Hindawi
Security and Communication Networks
Volume 2022, Article ID 4748528, 21 pages
https://doi.org/10.1155/2022/4748528

Research Article

An Enhanced Intrusion Detection System for IoT Networks Based on Deep Learning and Knowledge Graph

Xiuzhang Yang,[1,2] Guojun Peng,[1,2] Dongni Zhang,[1,2] and Yangqi Lv[1,2]

[1]*Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, Wuhan 430072, China*
[2]*School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China*

Correspondence should be addressed to Guojun Peng; guojpeng@whu.edu.cn

Received 24 December 2021; Revised 17 February 2022; Accepted 18 March 2022; Published 26 April 2022

Academic Editor: Weizhi Meng

Nowadays, the intrusion detection system (IDS) plays a crucial role in the Internet of Things (IoT) networks, which could effectively protect sensitive data from various attacks. However, the existing works have not considered multiview features fusion and failed to capture the semantic relationships among the anomalous requests. They are not robust and cannot detect the attack types in real-time. This paper proposes a lightweight intrusion detection system based on deep learning and knowledge graph. First, our system extracts semantic relationships and key features by knowledge graph and statistical analysis. Then, IoT network requests are converted into word vectors through multiview feature fusion and feature alignment. Finally, an attention-based CNN-BiLSTM model is designed to identify malicious request attacks, which can capture long-distance dependence and contextual semantic information. Experiment results show that the proposed model significantly outperforms the existing solution in the robustness of the model. Moreover, it can select more critical features for IDS to achieve better accuracy and lower the false alarm rate. Compared with the state-of-the-art systems, the proposed IDS achieves a higher detection accuracy of 90.01%. In addition, our system can detect various stealthy attack types (including DoS, Probe, R2L, and U2L) and extract semantic relationships among features.

1. Introduction

With various terminal devices and applications becoming interoperable through networks, the Internet of Things (IoT) systems improve the quality of life and enhance real-life intelligent devices [1]. The prominent devices in IoT networks include outdoor surveillance cameras, smart home devices, mobile user-worn devices, and industrial control services [2]. As a result, IoT devices are increasingly deployed in critical infrastructures. However, due to the intuitive design, IoT also faces many security threats and challenges, making the infrastructures suffer from serious attacks and undermining the integrity of sensitive data. Some incidents (e.g., Stuxnet [3], the widespread blackout in the Ukrainian power grid [4], SolarWinds supply chain attack [5], Colonial Pipeline suffering the ransomware attack [6]) have shown that network attacks on IoT devices can result in catastrophic consequences to the society, enterprises, and human life [7, 8]. Thus, effectively detecting security threats and protecting the IoT environment is crucial.

As an active defense technology, intrusion detection system (IDS) has gradually become a key technology to ensure network security. IDS can identify abnormal requests in the communication network, detect potential network threats, and generate alarms, thereby protecting the Internet and infrastructures in runtime. In this paper, we focus on constructing IDS for IoT networks. Meanwhile, machine learning (ML) has been widely used in security research recently, such as APT attack detection [9], personal privacy protection [10], malicious code analysis [11], cyberphysical system defense [8], and malicious traffic detection [12]. In order to detect malicious behaviors in large-scale network traffic of IoT, machine learning-based IDS has attracted

widespread attention. The goal of ML-based IDS is to learn a decision boundary that discriminates malicious network traffic from normal network traffic. In this process, it learns to distinguish different behaviors from the network traffic and host audit records. Traditional ML-based intrusion detection algorithms include support vector machine (SVM) [13], K-Nearest Neighbors (KNN) [14], Decision Tree (DT) [15], Random Forest (RF) [16], AdaBoost [17], K-Means [18], and Artificial Neural Networks (ANNs) [19].

However, these systems usually require a large amount of expert knowledge and labeled data to construct rules or models to detect abnormal requests [8]. Hence, they have poor applicability in large-scale intrusion detection systems. Also, traditional ML-based systems have not considered multiview features fusion and fail to capture the semantic relationships among the anomalous requests. The existing IDSs focus on constructing machine learning models and lack detailed feature extraction analysis [20], resulting in a poor ability to capture crucial features of IoT network traffic and detect unknown attack types (e.g., the zero-day attack). In addition, many machine learning algorithms are shallow learning algorithms, which have low detection throughput and suffer from evasion attacks that can easily bypass IDSs. Thus, they cannot efficiently defend IoT environments against various network attacks, especially when facing large-scale data [21]. They cannot detect the attack types in real time. Therefore, improving the design of current IDSs to provide suitable intrusion detection for IoT networks is urgent. Also, some features of different types of attacks often appear in network traffic at the same time, and traditional IDS ignores the semantic knowledge between features. Therefore, effectively capturing semantic-semantic relationships will help improve the accuracy and robustness of our IDS.

