Hindawi
Security and Communication Networks
Volume 2021, Article ID 5542543, 13 pages
https://doi.org/10.1155/2021/5542543

Research Article

A Security Log Analysis Scheme Using Deep Learning Algorithm for IDSs in Social Network

Ming Zhong, Yajin Zhou, and Gang Chen

Zhejiang University, Computer Science and Technology College, 310018 No. 38 Zheda Road, Xihu District, Hangzhou City, Zhejiang Province, China

Correspondence should be addressed to Ming Zhong; tracym@zju.edu.cn

Received 5 February 2021; Revised 20 February 2021; Accepted 13 March 2021; Published 23 March 2021

Academic Editor: Hao Peng

Due to the complexity of the social network server system, various system abnormalities may occur and in turn will lead to subsequent system failures and information losses. Thus, to monitor the system state and detect the system abnormalities are of great importance. As the system log contains valuable information and records the system operating status and users' behaviors, log data in system abnormality detection and diagnosis can ensure system availability and reliability. This paper discloses a log analysis method based on deep learning for an intrusion detection system, which includes the following steps: preprocess the acquired logs of different types in the target system; perform log analysis on the preprocessed logs using a clustering-based method; then, encode the parsed log events into digital feature vectors; use LSTM-based neural network and log collect-based clustering methods to learn the encoded logs to form warning information; lastly, trace the source of the warning information to the corresponding component to determine the point of intrusion. The paper finally implements the proposed intrusion detection method in the server system, thereby improving the system's security status.

1. Introduction

The development and rise of social networks have changed our way of life, realized the interconnection between people, accelerated information dissemination speed, and changed social communication. While enjoying the convenience that social networks bring to our lives, we also need to protect information. If there is a social network security issue, our personal information will be stolen. The data stored in social network servers and the services provided are themselves potential targets for various attacks. Due to their diversity and particularity, these attacks may have disastrous consequences. In this context, social network server security has become a major challenge, and research in this area is also increasing. The development of various tools and mechanisms ensures that the safety level is improved and meets modern life requirements. These methods include the IDS (intrusion detection system). The IDS is a tool used to detect network attack attempts and is used to identify abnormal activities and behaviors designed to interfere with the normal operation of the system [1].

Various logs are generated during the operation of the host system. The logs record the status of the computer during operation and various operations performed by the system. They are an excellent resource of information for online data monitoring and anomaly data detection. Therefore, the audit of system logs can be used as the host, an essential means of anomaly detection. There are already various security audit systems in the market, such as log audit systems and IDS. These systems can implement log collection, auditing, and abnormal behavior mining functions [2]. However, in the actual use, due to the differences in logs and the single backwardness of log auditing methods, these systems are often only suitable for specific types of hosts, and the abnormal behaviors that can be detected are not comprehensive and accurate.

Data mining capabilities have been further improved with AI and machine learning development, and research results based on machine learning into log auditing continue

to appear. However, for different host systems and processes, the types of logs generated are inconsistent. Traditional machine learning detection methods need to use different feature extraction methods for different types of logs, and users must have professional background knowledge to better extract logs' characteristic information. In reality, the types and grammar of logs are constantly updated, and a certain method cannot be directly applied to multiple systems, and it takes much workforce to update and match. The relatively advanced one is to use deep learning for anomaly detection. The increasing amount of system log data is sufficient for the deep learning model to perform learning processing. When the parameters are appropriate, there is almost no need to extract features manually, and the deep learning model can complete log detection well.

For this purpose, in this paper, we propose a log analysis method based on deep learning and apply it in the IDS to ensure the server's security. Figure 1 shows the scenario of attackers and defenders using the IDS. The attackers on the Internet will use various ways to intrude the social network server and then leak the users' privacy. The IDS monitors the operating conditions of the social network servers in accordance with certain security policies and discovers as much as possible various attack attempts, attack behaviors, or attack results to ensure the confidentiality, integrity, and availability of network system resources [3]. The proposed method in this paper will ensure the users' privacy for the social network. This paper's main work is to form a log parser, build the feature extraction neural network, and finally implement the IDS.

The rest of this paper is organized as follows. Section 2 summarizes the related work and emphasizes log analysis and AI methods. Section 3 proposes the deep learning methods for the IDS. Section 4 shows the experiments and results. Finally, Section 5 concludes the paper.

2. Related Works

2.1. Social Networks' Security. The social network is a platform for human beings to transmit information and conduct social conversations on the Internet. Many researchers show that, with the rapid development of social networks, the way of information exchange between Internet users has also undergone tremendous changes, and social networks play a more unceasingly important role in people's everyday life [4]. People can follow the latest developments of friends and celebrities on Sina Weibo or get other people's attention by sharing information on WeChat Moments. Social networks provide users with the following main functions:

(1) Creating and sharing user information which is public or semipublic within a certain scope

(2) Providing a list of users who can be contacted and provide users with a communication platform

(3) Online chatting, making friends, video sharing, blogs, online communities, music sharing, adding comments, etc.

(4) Providing open interfaces for application plug-in development

Simultaneously, social networks face increasingly serious security threats and have become key targets of attacks on the Internet. In order to actualize the purpose of information exchange and public communication, people have the will and capacity to put their information in public on various social network websites [5]. Without privacy protection and corresponding technical awareness, the safe use of social network websites will not be guaranteed. Coupled with the lack of relevant safety technology, laws, and regulations, safety hazards have gradually become prominent. We must attach great importance to the security of social networks [6].

Statistics and research on international relevant state departments of social networking security issues show as an application in the form of a computer network, security issues, social networks, and traditional computer network security problems have many similarities [7]. However, due to the openness of social networks, in addition to various traditional network information security threats, such as viruses, vulnerabilities, and Trojan horses, social networks also need to face many attacks against their characteristics [8]. It can be divided into three categories: user privacy and data security risks associated with leak triggered, the proliferation of spam and security risks caused by cyberattacks, Internet rumors and public opinion network security threats, and other security risks caused [9].

The security solutions of social networking sites can be implemented through appropriate technical means and management measures, and the effects of different technologies or management on security protection are minimal. The technical methods are as follows [10]. 1. Identity the authentication scheme. 2. Social network identity encryption scheme. 3. Data security management. 4. They grouped data access control. The construction of social network security management measures includes the following aspects: 1. strengthen the construction of website security; 2. strengthen users' awareness of data protection; 3. strengthen supervision of third parties; 4. strengthen legal supervision.

