ingegneria sismica
International Journal of Earthquake Engineering
Anno XLIII - Num. 1 - 2026

Research on Optimization Algorithm of Artificial Intelligence Network Security Management under Internet of Things Convergence Technology

Chengxin Wei[1] *and Kejiu Tan*[2,*]

[1] Information Technology Center, Guangxi University of Chinese Medicine, Nanning, Guangxi, 530200, China
[2] Modern Educational Technology Center, Hechi University, Hechi, Guangxi, 546300, China

SUMMARY: *In this paper, a cybersecurity posture assessment model based on the attention mechanism is put forward to assess the cybersecurity posture in terms of IoT fusion technology. In order to make progress in artificial intelligence-based cybersecurity posture prediction under the condition of IoT fusion, a prediction model named INGWO-LSTM is presented with the aid of an improved small-life realm gray wolf optimizer. The performance of INGWO-LSTM in cybersecurity posture prediction is compared with single prediction models. On the basis of the framework for assessing and predicting cybersecurity postures in this paper, an exploration is made into the effect of the optimization of network security management using practical cases. The results show that the proposed model can effectively recognize 13 out of 14 typical attack types currently employed. Except for Heartbleed and Web Attack-SQL Injection, the coverage rate of attack event sensing time is higher than 90% by the proposed method. Besides, the INGWO-LSTM prediction model is capable of presenting high optimization precision, consistency between the network structure and data, and greatly improved robustness. Based on the practical case study, it is shown that the attack type recognition accuracy of the INGWO-LSTM algorithm remains at a level higher than 90%, and the final prediction accuracy is 93.24%.*

KEYWORDS: *INGWO-LSTM; network security; situational assessment model; small habitat gray wolf optimization algorithm*

1 Introduction

With the fast development of ICT, the widespread application of IoT technology, and the increasing integration of computers and networks in every aspect of modern society [1, 2]. This deep integration not only greatly improves productivity and convenience of life, but also leads to an explosive growth of network traffic and attracts numerous attackers [3]. However, the increase in network traffic and complexity has also brought unprecedented challenges to network security. Taking advantage of the increasingly complex network environment, attackers have devised new types of attacks to bypass traditional security measures in order to gain financial benefits or damage critical infrastructures [4]. Traditional firewalls mainly filter network traffic through preset rules, which can block some of the known attacks, but are powerless against unknown threats and new attack methods that bypass the rules [5]. Intrusion detection systems rely on feature matching to detect intrusion behavior, which makes it difficult

*weicxx2025@163.com
https://doi.org/10.65102/is2026431
Published: 30 April, 2026

to accurately identify and respond in a timely manner in the face of massive amounts of data and ever-changing attack methods [6]. Moreover, traditional methods lack active defense capabilities and cannot effectively prevent attacks before they occur. When dealing with diversified and complex network threats, traditional security protection methods appear to be incompetent and unable to comprehensively safeguard network security.

Regarding Intrusion Detection Systems in the area of network security defense and detection, artificial intelligence plays an important role. It analyzes massive network traffic data through machine learning algorithms, learns normal network behavior patterns, and once it detects behaviors that significantly deviate from normal patterns, it can quickly judge them as intrusion behaviors and respond in a timely manner [7, 8]. According to Muneer, S et al [9], artificial intelligence technology had exhibited remarkable performance when detecting intrusions due to automatic feature learning from raw data, however, large amounts of labeled data and huge computing power were required for training the complicated model. In Vanin, P et al [10]'s view, the intrusion detection system was used for recognizing malicious activities by analyzing the network traffic, while machine learning algorithms applied in artificial intelligence models could detect the intrusions in network with great efficiency and precision. According to Patil, S et al [11], an intrusion detection system based on decision trees, random forest, and support vector machines had been created with an accuracy rate of 96.25%, as a result of applying the ensemble classifier with voting after training. Alladi, T et al [12] designed an intrusion detection system based on artificial intelligence technology, which was used for identifying and classifying the network traffic of vehicles as the cyber-security threat, with reference to the mobility properties of the vehicle and the real-time characteristic of the IoT network. Alohali, M et al [13] introduced a multimodal fusion intrusion detection system referred to as AIMMF-IDS.

The development of deep learning techniques has provided opportunities for researchers to explore new methods of network security protection in recent times. The deep learning models have automatic feature extraction from complex datasets, minimizing human involvement in feature engineering. They exhibit significant potential in handling multidimensional and non-linear data. Alazab, M et al [14] proposed a method based on the Moth-Flame optimizer algorithm that can be used to optimize the parameters of SVM to detect network traffic intrusion. As compared to the traditional SVM approach, this technique improves intrusion detection by tuning SVM parameters with an increase in accuracy and minimization of the false alarm rate. Jiang, H et al [15] developed the PSO-XGBoost model, using the search capabilities of the Particle Swarm Algorithm to optimize XGBoost parameters and improving the efficiency and accuracy of intrusion detection in networks. Zhang, L et al [16] proposed the Convolutional Neural Networks and Bidirectional Long Short-Term Memory Neural Networks model for mining the interactions of network intrusion datasets, using convolutional neural networks for image type data and long short-term memory neural networks for time series data. Yang, Y et al [17] proposed the unsupervised clustering and Deep Belief Network model for network traffic multi-classification. Using density peak clustering and feature clustering for dividing attacks into different classes before classification, the model is highly efficient in detecting DoS attacks.Zhang, Y et al [18] created an intrusion detection system with a combination of Genetic Algorithms with Deep Belief Networks, making it possible to adaptively modify the structure of the network to incorporate different attack types in the context of IoT systems. GSR, E. S et al [19] developed a feature fusion mechanism with deep learning for DDoS attack detection in cloud computing networks, creating a better representation of network traffic with deep neural networks through integration of feature extraction algorithms to enhance DDoS attacks detection capabilities. Hassan, M et al [20] suggested a hybrid deep learning method where they incorporated Convolutional Neural Networks with a Weighted Descent Long Short-Term

Memory Network for use in the detection of network-based attacks. Specifically, a deep CNN was used to obtain the key features of the large amounts of data while the pooled outputs were then fed into the WDLSTM in order to learn about the relationship between the features.

Ahmim, A et al [21] discussed the damaging impact of DDoS attacks on IoT networks and developed a deep learning model for detecting network intrusion as an attempt to overcome the shortcomings in using conventional machine learning techniques, which cannot effectively deal with the characteristics of actual traffic data and cannot detect DDoS attacks accurately. Beulah, J and Punithavathani, D [22] developed a feature selection method for identifying network intrusions using several feature selection models and combining their results to get the most important features, which can be applied to the problem of feature compression for any application domain. Blanco, R et al [23] presented a hybrid dimensionality reduction technique through the combination of Information Gain and PCA algorithm and built an intrusion detection model by integrating SVMs, IBLA, and MLP models into one model. The hybrid dimensionality reduction technique performs significantly better than individual algorithms in terms of performance and is proven to reliably recognize normal data from anomalies. A SAAE-DNN model, which combines the use of stacked autoencoders and deep neural networks in order to improve the feature extraction process of multiclassification intrusion detection was introduced by Tang, C et al [24]. Furthermore, unsupervised pre-training of the deep neural network was used to improve the classification capability of DNN. Duan, L et al [25] applied an improved artificial bee colony algorithm in optimizing the initial weights and thresholds of BP neural network to avoid convergence of the model at local optima and increase training speed. This resulted into a highly effective method with excellent intrusion detection performance. Wang, X and Wang, L [26] used the combination of autoencoder with an improved K-means algorithm, where the encoding part of the autoencoder reduces the dimensions of high dimensional network intrusion dataset, and then improved K-means algorithm can be applied to solve the problem. Seo, S et al [27] constructed an abnormal traffic detection model based on LSTM Networks, which predicts the future value of traffic data in advance in order to enhance the detection efficiency without decreasing the detection rate, but the method is still dependent on manual feature extraction.