To address the above problems, this paper proposes an enhanced intrusion detection system for IoT networks based on deep learning and knowledge graph. The main research problem in this paper is how to build a model to better learn the semantic relationship between network traffic features, so as to improve the robustness of IDS to traffic recognition in IoT networks. We mainly address this problem from the perspective of feature extraction and feature fusion. Therefore, we particularly devise two feature extraction methods: knowledge graph-based feature extraction and statistical analysis-based feature extraction. By this means, we can effectively extract the contextual semantic relationship of IoT network traffic and key features of different attack types, thereby increasing the similarity distance between benign and malicious samples. Furthermore, to solve feature ambiguity and the single perspective of traditional models, we design a multiview feature fusion and feature alignment algorithm to build more robust word vectors and construct a deep learning model to detect IoT requests. The contributions of our paper are summarized as follows:

(i) To our best knowledge, we are the first to design a multiview fusion model for intrusion detection of IoT networks. This model can extract the contextual semantic relationship through a knowledge graph, as well as crucial features through statistical analysis. Then, we make the IDS more robust and interpretable by feature fusion and alignment of the two feature extraction algorithms.

(ii) We present an attention-based Convolutional Neural Network and Bidirectional Long Short-term Memory (CNN-BiLSTM) model, which can capture the long-distance dependence and contextual semantic information. Moreover, our IDS relies on less prior knowledge and detects more attack types. The deep learning-based classifier can better detect attack types in runtime by adaptively updating the weights of different features, especially the features extracted by the knowledge graph.

(iii) We conduct a systematic comparison experiment with state-of-the-art systems. The results show that our system is effective and robust in detecting malicious requests of IoT and can accurately detect various stealthy attack types (including unknown attacks).

The rest of the paper is organized as follows. Section 2 summarizes the related works. Section 3 presents the proposed intrusion detection model for IoT networks in detail. Section 4 experimentally evaluates the performances of our model and Section 5 concludes the paper. Also, Table 1 summarizes all acronyms used in this paper for completeness and readability.

2. Related Work

In this section, we introduce the related work of intrusion detection systems for IoT networks, including rule-based systems, machine learning-based systems, and deep learning-based systems. Note that we introduce from the perspective of methodology, rather than discuss the following host-based IDS and network-based IDS.

2.1. Rule-based System. With the increasing complexity and stealthy of network attacks, intrusion detection system plays a significant role in detecting malicious requests. Traditional intrusion detection system mainly relies on expert knowledge and manual experience to extract traffic features by defining rules and feature databases. Then, such IDS analyzes network requests with statistical correlation algorithms to identify external attacks [22]. In 1987, Denning [23] first proposed a universal intrusion detection system and introduced it into computer security defense systems. Later, Lunt and Jaganna [24] designed an intrusion detection expert system based on domain knowledge. This system adaptively learned the normal behavior of each user through audit records and detected abnormal users in real time. After that, more rule-based intrusion detection systems appear, especially in the IoT ecosystem. Chimera [25] is a declarative query language for network traffic processing that combines the advantages of intrusion detection systems and SQL syntax with rules. The vNIDS [26] is an innovative architecture for network intrusion detection systems (NIDSs),

Table 1: List of the acronyms used in the manuscript.

Acronym	Definition	Acronym	Definition
ANN	Artificial Neural Network	ML	Machine Learning
BiLSTM	Bidirectional Long Short-Term Memory	M2-DAE	MultiModal Deep AutoEncoder
CNN	Convolutional Neural Network	NIDS	Network Intrusion Detection Systems
DL	Deep Learning	NIN	Network-in-Network
DoS	Denial of Service	RDP	Remote Desktop Protocol
DT	Decision Tree	ReLU	Rectified Linear Unit
FAR	False Alarm Rate	RF	Random Forest
FN	False Negative	RNN	Recurrent Neural Network
FP	False Positive	R2L	Remote to Local
IDS	Intrusion Detection System	SA	Statistical Analysis
IoT	Internet of Things	SVM	Support Vector Machine
KG	Knowledge Graph	TN	True Negative
KNN	K-Nearest Neighbors	TP	True Positive
LR	Logistic Regression	U2R	User-to-Root
LSTM	Long Short-Term Memory		

which achieves elastic security by configuring virtual NIDS as microservices and employing program slicing to partition the detection logic programs. Haugerud et al. [27] designed and implemented a lightweight elastic architecture NIDS, which constructed a flexible IDS in a docker container through network function virtualization and intelligent rule ordering.

However, rule-based systems rely heavily on various rules and expert experience, which will constantly update rules. Since massive network attacks with stealth and anti-detection, malicious requests often cannot be detected in time. In addition, the system ignores the semantic relationship between traffic features, making it easy for specific constructed and confused attack requests to bypass online firewalls or IDSs. Therefore, it cannot detect unknown attacks (e.g., zero-day attacks).