2.2. Deep Learning Algorithm for Security. Developments in information technology have required newer and better methods to analyze how these information systems work. Various machine learning methods study the principles of the device. Deep learning is treated as advanced technology and widely deployed in multiple embranchments, including pattern recognition, natural language processing, and network security. Due to the corresponding increment in the production of data, the ordinary machine learning methods deployed in the field of cybersecurity are gradually unable to work for intrusion detection in the network systems [11]. Therefore, the use of deep learning methods for big data analysis is an innovation that attempts to study the patterns of network connections to detect unauthorized access to computer networks.

Primarily, the system operates according to known principles, including two machine learning methods:

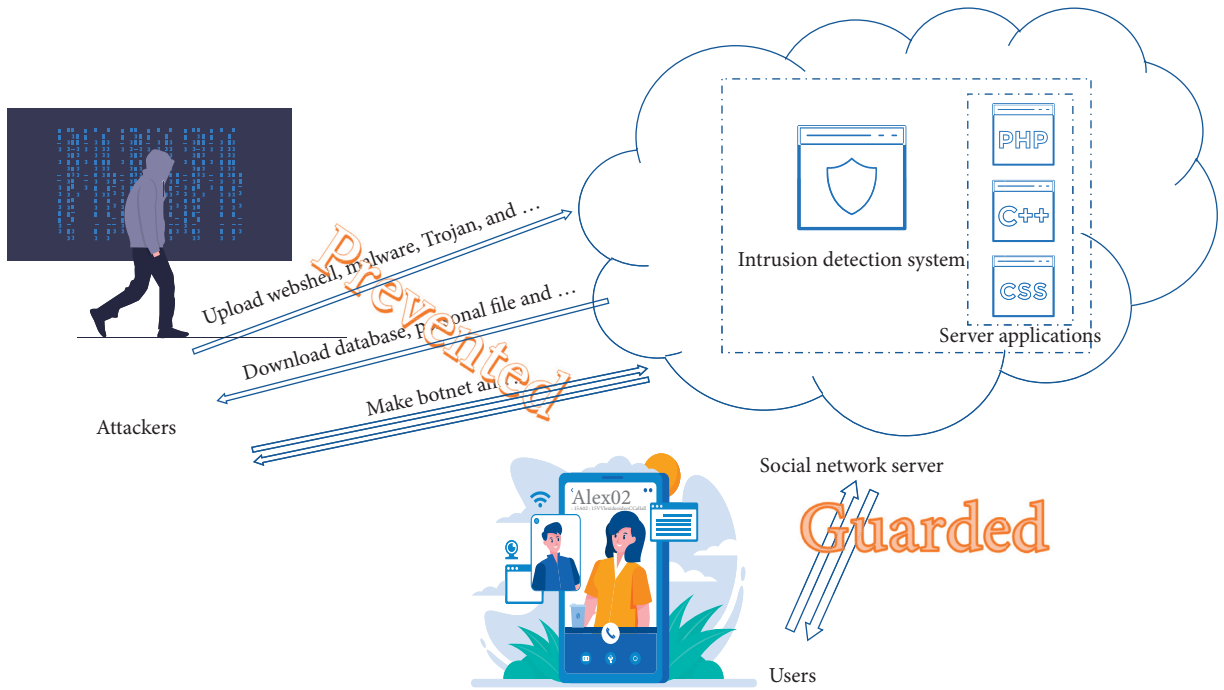

Figure 1: The scenario of attackers and defenders using the IDS in the social network server.

probabilistic and deterministic [12]. The deterministic machine learning method employs a small sample dataset and analyzes them to find any regular pattern deviation. IT experts will then evaluate this information and develop models for classifying the data and processing the results. Usually, the information in the model is compared with the baseline. Therefore, any abnormal data beyond the average level is deemed to be invasive [13].

Besides, the probabilistic method of machine learning takes another big progress because it can evaluate the patterns involved in the evaluation, and these patterns may not be able to get rid of deterministic analysis [14]. The whole system depends on the cluster detecting any strange characters related to the data. The system depends on the unsupervised operation, where the whole system operates independently, generating the map, and ultimately analyzes any abnormal behavior by the same machine, so this method is more effective because the evaluation is conclusive. To be precise, 90% of the problems can be solved by conservative estimate [15].

Deep learning has also recently gained attention because of its advantages. This method is dynamic because the system is predictable and can adapt to generated data. It is worth noting that this method uses the output of the top-down method and uses it as the input of the bottom-up method [16]. The model also uses linear models to extract features, and these linear models are used as essential functions of layers. These layers depend on each other to form a more in-depth system architecture [17].

2.3. Security Application for the Intrusion Detection System. Under the constant development of web technology, the web is used more and more widely. The frequency with which malicious hackers attack a website is usually proportionate to the websites' value. Even if the website's value is comparatively insignificant, it will face malicious test attacks by "script kiddies" or tests on various large-scale vulnerability scanners, just like a saying in the security industry: "there are only two kinds of people in the world, one knows that they have been hacked, and the other I do not know if I was hacked" [18].

At this time, the log analysis of the website is critical. As the role of the management manager, operation staff, and maintenance technician of the website, if they are not on to the real-time security status of the server, they will become the type of "unknown whether they were hacked" and result in losses. Of course, there is another situation where hacking caused economic losses [19]. At this time, we will also conduct log analysis and other emergency measures to try to recover the loss. In short, two of the most direct and most apparent purposes are to implement security log analysis. Self-security events occur on the server to understand, followed by an emergency analysis and evidence collection [20].

The development of deep learning technology has promoted the progress of intrusion detection research. It can use the hierarchical structure to achieve unsupervised functional learning and data pattern classification. The feature extractors and classifiers can be integrated into a framework without the need for security experts to extract features [21] manually. Deep learning methods can efficaciously deal with traffic data of the large-scale network. Compared with shallow traditional machine learning methods, it has higher efficiency and detection rate, but its training process is more complex, and the model's interpretability is poor.

At present, the IDS based on deep learning solves the following problems: (1) the abnormal traffic data in the

network is far less than the category imbalance problem of normal traffic data [22]; (2) the network data volume is large, the feature dimension increases, and the shallow machine learning technology is challenging to match. Massive high-dimensional data is detected, and it is challenging to extract nonlinear features from the data [23]; (3) improve the algorithm itself.