The above decisions about AI in cybersecurity protection often lack interpretability, which has many implications for cybersecurity. When the model detects abnormal behaviors or threats, security personnel have difficulty in understanding the basis of their judgments and are unable to determine whether the decisions are correct or not, which may lead to over-reliance or blind questioning of the model and affect the effectiveness of security protection [28-30]. Therefore, improving the interpretability of network security algorithms has become a hot research topic nowadays. Cybersecurity in the context of IoT has been seen to be very crucial and that using machine learning as the basis for Intrusion detection systems can improve the security of the IoT but decision making of these IDSs remains a black box hence increasing their interpretability is one way to proceed in the future, according to Mohale, V and Obagbuwa, I [31]. According to Zouhri, H and Idri, A [32], there was a creation of a unique empirical framework that allowed for the evaluation and comparison of filter and wrapper feature selection techniques on four different black boxes and realized that the use of interpretable models using global surrogates is a promising solution that allows to have a balance between the model's predictive power and its interpretability. Lv, M et al [33] developed a decision tree augmented deep attention network referred to as TAN and increased the explainability of the network intrusion detection system design. It has been discovered that TAN performs similarly to the existing deep learning networks but gives more explanation of its output. Djenouri, Y et al [34] have proposed a new approach of intrusion detection in the IoT setting where they normalize data collection/preprocessing by using min max approach together with recurrent neural networks having an attention module to compute Shapley values.

In this paper, for the purpose of analyzing the whole situation of network security posture, the entire dataset is divided into various kinds of posture elements categories. Features are extracted for every data by using the Bi-LSTM model and after that, the fused vector is gained via using an attention-based method for feature fusion. A 1D CNN is adopted to analyze network posture, and in order to solve the unbalanced data issue, the Focal loss is applied to regulate misclassified costs. As for network security posture prediction, an INGWO-LSTM prediction model is put forward, in which first of all, the initialization scheme and convergence factor of the Gray Wolf Optimizer are modified. In addition, small habitat technique is introduced to enhance its optimization performance. The INGWO-LSTM model is finally applied to a publicly available intrusion detection dataset, and hence the network attacks and corresponding security posture values are determined.

2 Analysis of the results of the cybersecurity posture assessment based on the attention mechanism

2.1 Data fusion posture assessment model based on attention mechanism

2.1.1 Experimental data set

The experiments in this paper use the CIC-IDS2024 dataset, which was formed by researchers by collecting network traffic from the network environment, and 69 features and attack type labels were extracted using the CICFlowMeter tool, which contains 14 attack types that are common today. In the design of testbed network architecture, the network is split up into two totally independent networks: the Victim-Network and the Attack-Network. Hosts use common operating systems, which include Windows, Linux, and Macintosh on various releases.

The change of network security status occurs in the attacked party, and this paper conducts experiments from the perspective of the attacked party. The basic information of the attacked party's network is shown in Table 1. It describes the basic information of Victim-Network servers, workstations and firewalls.

Table 1: Basic information of the attacked network

Device type	Operating system	IP address
Server	Windows Server 2024 (DC and DNS)	192.168.10.3
	Ubuntu 16 (Web Server)	192.168.10.50-205.174.165.68
	Ubuntu 12	192.168.10.51-205.174.165.66
Personal computer	Ubuntu 14.4 (32,64)	192.168.10.19-192.168.10.17
	Ubuntu 16.4 (32-64)	192.168.10.16-192.168.10.12
	Windows 7Pro	192.168.10.9
	Windows 8.1-64	192.168.10.5
	Windows Vista	192.168.10.8
	Windows 10 (Pro 32-64)	192.168.10.14-192.168.10.15
	Mac	192.168.10.25
Firewall	Fortinet	—

The traffic data captured includes normal traffic from 8:00 to 11:00 on October 1, 2024, as well as data from October 3 to October 6, from 8:00 to 18:00 each day, and the CIC-IDS2024 attack scenarios and attack types are shown in Table 2.

Table 2: CIC-IDS2024 Attack Status and Attack Types

Date	Time	Attack type
3 October 2024	8:20 – 10:20	FTP-Patator
	14:00 – 15:00	SSH-Patator
4 October 2024	9:47 – 10:10	DoS slowloris
	10:14 – 10:35	DoS Slowhttptest
	10:43 – 11:00	DoS Hulk
	11:10 – 11:23	DoS GoldenEye
	15:12 – 15:32	Heartbleed
5 October 2024	9:20 – 10:00	Web Attack-Brute Force
	10:15 – 10:35	Web Attack-XSS
	10:40 – 10:42	Web Attack-Sql Injection
	14:19 – 14:21, 14:33 - 14:35, 14:53 - 15:00, 15:04 - 15:45	Infiltration
6 October 2024	10:02 – 11:02	Botnet
	Firewall Rule on: 13:55 - 14:35, Firewall Rules off: 14:51 - 15:29	Port Scan
	15:56–16:16	DDoS LOIT

2.1.2 Cybersecurity posture characterization

As LSTM has been found to be performing well in sequence data analysis, this paper uses the Bi-LSTM neural network to extract features. The Bi-LSTM neural network takes care of posture elements in two different ways, and important feature information is extracted, which provides technical support for the subsequent network security posture that can be assessed more comprehensively.

The Bi-LSTM-based network posture feature extraction process is as follows: firstly, the data of posture elements over a period of time are combined to form $X_i = [f_1, f_2, f_3, \cdots, f_T]$, where $f_j \in R^{D_1}$ denotes the data of the j th time in the i th channel, and the dimension D_i. Subsequently, as inputs to the network model, the two LSTMs compute the inputs at each time step independently and generate a forward hidden state sequence $H_f = \{h_1, h_2, \cdots, h_N\}$ and a backward hidden state sequence $H_b = \{h_N, h_{N-1}, \cdots, h_1\}$. Finally, the output is obtained by combining the structures computed at both layers:

$$cf_i = BiLSTM\left(X_i\right) \tag{1}$$

where $cf_i \in R^d$ denotes the output feature vector of channel i and d is the dimension of the output vector of the Bi-LSTM layer.

2.1.3 Feature fusion

In this paper, we use the concat feature fusion method to combine the four dimensional feature vectors of cf_1, cf_2, cf_3, cf_4 with their dimensions d, which can be spliced into a new vector $F = \left[cf_1, cf_2, cf_3, cf_4\right]$. However, the splicing operation increases the dimensionality of the feature vectors and introduces some redundant information.

The feature vectors resulting from concat splicing are operated with the attention matrix to obtain the attention score:

$$e_i = F \cdot W_a \cdot w_i^a \tag{2}$$

where the attention weight matrix $W_a \in R^{d \times d}$, and $w_i^a \in R^d$ is the attention weight vector. Each attention weight is:

$$a_i = \frac{\exp(e_i)}{\sum_{j=1}^{n} \exp(e_j)} \tag{3}$$

The spliced feature vectors and attention weights are weighted and summed to obtain the final feature vector F_{att}:

$$F_{att} = \sum_{i=1}^{n} a_i cf_i \tag{4}$$

By fusing the features using the attention mechanism, the features that are more critical and meaningful to the assessment of cybersecurity posture can be further extracted, so as to more accurately reflect the actual operation of the cyberspace and the risks it faces, and give the corresponding posture assessment value.

2.1.4 Cybersecurity posture assessment models

In the cybersecurity posture assessment module, the fused data are feature extracted and modeled using a 1DCNN+BN+RELU network.

