Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6830372, 16 pages
https://doi.org/10.1155/2021/6830372

Research Article

LCHI: Low-Order Correlation and High-Order Interaction Integrated Model Oriented to Network Intrusion Detection

Shengwei Lei,[1] **Chunhe Xia**,[1] **and Tianbo Wang**[2]

[1]*Key Laboratory of Beijing Network Technology, Beihang University, Beijing 100191, China*
[2]*School of Cyber Science and Technology, Beihang University, Beijing, China*

Correspondence should be addressed to Tianbo Wang; wangtb@buaa.edu.cn

Received 16 June 2021; Revised 20 September 2021; Accepted 7 October 2021; Published 26 October 2021

Academic Editor: Zhihan Lv

Copyright © 2021 Shengwei Lei et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Network intrusion poses a severe threat to the Internet of Things (IoT). Thus, it is essential to study information security protection technology in IoT. Learning sophisticated feature interactions is critical in improving detection accuracy for network intrusion. Despite significant progress, existing methods seem to have a strong bias towards single low- or high-order feature interaction. Moreover, they always extract all possible low-order interactions indiscriminately, introducing too much noise. To address the above problems, we propose a low-order correlation and high-order interaction (LCHI) integrated feature extraction model. First, we selectively extract the beneficial low-order correlation between the same-type features by the multivariate correlation analysis (MCA) model and attention mechanism. Second, we extract the complicated high-order feature interaction by the deep neural network (DNN) model. Finally, we emphasize both the low- and high-order feature interactions and incorporate them. Our LCHI model seamlessly combines the linearity of MCA in modeling lower-order feature correlation and the nonlinearity of DNN in modeling higher-order feature interaction. Conceptually, our LCHI is more expressive than the previous models. We carry on a series of experiments on the public wireless and wired network intrusion detection datasets. The experimental results show that LCHI improves 1.06%, 2.46%, 3.74%, 0.25%, 1.17%, and 0.64% on the AWID, NSL-KDD, UNSW-NB15, CICIDS 2017, CICIDS 2018, and DAPT 2020 datasets, respectively.

1. Introduction

The COVID-19 pandemic in 2020 moved the main scenes of people's lives and work from offline to the Internet overnight. Based on "Cisco's 2021 Global Networking Trends Report [1]," 14.6 billion IoT devices will be connected to the network by 2022. The scale and complexity of IoT networks are rapidly expanding, and people are facing increasingly complex network attacks. Cisco checks 47 terabytes of network traffic every day, analyzes 28 billion flows, and records 1.2 trillion security incidents. The industrial control systems, networking equipment, and industrial cloud platforms are more vulnerable to intrusions under increasingly open network connections, causing lots of loss. Thus, it is essential to study information security protection technology in IoT. Among them, network intrusion detection is the foundation and core of ensuring IoT security.

Network intrusions detection consists of abuse-based and anomaly-based detection [2]. Abuse-based detection systems usually set rules to match network behavior. Thus, they get a high detection rate and low false-positive rate for known intrusions. However, unknown intrusions and their variants, zero-day attacks can easily evade them. Moreover, maintaining an updated rules database is a complex and labor-intensive task, as the rule setting needs a large amount of network security expertise. On the contrary, anomaly-based detection techniques are more promising for detecting zero-day and unknown intrusions [3], and they have become the mainstream.

At present, more effective methods of network intrusion detection are to use deep learning models. As a powerful approach to learning feature representation, deep learning models have the potential to learn sophisticated feature interactions. These models (e.g., convolutional neural network

(CNN) [4], recurrent neural network (RNN)/long short-term memory (LSTM) [5], DNN [6]) fitting the best classification curve through data mining and statistical analysis. Moreover, they are not constrained by expertise in network security; the profiles of legitimate behaviors are developed based on data mining and statistical analysis. Thus, they achieved a relatively satisfactory performance. However, there are some disadvantages to these models. For example, the CNN-based models are biased to the interactions between neighboring features, while RNN-based models are more suitable for data with sequential dependency. The existing deep learning models only capture the high-order feature interaction and always fully or partly ignoring the low-order interaction (e.g., multivariate correlation among features) [7]. However, in reality, multivariate correlations are widely present in intrusion data. For example, the identification, flag, and offset features in the network layer cooperate to guide IP fragmentation reorganization; the transport layer's SYN, FIN, ACK, PSH, and RST features work together to reflect the TCP status. Thus, ignoring the low-order feature correlation is not conductive to classification.

Meanwhile, some researchers have explored the feature correlation to distinguish between different intrusions. In [7, 8], researchers used the MCA method for accurate network traffic characterization by extracting the geometrical correlations between features. They extracted every correlation between every two different features. Dong et al. [9] proposed an intrusion detection model based on the MCA-LSTM from the temporal correlation features of intrusion data. From the above literatures, we notice that the current correlation analysis methods always extract the correlations between any two different features. There are some problems: (1) dimension disaster and too much noise: if they do not perform feature selection, any feature combination (i.e., correlation) is considered. Thus, the feature dimension of generated multivariate correlations is too large, resulting in dimension disaster and overfitting. Moreover, in these feature combinations, only a tiny part is meaningful; most are noise. In network intrusion data, there are usually protocol-type and statistical-type features. The correlation between the same types of features is strong. However, the correlation between different types of features is weak. Thus, generating the correlation between different types of features brings in too much noise. (2) Too much useful correlation loss: if they conduct the feature selection, thus many useful feature combinations will be lost due to the limited features. Moreover, the weak feature correlation between different types is still generated. Too much correlation loss and noise are not conducive to feature extraction.

In summary, the existing models are biased to low- or high-order feature interaction. And in the low-order correlation extraction, too much noise is generated since they seldom consider the difference of features and the influence of correlations generated by different types of features. To solve the above problems, we propose a low-order correlation and high-order interaction integrated feature extraction model. Following contributions have been made in this paper:

(1) To extract features more perfectly, we integrate the low-order correlation captured by the MCA model and the high-order interaction obtained by the DNN model

(2) To characterize low-order correlation more effectively, we comprehensively analyze the difference of features and divide them into different types. We selectively extract the useful low-order correlation between features in the same type, avoiding the dimension disaster and too much noise or correlation loss

(3) To consider the classification influence in generated correlations, we employ attention to estimate the importance of different correlations when incorporating their latent representations

(4) To evaluate the effectiveness and robustness of our LCHI model, we conduct a series of experiments on public wireless (e.g., AWID) and wire datasets (e.g., NSL-KDD, UNSW-NB15, CICIDS 2017, CICIDS 2018, and DAPT 2020). The experimental results show that our model has higher accuracy than the previous models

The rest of this paper is organized as follows: Section 2 describes the related works. In section 3, we state the problems of low- and high-order feature interaction extraction. Section 4 presents the LCHI model and key technologies. We present the theoretical analysis of our model and experimental evaluation in Section 5 and 6, respectively. Section 7 concludes this paper.

2. Related Works

To improve the performance of network intrusion detection, researchers try to extract features from different aspects. This section mainly explains the current situation from two perspectives: high-order feature interaction and low-order feature correlation.

2.1. High-Order Feature Interaction Extraction. The deep learning models are widely adopted to network intrusion detection to extract the sophisticated hidden features [10–15]. In intrusion detection, Wang et al. [16] proposed an SDAE-ELM-based integrated deep intrusion detection model to overcome the long training time and low classification accuracy and to achieve timely response to intrusion behavior. They also constructed a DBN-Softmax-based integrated mode for host intrusion detection. The experimental results show that the proposed models can effectively improve detection accuracy. Ge et al. [6] proposed a feed-forward neural network model with embedding layers (to encode high-dimensional categorical features) for multiclass classification. Using a second feed-forward neural networks model, they applied transfer learning to encode high-dimensional categorical features to build a binary classifier. Kasongo et al. [17] proposed a feed-forward deep neural network (FFDNN) wireless IDS system using a Wrapper-Based Feature Extraction Unit (WFEU). The WFEU used the Extra

Trees algorithm to generate a reduced optimal feature vector. They compared the WFEU-FFDNN to the standard machine learning algorithms that include random forest (RF), support vector machine (SVM), NaÃ¯ve Bayes (NB), decision tree (DT), and k-nearest neighbor (kNN) and got a better detection performance.

Some researchers adopt the RNN/LSTM and CNN models to process the temporal-spatial features of intrusion data. Zhang et al. [5] proposed a unified model combining multiscale convolutional neural network with long short-term memory (MSCNN-LSTM). The model processed the spatial and temporal features by the MSCNN and LSTM models, respectively. And then, the spatial-temporal features were employed to perform the classification. For the real-time detection of anomalies within the in-vehicle network, Khan et al. [18] developed LSTM based false information attack/anomaly detection model. They adopted the LSTM model to process the time-series data to get the sequential variation patterns. In the face of zero-day attacks, Sms et al. [19] applied RNN models to find complex patterns in attacks and generated similar ones. They demonstrated that RNNs are helpful to generate new, unseen mutants of attacks as well as synthetic signatures from the most advanced malware to improve the intrusion detection rate. The authors in [20] proposed a feature representation method to transform the raw flow-based statistical features into more discriminative representations and developed an ensemble of machine learning-based classifiers optimized to discriminate the malicious flows from the benign ones.