For the above problems, the current work is summarized as follows: (1) if an imbalanced dataset is used to build an intrusion detection model, it will seriously impact detection accuracy. Two methods are usually used to solve the imbalance problem in the data. They are a solution at the algorithm level (cost function method) and a solution at the data level (undersampling and oversampling methods) [24]. Some studies have used these two methods to deal with the imbalance of data categories in the IDS, but most research works have not considered the data imbalance. With the emergence of GANs with strong generation capabilities in recent years, new ideas have been provided to solve data category imbalance. GAN can solve this problem by generating new data. Therefore, the problem of data category imbalance is still currently studied as one of the hot issues [25]. (2) Most deep learning algorithms such as AE, DBM, DBN, LSTM, and CNN [26] have been used to solve this problem. In the deep learning comparison experiments cited in the article, it can be found that the detection accuracy of deep learning methods is usually excellent Traditional machine learning techniques, such as naive Bayes, decision tree, random forest, and support vector machine [27], reflect the effectiveness of deep learning detection methods [28]. However, some evaluation indicators such as the detection accuracy of binary classification and multiclassification problems based on deep learning IDS research still need to be improved [29]. For example, based on the accuracy of the multiclassification problem on the KDDTest + test set in the NSL-KDD dataset, the accuracy is roughly in the range of 79%–85%, while the accuracy of the multiclass problem on the KDDTest-21 test set is roughly 60%–69% [30]. Within the scope, there is still room for improvement. (3) Some studies have improved the deep learning algorithm to improve the model detection ability, such as using the SVM to replace the softmax function to improve the detection ability of the GRU model [31] and using a genetic algorithm to optimize the network structure of DBN [32].

3. System Description and Methodology

3.1. System Architecture. Each log line is generated by the output statement of the source code. For example, the log print statement of a process is printf ("accepted password for %s from %s port %d ssh," user, host, and port), and then, during the running of the program, it may generate a log sentence: "17 Nov 2020 19 : 12 : 48 combo sshd (pam) [1241]: accepted password for root from 10.21.234.33 port 8888 ssh." The logs printed by a source code are of the same type, and their codes are called log keys.

Adjacent or similar logs have a high degree of relevance. A log often depends on the previous logs. When a log breaks this logic, it means that an abnormal log execution path has occurred. Thus, we can regard the log execution sequence as a multiclassification problem [33]. The total number of log keys is certain. We regard it as K. In the training phase, we input the typical log execution sequence to generate a multiclassifier model. In the testing phase, we input the history of the most recent log key, and the output is the probability distribution of a log key. When the error between the sequence prediction result and the actual result is large, we can consider the log to be abnormal.

Figure 2 describes the procedure of the IDS. The input is a sequence of log data, and the output is the probability value. The log parser will parse the log data into the appropriate form and feed it to the neural network.

3.2. Principle of Log Parser. Logs can be divided into log keys and log parameters. We must first separate the two and parse the logs into the structure. The complete process of parsing logs is as follows (Algorithm 1).

3.3. Handling Feature Extraction. After the log parsing is completed, we have obtained the structured log of the system, but, at this time, the log key is only in the form of a string, and the parameter list elements are also strings, which cannot be directly used as the input of the deep learning model, so we need to use it characterized as a feature vector in the digital form. The feature extraction process converts the string into quantifiable numbers, thereby constructing a feature vector matrix. We use two different characterization methods for log keys and parameters due to their different formation methods and expressive meanings. Figure 3 shows the flow chart of the feature extraction.

3.3.1. Log Key Encoding. Since the log is output by the program's code or process, the code is constant, so the type of the output log is also constant, and the number is often not very large. Therefore, for the log key, it is directly coded using sequential numbers. For example, for log keys $K1$, $K2$, and $K3$, we directly characterize them as 1, 2, and 3.

3.3.2. Log Parameter Encoding. Unlike the log type, the parameter value is not generated by the template but dynamically generated according to the actual situation during the system operation, so it often has great uncertainty. The string type of the parameter value will be many and various. In many cases, the direct use of simple integer permutation codes will cause the linear length to be too large.

My approach is to extract all parameter lists, perform parameter preprocessing, and remove all punctuation marks and special characters. Because these characters are not used as criteria for parameter abnormalities, they may affect characters' accuracy. Then, all the parameter strings are recomposed to form a token list, the tokenizer module under the deep learning library Keras is used to process the strings, and the fit_on_texts method is used to learn the text dictionary, which is the mapping relationship between the corresponding words and numbers. Statistics such as word

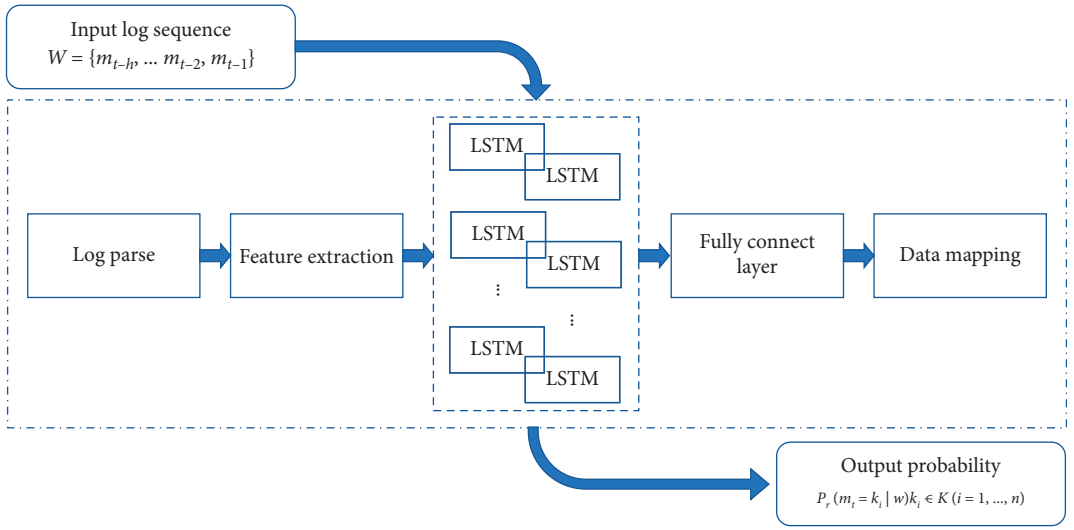

Figure 2: The flowchart of the IDS using LSTM.

Input: log input
Output: sequence output
(1) Initialization (log object, log template, line number list)
(2) Store all log objects to map
(3) Read log by STREAM
(4) Traverse the map to find the largest common subsequence
(5) **if** matching object is found **then**
(6) GOTO Line 11
(7) **else**
(8) GOTO Line 10
(9) **end if**
(10) Initialize the line of log into the list map
(11) Update the line log
(12) Update the template
(13) GOTO Line 3

Algorithm 1: Log parser.