2.2. Machine Learning-Based System. With the rapid advancement in statistical machine learning, machine learning-based intrusion detection system is widely deployed to protect IoT networks from various attacks [28]. This system uses the established and trained machine learning model to predict the malicious behavior of unknown network requests. Common methods include SVM, random forest, logistic regression, and K-Means. Whisper [21] is a real-time ML-based malicious traffic detection system utilizing frequency domain features to achieve high accuracy and throughput. Also, Whisper has good robustness because attackers cannot easily interfere with frequency domain features. Bitton and Shabtai [29] propose an intrusion detection algorithm based on clustering. This algorithm can protect the cyberphysical system from Remote Desktop Protocols (RDP) vulnerability attacks. Li et al. [30] design a model based on sustainable ensemble learning, which considers the accumulation of historical knowledge to realize incremental learning and improve the robustness of intrusion detection systems.

Compared with rule-based intrusion detection systems, machine learning-based systems can improve the model's accuracy and robustness, which is no longer limited to specific rules. However, because machine learning belongs to shallow learning, it lacks deep semantic knowledge and context analysis. Therefore, traditional IDS based on machine learning has some limitations in detecting complex, confusing, and hidden massive malicious requests and cannot detect the attack the first time.

2.3. Deep Learning-Based System. With the application of deep neural networks in various security fields [31], intrusion detection systems based on deep learning appear gradually. Representative models are mainly convolutional neural networks and recurrent neural networks (including LSTM). On the one hand, they can mine the deep information of network requests and improve detection accuracy through continuous learning of the model. On the other hand, they can avoid the limitations caused by artificial experience. LIO-IDS [32] is an intrusion detection system based on the LSTM classification model and promotion one-to-one technology, which is used to deal with infrequent network intrusions. Imrana et al. [33] developed a centralized intrusion detection system based on BiLSTM to identify attack types with fewer samples. DL-IDS [34] mixed CNN and LSTM models for intrusion detection. Li et al. [35] designed an IoT feature extraction and intrusion detection system for smart cities based on the deep migration learning model. Balakrishnan et al. [36] leveraged a deep belief network to enhance intrusion detection systems and scrutinize malicious activity in the IoT network. Moreover, BDL-IDS [37] was a big data-aware deep learning system, which can reduce the number of false alarms in the NSL-KDD dataset. Kasongo and Sun [38] proposed a feed-forward deep neural network wireless IDS system using a wrapper-based feature extraction unit. Ferrag et al. [39] compared the cybersecurity intrusion detection systems and datasets based on deep learning in detail and conducted a comparative study.

Furthermore, to solve the problem that existing neural networks need supervised training and label network traffic with the help of expert knowledge, Mirsky et al. [40] proposed an unsupervised online network intrusion detection

system called Kitsune. This system can distinguish normal and abnormal traffic patterns by building an ensemble of autoencoders. Bovenzi et al. [41] proposed a two-stage hierarchical network intrusion detection system, namely H2ID. The system performed anomaly detection via a novel lightweight solution based on MultiModal Deep AutoEncoder (M2-DAE) and achieved attack classification by soft-output classifiers.

However, the above systems lack the deep semantic relationship among traffic features and fail to analyze the correlation and difference between attack types through feature extraction and feature fusion. In addition, the lack of knowledge mapping to capture semantic information makes it impossible to sense hidden, unknown attacks (with small samples) and spoofing malicious traffic. Table 2 compares the characteristics of the three intrusion detection models in detail. It can be seen that the system in this paper can effectively extract semantic features through a knowledge graph, has better robustness, and can detect unknown types of attacks in real time.

3. System Design

In this section, we present the architecture of our proposed intrusion detection system for IoT networks and introduce how to extract features by knowledge graph and statistical analysis. Then, we describe the design and implementation of this IDS in detail.

3.1. Overview. The overall architecture of our IDS is depicted in Figure 1. This IDS consists of five phases: data preprocessing, feature extraction, feature fusion and alignment, model construction, and intrusion detection.

The proposed intrusion detection system is designed to increase the difference between normal and abnormal requests. As shown in Figure 1, our detection framework is comprised of five main phases. (1) The data preprocessing component can improve the quality of the dataset by processing the IoT network requests, which includes data cleaning, numerical conversion, log processing, normalized processing, and one-hot encoding. (2) The feature extraction component consists of two parts. First, the semantic relationship of different attack types is extracted through semantic feature analysis, cooccurrence calculation, and knowledge graph construction. Then, the key features of normal and malicious requests are mined through statistical analysis and power-law distribution. (3) The feature fusion and alignment component judges whether the feature should be weighted, aiming to improve the robustness and interpretability of the IDS by multiview fusion. In addition, this step uses the word embedding module to convert traffic features into word vectors. (4) The model construction component presents an attention-based CNN-BiLSTM model to capture the long-distance and contextual semantic features. (5) The intrusion detection component constructs a classifier through the softmax function, thereby detecting malicious requests (i.e., binary classification) and identifying attack types (i.e., multiclass classification).

3.2. Data Preprocessing. To obtain a high-quality dataset and solve the constraint problem of nonnumerical features, this paper carries out detailed data preprocessing to ensure the IoT intrusion detection experiment. This part mainly includes five processing, namely, data cleaning, numerical conversion, log processing, normalized processing, and one-hot encoding. The following part explains each phase in detail.