Other deep learning models are also applied to intrusion detection. Balasundaram et al. [21] proposed the novel bat extreme learning three-tier intrusion detection architecture based on novel bat optimized extreme learning machines to detect the various cyberattacks over the IoT network. The proposed system consists of scalable IoT deployable software and a data analyzer mechanism. Shitharth et al. [22] proposed an Intrusion Weighted Particle-Based Cuckoo Search Optimization (IWP-CSO) and Hierarchical Neuron Architecture-Based Neural Network (HNA-NN) techniques to detect the encrypted and external intrusions. Gu et al. [23] proposed an SVM and naive Bayes-based intrusion detection framework. They implemented the naive Bayes feature transformation to generate new data with high quality and then adopted an SVM classifier using the transformed data to build the intrusion detection model. The above models usually achieved better performance in intrusion detection. However, these methods commonly suffer from high false-positive rates because they only focus on the high-order feature interaction, intrinsically fully or partly neglecting the low-order correlation.

2.2. Low-Order Feature Correlation Extraction. Some researchers have explored the low-order feature correlation. Tan et al. [7] used the MCA for traffic characterization by extracting the geometrical correlations between two different features. And then they used the traditional statistical analysis method to construct a normal profile to distinguish normal traffic flow from DDoS flow. Yu et al. [24] exploited the flow correlation coefficient to distinguish the DDoS from the flash event. Like [7], they generated the correlations between any two different features, and the generated feature dimension was too large, resulting in dimension disaster easily. Jamdagni et al. [25] developed a refined geometrical structure-based analysis technique, where the Mahalanobis distance was used to extract the correlations between the selected packet payload features. To improve the low detection performance of network intrusion detection models caused by high-dimensional data, Dong et al. [9] proposed an MCA-LSTM intrusion detection model from the time correlation characteristics of intrusion data. They first made a feature selection process to select the optimal feature subsets based on the information gain. And then they used the MCA method to extract the correlations between different features. Finally, they adopted an LSTM model to extract the temporal features based on the correlations generated by the MCA. However, this method did not effectively integrate the low- and high-order feature interactions. Like [25, 26], the methods extracted the correlations from the selected feature subset, resulting in too much loss of useful correlations.

Factorization machines (FMs) [27, 28] are a popular solution that combines the advantages of SVM with factorization models. FMs model all interactions between variables using factorized parameters. It is a supervised learning approach that enhances the linear regression model by incorporating the second-order feature interactions. However, FM models feature interactions in a linear way, which can be insufficient for capturing the nonlinear and complex inherent structure of real-world data. To alleviate the IoT-dependent botnet attacks newfangled danger, Arul [29] suggested and estimated a deep nonlinear regression least-squares polynomial fit to recognize peculiar system traffic originating as of conceded IoT gadgets. However, the method generated too much noise.

From the above analysis, we notice that the current low-order feature correlation extraction methods usually extract possible correlations or interactions between any two different features, resulting in too much noise. Few consider the difference between different features when extracting low-order correlation, resulting in low detection performance.

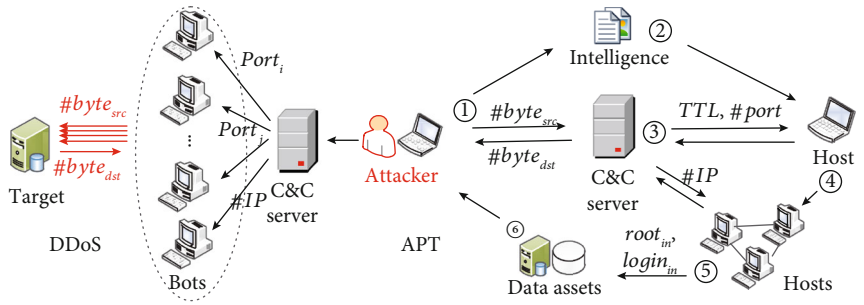
3. Problems Statement

The limited original features in practice are not enough to characterize different attacks. Thus, it is important for network intrusion detection to learn feature interactions. There are two types of feature interactions:

(1) Low-order feature interaction (FI_L).

$$\begin{cases} \mathrm{FI}_L = f_{\text{explicit}}(x_i, x_j), \\ f_{\text{explicit}} \in \{+, -, *, /\}, \end{cases} \tag{1}$$

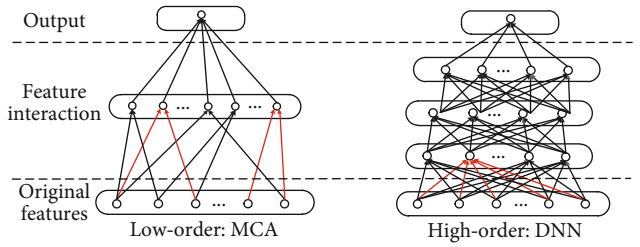
where the x_i, x_j are different features. The FI_L is easy to obtain and understand based on background knowledge. As shown in the left subfigure in Figure 1(a), the $\#\text{byte}_{\text{src}}/\#\text{byte}_{\text{dst}}$



Low-order feature interaction:
$FI_L = \{\#byte_{src}/\#byte_{dst}, port_i - port_j, \#byte_{src}/\#IP, \cdots\}$

High-order feature interaction:
$FI_H = g_{implicit}(\#byte_{src}, \#byte_{dst}, \#port, \#IP, TTL, root_{in}, login_{in}, \cdots)$

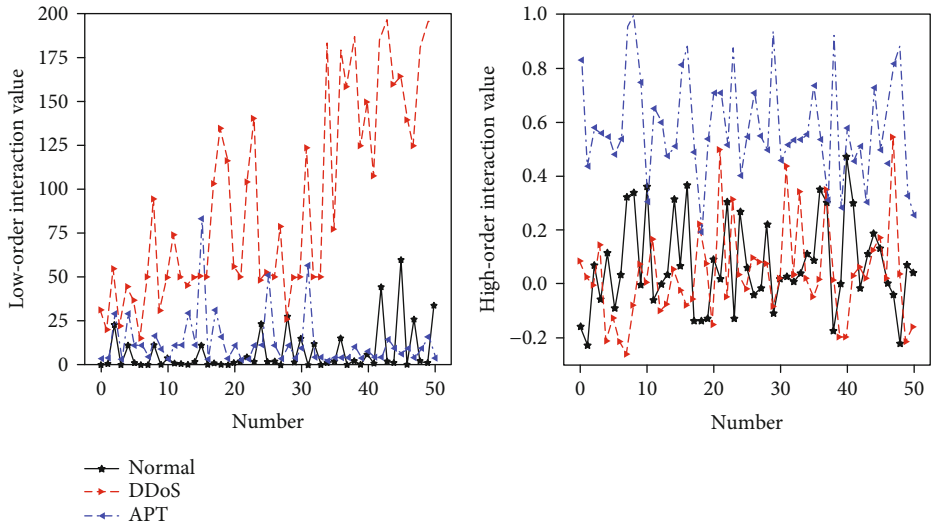
(a) The diagram in different attack scenarios



$FI_L = f_{explicit}(x_i, x_j)$, $f_{explicit} \in \{+, -, *, /\}$

$FI_H = g_{implicit}(x_i, x_j, x_k, \ldots)$, $g_{implicit} \in \{+, -, *, /, \xi\}*$

(b) Typical model of low- or high-order feature interaction



(c) Image effect of single low- or high-order feature interaction

Dataset	Low-order interaction			High-order interaction		
	PR	FM	MCA	CNN	LSTM	DNN
NSL-KDD	73.69	76.05	77.76	77.42	77.72	81.51
UNSW-NB15	70.13	74.29	76.14	75.17	76.20	77.04

(d) Detection accuracy (%) of single low- or high-order interaction

FIGURE 1: Current situation of feature interaction in network intrusion detection. The problems lie in (1) single low- or high-order feature interaction extraction and (2) too much noise while extracting all possible low-order feature interactions indiscriminately. In subfigure (c), the FI_L value is the correlation of $\#byte_{src}/\#byte_{dst}$ by the MCA model [7]; the FI_H value (interaction among the $\#byte_{src}$, $\#byte_{dst}$, #port, #IP, TTL, $root_{in}$, $login_{in}$, $\cdots$) is the 3-layer output of the DNN model. In subfigure (d), the PR denotes the polynomial regression [30], and the FM denotes the factorization machines [27].

Table 1: Detection results (%) of MCA method from the same, different, and all feature types.

Dataset	MCA types	Dimension	Model accuracy		
			CNN	LSTM	DNN
NSL-KDD	Same type	285	79.76	79.64	81.48
	Different types	535	78.59	78.04	78.45
	All types	820	78.27	78.45	80.09
UNSW-NB15	Same type	309	78.96	79.10	79.23
	Different types	552	76.28	78.31	77.04
	All types	861	78.19	78.24	78.48

directly reflects the principle of a DDoS attack. Since attackers often exploit bots to carry on the DDoS attack, we can infer that the $\text{port}_i - \text{port}_j$ and $\#\text{byte}_{\text{src}}/\#\text{IP}$ may effectively distinguish between normal and DDoS flows. However, among these FI_L, only a few are useful, and the others are noise. Thus, we need to selectively and effectively generate and evaluate meaningful FI_L.

(2) High-order feature interaction (FI_H).

$$\begin{cases} \text{FI}_H = g_{\text{implicit}}\left(x_i, x_j, x_k, \cdots\right), \\ g_{\text{implicit}} \in \{+, -, *, /, \xi\}^*, \end{cases} \tag{2}$$

where the parameter ξ contains $\odot, \otimes, \sum, \int$, log, etc. And the symbol $*$ in the upper right corner denotes the Kleene star operator. The FI_H is unknown and hard to understand, and it can be only extracted by deep learning models. As shown in the right subfigure of Figure 1(a), the interaction rules among features ($\#\text{byte}_{\text{src}}$, $\#\text{byte}_{\text{dst}}$, #port, #IP, TTL, root_{in}, login_{in}, $\cdots$) in the APT attack are unknown and hard to capture. In general, such feature interactions can be highly sophisticated, where both low- and high-order feature interactions should play important roles.