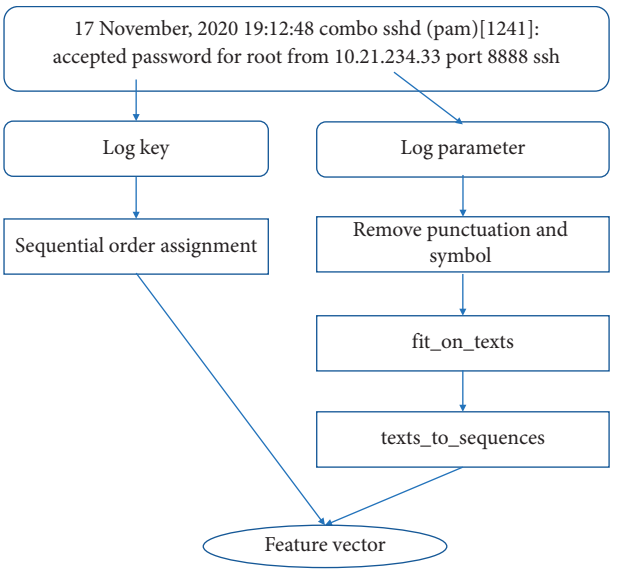

Figure 3: The flow chart illustrates the procedure of feature extraction.

frequency of parameter values. Then use the texts_to_sequences function of the text.Tokenizer module to convert the parameter text to a number, and use 0 to pad sequences of different lengths to the same length.

3.4. Anomaly Detection Scheme. Given a small number of regular log key sequences, and then, input the LSTM model for training. A detection model is obtained. When the system is running normally, the log sequence generated by the system is collected, a log key sequence is obtained after log analysis and feature extraction, and then the LSTM model is used to predict the possibility of a log key after the sequence. The next log in the situation has a greater probability of probability distribution; then, this log is considered a regular log. Otherwise, it is judged to be abnormal.

LSTM is a special type of RNN to avoid long-term dependence through deliberate design. Storing information for a long time is the default behavior of LSTM. All recurrent neural networks have chain repeating modules of neural networks. Different from the single neural network layer of RNN, there are four network layers in LSTM, and they interact in a very special way [34]. The LSTM core unit function process is shown in Figure 4, and the whole steps of the process are described as follows (Algorithm 2).

The first step of LSTM is to select what information to abandon from the cell state. This decision is made by the S-shaped network layer called the "forget gate layer." It receives h_{t-1} and x_t, and the output value is between 0 and 1 for each number in the cell state C_{t-1}. 1 means "accept this completely," and 0 means "ignore this completely:"

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right). \tag{1}$$

The next step is to determine which information needs to be stored in the cell state. This is divided into two parts. In the first part, an S-shaped network layer called the "input gate layer" determines which information needs to be updated. In the second part, a tanh network layer creates a new candidate value vector-$\widetilde{C}_t$, which can be used to add to the cell state. In the next step, we combine the above two parts to generate an update to the state:

$$\begin{aligned} i_t &= \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right), \\ \widetilde{C}_t &= \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right). \end{aligned} \tag{2}$$

Now, update the old cell state C_{t-1} to C_t. The previous steps have already decided what to do, and we just need to do it. We multiply the old state by f_t to forget what we decided to forget. Then, we add $i_t * \widetilde{C}_t$, which is the new candidate value, which is scaled proportionally according to the updated value we decide for each state:

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t. \tag{3}$$

Finally, we need to calculate the output value. The output depends on our cell state, but will be a "filtered" version. Firstly, we run the S-shaped network layer to determine which parts of the cell state can be output. Then, we input the cell state into tanh (adjust the value between −1 and 1) and multiply it with the output value of the S-shaped network layer so that we can output the points we want to output:

$$\begin{aligned} o_t &= \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right), \\ h_t &= o_t * \tanh\left(C_t\right). \end{aligned} \tag{4}$$

In the training phase, the model needs to find an appropriate weight distribution so that the ultimate output data of the LSTM sequence generates the required labels and outputs it along with the input in the training dataset. In the training procedure phase, each input and output use the gradient descent method to find the minimum loss to update these parameters' weights. The input is a log sequence, and the output is the log key value immediately following this log sequence. In training, the loss function used by the log key is categorical cross-entropy loss, and the parameter uses the mean square loss to measure the error.

In the detection stage, we use a layer containing x LSTM blocks to predict the output of an input sequence w and add each log key of w to the LSTM block corresponding to this layer. We use multiple hidden layers, and the previous hidden state will be used as the input in the next layer's core blocks, turning the model into a deep neural network, and the input layer will come from all log key types K. The n logs are encoded as one-hot vectors. The output layer uses a standard polynomial logistic function (standard multinomial logistic function) to convert the output into a probability distribution function. When the model predicted log key and the actual log key have an enormous difference, which exceeds the threshold we defined, it is determined that the log has an abnormal execution path.

The model architecture of anomaly detection based on the LSTM is shown in Figure 5. The figure's top shows an LSTM module, and the repeated LSTM blocks make up the entire architecture. Each LSTM module will record a state as a fixed-dimensional vector. The LSTM module's state from the previous time step and its external input S_{t-i} will be used as the next LSTM module's input to calculate the new state and output. This method ensures that the log information in the log sequence can be passed to the next in an LSTM block [35].

Figure 4 shows a sequence of the core blocks that form an expanded form of the cyclic pattern. Each block retains a hidden state $Ht - i$ and a state vector C_{t-i}, both of which are transferred to the next block, thus initializing its state. For each log key data in the input sequence w, we use an LSTM core block. Therefore, a layer is composed of h-expanded LSTM core blocks. In an LSTM core block, the input m_{t-i} and the output H_{t-i-1} of the previous block can be determined:

The degree to which the previous unit state C_{t-i-1} is retained in the state C_{t-i}

How the current layer input and previous layer output affects the state

The establishment of the output H_{t-i}

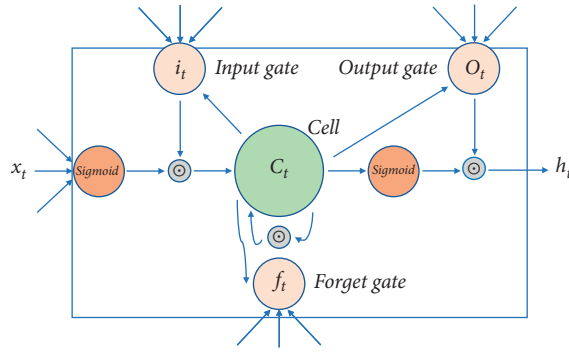

Figure 4: The function process of the LSTM core unit.

Input: input sequence
Output: prediction
(1) **while** BatchNotFinished **do**
(2) InitializeParameters(normal_log, sequence_window, num_layers, hidden_size)
(3) Connect the previous hidden state with the current input − >combine
(4) Put the combineinto forget layer, DELETE irrelevant data
(5) Create a candidate layer using combine < −cell state
(6) combine − >input layer, decide candidate layer data
(7) Calculate the vector using forget, candidate and input layers
(8) Calculate the current output
(9) Update the new hidden state
(10) **end while**
(11) Output the prediction

Algorithm 2: Anomaly detection.