Note that we apply the NSL-KDD dataset in our experiment, which is an improvement of the KDD Cup 99 dataset [42]. This dataset is the traffic request generated by MIT Lincoln Laboratory using IoT devices to simulate attacks in the real-time environment. Thus, it is a relatively authoritative dataset. The entire NSL-KDD dataset can be described in Figure 2. It includes 41 traffic attribute features and one label feature. Among them, the attribute feature contains four categories: intrinsic feature, content feature, time-based feature, and host-based feature, which can be formulated as (1), where x_i corresponds to the feature i. Moreover, the label feature is divided into normal network traffic and abnormal network traffic. The abnormal network traffic includes four types (i.e., DoS, Probe, R2L, and U2R), covering a total of 39 different attack types (e.g., Nmap, Smurf, Sqlattack, and Spy). The detailed feature description and data distribution will be introduced in the following experimental section.

$$X = (x_1, x_2, \ldots, x_{40}, x_{41}). \tag{1}$$

Data cleaning can standardize the dataset and remove redundant data. Typical operations include filling in gaps and deleting duplicate values. In this paper, data cleaning is performed on both the training set and the testing set. Since the deep learning model requires a vector of real numbers as input embedding, it is necessary to convert symbolic features into numerical features. Thus, we perform numeric conversion processing, encoding some strings from 0 to convert them to the corresponding numeric value. For example, the types of protocol (i.e., TCP, UDP, and ICMP) will be converted into corresponding numeric values (i.e., 0, 1, and 2). In the case of the NSL-KDD dataset, the nonnumeric features that require numerical transformation are protocol type, service type, flag, and category type.

To enhance the performance of the proposed IDS, we will execute log processing and normalized processing. The former can effectively reduce the difference among traffic features in the dataset. For example, the NSL-KDD dataset counts the size of requests sent from the source host to the target host, denoted as src_bytes, and the maximum value is 1,379,963,888. Obviously, this value is an outlier, which will seriously affect our experimental results. Therefore, we need to execute the log function to reduce dimensionality. By this means, it can reduce feature values to the same or similar granularity, and the processed result is 9.1399. In this paper, five features (i.e., duration, src_bytes, dst_bytes, num_compromised, and num_root) of the NSL-KDD dataset will be processed by the log function, whose calculation is shown in equation (2). The results are shown in Table 3.

Table 2: Comparing the existing intrusion detection systems.

Category	Related work	Techniques	Robust detection	Knowledge graph	Real-time detection	Unknown attack detection	Semantic feature extraction
Rule-based System	Chimera [25]	Rules SQL syntax	×	×	✓	×	×
	vNIDS [26]	virtual NIDS	×	×	✓	×	×
	Haugerud et al. [27]	rule ordering virtualization	×	×	✓	×	×
ML-based System	Whisper [21]	Clustering frequency domain	✓	×	✓	✓	×
	Bitton et al. [29]	Clustering T-SNE	✓	×	✓	×	×
	Li et al. [30]	ensemble learning	✓	×	✓	×	×
DL-based System	LIO-IDS [32]	LSTM	✓	×	✓	×	×
	Imrana et al. [33]	BiLSTM	✓	×	✓	×	×
	DL-IDS [34]	CNN-LSTM	✓	×	✓	✓	×
	Our IDS	KG + SA CNN BiLSTM	✓	✓	✓	✓	✓

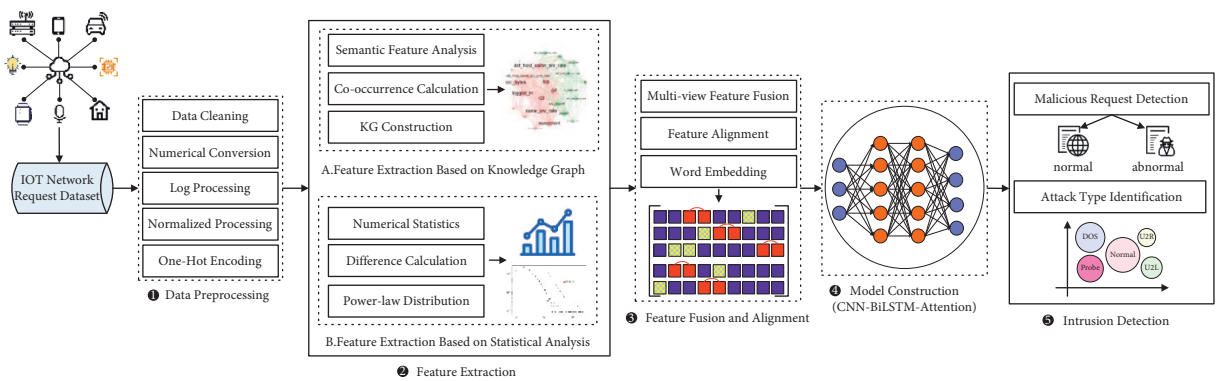


Figure 1: The architecture of our proposed intrusion detection system for IoT.