First, feature extraction is performed on the sequence data using 1D convolution. Second, to speed up the training of the model and reduce the risk of overfitting use BN for normalization. Finally, the feature matrix M after convolution is output using ReLU function for its activation:

$$M^l = \mathrm{Re}LU^l\left(BN^l\left(conv1D\left(F_{att}\right)\right)\right) \tag{5}$$

l denotes the layer order. Subsequently, the output features are sampled using the maximum pooling layer:

$$gm = MaxPooling\left(M^l\right) \tag{6}$$

Finally, the features are fully connected and the softmax function is used to output the level of the network security posture in the time period. In addition, in order to better respond to the network posture situation, the obtained posture category belongs to the probability of processing to generate the corresponding posture value, to facilitate the subsequent experimental comparison and evaluation, the specific posture value calculation formula is as follows:

$$value = \sum a_i \cdot x_i \tag{7}$$

where a_i is the standardized value of the security posture corresponding to the category and x_i is the probability of the category predicted by the neural network.

Focal Loss is a loss function designed to address the category imbalance problem. It introduces an adjustable parameter γ to regulate the degree of suppression applied to easy-to-categorize samples. For hard-to-categorize samples, Focal Loss retains the original loss value, while for easy-to-categorize samples, it reduces their weights accordingly. The Focal Loss formula is as follows:

$$FL(p_t) = -(1 - p_t)^{\gamma} \log(p_t) \tag{8}$$

where p_t is the probability of the network output, and γ is a tunable parameter that allows the trained network model to focus more on difficult-to-classify samples by setting γ.

2.2 Simulation results for cybersecurity posture assessment

2.2.1 Experimental procedure

The 5-day traffic dataset is filtered to find out the basic information of network security posture ID, Timestamp, DIP, DP, Attack for each traffic. the experiment takes the traffic data with t = 500s as the time unit as the sample, and the overlapping time difference between two neighboring samples is 50s, which forms the sample of the system layer; according to the IP address information of the devices in the server zone and the PC zone, the division of the Based on the IP address information of the devices in the server area and the personal computer area, each sample in the system layer is divided into a server area subnet sample and a personal computer area subnet sample, which are used as inputs to the subnet layer; each host corresponds to a unique IP address, and each sample in the subnet is divided into a host sample, which is used as inputs to the host layer; different services correspond to different port numbers, and the host sample filters the traffic of the same port number as a sample in the service layer. The parameters in the evaluation process of each layer are set as follows:

(1) Service layer

The attack influence settings are shown in Table 3.

Table 3: Attack influence settings

Order number	Attack type	Impact of Attack
1	FTP-Patator	0.5
2	SSH-Patator	0.5
3	DoS slowloris	0.9
4	DoS Slowhttptest	0.9
5	DoS Hulk	0.9
6	DoS GoldenEye	0.9
7	Heartbleed	0.9
8	Web Attack-Brute Force	0.7
9	Web Attack-XSS	0.7
10	Web Attack-Sql Injection	0.7
11	Infiltration	0.9
12	Botnet	0.5
13	Port Scan	0.4
14	DDoS LOIT	0.9

(2) Host layer

Operating system vulnerability A is set to Linux: 0.6, Windows: 0.9, Macintosh: 0.4.

(3) Subnet layer and System layer

The distribution of network security posture assessment results is: 1/4=11.0527, 1/2=18.0013, 3/4=44.2061. The value of h in the posture affiliation function is set to 44.2061. On the other hand, the current work replicates the CNN method with machine learning and data analysis and the hierarchical analysis method AHP with quantitative analysis in the CICIDS2024 dataset. In the current experiment, the time window used is one day, with results from the experiment expressed in numeric form. To ensure effective comparison of experimental results and variable control, this paper sets the same time window t = 500s on the three methods to reproduce the quantitative method of cybersecurity posture assessment of the two methods, and finally obtains the value of cybersecurity posture in the range of [0,1] through the mapping of the posture affiliation function.

2.2.2 Analysis of experimental results

This paper evaluates the effect of the model from two aspects: on the one hand, the comparison is made from the model's ability to prejudge the network danger state, with the ability to prejudge the danger state, that is, when the attack event occurs (before), the SA obtained from the evaluation result should reach the danger value 1, i.e., the perceived network danger state. On the other hand, a comparison is made in terms of the coverage degree of the model on the perceived time of the attack event, that is, the network danger state obtained by the evaluation model should cover the time period when the real attack event occurs, i.e., the time period corresponding to SA=1 accounts for the percentage of the time period in the shaded area, and the higher the percentage is, the more accurate the model is for the perceived time of the attack event.

Prediction performances for the three approaches used in the posture assessment experiment are provided in Table 4. The total number of attack types is 14; however, the proposed approach detects 13 types, while CNN and AHP detect 4 and 5 attacks, respectively. In addition, in case of an attack type that cannot be detected by the proposed approach, that is, during Web Attack-SQL Injection, the maximum value of security posture SA calculated using CNN and AHP does not exceed 0.6, but for the proposed approach, it is close to 1. In the state of danger, the proposed approach calculates the security posture SA close to 1.

Table 4: Comparison of the prediction effect of the situation assessment experiment

Order number	Attack type	Predict attack events		
		Our method	CNN	AHP
1	FTP-Patator	√	×	×
2	SSH-Patator	√	×	×
3	DoS slowloris	√	√	√
4	DoS Slowhttptest	√	√	√
5	DoS Hulk	√	×	√
6	DoS GoldenEye	√	√	√
7	Heartbleed	√	×	×
8	Web Attack-Brute Force	√	×	×
9	Web Attack-XSS	√	×	×
10	Web Attack-Sql Injection	×	×	×
11	Infiltration	√	×	×
12	Botnet	√	×	×
13	Port Scan	√	×	×
14	DDoS LOIT	√	√	√

The analysis outcomes of the performance on coverage are shown in Table 5. The attacks Heartbleed and Web Attack-SQL Injection rarely happen and have short periods, leading to low sensing time coverage of all three techniques. In the cases of the other types of attacks, except for these two types, the proposed approach has sensing time coverage more than 90%, while both of the other two approaches have low sensing time coverage. Additionally, the types of DoS and PortScan have high attack volume and duration, which produce high sensing time coverage. Among the 14 types of attacks, the sensing time coverage of the proposed method is higher than those of both CNN and AHP, which suggests that the proposed method can capture more accurate situational signals in terms of time.

Table 5: Comparison of Coverage Effect of Situation Assessment Experiment

Order number	Attack type	Attack event perception time coverage(%)		
		Our method	CNN	AHP
1	FTP-Patator	100	0	0
2	SSH-Patator	100	0	0
3	DoS slowloris	100	100	100
4	DoS Slowhttptest	100	100	100
5	DoS Hulk	100	100	100
6	DoS GoldenEye	100	44.14	43.27
7	Heartbleed	37.51	0	9.15
8	Web Attack-Brute Force	100	0	0
9	Web Attack-XSS	100	0	0
10	Web Attack-Sql Injection	0	0	0
11	Infiltration	92.16	62.94	65.58
12	Botnet	93.63	0	0
13	Port Scan	95.91	40.59	51.92
14	DDoS LOIT	99.85	97.66	98.87

3 INGWO-LSTM-based network security posture prediction model and results

3.1 Cybersecurity posture prediction model based on INGWO-LSTM

3.1.1 Improved optimization algorithm for gray wolves in small habitats

(1) Initialization of Population Based on Backward Learning

In the traditional GWO, the population is initialized randomly within the search space, leading to non-uniformity. The lack of uniformity in the population initialization process leads to the inability of the initial population to span the whole search space, thereby significantly reducing the effectiveness and accuracy of the optimization process. In order to enhance the uniformity and diversity of the population initialization process, an initialization technique based on backward learning has been employed in this work. It helps in generating the inverse solutions to enhance the diversity of the population initialization process.