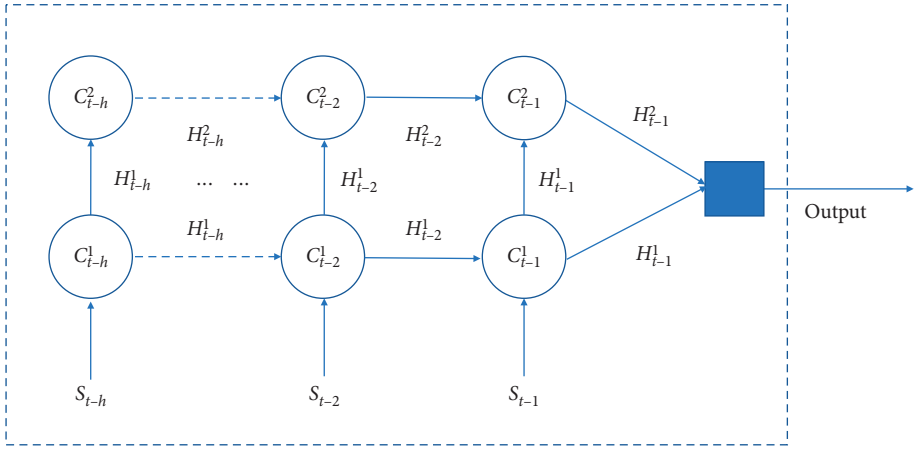

Figure 5: The detection model using LSTM.

LSTM is determined to use a combination of gate functions. These functions resolve the state by influencing the state reserved by the previous LSTM block, the previous block's output information, and the current block's input information flow. A set will learn the parameters of each gate of weights.

4. Result and Analysis

To solve the security problem of social network servers, we propose an IDS based on log analysis. The proposed method in this paper is composed of log analysis, feature extraction, and classification detection. The feature extraction part uses

the LSTM neural network so that the IDS can better extract the feature information hidden in the log and achieve better detection results. This section will analyze the dataset, evaluation method, and experimental results.

The HDFS audit log (audit log) reflects the user operations on HDFS. The detailed information includes the operation's success or failure, user name, client address, operation command, and operation directory. For each user's operation, the NameNode will organize the information in the form of key-value pairs into a log in a fixed format and then record it in the audit.log file. Through the audit log, we can view various operating conditions of HDFS in real time, track various misoperations, and do some indicator monitoring.

The dataset uses the public HDFS log dataset. The HDFS data contains 11,175,629 log messages, which are Hadoop logs collected from the Amazon EC2 platform, and are marked as normal or abnormal by experts in the relevant field [36]. The parsed dataset is shown in Table 1.

4.1. Numerical Standardization. First, calculate the average value and average absolute error of each attribute:

$$\overline{x}_k = \frac{1}{n}\sum_{i=1}^{n} x_{ik},$$
$$S_A = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_{ik} - \overline{x}_k\right)^2}. \tag{5}$$

Thereinto, x_k stands for the average value of the attribute k, S_k stands for the average absolute error of the attribute k, and x_{ik} stands for the attribute k of the record i. Then, standardize each data record:

$$Z_{it} = \frac{x_{it} - \overline{x}_i}{S_t}. \tag{6}$$

Thereinto, Z_{ik} stands for the attribute k of the record i after standardization.

4.2. Numerical Normalization. Normalize each value after standardization to the section $[0, 1]$:

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}. \tag{7}$$

Thereinto, $x_{\max}$ stands for the maximum value of the sample data, and $x_{\min}$ stands for the minimum value [37].

4.3. TF-IDF Term Weighting. In the large text corpus, some words appear very many, and they carry a small amount of information. We cannot directly use these words' frequency in the classifier, which will reduce the terms that we are interested in, but the frequency is minimal. We need to further re-weight the feature frequency of the feature into a floating-point number to facilitate the classifier's use. This step is completed by TF-IDF conversion [38].

If a word is more common, the denominator is larger, and the inverse document frequency is smaller and closer to 0. The reason for adding 1 to the denominator is to avoid the denominator being 0 (that is, all documents do not contain the word). The operator log means taking the logarithm of the obtained value:

$$\mathrm{tf}_{ij} = \frac{n_{i,j}}{\sum_k n_{k,j}},$$
$$\mathrm{idf}_i = \log\frac{|D|}{\left|\left\{j:\, t_i \in d_j\right\}\right|}. \tag{8}$$

A high word frequency in a particular document and a low word frequency in the entire document collection can produce a high-weight TF-IDF. Therefore, TF-IDF tends to sift through universal words and keep meaningful words. That is, TF-IDF is actually tf $*$ idf.

4.4. Word2Vec. The training process of word2vec is to train an external neural network to map each word in the training set to a vector space of a specified dimension [39].

The basic unit of word2vec vectorization is words. Each word is mapped to a vector of a specified dimension, and all words form a word sequence (sentence) to become a vector matrix (the number of words x in the specified word2vec embedding dimension). However, the input required by the machine learning algorithm is a one-dimensional tensor. Therefore, we also need to perform feature processing using the word vector table to perform feature encoding on the original corpus via the TF-IDF method.

Precision and Recall rate are used in this paper as indicators:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}},$$
$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{9}$$

TP. True Positive relates to the num of correctly classified as malicious

TN. True Negative relates to the num of correctly classified as benign

FP. False Positive relates to the num of mistakenly classified as malicious

FN. False Negative relates to the num of mistakenly classified as benign

$$F1 - \mathrm{Score} = \frac{2 * \mathrm{Precision} * \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}. \tag{10}$$

The F1 − score is defined as the weighted harmonic mean of the test's precision and recall [40]. The calculation of the F1 − score takes into account the precision and recall of the test. The precision, also known as a positive predictive value, is the proportion of positive results that really indicate a positive. The recall rate (also called sensitivity) is the ability to test to identify a positive result to correctly obtain an accurate positive rate. The F1 − score can reach the best value, that is, the precision and recall rate are 1. The worst

Table 1: Parsed log structure.

Type	Data
Date	17 November, 2020, 19 : 12 : 48
Module	Combo
Process	sshd (pam)[1241]
Log content	Accepted password for root from 10.21.234.33 port 8888 ssh
Log key	Accepted password for * from * port * ssh
Parameter list	("root," "10.21.234.33," "8888")

Table 2: Hardware and software environment.