To distinguish the low-order feature interaction from the high-order feature interaction, we usually call the low-order interaction the low-order correlation. The low- and high-order feature interactions are both useful for distinguishing different attacks. However, existing research does not make full use of them. There are two problems:

Problem 1: the existing works are all biased to single low- or high-order feature interaction. However, the detection accuracy of single low- or high-order interaction is not satisfied, as shown in Figure 1(d). The environment setup is the same as Section 6. The typical low- or high-order feature interaction model is shown in Figure 1(b). Figure 1(c) represents the image effect of a single low- or high-order feature interaction in the normal, DDoS, and APT attacks. Without the high-order interaction, we can also distinguish between the DDoS and normal flows, as shown in the left subfigure of Figure 1(c). However, the APT attack is much more complex, and it is hard to distinguish only by the low-order interaction. Referring to the left subfigure of Figure 1(c), the APT flow is similar to the normal flow. Without the low-order interaction, we can also distinguish between the APT and normal flows based on the single high-order interaction, as shown in the right subfigure of Figure 1(c). However, lacking harmonization of low-order interaction, sometimes deep learning models overfit the complex and unknown APT attack as the samples are rare, resulting in the DDoS and normal flows intertwined, as shown in the right subfigure Figure 1(c).

Problem 2: the low-order correlation methods introduce too much noise as they always extract all possible correlations between any two different features *indiscriminately*. Among these correlations, only a few parts are useful; most are noise. Network intrusion data contains different types of features (e.g., protocol-type and statistical-type). The correlation between features of the same type is strong. However, the correlation between different types of features is weak and easy to become noise. Table 1 shows the effects of correlations generated between features in the same type, different types, and all types on the CNN, LSTM, and DNN models. The experimental setup is the same as Section 6. The accuracy in the same type is much higher than the others, and the feature dimension is lower. The accuracy in different types is much lower, and the feature dimension is not low. Due to the adverse effect of the correlations generated by different types, the accuracy of all types is limited. Furthermore, the feature dimension of all types is the highest, which needs much more training time.

4. LCHI Feature Extraction Model

4.1. Framework of the LCHI Model. Figure 2 shows the overall architecture of the LCHI model. The processes are as follows: Firstly, we extract the low-order feature correlation. We propose a feature division-multivariate correlation analysis (FD-MCA) method. We divide the network intrusion features into different types based on the characteristics of intrusion data (e.g., protocol-type features and statistical-type features). Based on the network background knowledge, we classify the correlated features into the same type as much as possible. Thus, the correlation between the same types is strong (e.g., SYN/FIN/ACK/PSH/RST), and the correlation between different types is weak. And then extract the low-order correlation between features only in the same type by the MCA method. In addition, we further fuse the correlations through an attention mechanism, considering the different effects of correlations. We, respectively, leverage the feature vectors of correlations obtained from different types into the same dimensional dense embedding of a common latent space. It is to lay the foundation for the following

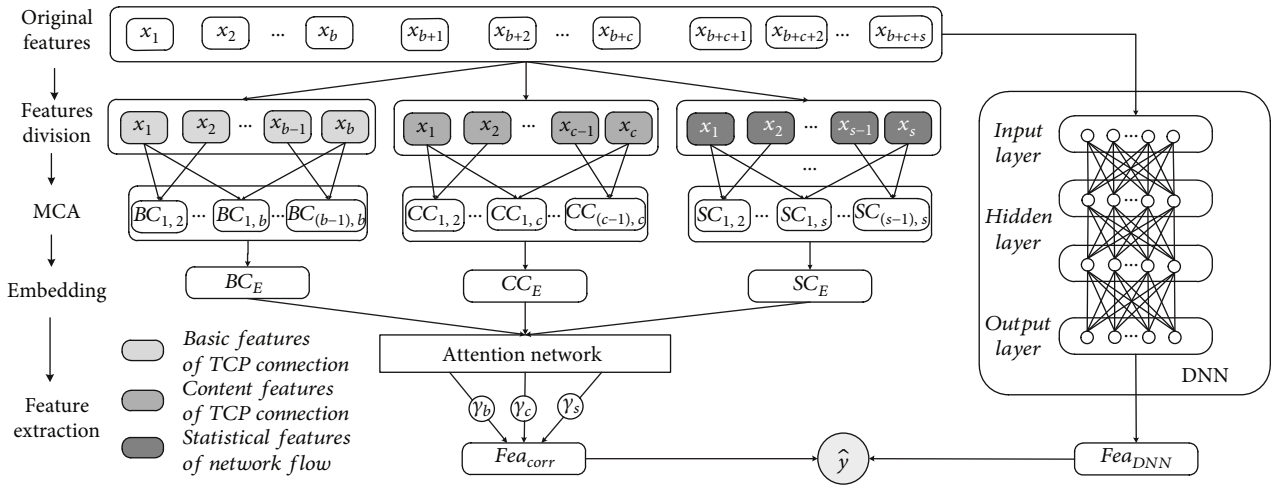


Figure 2: The framework of the LCHI model.

attention mechanism. Then, considering the different effects of generated correlations, we fuse the correlations through an attention mechanism. Secondly, we extract the high-order feature interaction based on the DNN model due to the full-connection characteristics. Finally, we incorporate the low-order correlation and high-order interaction to classify different attacks.

4.2. Low-Order Correlation Extraction by MCA and Attention

4.2.1. FD-MCA Method. The network intrusion data we usually obtain mainly contains three-type features:

(1) *The basic features of TCP connection:* These features are obtained from the protocol header fields and cooperate to guide network data transmission (e.g., protocol, service)

(2) *The content features of TCP connection:* These features are usually embedded in the data payload of the packet and reflect the content characteristics of intrusion behavior (e.g., num_failed_logins and num_root)

(3) *The statistical features of Network flow:* These features are extracted from the network operation process and usually indicate the network operation status (e.g., same_srv_rate and dst_host_count)

Since the gap is vast between different types of features, the correlation between different type features may become the noise and should be ignored. Even though there are some correlations between the different type features that we do not know, the noncorrelated situations far exceed the correlated cases among all the feature correlations. Thus, the useful correlations generated are far less than the noise generated. Therefore, we divide the intrusion data into different types as shown above and only extract the multivariate correlations between the same-type features. It not only avoids introducing too many useless correlations but also generates useful correlations as much as possible. We apply the MCA [7] to extract the correlations by constructing the triangle area map (TAM).

The network intrusion data is $D = \{d_1, d_2, \cdots, d_n\}$, and $d_i = [x_1^i x_2^i \cdots x_m^i]^T$, $(1 \le i \le n)$ denotes the ith m-dimension record. We first extract the geometrical correlation between the same-type features and map each record d_i to TAM^i. Triangle area mapping is used to extract the correlation between the jth and kth features. Then, we project the record d_i on the (j,k)-th two-dimension Euclidean:

$$y_{i,j,k} = \left[\varepsilon_j \varepsilon_k\right]^T d_i = \left[f_j^i f_k^i\right]^T, \tag{3}$$

where $1 \le j \le m, 1 \le k \le m$, $\varepsilon_j = [e_{j,1}, e_{j,2}, \cdots, e_{j,m}]^T$, and $\varepsilon_k = [e_{k,1}, e_{k,2}, \cdots, e_{k,m}]^T$. $e_{j,j} = 1$, $e_{k,k} = 1$, and the others are 0. The $y_{i,j,k}$ is defined as a point on the Cartesian coordinate system in the (j,k)-th 2D Euclidean subspace with coordinate (f_j^i, f_k^i). Then, we build a triangle $\Delta f_j^i O f_k^i$ formed by the origin and the projected points of the coordinate (f_j^i, f_k^i) on the j-axis and k-axis. The triangle area $Tr_{j,k}^i$ is defined as

$$Tr_{j,k}^i = \begin{cases} \left(\left\| \left(f_j^i, 0\right) - (0,0) \right\| \times \left\| \left(0, f_k^i\right) - (0,0) \right\| \right)/2, & j \ne k \\ 0, & j = k \end{cases}. \tag{4}$$

Thus, we construct the correlation of TAM in every feature type, and all the triangle areas are arranged on the map with respect to their indexes. $Tr_{j,k}^i$ denotes the jth raw and the kth column of the TAM^i. According to Equation (2), $Tr_{j,k}^i = Tr_{k,j}^i$ and $Tr_{j,k}^i = 0$ when $j = k$. Thus, the TAM is symmetric:

$$TAM^i = \begin{bmatrix} 0 & Tr^i_{1,2} & \cdots & Tr^i_{1,m} \\ Tr^i_{2,1} & 0 & \cdots & Tr^i_{2,m} \\ \cdots & \cdots & \cdots & \cdots \\ Tr^i_{m,1} & Tr^i_{m,2} & \cdots & 0 \end{bmatrix}. \quad (5)$$

Thus, we only take the upper triangle of the TAM:

$$\mathrm{TAM}^i_{\mathrm{upper}} = \left\{ Tr^i_{j,k} \right\}, 1 \le j < k \le m. \quad (6)$$

Finally, we get the generated correlations from different feature types: correlations between the basic features of TCP connection (BC), correlations between the content features of TCP connection (CC), and correlations between the statistical features of network flow (SC). As shown in Figure 2, the original feature dimension is $m = b + c + s$. The dimension of nonrepetitive and nonzero generated correlations by the existing MCA method is C^2_{b+c+s}. However, in our FD-MCA method, the dimension is much lower, only $C^2_b + C^2_c + C^2_s$.