Step1: Randomly initialize N gray wolves X_{ij} to form a positive population Pop in a random search space of range $[u,l]$, where N is the number of individuals in the population, D is the dimension of the problem solution, and u_i and l_i are the upper and lower bounds on the value of the j th dimension of the problem solution, respectively. That is:

$$X_{ij} = l_j + rand[0,1]\times\left(u_j - l_j\right), i\in[1,N], j\in[1,D] \tag{9}$$

Step2: Generation of reverse population $rePop$. The main task of this process is to generate the inverse point of each individual gray wolf. For example, if there exists a point $X_i = \left(X_{i1}, X_{i2}, \ldots, X_{iD}\right)$ in D-dimensional space, where $X_{ij} \in \left[l_j, u_j\right], j = 1, 2, \ldots, D$, then its reverse point would be $X_i{}' = \left(X'_{i1}, X'_{i2}, \ldots, X'_{iD}\right)$, where $X'_{ij} = l_j + u_j - X_{ij}$.

Step3: Combine the forward population Pop and the reverse population $rePop$ into a large population with a population size of $2 \times N$, and then select the gray wolf individuals in the top N of the fitness ranking as the initial population of the algorithm.

(2) Nonlinearization of convergence factor

In the GWO algorithm, $|\vec{A}|$ is mainly used to control the search process of the algorithm, when $|\vec{A}| > 1$, in order to find a better prey, the gray wolf group will expand the search scope, the algorithm carries out a global search; when $|\vec{A}| < 1$, in order to complete the hunting task, the gray wolf group will narrow the search scope, and carry out local search. It can be seen that the algorithm is directly affected by $\vec{A}$. The value of $\vec{A}$ is calculated through the convergence factor a, so the convergence factor a indirectly influences the overall performance of the algorithm. In the traditional GWO algorithm, the convergence factor a decreases linearly with the number of iterations, and it serves to balance the global and local search capabilities of the algorithm, however, it has been shown in the literature that different ways of updating an important parameter such as the convergence factor a can greatly affect the performance of the algorithm, and the linear updating method is often not the most effective one. Therefore, a new nonlinear convergence factor updating strategy is used in this paper:

$$a = 2 - 2\sqrt{1 - \left(\frac{t_{\max} - t}{t_{\max}}\right)^2} \tag{10}$$

where $t_{\max}$ is the maximum number of iterations of the algorithm, while the current number of iterations is denoted by t.

(3) Minor habitat evolution

In order to effectively improve the optimization ability of the algorithm, this paper introduces the most widely used small habitat algorithm based on the sharing mechanism in the later stage of GWO optimization. In the process of algorithm improvement, according to the definition of small habitat algorithm of sharing mechanism, the sharing function is used to reflect the similarity degree between any two gray wolf individuals in the wolf pack. In this paper, $f_{share}\left(d_{ij}\right)$ is used to represent the sharing degree between gray wolf individuals i and j. i.e:

$$f_{share}\left(d_{ij}\right) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{sh}}\right)^{\alpha} & d_{ij} < \sigma_{sh} \\ 0 & otherwise \end{cases} \tag{11}$$

where d_{ij} denotes some kind of relationship between individuals i and j in a wolf pack, σ_{sh} is the radius of the niche, and α is used to control the shape of the sharing function, which in this paper $\alpha = 1$. That is:

$$\sigma_{sh} = \sum_{i=1}^{m} \frac{d_i}{m} \tag{12}$$

$$d_i = \min\left(\left\| X_i - X_j \right\|\right) \tag{13}$$

The degree of sharing S is also known as the small habitat number, and its magnitude reflects the degree to which individual gray wolves share in the pack. S_i is defined as the sum of the values of the sharing function between an individual gray wolf i and each of the other individuals in the pack, where N denotes the size of the population:

$$S_i = \sum_{j=1}^{N} f_{share}\left(d_{ij}\right) \tag{14}$$

The adjusted fitness value $F'(i)$ of an individual i in the population is obtained by dividing its original fitness $F(i)$ by the sharing degree S_i. That is:

$$F'(i) = \frac{F(i)}{S_i} \tag{15}$$

3.1.2 INGWO-LSTM based network security posture prediction

(1) Architecture of INGWO-LSTM-Based Network Security Posture Prediction Model

The architecture of network security posture prediction model using INGWO-LSTM approach is shown in Fig. 1. It involves the combination of efficient parameter optimization ability of INGWO and effective time-series data analysis of LSTM to achieve the prediction of network security posture effectively.

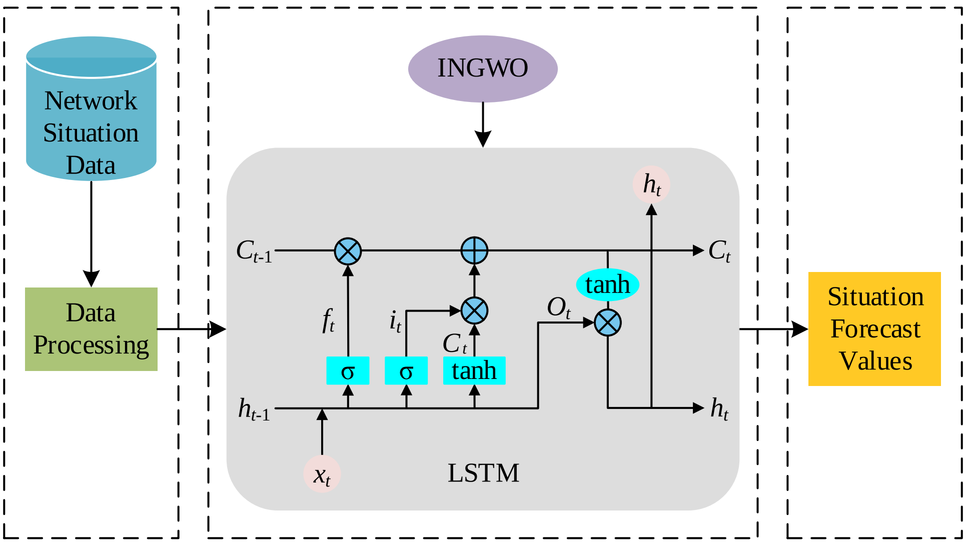


Figure 1: Cybersecurity Situation Prediction Model Based on INGWO-LSTM

(2) Prediction of Cyber Security Posture using INGWO-LSTM

While the LSTM neural network is proven to be very effective when dealing with time series data, the number of parameters to train the model is relatively large, and the choice of the number of hidden layer nodes and batch size is relatively subjective. Compared to debugging the network based on experience during model training, the INGWO algorithm proposed in this paper utilizes its strong global search capability to eliminate the subjective factors in the selection of parameters for LSTM neural networks, thus enhancing the performance of LSTM in prediction. The specific steps for network security posture prediction using LSTM optimized by INGWO are as follows:

Step 1: Network security posture value data pre-processing. Considering that the distribution range of network security posture values is quite large and significantly impacts the LSTM network training and decision speed, data normalization is needed to get rid of the effect of data distribution range and confine normalized data to [0, 1].

Step 2: Multidimensional representation of the network security posture data for easy network training and prediction. The sliding window technique can be employed for the reconstruction of one-dimensional posture data, and the reconstructed data can be divided into training and test datasets.

Step 3: Initialization of the gray wolf position vector based on the range of the network hyperparameters to optimize.

Step4: Initialize the parameters of INGWO algorithm.

Step 5: The initial gray wolf position vector is assigned to the training hyperparameters in LSTM after reverse coding, and the training samples are used to train the model. INGWO uses $1/RMSE$ as the fitness function in the optimization process of LSTM. The fitness values of all individuals in the wolf pack are obtained after each training, and the three gray wolves with the highest fitness are labeled as the head wolf α, β, and δ respectively.