No.	Type	Description
1	OS and version	Ubuntu 20.04
2	Experiment platform	Tensorflow 2.0
3	CPU	Intel Core-i7 9700 K
4	Units of CPU	8 cores 8 threads
5	Memory	DDR4 32 GB
6	Graphics card	GTX1070 8 GB
7	Storage	2 TB Nvme SSD

F1 − score means that the lowest precision and lowest recall rate should be 0.

The test environment is shown in Table 2, and the parameters' description is shown in Table 3.

The parameter normal_log and sequence_window represent the sensitivity of the IDS, and the adjustment of these two parameters is the core mission of the experiment. The parameter num_layers and hidden_size reflect the performance of the IDS, and the adjustment should both consider the effect and overhead. The following parameter values normal_log = 10, sequence_window = 10, num_layers = 2, and hidden_size = 64 are used by default to train the deep learning model. normal_log represents the range of normal conditions in the prediction output (the threshold of the normal logs), sequence_window is the sequence window size, and num_layers and hidden_size represent the number of hidden layers and the size of hidden layers in the LSTM model. Each of the four parameters will be adjusted in these experiments; with the target parameter's adjustment, the other three parameters will remain in the default value. To evaluate the model, the F1 − score and the all-around performance should be concerned. Increasing the value of sequence_window, num_layers, or hidden_size will increase the model's complexity, increasing the training cycle and system overhead. Variable adjustments are taken as the parameters, and the results are as follows.

In Figure 6, the compared parameter is the normal_log, which refers to the several top numbers of the prediction output sequence. The range of this parameter chooses from 6 to 12 because the performance will be deficient when the value is under 6, and the overhead will be high when the value is above 12. It can be concluded from the figure when the parameter is 11, the F1 − score reaches the highest point and slightly decreases when the parameter is over 11. In Figure 7, the compared parameter is the size of the sequence_window, which refers to the length of the input sequence, The range of this parameter chooses from 6 to 13 for the reason of the efficiency and cost. In the experiment, the F1 − score is both at a high point when the parameter is 9 and 13. However, the precision and recall are closer when the parameter is 13. Therefore, the sequence_window value should be 13.

In Figure 8, the compared parameter is the amount of the hidden layers (num_layers), which relates to the increase of the neural network's resolution and complexity. The range of this parameter chooses from 1 to 5 concerning the balancing of the profit and cost. It can be concluded from the figure that the F1 − score and the performance of the model become more balanced when the parameter is 2. In Figure 9, the compared parameter is the size of each hidden layer (hidden_size), which represents the amount of the node units in the hidden layer. The value of this parameter is according to the convergence, scale of the input and output layer, and the training samples. Here, we tested from 16 to 512 nodes of each hidden layer, and the figure indicates that the F1 − score has already reached a higher point when the value of this parameter is 64. And, when the parameter increases from 64, the indicator does not vary much, but the overhead of the system increases obviously. Thus, it comes to the conclusion that the parameter should be 64 in this experiment.

In conclusion, the model achieves the best performance when normal_log = 11, sequence_window = 13, num_layers = 2, and hidden_size = 64.The further comparative study showed that the proposed method has better performance than other deep learning algorithms and traditional machine learning algorithms. This paper selects MLP [41], RBM [42], SVM [43], and naive Bayes algorithms [44] for comparative analysis, uses comprehensive F1-score indicators for comparison, and uses training samples of different magnitudes. As shown in Figure 10, the algorithm proposed in this paper achieves a more accurate recognition rate than traditional machine learning algorithms and has better results than ordinary deep learning algorithms.

Table 3: Parameters' description.

No.	Parameter	Description	Adjustment range
1	normal_log	The threshold of the normal logs	6–13
2	sequence_window	The detection window size	6–13
3	num_layers	The number of hidden layers	1–6
4	hidden_size	The number of memory units	32–512

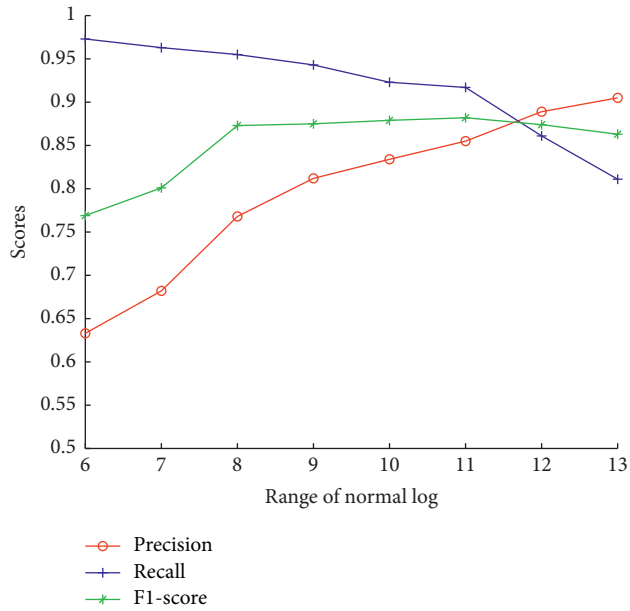

Figure 6: Scores with normal_log variation.

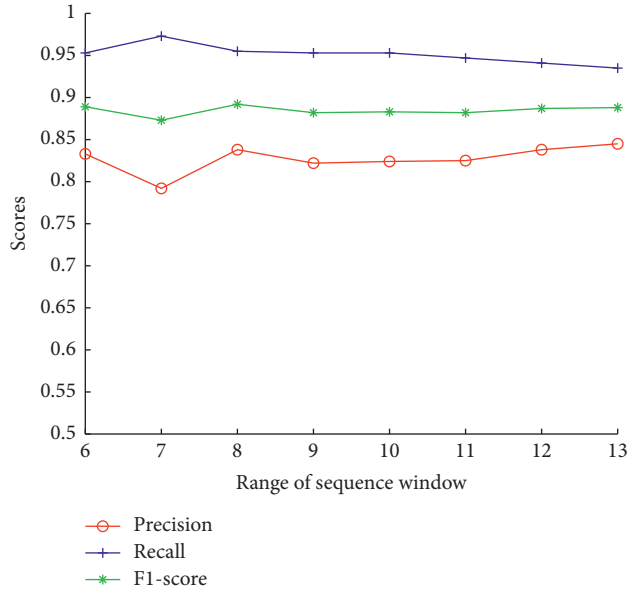

Figure 7: Scores with sequence_window variation.