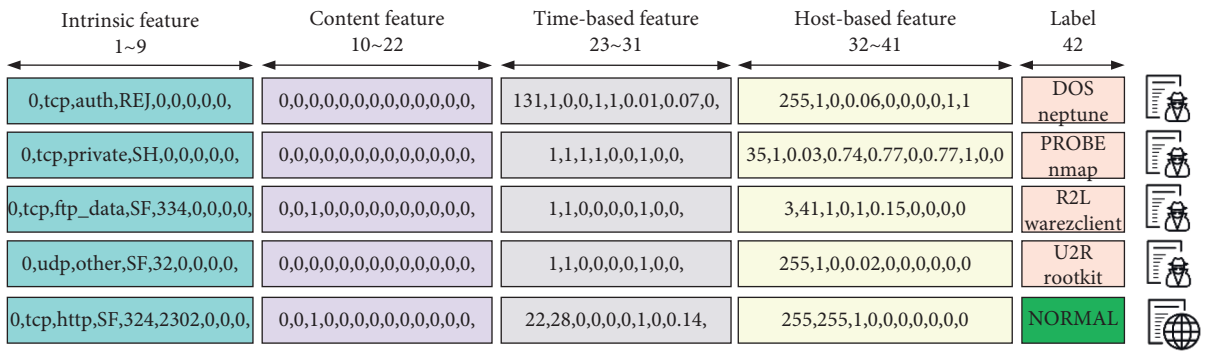


Figure 2: The description of the features in the NSL-KDD dataset.

Table 3: The result of log processing in the NSL-KDD dataset.

No.	Feature	Scope before processing	Scope after processing
1	duration	[0, 58329]	[0, 4.7659]
5	src_bytes	[0, 1379963888]	[0, 9.1399]
6	dst_bytes	[0, 1309937401]	[0, 9.1173]
13	num_compromised	[0, 7479]	[0, 3.8738]
14	num_root	[0, 7468]	[0, 3.8732]

$$x_i' = \log(x_i). \tag{2}$$

Moreover, data normalization converts the feature value into a suitable range, which can effectively eliminate the problem of data imbalance and preference for larger values. This paper mainly uses the Min-Max scaling to normalize the values of different features between 0 and 1, and the Min-Max scaling is described as

$$x_{ij}' = \frac{x_{ij} - x_i^{\min}}{x_i^{\max} - x_i^{\min}}, \tag{3}$$

where i is a feature in the dataset, j is a record of the dataset, and $x_i^{\max}$ and $x_i^{\min}$ are the maximum and minimum values of the feature i. Hence, we can map every continuous feature's values within the range of (0, 1) and effectively characterize the importance and distribution of the corresponding features. However, to preserve the authenticity and relationship of the original network requests, this paper does not call the Z-score function for standardized processing.

Finally, we perform the one-hot encoding to convert the category label of the NSL-KDD dataset to a unique category, which assigns the current category bit to 1 and other low bits to 0. For example, the DoS label in Figure 2 will be converted to the form [1, 0, 0, 0, 0]. By this, our deep learning model can receive better input vectors for training. In short, we show how the proposed IDS implements data preprocessing in five steps, and then, we will introduce the feature extraction part in detail (the second step in Figure 1).

3.3. Feature Extraction Based on Knowledge Graph. Without prior knowledge, it is difficult to obtain the correlation between features, especially facing unknown network attacks. Before training, it is important for our IDS model to reduce the noise of redundant features and to mine the semantic relationships. Therefore, this paper proposes a feature extraction method based on the knowledge graph. The method can construct a knowledge graph through semantic feature analysis and cooccurrence calculation. By this, it can extract the critical feature pairs of normal and abnormal requests (including four types of attacks) of the NSL-KDD dataset. Algorithm 1 shows the feature extraction method based on the knowledge graph, which is designed to select crucial feature pairs of different types.

In Algorithm 1, the output is the set M, which covers the four categories (i.e., intrinsic feature, content feature, time-based feature, and host-based feature) of the NSL-KDD dataset, and completes the intrusion detection classification tasks from a global perspective. Through this processing, we can effectively count out the semantic relations of the representative features of different attack types, thereby providing better support for subsequent classification. For example, the <scr_bytes, dst_host_same_srv_rate> and <tcp, same_src_rate> features often exist in R2L attacks at the same time.

3.4. Feature Extraction Based on Statistical Analysis. The knowledge graph highlights the semantic relationship of features and improves the robustness of intrusion detection by calculating features that often appear in pairs in the same attack type. However, some individual features also make an important contribution to intrusion detection systems. For instance, in the Probe attack of the NSL-KDD dataset, the Nmap command is repeatedly called to perform port or IP scanning. Thence, the corresponding number of network requests or certain features exceeds other types of attacks, even reaching a threshold. To this end, we propose a feature extraction method based on statistical analysis. This method uses mathematical statistics to calculate the average, median, and mode of different features in the training set. Further, this paper selects the average as the threshold of statistical features, and the difference between normal requests and abnormal requests is calculated as shown in

$$d_i = \sum_{i=1}^{N} \sum_{j=1}^{N} \left| \overline{x_i} - \overline{x_j} \right|. \tag{4}$$