Step 6: Update the position of individual gray wolves in the population. Once the update is completed, recalculate the fitness values of all gray wolf individuals in the population, and then determine the new three head wolves based on the calculation results.

Step7: Interpret whether the algorithm has reached the termination condition, if the termination condition has been reached then continue to execute Step8. otherwise return to Step3 to start a new round of loop iteration.

Step8: Assign the position vectors of the gray wolves to the network hyperparameters of the LSTM after reverse coding.

Step 9: Input training data into the neural network model for training. When the training stop condition is reached during this process, the learning of the neural network is halted and the trained network model is saved.

Step10: Input test samples into the trained network model in Step8 for prediction.

3.2 INGWO-LSTM based network security posture prediction results

3.2.1 Experimental preparation

(1) Experimental environment

Hardware environment: CPU model is Intel Core i5-10300H 3.20GHz, memory model is 16G DDR42933Hz, hard disk model is KINGSTON OM8PCP3512F-AB 500G; software environment: PyCharm Professional 2024 is used as the development environment, and Python 3.6.5 and TensorFlow are utilized as the programming language and neural network computing platform, respectively.

(2) Data Processing

In order to validate the efficiency of the proposed INGWO-LSTM hybrid prediction model in real-world engineering projects, the IoT data collected from Province J in real time will be chosen as the target of prediction and analysis. The dataset of Province J's cybersecurity situation consists of 20,031 samples collected once every hour from 2022 to 2024. The last 200 samples are reserved for testing, and the rest are used for training purposes. Meanwhile, both the training and testing datasets will be normalized. In this experiment, COD index data will be predicted in the dataset. The number of data in the dataset is a total of 5,024 samples. In order to examine the effectiveness of the short, medium, and long-term prediction of the model, the preprocessing dataset is split into four groups. The number of training sets for Groups 1, 2, 3, and 4 are 5,003, 5,012, 5,009, and 5,020, respectively, while their test sets are 25, 50, 75, and 100.

When there are abnormal samples in the dataset, the strategy adopted in this paper is deletion and chronological sorting. In order to reduce the impact of different scales of data features on computational results, the training and testing datasets will be normalized separately using the following formula:

$$x_{norm} = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{16}$$

where n denotes the total number of samples in the dataset; x_i denotes the observation value at the i th moment; x_{norm} denotes the value after normalization of x_i, and $x_{\min}$ and $x_{\max}$ denote the minimum and maximum values in the whole dataset, respectively.

(3) Model quantitative indicators

To compare and analyze the prediction performance of each model from multiple dimensions, four quantitative indicators are used to evaluate the prediction results: mean absolute error (MAE), mean absolute percentage error (MAPE), root mean square error (RMSE), and coefficient of determination R^2. Among these, MAE, MAPE, and RMSE quantify the prediction error of the model, while R^2 measures prediction accuracy. The formula is as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2}{\sum_{i=1}^{n}\left(\bar{y}_i - y_i\right)^2} \tag{17}$$

where $\hat{y}_i$ denotes the i th predicted value, y_i denotes the i th true value, and n denotes the total number of data samples in the test set.

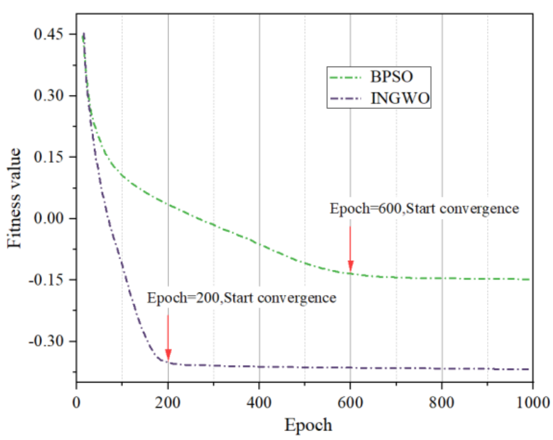
(4) Comparison of models

To verify the prediction performance of the INGWO-LSTM model against a single prediction model, INGWO-LSTM is compared with the traditional machine learning SVR model, RNN, and LSTM models.

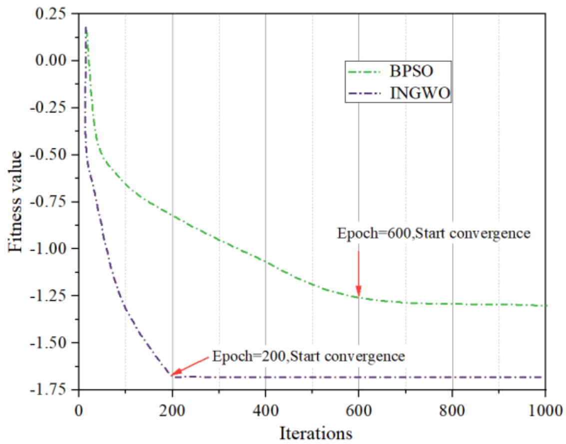
3.2.2 INGWO algorithm validation and result analysis

To test the optimization efficiency of the proposed INGWO algorithm, two standard benchmark functions, namely Ackley function and Sphere function, are selected to conduct optimization experiments. The parameters set for INGWO algorithm in this paper are as follows: learning factors c1 = c2 = 1, swarm size = 25, particle search dimension = 7, and maximum iteration number = 1000.

In Figs. 2(a) and 2(b), the comparative results between INGWO optimization search and BPSO optimization search algorithms are demonstrated, where Fig. 2(a) corresponds to the standard function Ackley, and Fig. 2(b) corresponds to the standard function Sphere. The performance of INGWO optimization algorithm is superior to that of the traditional BPSO algorithm in optimizing optimal solutions from both benchmarks. In particular, the INGWO algorithm takes only about 200 iterations to converge, but BPSO optimization algorithm converges at about 600 iterations. From the above results, it could be known that under particle search dimension of 7, the INGWO algorithm performs better than BPSO in optimization searching with faster speed and higher accuracy.



(a)Ackley



(b)Sphere

Figure 2: Comparison of Optimization Performance between INGOW and BPSO

3.2.3 INGWO-LSTM algorithm validation and result analysis

(1) Analysis of results with a single prediction model

Table 6 compares the predictive results of the two models, while the dynamic change process of predictive results for both models can be seen in Fig. 3. It can be found that all predictive models are able to recognize the existing patterns within the datasets, among which, the prediction results of INGWO-LSTM fit the observations significantly better than the LSTM model.INGWO-LSTM reduces 50.69%, 70.68%, and 51.26% in the three error evaluation indexes, namely, MAE, MAPE, and RMSE, respectively, compared with LSTM, and the R^2 value is improved by 9.71% compared to LSTM, which shows that INGWO-LSTM model has better prediction effect.