4.2.2. Attention Mechanism for the Generated Correlations. This section evaluates the different impacts of generated correlations. Network intrusion data contains various low-order correlations. However, these three correlations have their emphasis and are not equally useful for distinguishing attacks. Different attacks have different weights on these correlations. Our model can be hindered by its modeling of these correlations with the same weight if we ignore the differences. Therefore, we improve our model by discriminating the importance of these correlations. The attention mechanism discriminates the importance of different components when compressing them into a single representation.

Since the dimensions of generated correlations are not equal, we first transform them into the same dimension. The original dimensions of BC, CC, and SC are C^2_b, C^2_c, and C^2_s, respectively. The dimension unification is as follows:

$$\begin{cases} BC_E = w_{BC} BC + b_{BC}, \\ CC_E = w_{CC} CC + b_{CC}, \\ SC_E = w_{SC} SC + b_{SC}, \end{cases} \quad (7)$$

where $w_{BC}, w_{CC}, w_{SC}, b_{BC}, b_{CC}$, and b_{SC} are parameters.

The single representation of correlations is as follows:

$$Fea_{corr} = \gamma_b BC_E + \gamma_c CC_E + \gamma_s SC_E, \quad (8)$$

where γ_b, γ_c, and γ_s are the attention scores for correlations BC_E, CC_E and SC_E, respectively. $BC_E, CC_E, SC_E \in \mathbb{R}^d$. Fea_{corr} denotes the fused features. Here, the attention scores are calculated via a two-layer attention network:

$$\begin{cases} \gamma'_b = w_c^T \sigma(W_c BC_E + b_c) + c_c, \\ \gamma'_c = w_c^T \sigma(W_c CC_E + b_c) + c_c, \\ \gamma'_s = w_c^T \sigma(W_c SC_E + b_c) + c_c, \end{cases} \quad (9)$$

where $W_c \in \mathbb{R}^{t \times d}$, $b_c \in \mathbb{R}^t$ is the first layer parameters, $w_c \in \mathbb{R}^t$, $c_c \in \mathbb{R}$ is the second layer parameters, and t is the size of hidden layer, which we call attention factor. The symbol σ is the activation function, and we use the ReLU, which empirically shows good performance. The attention scores are normalized through the softmax function, a common practice by previous work, as follows:

$$\gamma_b = \frac{\exp\left(\gamma'_b\right)}{\sum \exp\left(\gamma'\right)}, \gamma_c = \frac{\exp\left(\gamma'_c\right)}{\sum \exp\left(\gamma'\right)}, \gamma_s = \frac{\exp\left(\gamma'_s\right)}{\sum \exp\left(\gamma'\right)}, \quad (10)$$

where $\sum \exp(\gamma') = \exp(\gamma'_b) + \exp(\gamma'_c) + \exp(\gamma'_s)$.

4.3. High-Order Interaction Extraction by DNN. In this section, we introduce a deep learning model to extract the high-order feature interaction. Due to the fully connected characteristic of the DNN model, we employ a DNN to extract the high-order feature interaction in an implicit and nonlinear manner, as follows:

$$z^l = \sigma\left(w^l_d z^{l-1} + b^l_d\right), \quad (11)$$

where z^0 denotes the original features. l denotes the hidden layer, and L is the number of hidden layers. w_d and b_d are the parameters. $Fea_{DNN} = z^L$ denotes the output of the DNN. It is worth mentioning that the FD-MCA part and the DNN part share the same input, which may enable learning both low-order and high-order feature interactions from original features.

4.4. Fusion of the LCHI. According to the literature [31], considering the low- and high-order feature interactions simultaneously brings additional improvement over the cases of considering either alone in recommender systems. Inspired by this, we further incorporate the low-order correlation $\mathrm{Fea}_{\mathrm{corr}}$ and high-order interaction $\mathrm{Fea}_{\mathrm{DNN}}$ for the resulting output of the LCHI model:

$$\hat{y} = \sigma\left(w^T_{\mathrm{corr}} \mathrm{Fea}_{\mathrm{corr}} + w^T_{\mathrm{dnn}} \mathrm{Fea}_{\mathrm{DNN}} + b\right), \quad (12)$$

where $w_{\mathrm{corr}}, w_{\mathrm{dnn}}$ and b are the parameters. The loss function of the LCHI model is

$$\mathrm{Loss} = -\frac{1}{n} \sum y^L \log(\hat{y}) + \left(1 - y^L\right) \log(1 - \hat{y}), \quad (13)$$

where n is the number of records, y^L denotes the label of the original input sample, and $\hat{y}$ denotes the predicted result.

5. Comparison of Existing Models Theoretically

This section compares the proposed LCHI model with the typical low- (e.g., PR, FM, and MCA) and high-order interaction models (e.g., CNN, LSTM, and DNN). The characteristics of each model are shown in Table 2.

5.1. CNN. The kernel convolution operation is $y^l_{\text{local}} = \sigma(w^l x^{l-1}_{\text{local}} + b^l)$, where x^{l-1}_{local} and y^l_{local} are the local input and output of layer l. Thus, the local perception characteristic of the model makes it more inclined to the interaction between neighbor features, resulting in the loss of interactions between features that are separated by a larger distance.

5.2. LSTM. The kernel process is $h_t = \sigma(W_o[h_{t-1}, x_t] + b_o) * \tanh(C_t)$. The current state h_t is affected by the current input x_t and previous state h_{t-1} passed down. Thus, the model is more suitable for time series data with sequential dependency. However, the timing relationship is not obvious in intrusion data sometimes.

5.3. DNN. As shown in Equation (11), the model is much suitable to capture the high-order interaction due to the full connected characteristic. However, the CNN, LSTM, and DNN models all lose the low-order feature interaction.

5.4. PR. $\hat{y}_{PR} = (b_0 + b_1x_1 + b_2x_2 + \cdots + b_mx_m)^d$. Thus, lots of new features are generated when obtaining the feature interactions. However, the PR may introduce some noise and cause dimension disaster (i.e., the dimension is C^m_{m+d}) when the original feature dimension m or the value d is large. Thus, d is usually set as 1 or 2. And we show the result of low-order feature interaction by PR when $d = 2$ in Table 3.

5.5. FM [26]. $\hat{y}_{FM} = \langle w, x\rangle + \sum_{i=1}^{m}\sum_{j=i+1}^{m}\hat{w}_{ij}x_ix_j$. The formal reflects the linear regression, and the latter denotes the pair-wise feature interactions. Compared with our LCHI model, the FM can be hindered by its modeling of all feature interactions with the same weight, as not all feature interactions are equally useful.

5.6. MCA [7] and MCA-LSTM [9]. As shown in Equation (4), the MCA can capture the low-order correlation, and the MCA-LSTM can capture both low- and high-order features. However, they all ignore the background knowledge and capture the correlations between distinct features, introducing too much noise. In addition, the MCA-LSTM conducts the feature selection, losing too many valuable correlations.

When extracting the low-order feature correlation, the MCA is more balanced compared with the PR and FM. While the PR generates features with higher dimensions and introduces more noise, the FM is more suitable for features with high sparsity. When modeling the high-order feature interaction, the CNN and LSTM models cause more loss, and the fully connected characteristic of the DNN model is more conducive to achieving this goal. Therefore, we adopt the MCA and DNN to extract low- and high-order feature interactions, respectively. The detection results in Table 3 show the advantages of the MCA and DNN models in low- and high-order feature interactions, respectively. The experimental environment is the same as Section 6.

Table 2: Comparison of the low- or high-order interaction models.

Model	Low-order interaction	High-order interaction	Background knowledge	Few noise or loss
CNN	×	√	×	×
LSTM	×	√	×	×
DNN	×	√	×	√
PR	√	√	×	×
FM	√	×	×	×
MCA	√	×	×	×
MCA-LSTM	√	√	×	×
LCHI	√	√	√	√

Table 3: Average detection accuracy (%) of different low- and high-order feature interaction models in the CICIDS 2018 and AWID datasets.

Dataset	Low-order feature interaction			High-order feature interaction		
	PR	FM	MCA	CNN	LSTM	DNN
AWID	95.10	95.48	96.10	96.51	97.13	98.54
CICIDS 2018	96.23	96.38	97.38	97.29	95.14	98.44

5.7. Summarizations. The LCHI not only captures both the low- and high-order feature interactions simultaneously but also selectively generates useful low-order correlation. It makes full use of background knowledge to avoid too much noise or information loss. Moreover, the LCHI also evaluates the importance of different feature correlations by attention.

6. Model Evaluation

This section describes the model evaluation. We evaluate the effectiveness of our LCHI model from the following perspectives:

(1) Detection performance of the LCHI model: we first compare the overall detection performance of the LCHI model with existing models and then we study the hyper-parameters of the LCHI model

(2) Effectiveness of the FD-MCA method: we compare the detection performance of our FD-MCA method with the existing MCA methods (i.e., MCA with the full features and selected features)

(3) Superiority of the attention mechanism: we compare the detection performance of our attention mechanism for the different low-order correlations with the directly spliced method for the correlations

Table 4: The value distributions of the AWID, NSL-KDD, UNSW-NB15, CICIDS 2017, CICIDS 2018, and DAPT 2020 datasets.