In (4), $\overline{x_i}$ and $\overline{x_j}$ represent the average value of feature x's two classification types (i.e., i and j). The entire equation calculates a sum of the average differences of different features in network requests. The number N represents five types (normal, DoS, Probe, R2L, and U2R) in the NSL-KDD dataset. Through the above processing, we can maximize the differences between various categories. Then, we use the sorting algorithm to process the four feature categories (i.e., intrinsic feature, content feature, time-based feature, and host-based feature) and identify the two largest values of each category. Finally, eight key features based on statistical analysis are formed, which can further enhance the performance of our IDS. In addition, in the later deep learning classification model, when these features exceed a defined threshold (shown in the previous calculation), they are weighted to increase the gradient descent speed of the neural network.

Moreover, we use the power-law distribution to analyze the NSL-KDD dataset statistically, as shown in (5). As a result, the whole data presented a long-tail distribution, which often appears in some specific cyberattacks. Also, this stage further verifies that the intrusion detection system conforms to the social law and can map the crucial features from the statistical view.

$$P(k) \sim Cx^{-\alpha}. \tag{5}$$

3.5. Feature Fusion and Alignment. In this module, we design and implement a multiview feature fusion and alignment method. In recent years, multimodel deep learning methods have been gradually applied to intrusion detection. Aceto et al. [43] proposed a novel multimodal deep learning framework for encrypted traffic classification, named MIMETIC. The method can improve the performance of mobile traffic classification by learning intramodality and intermodality dependencies, thereby utilizing complementary views to identify traffic. Bu et al. [44] designed a neural network model with deep and parallel network-in-network (NIN) structures for classifying encrypted network traffic.

Input: Feature vector set $D = \{x_{ij}\}$ of the dataset, where i is the i^{th} network request, j is the j^{th} feature. Feature set $F = \{f_i, i = 1, 2, \ldots, n\}$.
Output: The selected feature subset $M = \{\langle f_i, f_j \rangle\}$.
(1) Initialization: set $M = \varnothing$.
(2) **for** each $x_{ij} \in D$ **do**
(3) Feature conversion by $v_{ij} = \begin{cases} 0, \text{ if } x_{ij} \le 0 \\ 1, \text{ if } x_{ij} > 0 \end{cases}$
(4) **end for**
(5) **for** each $f_i \in F$ **do**
(6) Calculate pairwise feature pairs and judge whether the features cooccur.
(7) NumPairs$(\langle f_i, f_j \rangle) = \begin{cases} n_{ij} + 0, \text{if there is no cooccurrence relationship} \\ n_{ij} + 1, \text{ if features } \mathrm{f_i} \text{ and } \mathrm{f_j} \text{ are cooccurring} \end{cases}$
(8) **end for**
(9) **for** each $\langle f_i, f_j \rangle$ from NumPairs **do**
(10) Use feature semantic cooccurrence relationship to construct a knowledge graph.
(11) (a) Build an entity feature set $E = \{e_1, e_2, \ldots, e_m\}$.
(12) (b) Build a relational feature set $R = \{r_{ij}, i, j = 1, 2, \ldots, m\}$, where $\mathrm{r_{ij}}$ is the weight of entity e_i and e_j.
(13) (c) Calculate the cooccurrence threshold set T of different types of features.
(14) (d) Build a knowledge graph of the IDS dataset.
(15) **end for**
(16) **for** each feature category **do**
(17) **for** each classification type **do**
(18) (a) Calculate the critical feature pairs of four features (i.e., intrinsic feature, content feature, time-based feature, and host-based feature) by sorting algorithm, which covers all classification types (e.g., DoS, Probe, R2L, U2R).
(19) (b) M = TopFeaturePairs$(\langle f_i, f_j \rangle)$
(20) Output key feature pairs based on the knowledge graph.
return M

Algorithm 1: Feature extraction based on the knowledge graph.

NIN can adopt a micronetwork after each convolutional layer to enhance local modeling and improve classification accuracy.

However, the above methods only learn features by fusing or combining different deep learning models, ignoring the feature extraction and feature fusion of network traffic in intrusion detection. Also, their time overhead and model complexity are high. Compared with these methods, this paper implements feature fusion and feature alignment in the feature extraction stage. This multiview feature fusion method belongs to a lightweight intrusion detection feature preprocessing, which effectively combines the advantage of two feature extraction methods. On the one hand, a knowledge graph-based feature extraction algorithm can mine the semantic relevance between features and extract four feature pairs. On the other hand, a statistical analysis-based feature extraction algorithm can select a single feature, which makes an essential contribution to our IDS and extracts eight key features.

Actually, the feature extraction from the two views proposed in this paper is better than the traditional IDS. The latter tends to ignore the correlation between features and select features only from a single view, such as using statistical features or weak correlation algorithms (e.g., principal component analysis or statistical frequency).