Table 6: The prediction results of each model

Model	LSTM	INGWO-LSTM
MAE	3.6916	1.8204
MAPE	0.0614	0.0180
RMSE	4.3277	2.1092
R^2	0.9102	0.9985

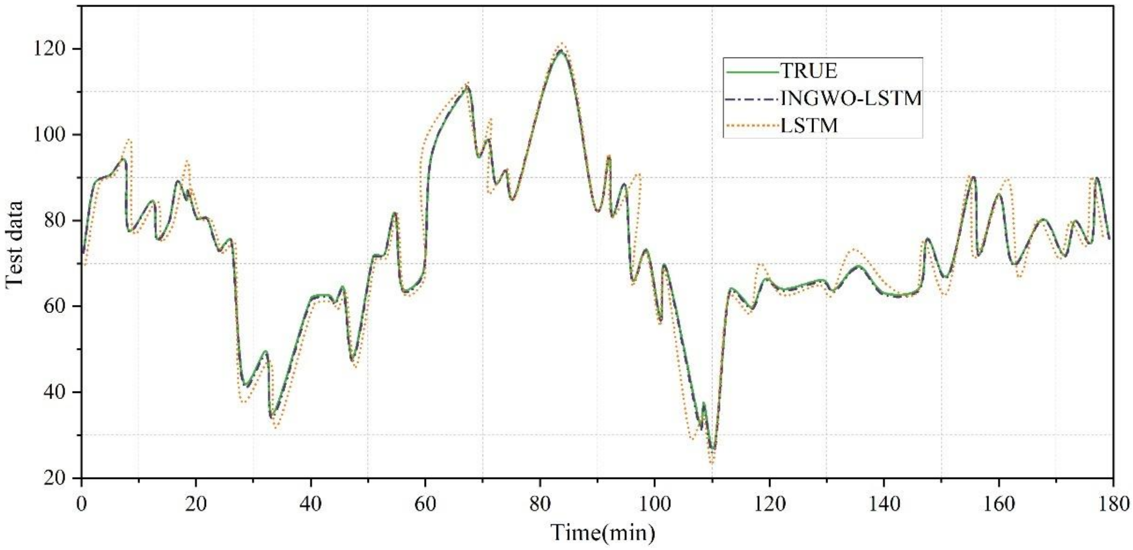


Figure 3: The Process of the Prediction Results of Two Models with Time

In order to further investigate the distinction between INGWO-LSTM and LSTM prediction results, weekly prediction results can be split into three parts for the first three days, the middle three days, and the last day, analyzed under four criteria as illustrated in Table 7. It can be seen that the distinction between the prediction results of LSTM and INGWO-LSTM is relatively small for the middle three days, but the prediction results of LSTM in the first 3 days and the last day have more oscillating curves over time and are farther away from the observed values, which indicates that LSTM has more errors than INGWO-LSTM.

The quantitative results of INGWO-LSTM are better than those of LSTM model in all four evaluation indexes in the three prediction time periods, and the prediction errors of both models improve as well as the prediction accuracies decrease with the prolongation of the prediction period, i.e., the prediction performances of both models are reduced. Overall, the prediction performance of INGWO-LSTM decreases more slowly than that of LSTM, which has better application prospects and reference value.

Table 7: Quantitative results of the four indicators

Time	Model	MAE	MAPE	RMSE	R^2
First 3 days	LSTM	3.4707	0.0503	4.1731	0.9096
	INGWO-LSTM	1.6335	0.0227	1.9346	0.9998
3 days in between	LSTM	3.1165	0.0568	3.6945	0.8676
	INGWO-LSTM	2.0523	0.0195	2.5854	0.9955
Last day	LSTM	5.2388	0.0692	6.0307	0.1957
	INGWO-LSTM	2.3559	0.0219	2.4235	0.9205

(2) Analysis of results with combined prediction models

Prediction results in all four data groups are provided in Table 8. For each of the four data groups, the errors generated by INGWO-LSTM are smaller than the errors made by the LSTM algorithm. By comparison with LSTM, INGWO-LSTM decreases RMSE by 17.5% to 22.14% and MAE by 23.73% to 45.52%. In the fourth data group especially, INGWO-LSTM decreases RMSE and MAE by 20.42% and 45.52% respectively when compared with LSTM. From the results of the two models, one can find that the single LSTM model cannot perform well. This is because once the parameters of a single LSTM model have been set, their values will never change during the following data processing procedure. Therefore, this LSTM model cannot discover any potential patterns in the data during the training process or modify its structure according to the data; consequently, the prediction error becomes very large. However, the INGWO-LSTM algorithm is much more stable during the optimization procedure and reaches the best candidate solution quickly. In addition, the accuracy of the results can be greatly improved in this way, and finally a better network structure corresponding to the data can be obtained.

Table 8: The prediction results on the four sets of data

Group	Metric	LSTM	INGWO-LSTM
Group 1	RMSE	6.8185	5.6252
	MAE	5.8224	4.3059
Group 2	RMSE	8.6191	7.1208
	MAE	7.1859	4.5208
Group 3	RMSE	9.492	7.3908
	MAE	8.1429	6.2103
Group 4	RMSE	9.7569	7.7644
	MAE	8.567	4.6675

4 Artificial Intelligence Cybersecurity Management Optimization Case Study

4.1 Data set description and preprocessing

Experimental environment: application of MAC 12.0.1 operating system, Python 3.6 as the development language, PyCharm 2019.1 as the environment, using the internal library environment Numpy==1.18.1, Keras==2.3.1, Scikit_learn==0.24.1, Pandas==0.25.3. Tensorflow==1.15, etc. The important parameters of the model in the experiment were set to units=24, time_steps=1, batch_size=1024.

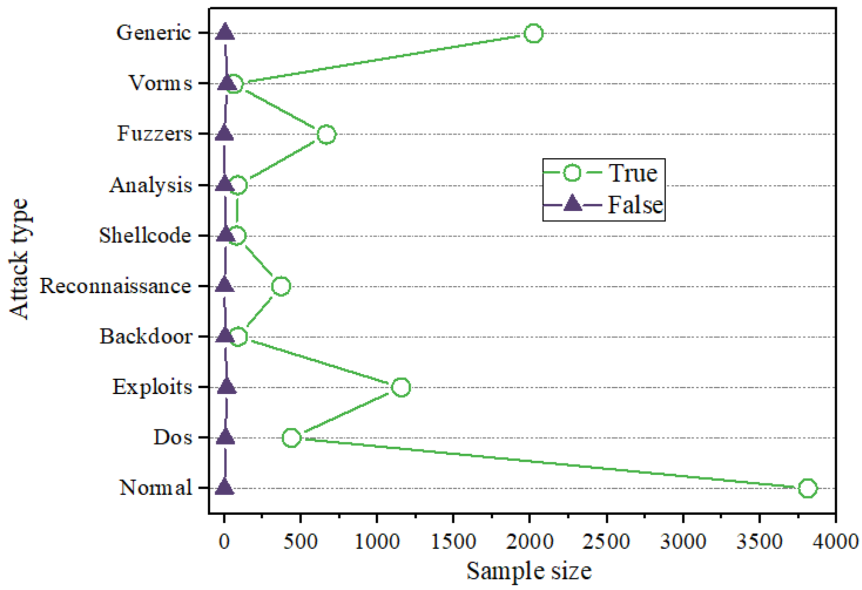
Experimental data: in order to describe the actual operation of the network, this paper uses the UNSW-NB15 public dataset as the experimental data. It is a network attack dataset established by the Australian Network Center. It includes nine types of cyber-attacks, relying on tools such as Argus, Bro IDS and other tools to construct a multi-class algorithm, which results in 49 columns of characteristic attributes with corresponding labeled values. The dataset has a continuous feature called Stime on the time axis. For prediction of this type of data, an LSTM time series prediction algorithm will be chosen since this approach can deal with datasets with temporal relationships. The effect of the network operations from the previous time can affect the posture assessment of the next time, making the prediction more realistic.

Experimental data preprocessing: Preprocess the data. First, extract the unique feature values of each group of records in the dataset. The unique feature values in UNSW are "proto", "service", and "state". Track new virtual data sequences based on unique string records in the feature values. Normalize the data sample features to further learn the relationship between features and labels and store them separately. The newly created dataset is further used in INGWO-LSTM prediction models in the next phase of the assessment and prediction of network posture.