Dataset	Attack type	Training set	Testing set
AWID	Normal	326311	104657
	Injection	13220	4294
	Impersonation	9853	3083
	Flooding	9731	3081
NSL-KDD	Normal	67343	9711
	DoS	45927	7458
	Probe	11656	2421
	R2L	995	2750
	U2R	52	200
UNSW-NB15	Normal	56000	37000
	Generic	40000	18871
	Exploit	33393	11132
	Fuzzers	18184	6062
	DoS	12264	4089
	Reconnaissance	10491	3496
	Analysis	2000	677
	Backdoor	1746	583
	Shellcode	1133	378
	Worm	130	44
CICIDS 2017	Benign	65093	32625
	DDoS	85404	42623
CICIDS 2018	Benign	445212	222414
	DoS-SlowHTTPTest	128779	64581
DAPT 2020	Benign	6003	2852
	Establish foothold	5621	2967
	Reconnaissance	34	10

6.1. Dataset Description and Metrics. This section aims to demonstrate the datasets and performance metrics. To illustrate the robustness and applicability of our LCHI model, we use the public wireless network intrusion dataset of Aegean WiFi intrusion dataset (AWID) [32] and the wired network intrusion datasets of NSL-KDD [33], UNSW-NB15 [34], CICIDS 2017 [35], CICIDS 2018 [36], and DAPT 2020 for APT [37] to evaluate the model.

The AWID dataset contains one type of normal data and three instances of typical intrusion data. Each sample contains a 154-dimension feature and a class label. Compared with the KDD Cup 99, NSL-KDD, and UNSW-NB15 datasets, which are all general-purpose datasets commonly used for IDS research, the AWID is solely generated from wireless network traffic. Unlike intercepting data from the traditional TCP/IP communication protocol, traces in the AWID were not artificially generated. They were naturally produced from a wireless local area network and were more in line with the actual situation. As the total number of AWID dataset is too large, in this study, we use 20% of the reduced AWID dataset (i.e., AWIDCLS-R-Trn and AWID-CLS-Tst), and Table 4 depicts the value distribution of the selected AWID dataset.

The NSL-KDD dataset contains one type of normal network flow and four types of intrusion data. Each sample contains a 41-dimension feature and a class label. The UNSW-NB15 dataset contains one type of normal network flow and nine types of intrusion data. Each sample contains a 42-dimension feature and a class label. In the CICIDS 2017 dataset, a part of the dataset "Friday-WorkingHours-Afternoon" is used to validate our model's effectiveness. And the selected dataset contains 97718 records of benign traffic and 128027 records of DDoS network traffic, and each sample consists of a 78-dimension attribute and a class label. In the CICIDS 2018 dataset, a part of the dataset "data cicids2018 reconnaissance" is used for our model. And the selected dataset contains 667626 records of benign traffic and 193360 records of DoS-SlowHTTPTest traffic, and each sample has a 79-dimension attribute and a class label. In the DAPT 2020 dataset, we use the "data_custom_wednesday" dataset to test our model. And the selected dataset contains 8855 records of benign traffic, 8588 records of establish foothold traffic, and 44 records of reconnaissance traffic, and each sample has an 83-dimension attribute and a class label. Unlike the previously addressed datasets, these three datasets do not divide the data into training and testing groups. Thus, we randomly select 66% of the dataset for training and use the remaining 34% for testing. The value distributions of these datasets are shown in Table 4.

In this study, we use the following four indicators to evaluate detection performance: accuracy (ACC), F1-measure (F1), false-negative rate (FNR), and false-positive rate (FPR). The calculation formulas are as follows:

$$\begin{cases} \mathrm{ACC} = \dfrac{\mathrm{Num}'_{\mathrm{Normal}} + \sum_{\mathrm{intrusion}\in\mathrm{Intrusion}} \mathrm{Num}'_{\mathrm{intrusion}}}{\mathrm{Num}_{\mathrm{total}}}, \\ \mathrm{F1} = \dfrac{2 \times (\mathrm{TP}/\mathrm{TP} + \mathrm{FP}) \times (\mathrm{TP}/\mathrm{TP} + \mathrm{FN})}{(\mathrm{TP}/\mathrm{TP} + \mathrm{FP}) + (\mathrm{TP}/\mathrm{TP} + \mathrm{FN})}, \\ \mathrm{FNR} = \dfrac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}}, \\ \mathrm{FPR} = \dfrac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}}, \end{cases} \tag{14}$$

where Num'_i denotes the number of i-type intrusion successfully detected. $\mathrm{Num}_{\mathrm{total}}$ denotes the number of all the samples. TP, TN, FP, and FN denote the true positive, true negative, false positive, and false negative, respectively. The higher the ACC and F1 and the lower the FNR and FPR, the better the detection performance.

In this study, we use the "GTX 1080Ti four-card E5-2650, v2 32 cores, memory 64G, 440GB SSD, 4TB hard drive" to deploy our LCHI model and conduct the experiments in Python 3.7 and TensorFlow 1.14.

6.2. Comparison of Detection Performance. In this section, we first compare the overall detection performance of our LCHI model with previous intrusion detection models. And then we study the hyper-parameters of our LCHI model.

Table 5: Average detection performance (%) of different methods for multiclassification on the AWID, NSL-KDD, UNSW-NB15, CICIDS 2017, CICIDS 2018, and DAPT 2020 datasets.

Dataset	Model	ACC	F1	FNR	FPR
AWID	SVM	96.33	97.70	1.92	0.27
	BN	94.88	87.25	1.96	2.67
	RF	96.13	72.71	40.02	0.00
	CNN	96.51	78.79	28.69	0.97
	LSTM	97.13	83.74	27.90	0.001
	DNN	98.54	91.61	15.47	0.002
	MCA	96.10	80.56	24.75	0.36
	LCHI	99.60↑	99.68↑	1.02↓	0.005
NSL-KDD	SVM	56.90	66.95	39.29	27.79
	BN	61.80	78.13	34.73	2.39
	RF	76.05	75.69	37.68	3.09
	CNN	77.42	84.57	22.90	6.90
	LSTM	77.72	83.70	23.30	8.70
	DNN	81.51	85.35	21.50	7.20
	MCA	77.76	82.38	27.34	4.93
	LCHI	83.97↑	89.40↑	15.40↓	6.16
UNSW-NB15	SVM	69.70	80.90	17.10	27.00
	BN	57.06	76.59	26.12	23.34
	RF	72.29	82.12	16.25	24.78
	CNN	75.71	83.04	16.70	21.10
	LSTM	76.20	83.54	17.20	18.90
	DNN	77.04	88.57	14.32	9.53
	MCA	76.14	87.68	17.05	7.66
	LCHI	80.78↑	92.97↑	8.00↓	7.25↓
CICIDS 2017	SVM	98.41	98.17	2.05	1.23
	BN	92.22	90.33	16.25	1.29
	RF	99.16	99.03	1.92	0.005
	CNN	97.78	97.37	4.92	0.16
	LSTM	98.07	97.73	4.40	0.04
	DNN	99.53	99.46	0.04	0.80
	MCA	98.52	98.70	0.43	2.85
	LCHI	99.78↑	99.81↑	0.28	0.13
CICIDS 2018	SVM	96.60	91.86	14.78	0.09
	BN	96.15	92.13	0.00	4.96
	RF	97.10	93.15	12.82	0.00
	CNN	97.29	93.59	12.04	0.00
	LSTM	95.14	87.90	21.59	0.00
	DNN	98.44	96.40	6.95	0.00
	MCA	97.38	93.55	11.97	0.05
	LCHI	99.61↑	99.14↑	1.55	0.05
DAPT 2020	SVM	97.88	97.85	1.93	2.28
	BN	88.42	87.44	22.16	0.20
	RF	98.10	98.09	0.14	3.59
	CNN	96.48	96.34	5.33	1.78
	LSTM	96.74	97.16	3.68	1.88
	DNN	98.47	98.46	0.39	2.62
	MCA	96.29	97.12	2.01	3.79
	LCHI	99.11↑	99.12↑	1.17	0.60

6.2.1. Overall Detection Performance. This section describes the overall performance of our LCHI model with existing models. In the AWID dataset, since the feature dimension is too large, we obtained 72 features after removing some all-zero features. And then we selected 10 protocol-type features as the basic features, and the remaining are considered statistical features. In the NSL-KDD dataset, the features were divided into three types: the basic features of TCP connection (1st–9th dimensions), the content features of TCP connection (10th–22nd dimensions), and the statistical features of network flow (23rd–41st dimensions). The UNSW-NB15 dataset was also divided into three types: the 1st–14th, 15th–22nd, and the remaining. In the CICIDS 2017 dataset, we selected 12 protocol-type features as the basic features and the remaining as the statistical features. In the CICIDS 2018 dataset, we obtained 68 features after removing some all-zero features as the feature dimension is too large. We selected 10 protocol-type features as the basic features and the remaining as the statistical features. In the DAPT 2020 dataset, we selected 11 protocol-type features as the basic features, and the remaining are considered statistical features.

In our LCHI model, the DNN consisted of three hidden layers, with 64 neurons per layer. ReLU activation was adopted for our model. We set dropout = 0.2 and $d = t = 64$ as the unified dimension size and the attention sizes. Table 5 shows the average detection results of the LCHI model and previous models (i.e., SVM, BN, RF [38], CNN, LSTM, DNN, and MCA [7]).

We can see that in these six datasets, our LCHI model has higher ACC and F1 than the other models. For the FNR and FPR, our model maintains a much lower value. The details are as follows.