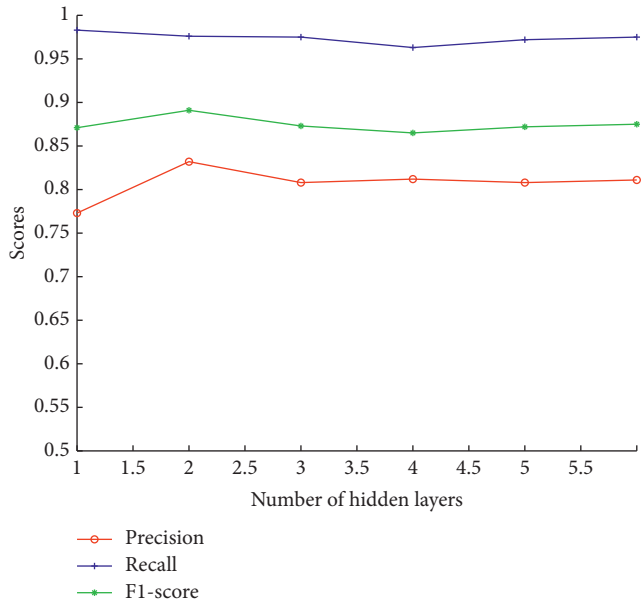

Figure 8: Scores with num_layers variation.

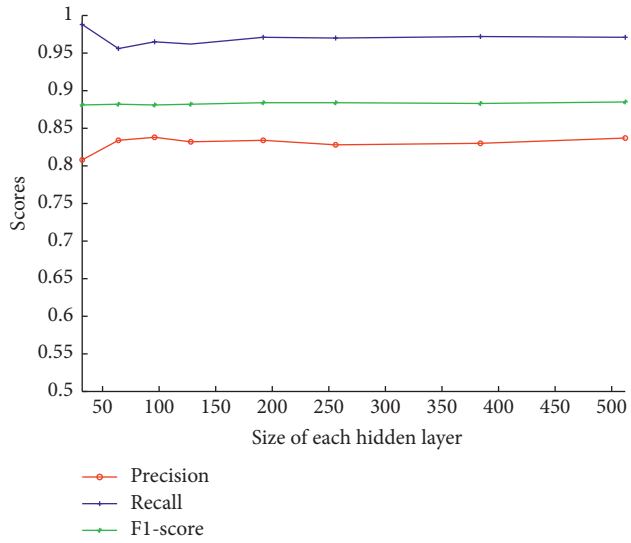

Figure 9: Scores with hidden_size variation.

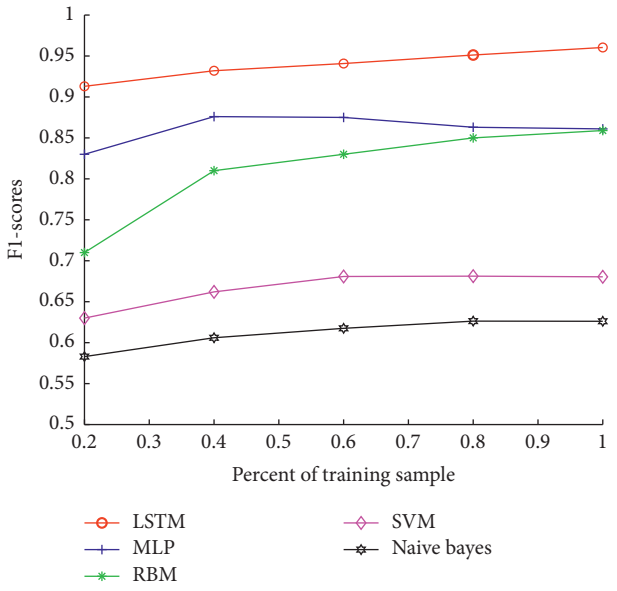

Figure 10: The comparison with other algorithms.

5. Conclusion and Future Works

Social networking sites are often faced with server security risks. The exposure of social network servers to the ubiquitous Internet environment leaves massive attacks everywhere. Applying deep learning-based log analysis methods, IDSs can detect potential cyber-attacks and prevent the social network server from being unsecured. This paper first analyzes the features of social network servers and summarizes the security issues existing in social network servers. Afterwards, the solution to the security issue was discussed, and the social network website server security protection was selected as the entry point, and the existing protection solutions were summarized. After comparing and analyzing the advantages and disadvantages, a log detection scheme based on deep learning is proposed to protect social network servers.

In this paper, a security log analysis method based on the IDS is proposed by using deep learning algorithms. The proposed IDS has three parts: log parser, feature extraction, and anomaly detection. The log analysis method combines deep learning algorithms with traditional black and white lists, rule matching, and statistical strategies to complement each other's advantages. Traditional analysis methods have good identification capabilities for known and common attack behaviors, while deep learning algorithms have an adaptive ability to detect unknown and new attack behaviors and distinguish different types of attacks. The performance of deep learning is better, and the overhead is lower. In the experiments, the proposed method is proved to have a significant improvement on the detection capabilities, which has a preferable $f1-$ score than other comparative methods. Nevertheless, this paper mainly focuses on classical datasets and algorithms but has not made further comparisons with new datasets and algorithms. More datasets and algorithms will be employed in the future work to conduct experiments, and the proposed scheme will be optimized.

Data Availability

The HDFS data used to support the findings of this study have been deposited in the Wei Xu repository (https://github.com/logpai/loghub/tree/master/HDFS).

Conflicts of Interest

There are no conflicts of interest to declare.

References

[1] R. A. Kemmerer and G. Vigna, "Intrusion detection: a brief history and overview," *Computer*, vol. 35, pp. supl27–supl30, 2002.

[2] W. Pieters, Z. Lukszo, D. Hadziosmanovic, and J. Van Den Berg, "Reconciling malicious and accidental risk in cyber security," *Journal of Internet Services and Information Security*, vol. 4, pp. 4–26, 2014.

[3] G. Vigna, W. Robertson, V. Kher, and R. A. Kemmerer, "A stateful intrusion detection system for world-wide web servers," in *Proceedings of the 19th Annual Computer Security Applications Conference*, pp. 34–43, Las Vegas, ND, USA, December 2003.

[4] D. Fincher, A. Sorkin, T. Reznor, S. Rudin, and B. Mezrich, "The social network," Sony Pictures Home Entertainment USA, Culver City, CL, USA, 2010.

[5] H. Peng, Z. Qian, Z. Kan, H. Ye, Z. Fang, and D. Zhao, "Security assessment for cascading failures of cyber-physical systems under target attack strategy," in *Proceedings of the International Conference on Frontiers in Cyber Security*, pp. 315–327, Tianjin, China, November 2020.

[6] S. Rathore, P. K. Sharma, V. Loia, Y.-S. Jeong, and J. H. Park, "Social network security: issues, challenges, threats, and solutions," *Information Sciences*, vol. 421, pp. 43–69, 2017.