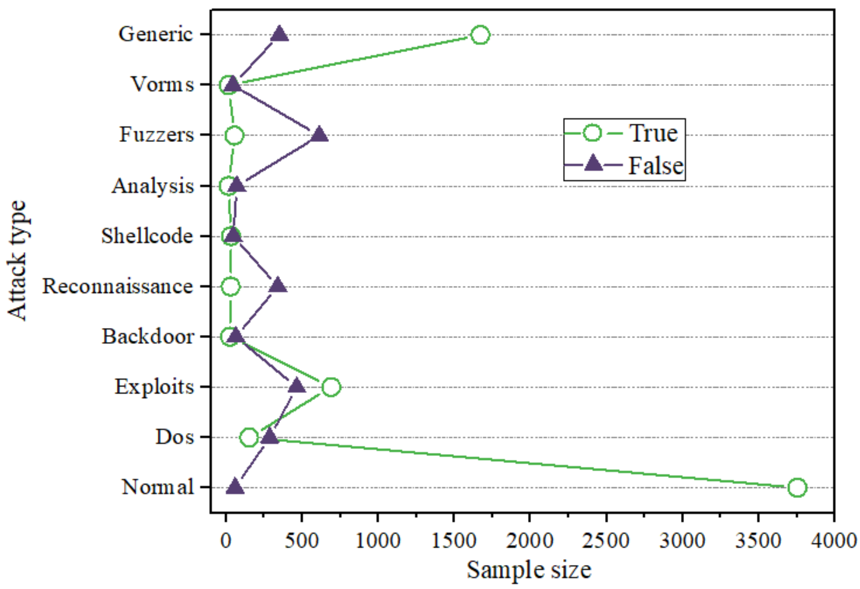
4.2 Experiments and Analysis of Results

4.2.1 Analysis and Comparison of Attack Type Identification Results

For testing the validity of the network security posture assessment model using Bi-LSTM suggested in this study, ten random attacks including Generic will be tested. Vorms and Fuzzers in CIC-IDS2024 dataset, and compared and analyzed the recognition effect of this paper's model, Bi-LSTM, with the LSTM model for the categories of cyber-attacks. The recognition accuracy of each type of attack is shown in Fig. 4, where (a) and (b) represent Bi-LSTM algorithm and LSTM algorithm respectively. Earlier techniques for the use of LSTMs put forth by previous researchers are highly affected by the number of samples. With the attack categories that have few samples, such as Vorms, the recognition accuracy suffers greatly and instability is experienced. Despite the fact that the modeling method suggested in this paper is also affected by the number of samples, the bi-directional LSTM manages to achieve 72.13% recognition accuracy for Vorms, which is much better than the former. It can also be found that the Normal type possesses the highest classification accuracy in both models due to the fact that it has the most data during training and testing. Not only that, the accuracy of Bi-LSTM algorithm for all attack types except Vorms is above 90%. In the case of Shellcode attacks (88), for example, it has only 78 training and 10 testing records. The classification results are relatively poor due to lack of samples. It is concluded that the more record entries for each network attack, the clearer this model grasps the features of that attack method, and the recognition accuracy will be improved accordingly.



(a) Bi-LSTM



(b) LSTM

Figure 4: Identification accuracy of various attacks

4.2.2 Probability of attack

The probability of the threat event is mainly derived via prediction based on the INGWO-LSTM time series forecasting algorithm, and its impact effect is measured through the loss functions in the process of training datasets. Two loss functions adopted in this case study are L1 and L2; they measure the difference between the predicted results and the actual observation results of models. Specifically,

$$\begin{aligned} &L_2\ loss: L(y,\hat{y}) = \omega(\theta)(\hat{y}-y)^2 \\ &L_1\ loss: L(y,\hat{y}) = \omega(\theta)\left|\hat{y}-y\right| \end{aligned} \tag{18}$$

L_1 and L_2 represent two types of loss function computation. L_1 is used to address the gradient problem, while L_2 is applied to handle dataset anomalies. Taking both into consideration, this paper adopts the L_2 formula to calculate the loss function and evaluate the model training results. In the formula, $\omega(\theta)$ is the weight of the true value, y is the true value, and $\hat{y}$ is the model predicted value.

Based on the existing INGWO-LSTM model, predictions will be made regarding the probability of threat occurrence. The distribution chart of the predicted value and the performance of fitting in the test set can be seen in Fig. 5. With the closer predicted value to 1, the higher the possibility that there might occur an attack. Conventionally, there exists an attack probability criterion of 0.5, which means that if the predicted result is larger than 0.5, the result will be classified into category 1, and the value equal to or smaller than 0.5 will fall into category 0. According to the prediction procedure implemented on the existing dataset, accuracy is employed to evaluate fitting performance, and the final prediction accuracy is 93.24%. In this case, the predicted value is the probability of occurrence of the threat. This way, the prediction procedure brings some probabilistic features into threat classification.

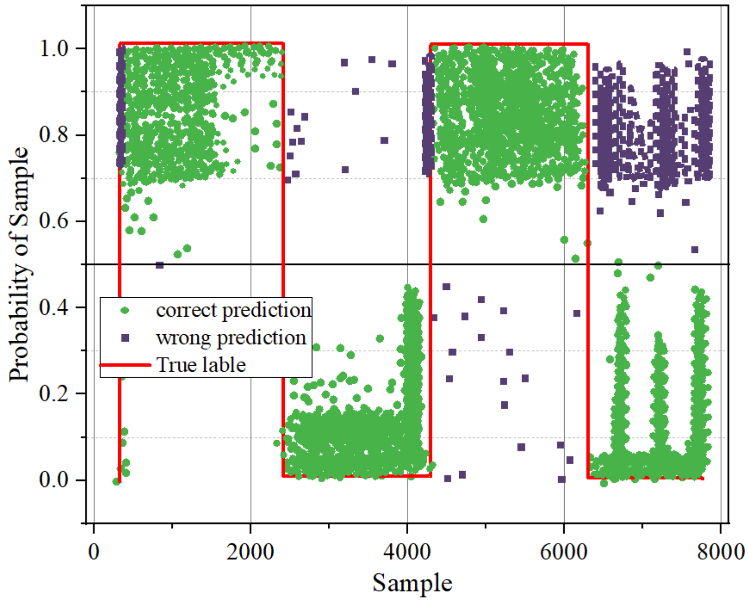


Figure 5: The distribution of specific predicted values and fitting effects in the test set

5 Conclusion

This paper constructs a data fusion posture assessment model grounded in attention mechanism and a cybersecurity posture prediction model based on INGWO-LSTM, developed to identify and predict AI cybersecurity posture under IoT fusion technology. The results indicate that:

Among 14 kinds of attacks, this paper's method predicts 13 kinds of attacks, and except for Heartbleed and Web Attack-Sql Injection attacks, which have fewer attack occurrences and shorter attack occurrence time, this paper's method achieves more than 90% coverage of the perceived time of the rest of the attack events, which is significantly better than the other 2 comparative methods, which indicates that this paper's method can in the time dimension react to more accurate posture signaling.

The INGWO-LSTM model predicts the network security posture better, and its values of MAE, MAPE, and RMSE are reduced by 50.69%, 70.68%, and 51.26% than the LSTM model, and the value of R^2 is improved by 8.84%. The results of the real case study also show that the INGWO-LSTM model's prediction accuracy of AI network security posture under IoT convergence technology is also above 90%, which proves that the network security management optimization algorithm proposed in this paper has high application value.

About the Author

Chengxin Wei, who holds a Master's degree, is currently serving as a Senior Engineer at the Information Technology Center of Guangxi University of Chinese Medicine. His primary research areas include network security and protection technologies, AI-driven network defense mechanisms, data privacy protection in edge computing environments, and the application of artificial intelligence technology in the field of network security management.

Kejiu Tan (Menber, IEEE) received the B.S. in Computer Science and Technology from Guangxi University in 2008 and an M.S. in Engineering from Hubei University of Technology in 2014. He is currently a Senior Engineer at the Modern Educational Technology Center, Hechi University. His research interests include network and information security, and data mining.