Next, we need to introduce multiview feature fusion and feature alignment processing for these features so that the following deep learning model can be better transformed into input word vectors. The whole calculation process is shown in Algorithm 2. The algorithm steps are as follows. (1) Convert the feature pairs identified by the knowledge graph into sequences, and perform alignment and deduplication processing. (2) Take the union operation on the features extracted from the knowledge graph and statistical analysis, and perform feature combination. (3) Assign different weights to the features of the two types of fusion, which will be used as the initial weights of the neural network model.

Algorithm 2 generates the weight set W through feature fusion and feature alignment. Then, we use word embedding to convert the network traffic into word vectors, which is the input of neural networks. Next, the set W will initialize the weight parameters of the corresponding feature vectors. In other words, if feature f_i belongs to the critical feature selected by our feature extraction algorithm (i.e., from the set U), the weight corresponding to the feature will be multiplied by for addition. Otherwise, the other weight remains unchanged. Through the above processing, we can improve the robustness of IDS and increase the gradient descent speed of neural networks. Thus, this stage will significantly contribute to the following intrusion detection and malicious request classification tasks.

3.6. Model Construction. Now, we construct an attention-based CNN-BiLSTM algorithm to learn the traffic features obtained from the proposed feature extraction and feature fusion module. In this model, CNN is similar to a feature extractor, which can reduce computing resources and is suitable for IoT devices (with limited resources). Moreover, LSTM performs classification operations based on the serialized feature information given by CNN, similar to a classifier. For example, if only one TCP handshake packet is

Input: Feature set $F = \{f_i, i = 1, 2, \ldots, n\}$. The feature subset $M = \{\langle f_i, f_j \rangle\}$ is selected by a knowledge graph. The feature subset $F_{SA} = \{f_k, k = 1, 2, \ldots, m\}$ is selected by statistical analysis.
Output: $W = \{w_i, i = 1, 2, \ldots, n\}$ is the weight parameter to be multiplied.
(1) Initialization: set $W = \varnothing$.
(2) Perform feature alignment on the feature pair M, and convert it into a sequence.
(3) **for** each $\langle f_i, f_j \rangle \in M$ **do**
(4) F_{KG} = FeaturesOfKG.append(f_i, f_j)
(5) F_{KG} = AlignmentBySort(F_{KG})
(6) **end for**
(7) Fuse the features extracted from the two views of KG and SA.
(8) $U = F_{KG} \cup F_{SA}$
(9) **for** each $f_i \in F$ **do**
(10) Weight calculation by $w_i = \begin{cases} w_i & \text{if } f_i \notin U \\ \alpha \bullet w_i & \text{if } f_i \in U \end{cases}$
(11) **end for**
(12) Output the weight set W.
return W

Algorithm 2: A multiview feature fusion and feature alignment approach.

analyzed in intrusion detection, it is difficult to judge whether it is a port scan. However, when multiple data packets are serialized and the LSTM network is used for judgment, it can be more accurately judged that these data packets are from port scanning attacks, with the reason that the LSTM network can learn context information and serialized features. Thus, this paper designs an attention-based CNN-BiLSTM model to reduce the computing resources and improve the result of IDS in IoT networks. Figure 3 summarizes the architecture of the attention-based CNN-BiLSTM model, which consists of six phases: (1) word embedding layer, (2) convolutional layer, (3) pooling layer, (4) BiLSTM layer, (5) attention mechanism layer, and (6) fully connected layer.

3.6.1. Word Embedding Layer. We perform word embedding processing on the extracted features and weight information. In this paper, Word2Vec is used to implement word embedding processing. The feature information of network traffic is converted into a vector, which will be used as the input of the subsequent deep learning model. Therefore, the attention-based CNN-BiLSTM model can be trained more quickly, making our model learn the feature distribution efficiently.

3.6.2. Convolutional Layer. Convolutional Neural Network (CNN) is designed to extract local features by scrolling the convolution kernel. In IoT intrusion detection, a CNN model can effectively highlight the key features of network requests. In this paper, we construct a convolutional layer to extract import elements (e.g., some features that significantly impact DoS attacks). The convolution kernel convolves the input word vector matrix, and its filter will select different features (step 2 in Figure 3). The convolutional layer calculation is as shown in

$$h_i^d = f\left(w_d \cdot V_i + b_d\right), \tag{6}$$

where V_i is a word vector of network request feature in (6), and $V_i \in R^{n \times k}$, n is the number of features, k is the dimension of the word vector, w_d is a convolution kernel of size d, and b_n is the bias vector. Here, f is an activation function (e.g., ReLU), and the obtained feature is denoted as h_i^d. After the convolution processing, the local feature set H is obtained by mapping, and we can write the following equation:

$$H_d = \left\{h_1^d, h_2^d, \ldots, h_{n-d+1}^d\right\}. \tag{7}$$

3.6.3. Pooling Layer. In this stage, we sample the output vector of the convolution process and calculate the optimal solution of local features, thereby reducing the dimension of our features and maintaining the core features of the network traffic (step 3 in Figure 3). This paper uses Max Pooling technology to pool features as (8), which can calculate the most critical feature of malicious or normal requests.