In the AWID dataset, our model gets the higher ACC with a 99.60% rate, which is close to 100%. And the second largest ACC is 98.54%. Thus, our model improves the ACC by 1.06%. Meanwhile, the FNR of our model is all lower than the other models. Although the FPR of our model is not the lowest, the FPR is only 0.005%.

In the NSL-KDD dataset, the ACC of our LCHI model is 83.97%, while the second highest is 81.51% in the DNN model. We improve the ACC by 2.46%. For the FNR and FPR, our model has the lower FNR. Although the FPR of our model is higher than that of BN and FR, it maintains a much lower value.

In the UNSW-NB15 dataset, the FNR and FPR of our model are lower than others. For the ACC, our model achieves the best outcome with an 80.78% rate, followed by DNN and LSTM, with 77.04% and 76.20%, respectively. We improve the ACC by 3.74%.

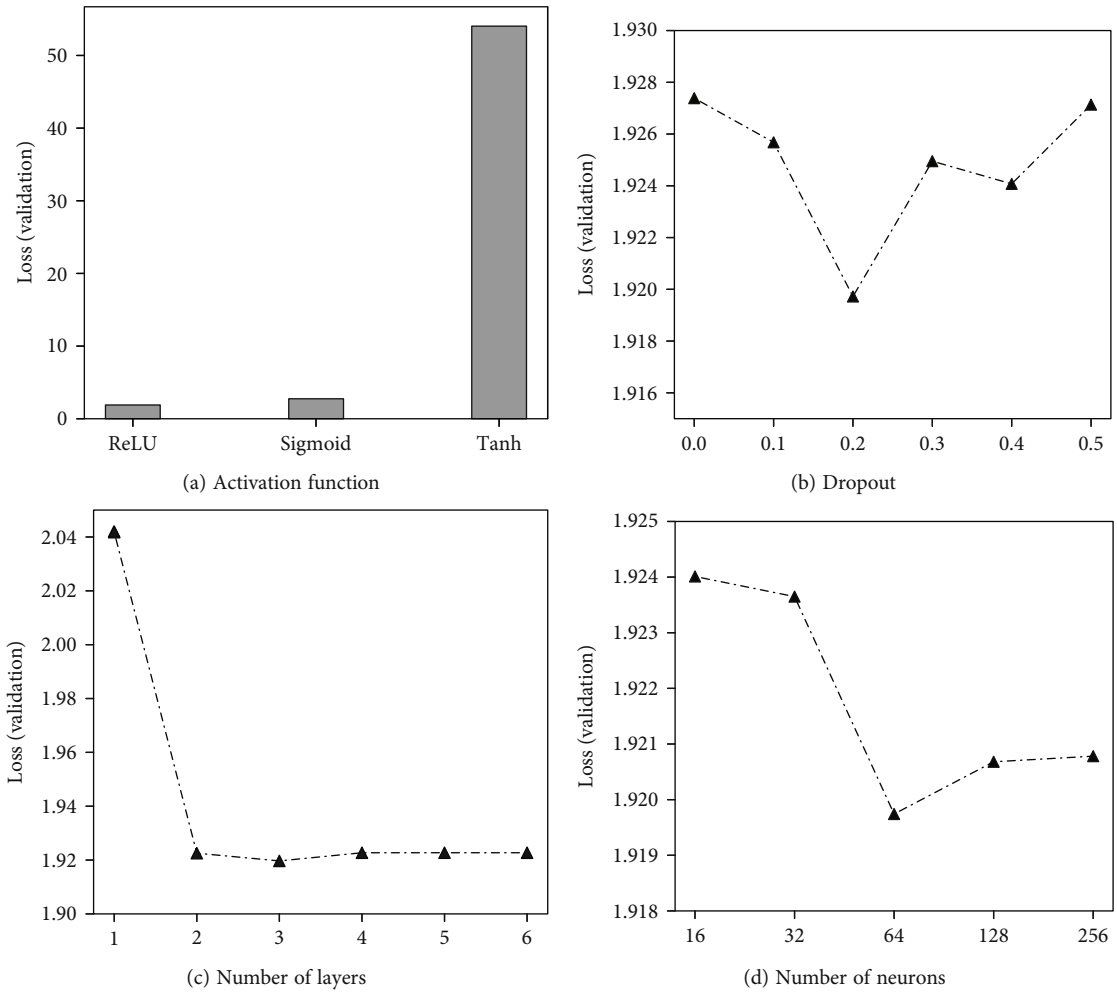


Figure 3: Losses of different parameters on the UNSW-NB15 dataset.

In the CICIDS 2017 dataset, our model achieves the best outcome for the ACC metric with a 99.78% rate, followed by DNN and RF, with 99.53% and 99.16%, respectively. The FNR and FPR in our model are not the lowest. However, the values are only 0.28% and 0.13%, respectively. Under the premise of enhancing ACC and F1, our model maintains a minimal value on the FNR and FPR.

In the CICIDS 2018 dataset, our model achieves the best ACC with a 99.61% rate, followed by DNN and CNN, with 98.44% and 97.29%, respectively. The ACC of our model is close to 100%. The FNR and FPR of our model are not the lowest. However, the values are only 1.55% and 0.05%, respectively. Under the premise of enhancing ACC and F1, our model maintains minimal FNP and FPR.

In the DAPT 2020 dataset, we can see that the ACC and F1 in our LCHI model are higher than the other models, with 99.11% and 99.12%. Although the FNR and FPR are not always lower than the other models, their values are maintained very low, only 1.17% and 0.60%, respectively.

Our model not only extracts the multivariate correlations between the same feature types but also integrates the low-order interaction and high-order interaction features. While extracting the effective correlation features, we try our best to avoid feature redundancy or too much information loss. The experimental results verify the effectiveness of our LCHI model.

6.2.2. Impact of Hyperparameters and Statistical Analysis. We then study the impact of different hyperparameters of our LCHI model on the UNSW-NB15 dataset. The order is (1) activation functions, (2) dropout rate, (3) number of hidden layers, and (4) number of neurons per layer. The losses of different parameters are shown in Figure 3.

The ReLU, Sigmoid, and tanh are the most common activation functions for deep learning models. In this paper, we compare the performance of deep learning model when applying these three different activation functions. As shown in Figure 3(a), ReLU is more appropriate than Sigmoid and tanh in the LCHI model when the other parameters are the same.

Dropout refers to the probability that a neuron is kept in the network. It is a regularization technique to compromise the precision and the complexity of the neural network. We set the dropout to be 0, 0.1, 0.2, 0.3, 0.4, and 0.5. As shown in Figure 3(b), the model reaches the best performance when the dropout is set to 0.2.

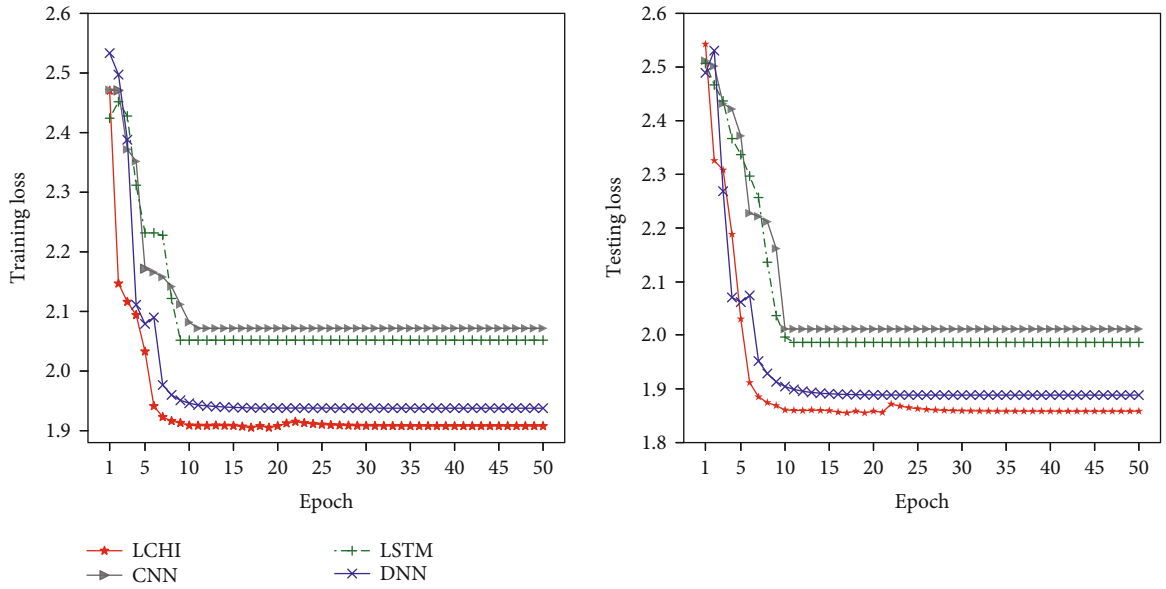


Figure 4: Training and testing losses of different models on the UNSW-NB15 dataset.

As presented in Figure 3(c), an increasing number of hidden layers improve the performance of the models at the beginning. However, their performance degrades if the number of hidden layers keeps increasing. This phenomenon is because of overfitting. The loss is lowest when the number of hidden layers is set to 3.

When other factors remain the same, increasing the number of neurons per layer introduces complexity. We can observe from Figure 3(d) that increasing the number of neurons does not always benefit. For instance, the loss is the lowest when the number of neurons is 64. The model performs worse when we increase the number of neurons from 64 to 256 because an overcomplicated model is easy to overfit. In our tests, 64 neurons per layer is a good choice.

