 4. (only 45 samples are listed here). Additionally, a security situation prediction curve was plotted for the test set, comparing the actual security situation values with the predicted values from the improved SSA-LSTM algorithm. The NSSP output comparison for the test set is shown in Figure 14. The results indicate that the proposed prediction algorithm can effectively predict the network security situation for the coming period.

Table 4: Improve the analysis of SSA-LSTM test set error

Number	Relative error	Number	Relative error	Number	Relative error
1	8.24%	16	-4.18%	31	5.98%
2	-9.27%	17	4.69%	32	-5.16%
3	-8.51%	18	-0.14%	33	2.74%
4	6.04%	19	-9.12%	34	-8.27%
5	-5.78%	20	-0.24%	35	-4.77%
6	-5.23%	21	0.44%	36	0.28%
7	9.72%	22	7.91%	37	-10.37%
8	2.19%	23	9.23%	38	-0.82%
9	3.41%	24	5.56%	39	-8.26%
10	0.44%	25	-8.41%	40	2.45%
11	-4.9%	26	-9.14%	41	-1.21%
12	-2.84%	27	-10.58%	42	2.55%
13	10.67%	28	-7.84%	43	2.71%
14	-0.02%	29	4.64%	44	-5.84%
15	4.60%	30	1.90%	45	-10.83%

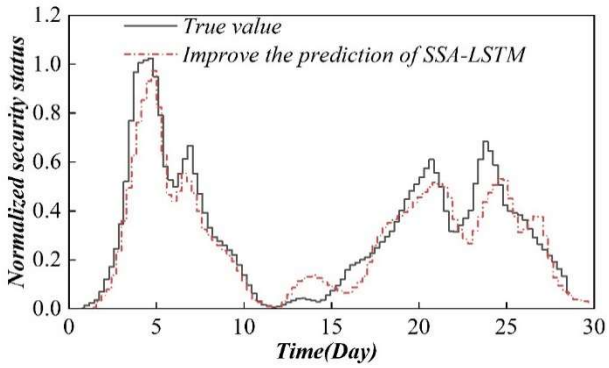

Figure 14: The NSSP output is compared to the test set

VI. Conclusion

Cybersecurity machine applications are now widely used and play a particularly important role, but they cannot quickly and directly reflect changes in the cybersecurity situation. To help network administrators with analysis, cybersecurity situation assessment was developed.

In terms of NSSE, this paper establishes a cybersecurity situation assessment model. When creating the model, a neural network system and a fuzzy reasoning system were combined, and an adaptive neural fuzzy reasoning system was used to achieve data fusion from multiple information sources. This system not only possesses the imitation capabilities and self-adaptive nature of neural networks but also complements classical fuzzy systems. By quantitatively calculating cybersecurity status values at different levels—service level, host level, and network system level—and displaying threat states at each level in a layered manner, this evaluation model provides more comprehensive assessment information and more accurate results compared to traditional methods.

In terms of NSSP, addressing the issues of poor applicability and low accuracy of traditional network security situation prediction algorithms, an improved SSA-LSTM neural network-based network security situation prediction algorithm was proposed. The optimized prediction model of the improved SSA algorithm outperforms the original model, with an average relative error stabilizing around 5.29%, enabling accurate prediction of the network security situation for the coming period.

In the next phase of work, this paper will continue to focus on theoretical research in the field of cybersecurity situation awareness. To address the current issue of insufficiently intuitive data presentation, a visualization-based intelligent information display platform will be constructed to accurately and clearly present cybersecurity situation information through more intuitive means, which will also facilitate the effective prevention and control of cybersecurity attack incidents.

References

[1] Hong, H. J., El-Ganainy, T., Hsu, C. H., Harras, K. A., & Hefeeda, M. (2017). Disseminating multilayer multimedia content over challenged networks. IEEE Transactions on Multimedia, 20(2), 345-360.

[2] Nauman, A., Qadri, Y. A., Amjad, M., Zikria, Y. B., Afzal, M. K., & Kim, S. W. (2020). Multimedia Internet of Things: A comprehensive survey. Ieee Access, 8, 8202-8250.

[3] Xu, G., Ngai, E. C. H., & Liu, J. (2017). Ubiquitous transmission of multimedia sensor data in Internet of Things. IEEE Internet of Things Journal, 5(1), 403-414.

[4] Seng, K. P., & Ang, L. M. (2018). A big data layered architecture and functional units for the multimedia Internet of Things. IEEE Transactions on Multi-Scale Computing Systems, 4(4), 500-512.

[5] Piepponen, A., Ritala, P., Keränen, J., & Maijanen, P. (2022). Digital transformation of the value proposition: A single case study in the media industry. Journal of Business Research, 150, 311-325.