[7] H. Peng, W. Peng, D. Zhao, and W. Wang, "Impact of the heterogeneity of adoption thresholds on behavior spreading in complex networks," *Applied Mathematics and Computation*, vol. 386, Article ID 125504, 2020.

[8] F. Amato, A. Castiglione, A. De Santo et al., "Recognizing human behaviours in online social networks," *Computers & Security*, vol. 74, pp. 355–370, 2018.

[9] Y. Hori, W. Claycomb, and K. Yim, "Guest editorial: frontiers in insider threats and data leakage prevention," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 3, 2012.

[10] J. Nagy and P. Pecho, "Social networks security," in *Proceedings of the 2009 Third International Conference on Emerging Security Information, Systems and Technologies*, pp. 321–325, Athens, Greece, June 2009.

[11] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, p. 122, 2019.

[12] F. Ertam, "An efficient hybrid deep learning approach for internet security," *Physica A: Statistical Mechanics and Its Applications*, vol. 535, Article ID 122492, 2019.

[13] M. A. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for internet of things (IoT) security," *IEEE Communications Surveys & Tutorials*, vol. 21, 2020.

[14] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, and M. Marchetti, "On the effectiveness of machine and deep learning for cyber security," in *Proceedings of the 2018 10th*

International Conference on Cyber Conflict (CyCon), pp. 371–390, Tallinn, Estonia, May 2018.

[15] W. Guo, D. Mu, J. Xu, P. Su, G. Wang, and X. L. Xing, "Explaining deep learning based security applications," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 364–379, Toronto, Canada, October 2018.

[16] X. Ling, S. Ji, J. Zou et al., "Deepsec: a uniform platform for security analysis of deep learning model," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*, pp. 673–690, San Francisco, CA, USA, May 2019.

[17] M. Alazab and M. Tang, *Deep Learning Applications for Cyber Security*, Springer, Berlin, Germany, 2019.

[18] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, Article ID 102419, 2020.

[19] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges," *Archives of Computational Methods in Engineering*, vol. 33, pp. 1–33, 2020.

[20] H. Peng, Z. Kan, D. Zhao, and J. Han, "Security assessment for interdependent heterogeneous cyber physical systems," *Mobile Networks and Applications*, vol. 42, pp. 1–11, 2019.

[21] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pp. 21–26, New York , NY, USA, May 2016.

[22] H. J. Liao, C. H. R. Lin, Y. C. Lin, and K. Y. Tung, "Intrusion detection system: a comprehensive review," *Journal of Network and Computer Applications*, vol. 36, pp. 16–24, 2013.

[23] T. Booth and K. Andersson, "Network security of internet services: eliminate DDoS reflection amplification attacks," *Journal of Internet Services and Information Security (JISIS)*, vol. 5, pp. 58–79, 2015.

[24] C. Atapour, I. Agrafiotis, and S. Creese, "Modeling Advanced Persistent Threats to enhance anomaly detection techniques," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 9, 2018.

[25] S. R. Chhetri, A. B. Lopez, J. Wan, and M. A. Al Faruque, "Gan-sec: generative adversarial network modeling for the security analysis of cyber-physical production systems," in *Proceedings of the 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 770–775, Florence, Italy, March 2019.

[26] E. Aminanto and K. Kim, "Deep learning in intrusion detection system: an overview," in *Proceedings of the 2016 International Research Conference on Engineering and Technology (2016 IRCET). Higher Education Forum*, Seoul, South Korea, January 2016.

[27] C. F. Tsai, Y. F. Hsu, C. Y. Lin, and W. Y. Lin, "Intrusion detection by machine learning: a review," *Expert Systems with Applications*, vol. 36, pp. 11994–12000, 2009.

[28] S. S. Roy, A. Mallik, R. Gulati, M. S. Obaidat, and P. V. Krishna, "A deep learning based artificial neural network approach for intrusion detection," in *Proceedings of the International Conference on Mathematics and Computing*, pp. 44–53, Haldia, India, January 2017.

[29] G. Meena and R. R. Choudhary, "A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA," in *Proceedings of the 2017 International Conference on Computer, Communications and Electronics (Comptelix)*, pp. 553–558, Manipal, India, April 2017.

[30] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *Proceedings of the 2016 15th IEEE international conference on machine learning and applications (ICMLA)*, pp. 195–200, Anaheim, CL, USA, December 2016.

[31] A. F. M. Agarap, "A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data," in *Proceedings of the 2018 10th International Conference on Machine Learning and Computing*, pp. 26–30, Macau, China, March 2018.

[32] L. Stepanov, A. Parinov, L. Korotkikh, and A. Koltsov, "Approach to estimation of level of information security at enterprise based on genetic algorithm," *Journal of Physics: Conference Series*, vol. 1015, Article ID 032141, 2018.

[33] Q. Fu, J. G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *Proceedings of the 2009 ninth IEEE international conference on data mining*, pp. 149–158, Miami, FL,USA,, December 2009.

[34] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," 1999.

[35] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proceedings of the Thirteenth annual conference of the international speech communication association*, Portland, OR, USA, September 2012.

[36] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pp. 117–132, Big Sky, MT, USA, October 2009.

[37] M. Zhong, Y. Zhou, and G. Chen, "Sequential model based intrusion detection system for IoT servers using deep learning methods," *Sensors*, vol. 21, 2021.

[38] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, pp. 133–142, Washington D.C., USA, August 2003.

[39] Y. Goldberg and O. Levy, "Word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method," 2014, https://arxiv.org/abs/1402.3722.

[40] R. Joshi, "Accuracy, precision, recall & f1 score: interpretation of performance measures," 2016.

[41] P. Barapatre, N. Tarapore, S. Pukale, and M. Dhore, "Training MLP neural network to reduce false alerts in IDS," in *Proceedings of the 2008 International Conference on Computing, Communication and Networking*, pp. 1–7, Dalian, China, July 2008.

[42] M. Mayuranathan, M. Murugan, and V. Dhanakoti, "Best features based intrusion detection system by RBM model for detecting DDoS in cloud environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 23, pp. 1–11, 2019.

[43] J. Wang, X. Hong, R. R. Ren, and T. H. Li, "A real-time intrusion detection system based on PSO-SVM," in *Proceedings of the 2009 International Workshop on Information Security and Application (IWISA 2009)*, p. 319, Busan, Korea, August 2009.

[44] S. Mukherjee and N. Sharma, "Intrusion detection using naive bayes classifier with feature reduction," *Procedia Technology*, vol. 4, pp. 119–128, 2012.