He is a member of China Computer Federation (CCF).

References

[1] Salazar, Z., Nguyen, H. N., Mallouli, W., Cavalli, A. R., & Montes de Oca, E. (2021, August). 5greplay: A 5g network traffic fuzzer-application to attack injection. In Proceedings of the 16th International Conference on Availability, Reliability and Security (pp. 1-8).

[2] Butun, I., Österberg, P., & Song, H. (2019). Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures. IEEE Communications Surveys & Tutorials, 22(1), 616-644.

[3] Abu Al-Haija, Q., & Al-Badawi, A. (2021). Attack-Aware IoT network traffic routing leveraging ensemble learning. Sensors, 22(1), 241.

[4] Rajić, B., Stanisavljević, Ž., & Vuletić, P. (2023). Early web application attack detection using network traffic analysis. International Journal of Information Security, 22(1), 77-91.

[5] Singh, L., & Singh, R. (2024). Comparative analysis of traditional firewalls and next-generation firewalls: A review. Latest Trends in Engineering and Technology, 15-27.

[6] Garuba, M., Liu, C., & Fraites, D. (2008, April). Intrusion techniques: Comparative study of network intrusion detection systems. In Fifth international conference on information technology: new generations (itng 2008) (pp. 592-598). IEEE.

[7] Zhang, C., Jia, D., Wang, L., Wang, W., Liu, F., & Yang, A. (2022). Comparative research on network intrusion detection methods based on machine learning. Computers & Security, 121, 102861.

[8] Pawlicki, M., Choraś, M., & Kozik, R. (2020). Defending network intrusion detection systems against adversarial evasion attacks. Future Generation Computer Systems, 110, 148-154.

[9] Muneer, S., Farooq, U., Athar, A., Ahsan Raza, M., Ghazal, T. M., & Sakib, S. (2024). A critical review of artificial intelligence based approaches in intrusion detection: A comprehensive analysis. Journal of Engineering, 2024(1), 3909173.

[10] Vanin, P., Newe, T., Dhirani, L. L., O'Connell, E., O'Shea, D., Lee, B., & Rao, M. (2022). A study of network intrusion detection systems using artificial intelligence/machine learning. Applied Sciences, 12(22), 11752.

[11] Patil, S., Varadarajan, V., Mazhar, S. M., Sahibzada, A., Ahmed, N., Sinha, O., ... & Kotecha, K. (2022). Explainable artificial intelligence for intrusion detection system. Electronics, 11(19), 3079.

[12] Alladi, T., Kohli, V., Chamola, V., Yu, F. R., & Guizani, M. (2021). Artificial intelligence (AI)-empowered intrusion detection architecture for the internet of vehicles. IEEE Wireless Communications, 28(3), 144-149.

[13] Alohali, M. A., Al-Wesabi, F. N., Hilal, A. M., Goel, S., Gupta, D., & Khanna, A. (2022). Artificial intelligence enabled intrusion detection systems for cognitive cyber-physical systems in industry 4.0 environment. Cognitive Neurodynamics, 16(5), 1045-1057.

[14] Alazab, M., Khurma, R. A., Awajan, A., & Camacho, D. (2022). A new intrusion detection system based on Moth–Flame Optimizer algorithm. Expert Systems with Applications, 210, 118439.

[15] Jiang, H., He, Z., Ye, G., & Zhang, H. (2020). Network intrusion detection based on PSO-XGBoost model. IEEE Access, 8, 58392-58401.

[16] Zhang, L., Huang, J., Zhang, Y., & Zhang, G. (2020, October). Intrusion detection model of CNN-BiLSTM algorithm based on mean control. In 2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS) (pp. 22-27). IEEE.

[17] Yang, Y., Zheng, K., Wu, C., Niu, X., & Yang, Y. (2019). Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks. Applied Sciences, 9(2), 238.

[18] Zhang, Y., Li, P., & Wang, X. (2019). Intrusion detection for IoT based on improved genetic algorithm and deep belief network. IEEe Access, 7, 31711-31722.

[19] GSR, E. S., Ganeshan, R., Jingle, I. D. J., & Ananth, J. P. (2023). FACVO-DNFN: Deep learning-based feature fusion and Distributed Denial of Service attack detection in cloud computing. Knowledge-Based Systems, 261, 110132.

[20] Hassan, M. M., Gumaei, A., Alsanad, A., Alrubaian, M., & Fortino, G. (2020). A hybrid deep learning model for efficient intrusion detection in big data environment. Information Sciences, 513, 386-396.

[21] Ahmim, A., Maazouzi, F., Ahmim, M., Namane, S., & Dhaou, I. B. (2023). Distributed denial of service attack detection for the Internet of Things using hybrid deep learning model. IEEE Access, 11, 119862-119875.

[22] Rene Beulah, J., & Shalini Punithavathani, D. (2018). A hybrid feature selection method for improved detection of wired/wireless network intrusions. Wireless Personal Communications, 98(2), 1853-1869.

[23] Blanco, R., Malagón, P., Briongos, S., & Moya, J. M. (2019, August). Anomaly detection using gaussian mixture probability model to implement intrusion detection system. In International Conference on Hybrid Artificial Intelligence Systems (pp. 648-659). Cham: Springer International Publishing.

[24] Tang, C., Luktarhan, N., & Zhao, Y. (2020). SAAE-DNN: Deep learning method on intrusion detection. Symmetry, 12(10), 1695.

[25] Duan, L., Han, D., & Tian, Q. (2019). Design of intrusion detection system based on improved ABC_elite and BP neural networks. Computer Science and Information Systems, 16(3), 773-795.

[26] Wang, X., & Wang, L. (2017, December). Research on intrusion detection based on

feature extraction of autoencoder and the improved k-means algorithm. In 2017 10th International Symposium on Computational Intelligence and Design (ISCID) (Vol. 2, pp. 352-356). IEEE.

[27] Seo, S., Han, S., Park, J., Shim, S., Ryu, H. E., Cho, B., & Lee, S. (2021). Hunt for unseen intrusion: Multi-head self-attention neural detector. IEEE Access, 9, 129635-129647.

[28] Ahmad, J., Latif, S., Khan, I. U., Alshehri, M. S., Khan, M. S., Alasbali, N., & Jiang, W. (2025). An interpretable deep learning framework for intrusion detection in industrial Internet of Things. Internet of Things, 101681.

[29] Islam, M. T., Syfullah, M. K., Rashed, M. G., & Das, D. (2024). Bridging the gap: advancing the transparency and trustworthiness of network intrusion detection with explainable AI. International Journal of Machine Learning and Cybernetics, 15(11), 5337-5360.

[30] Keshk, M., Koroniotis, N., Pham, N., Moustafa, N., Turnbull, B., & Zomaya, A. Y. (2023). An explainable deep learning-enabled intrusion detection framework in IoT networks. Information Sciences, 639, 119000.

[31] Mohale, V. Z., & Obagbuwa, I. C. (2025). Evaluating machine learning-based intrusion detection systems with explainable AI: enhancing transparency and interpretability. Frontiers in Computer Science, 7, 1520741.

[32] Zouhri, H., & Idri, A. (2025). Feature selection and global interpretability of black-box classification for intrusion detection. Neural Computing and Applications, 1-32.

[33] Lv, M., Gan, S., Xu, K., Chen, T., Zhu, T., & Chen, J. (2025). An Interpretable Network Intrusion Detection Model via Decision Tree Enhanced Deep Attention Network. IET Information Security, 2025(1), 5552833.

[34] Djenouri, Y., Belhadi, A., Srivastava, G., Lin, J. C. W., & Yazidi, A. (2023). Interpretable intrusion detection for next generation of Internet of Things. Computer Communications, 203, 192-198.