The above experiments show that our LCHI model gets lower loss when the activation function is ReLU, and the dropout is 0.2. The number of neurons per layer is set to 64, and the number of layers is set to 3. These parameters can effectively extract features, and the model is not too complicated, thus avoiding overfitting or long training time. Finally, we display the training loss and testing loss of different epochs for different models, as shown in Figure 4. We can see that our LCHI model gets lower loss than the other models both on the training and testing stages, indicating that LCHI can better fit the data and lead to more accurate detection.

Table 6 shows the training time per epoch of different models on the UNSW-NB15 and AWID datasets. In contrast to the LSTN, DNN, and MCA, the training time of our LCHI model is longer. Although the training time of these models is lower than the LCHI model, they all get lower detection accuracy (see Table 5). Thus, under the condition of not long training time, our LCHI model has higher detection accuracy and practical application value.

6.3. Effectiveness of the FD-MCA Method. To evaluate the effectiveness of the FD-MCA method, we conducted a series of experiments on different correlation extraction methods. We extracted the multivariate correlations between every two different features from the full features (i.e., full MCA), from the selected features (i.e., selected MCA), and the features in the same type (i.e., FD-MCA). In these tests, we employed the feature selection algorithm of literature [17] and selected 20, 20, 22, and 28 features for the AWID, NSL-KDD, UNSW-NB15, and CICIDS 2018 datasets, respectively.

Table 6: Training time per epoch of different detection methods for multiclassification on the UNSW-NB15 and AWID datasets.

Dataset	Model	Training time (second)
UNSW-NB15	CNN	62.12
	LSTM	4.52
	DNN	2.95
	MCA	2.24
	LCHI	5.86
AWID	CNN	202.56
	LSTM	4.93
	DNN	4.22
	MCA	2.49
	LCHI	6.62

The detection results are shown in Figure 5. In these four datasets, we can see that the ACC and F1 in our FD-MCA method are both higher than those in the full MCA and selected MCA methods. Comparing the feature dimension of our FD-MCA method, the dimension of full MCA is much higher than the others, and the dimension of selected MCA is much lower than the others. For the full MCA, although it generated the most correlations, the ACC and F1 are lower than others. The noise may be more than the useful correlations in these generated correlations, especially

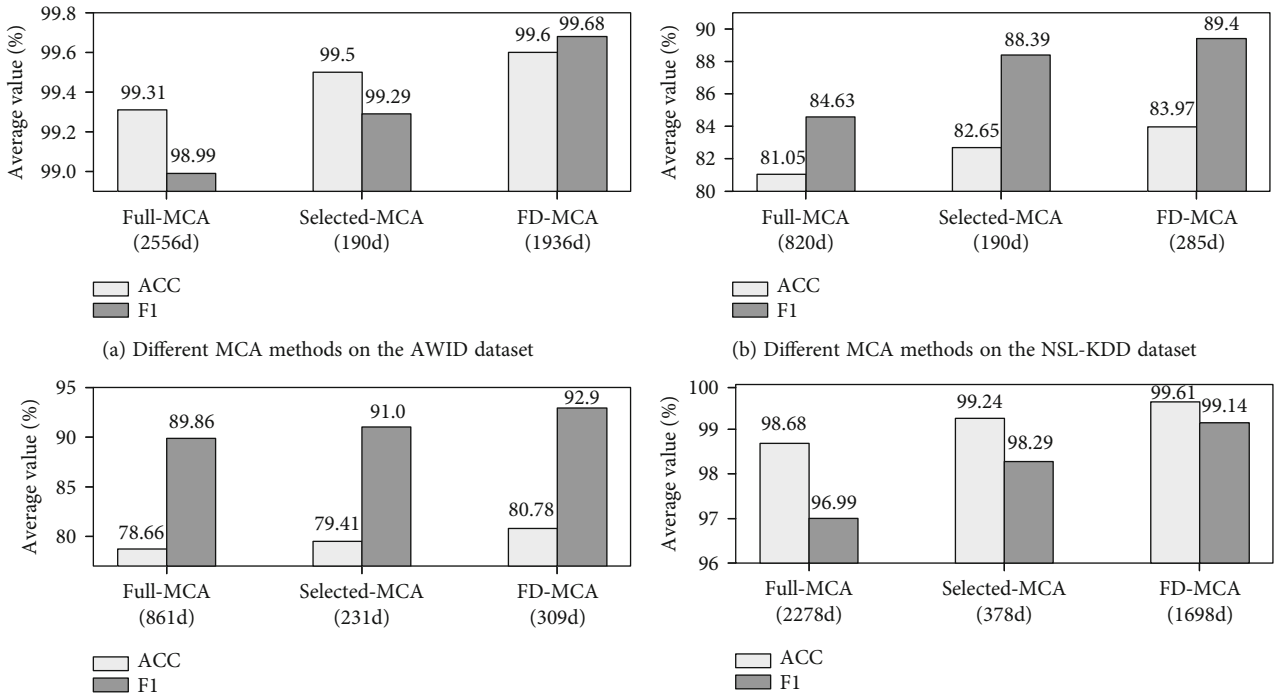


Figure 5: Detection results of different MCA methods. The number after the model bracket denotes the feature dimension (e.g., full MCA (2278d) means that there are 2278 dimension features generated by the full MCA method).

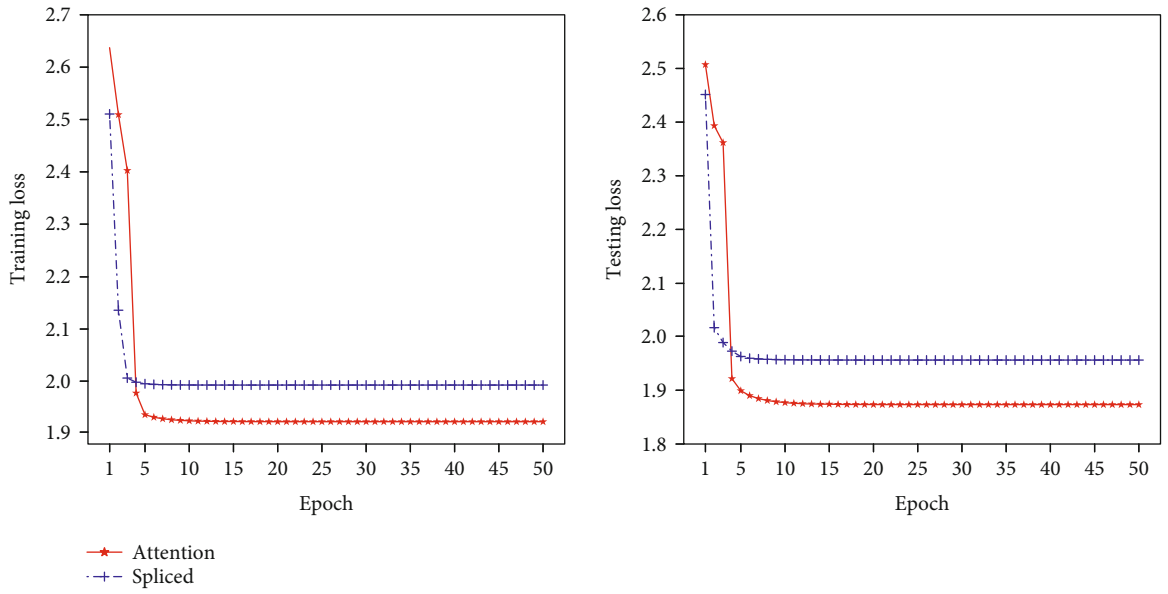


Figure 6: Training and testing losses of each epoch on the UNSW-NB15 dataset.

the correlations generated between different feature types. For the selected MCA, the selected features are much less than the original features. Thus, some useful features had been removed, resulting in much loss of useful correlations. Our FD-MCA method avoided the defects of the above two methods and only generated the correlations between the same feature type without removing any original feature. Under the guidance of background knowledge, our FD-MCA method can avoid excessive noise. The experimental results show that our method is superior to the others.

6.4. Superiority of the Attention Mechanism. This section compares the performance of different processing methods for the generated low-order correlations. Our model uses

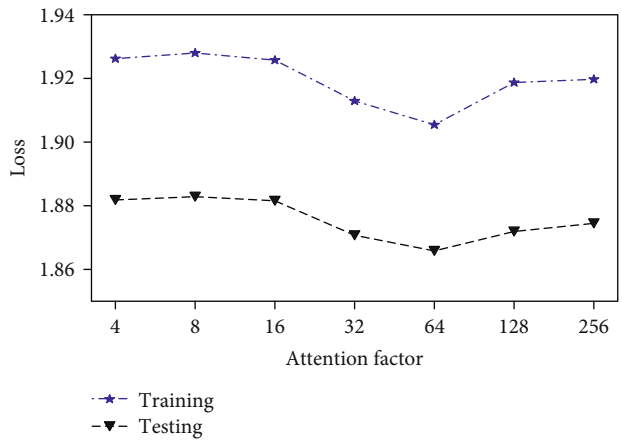


Figure 7: Training and testing losses of different attention factors on the UNSW-NB15 dataset.

Table 7: Average detection results (%) for different low-order correlations processing methods for multiclassification on the AWID, NSL-KDD, UNSW-NB15, and CICIDS 2018 datasets.