[6] Zhang, Z., & Gupta, B. B. (2018). Social media security and trustworthiness: overview and new direction. Future Generation Computer Systems, 86, 914-925.

[7] Yang, W., Wang, S., HuHu, J., & Karie, N. M. (2022). Multimedia security and privacy protection in the internet of things: research developments and challenges. International Journal of Multimedia Intelligence and Security, 4(1), 20-46.

[8] Yang, J., He, S., Lin, Y., & Lv, Z. (2017). Multimedia cloud transmission and storage system based on internet of things. Multimedia Tools and Applications, 76, 17735-17750.

[9] Lien, C. W., & Vhaduri, S. (2023). Challenges and opportunities of biometric user authentication in the age of iot: A survey. ACM Computing Surveys, 56(1), 1-37.

[10] Singh, A. K., Kundur, D., & Conti, M. (2024). Introduction to the special issue on integrity of multimedia and multimodal data in Internet of Things. ACM Transactions on Multimedia Computing, Communications and Applications, 20(6), 1-4.

[11] Noman, H. A., & Abu-Sharkh, O. M. (2023). Code injection attacks in wireless-based Internet of Things (IoT): A comprehensive review and practical implementations. Sensors, 23(13), 6067.

[12] Makhdoom, I., Abolhasan, M., Lipman, J., Liu, R. P., & Ni, W. (2018). Anatomy of threats to the internet of things. IEEE communications surveys & tutorials, 21(2), 1636-1675.

[13] Rizvi, S., Orr, R. J., Cox, A., Ashokkumar, P., & Rizvi, M. R. (2020). Identifying the attack surface for IoT network. Internet of Things, 9, 100162.

[14] Mohindru, V., & Garg, A. (2020, March). Security attacks in internet of things: A review. In The International Conference on Recent Innovations in Computing (pp. 679-693). Singapore: Springer Singapore.

[15] Lv, Z., Qiao, L., & Song, H. (2020). Analysis of the security of internet of multimedia things. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 16(3s), 1-16.

[16] Hurrah, N. N., Parah, S. A., Loan, N. A., Sheikh, J. A., Elhoseny, M., & Muhammad, K. (2019). Dual watermarking framework for privacy protection and content authentication of multimedia. Future generation computer Systems, 94, 654-673.

[17] Mishra, D., Vijayakumar, P., Sureshkumar, V., Amin, R., Islam, S. H., & Gope, P. (2018). Efficient authentication protocol for secure multimedia communications in IoT-enabled wireless sensor networks. Multimedia Tools and Applications, 77, 18295-18325.

[18] Dhar, S., Khare, A., & Singh, R. (2023). Advanced security model for multimedia data sharing in Internet of Things. Transactions on Emerging Telecommunications Technologies, 34(11), e4621.

[19] Zhu, G., Li, X., Zheng, C., & Wang, L. (2022). Multimedia fusion privacy protection algorithm based on iot data security under network regulations. Computational Intelligence and Neuroscience, 2022(1), 3574812.

[20] Park, N., & Lee, D. (2018). Electronic identity information hiding methods using a secret sharing scheme in multimedia-centric internet of things environment. Personal and Ubiquitous Computing, 22, 3-10.

[21] Singh, I., & Lee, S. W. (2022). Self-adaptive and secure mechanism for IoT based multimedia services: a survey. Multimedia Tools and Applications, 81(19), 26685-26720.

[22] Luo, M., & Zhao, R. (2018). Fuzzy reasoning algorithms based on similarity. Journal of intelligent & fuzzy systems, 34(1), 213-219.

[23] Liu, S., Wang, S., Liu, X., Dai, J., Muhammad, K., Gandomi, A. H., ... & de Albuquerque, V. H. C. (2022). Human inertial thinking strategy: A novel fuzzy reasoning mechanism for IoT-assisted visual monitoring. IEEE Internet of Things Journal, 10(5), 3735-3748.

[24] Bowen Liu, Junbin Chen, Xiaoguang Zhang & Zhenwei Wang. (2025). Usage of the dwarf mongoose optimization-based ANFIS on the static strength of seasonally frozen soils. Journal of Engineering and Applied Science,72(1),71-71.

[25] Hoang Tran, Tian Zhou, Zeli Tan, Yilin Fang & L. Ruby Leung. (2025). Improving the prediction of daily reservoir releases over the CONUS using conditioned LSTM. Journal of Hydrology,661(PC),133750-133750.

[26] Zongyao Wang, Qiyang Peng, Wei Rao & Dan Li. (2025). An improved sparrow search algorithm with multi-strategy integration. Scientific Reports,15(1),3314-3314.