Dataset	Method	ACC	F1
AWID	Spliced	98.62	91.03
	Attention	98.88↑	91.35↑
NSL-KDD	Spliced	78.87	84.69
	Attention	79.16↑	85.75↑
UNSW-NB15	Spliced	79.59	90.06
	Attention	80.33↑	91.27↑
CICIDS 2018	Spliced	97.86	94.01
	Attention	97.97↑	95.36↑

the attention mechanism to fuse these different correlations (i.e., attention). In addition, we can also directly splice these correlations (i.e., spliced). We compared these two methods to verify the superiority of our attention method. As shown in Figure 6, we observe that the attention's training and testing losses are much lower than the spliced form, indicating that the attention method can better fit the data and more accurately detect. Figure 7 shows the loss of attention w.r.t. different attention factors. The attention's performance is relatively stable across attention factors. Thus, the experimental results justify the rationality of attention's design that estimates the importance score of different correlations.

We then adopted the DNN model to train these two processing methods on the AWID, NSL-KDD, UNSW-NB15, and CICIDS 2018 datasets. The detection results are shown in Table 7. In these four datasets, we can see that the ACC and F1 in our attention method are higher than those in the spliced method. The ACC has proved by 0.26%, 0.29%, 0.74%, and 0.11% in these four datasets, respectively. Compared with the spliced method, the attention method takes into consideration the effect of different low-order correlations. These low-level correlations have different distinguishing capabilities for different attacks and help distinguish the subtle differences between different attacks. Splicing the correlations loses these tiny differences. The experimental results verify the superiority of our attention mechanism.

7. Conclusions

This paper proposes a novel LCHI model to learn the low-order feature correlation and high-order feature interaction simultaneously. Firstly, we consider the difference of features and divide them into different types. Since the feature correlation between the same types is much stronger than between different types, we selectively generate the low-order correlation between the same-type features to avoid too much noise. We do not conduct the feature selection to avoid too much correlation loss. Moreover, we consider the influence of different correlations, and thus we adopt the attention mechanism on them. Secondly, we employ the DNN model to extract the sophisticated high-order feature interaction. Finally, we incorporate the low-order feature correlation and high-order feature interaction. Above all, our LCHI model seamlessly combines the linearity of MCA in modeling lower-order feature correlation and the nonlinearity of DNN in modeling higher-order feature interaction. Conceptually, LCHI is more expressive than the previous models. The experimental results indicate that our model gets higher accuracy than the others. In the future, more effective integration of low- and high-order feature interactions in real IoT networks will be our next task.

Data Availability

The data used to support the findings of this study are included within the article. We have described the data and their references in detail in this article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 61902013, No. U1636208, No.61862008) and the Beihang Youth Top Talent Support Program (Grant No. YWF-21-BJ-J-1039).

References

[1] https://blogs.cisco.com/tag/2021networkingtrends.

[2] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: a systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, article e4150, 2021.

[3] M. Zhang, L. Wang, S. Jajodia, and A. Singhal, "Network attack surface: lifting the concept of attack surface to the network level for evaluating networks' resilience against zero-day attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 310–324, 2021.

[4] K. Huang, X. Liu, S. Fu, D. Guo, and M. Xu, "A lightweight privacy-preserving CNN feature extraction framework for mobile sensing," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1–1455, 2020.

[5] J. Zhang, Y. Ling, X. Fu, X. Yang, G. Xiong, and R. Zhang, "Model of the intrusion detection system based on the integration of spatial- temporal features," *Computers & Security*, vol. 89, article 101681, 2020.

[6] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Towards a deep learning-driven intrusion detection approach for internet of things," *Computer Networks*, vol. 186, article 107784, 2021.

[7] Zhiyuan Tan, A. Jamdagni, Xiangjian He, P. Nanda, and Ren Ping Liu, "A system for denial-of-service attack detection based on multivariate correlation analysis," *IEEE Transactions on Parallel & Distributed Systems*, vol. 25, no. 2, pp. 447–456, 2014.

[8] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "Denial-of- service attack detection based on multivariate correlation analysis," in *Neural Information Processing*, pp. 756–765, Springer, Berlin, Heidelberg, 2011.

[9] R. H. Dong, X. Y. Li, Q. Y. Zhang, and H. Yuan, "Network intrusion detection model based on multivariate correlation analysis – long short-time memory network," *IET Information Security*, vol. 14, no. 2, pp. 166–174, 2020.

[10] X. Chen, C. Li, D. Wang et al., "Android HIV: a study of repackaging malware for evading machine-learning detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, no. 1, pp. 987–1001, 2020.

[11] G. Lin, S. Wen, Q.-L. Han, J. Zhang, and Y. Xiang, "Software vulnerability detection using deep neural networks: a survey," *Proceedings of the IEEE*, vol. 108, no. 10, pp. 1825–1848, 2020.

[12] T. Wang, C. Xia, X. Li, and Y. Xiang, "Epidemic heterogeneity and hierarchy: a study of wireless hybrid worm propagation," *IEEE Transactions on Mobile Computing*, pp. 1–18, 2020.

[13] J. Zhang, L. Pan, Q. L. Han, C. Chen, S. Wen, and Y. Xiang, *Deep learning based attack detection for cyber-physical system security: a survey*, IEEE/CAA Journal of Automatica Sinica, 2021.

[14] T. Wang, C. Xia, S. Wen, H. Xue, Y. Xiang, and S. Tu, "SADI: a novel model to study the propagation of social worms in hierarchical networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 1, pp. 142–155, 2019.

[15] T. Wang, C. Xia, Z. Li, X. Liu, and Y. Xiang, "The spatial-temporal perspective: the study of the propagation of modern social Worms," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2558–2573, 2017.

[16] Z. Wang, Y. Liu, D. He, and S. Chan, "Intrusion detection methods based on integrated deep learning model," *Computers & Security*, vol. 103, article 102177, 2021.

[17] S. Kasongo and Y. Sun, "A deep learning method with wrapper based feature extraction for wireless intrusion detection system," *Computers & Security*, vol. 92, article 101752, 2020.

[18] Z. Khan, M. Chowdhury, M. Islam, C.-Y. Huang, and M. Rahman, "Long short-term memory neural networks for false information attack detection in software-defined in-vehicle network," 2019, http://arxiv.org/abs/1906.10203.

[19] S. M. Sohi, J. P. Seifert, and F. Ganji, "RNNIDS: enhancing network intrusion detection systems through deep learning," *Computers & Security*, vol. 102, article 102151, 2021.

[20] A. J. Siddiqui and A. Boukerche, "TempoCode-IoT: temporal codebook-based encoding of flow features for intrusion detection in Internet of Things," *Cluster Computing*, vol. 24, no. 1, pp. 17–35, 2021.

[21] J. Balasundaram and P. M, "A novel optimized bat extreme learning intrusion detection system for smart internet of things networks," *International Journal of Communication Systems*, vol. 34, no. 7, article e4729, 2021.

[22] S. S and P. W. D, "An enhanced optimization based algorithm for intrusion detection in SCADA network," *Computers & Security*, vol. 70, pp. 16–26, 2017.

[23] J. Gu and S. Lu, "An effective intrusion detection approach using SVM with naive Bayes feature embedding," *Computers & Security*, vol. 103, no. 3, article 102158, 2021.

[24] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, and F. Tang, "Discriminating DDoS attacks from flash crowds using flow correlation coefficient," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1073–1080, 2012.

[25] A. Jamdagni, Z. Tan, X. He, P. Nanda, and R. P. Liu, "RePIDS: a multi tier real-time payload-based intrusion detection system," *Computer Networks*, vol. 57, no. 3, pp. 811–824, 2013.

[26] S. Lei, C. Xia, Z. Li, X. Li, and T. Wang, "HNN: a novel model to study the intrusion detection based on multi-feature correlation and temporal-spatial analysis," *IEEE Transactions on Network Science and Engineering*, pp. 1–18, 2021.

[27] S. Rendle, "Factorization machines," in *2010 IEEE International Conference on Data Mining*, pp. 995–1000, Sydney, NSW, Australia, 2010.

[28] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, "Attentional factorization machines: learning the weight of feature interactions via attention networks," 2017, http://arxiv.org/abs/1708.04617.

[29] E. Arul, "Deep nonlinear regression least squares polynomial fit to detect malicious attack on IoT devices," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 769–779, 2021.

[30] M. Zhao and Z. Ma, "From polynomial fitting to kernel ridge regression: a generalized difference filter for encoder signal analysis," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 9, pp. 6212–6220, 2020.

[31] H.-T. Cheng, L. Koc, J. Harmsen et al., "Wide & deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pp. 7–10, Boston MA USA, 2016.

[32] C. Kolias, V. Kolias, and G. Kambourakis, "TermID: a distributed swarm intelligence-based approach for wireless intrusion detection," *International Journal of Information Security*, vol. 16, no. 4, pp. 401–416, 2017.

[33] 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 102419, 2020.

[34] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, Canberra, ACT, Australia, 2015.

[35] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a reliable intrusion detection benchmark dataset," *Software Networking*, vol. 2017, no. 1, pp. 177–200, 2017.

[36] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP*, pp. 108–116, Funchal, Madeira, Portugal, January 2018.

[37] S. Myneni, A. Chowdhary, A. Sabur et al., "DAPT 2020 - constructing a benchmark dataset for advanced persistent threats," in *Deployable Machine Learning for Security Defense*, pp. 138–163, Springer, Cham, 2020.

[38] P.-F. Marteau, "Random partitioning forest for point-wise and collective anomaly detection - application to network intrusion detection," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2157–2172, 2021.