$$M_i = \max\{H_d\}. \tag{8}$$

After the pooling layer extracts essential features, the obtained features are added to the subsequent network model. Finally, the output vector S formed by the combination of this step is defined by

$$S = \{M_1, M_2, \ldots, M_n\}. \tag{9}$$

3.6.4. BiLSTM Layer. Long Short-Term Memory (LSTM) is a variant of Recurrent Neural Network (RNN), a sequence processing model. The model retains the key information and forgets the secondary information through the memory unit and gate structure. A Bidirectional LSTM (BiLSTM) model comprises a forward LSTM and a backward LSTM, which encodes features from the front and back directions. This paper uses BiLSTM to extract contextual semantic information and capture the long-distance dependence of network traffic (as shown in Figure 3). Taking DoS attacks as an example, this model can effectively identify the

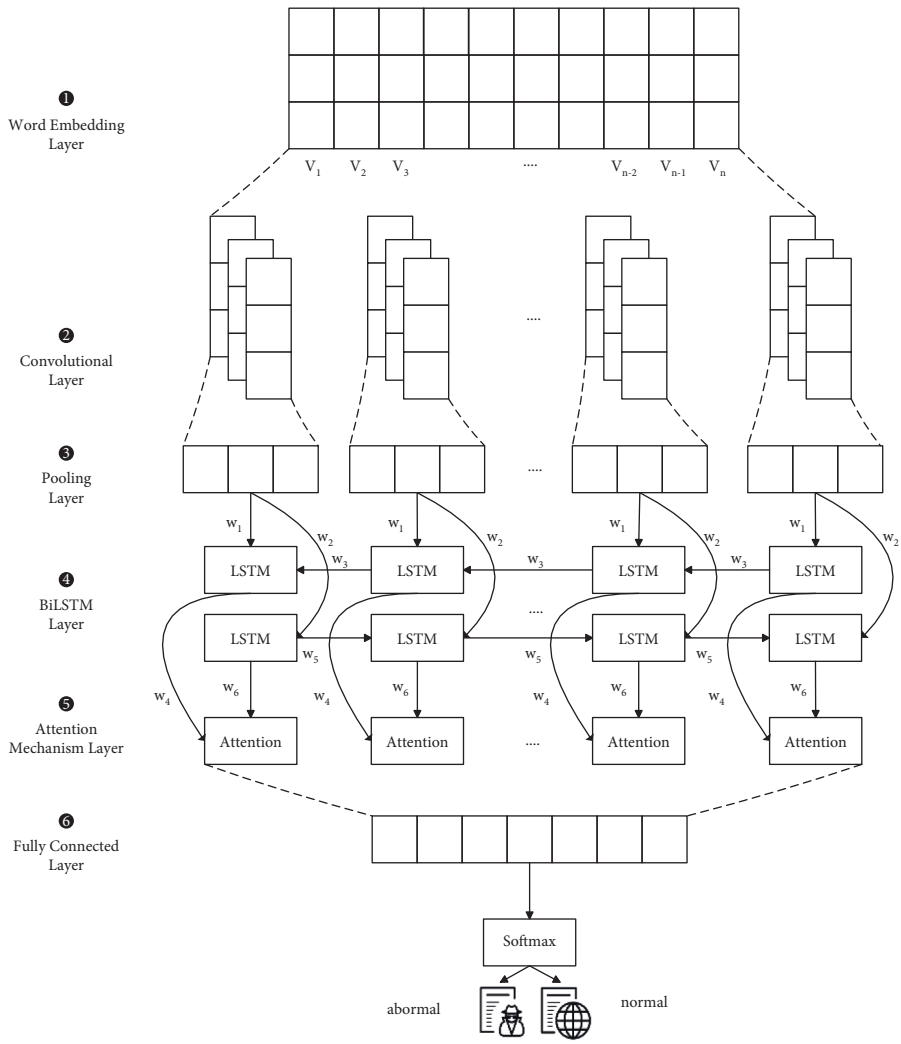


Figure 3: The architecture of an attention-based CNN-BiLSTM model.

relationship between the number of host connections and a specific flag feature of the connection rejection error (e.g., REJ). Obviously, this processing can maintain contextual semantic information, and the BiLSTM is a coarse-grained intrusion detection model. In this paper, the S vectors processed by the CNN model are used as the input of the BiLSTM model. They will be connected to form the CNN-BiLSTM model and complete the IoT intrusion detection task. The structure of BiLSTM includes the forget gate, input gate, output gate, and memory unit, which can be defined as follows:

$$\begin{cases} \overrightarrow{h_t} = f\left(w_1 \cdot s_t + w_2 \cdot \overrightarrow{h_{t-1}}\right), \\ \overleftarrow{h_t} = f\left(w_3 \cdot s_t + w_5 \cdot \overleftarrow{h_{t+1}}\right), \\ y_t = g\left(w_4 \cdot \overrightarrow{h_t} + w_6 \cdot \overleftarrow{h_t}\right). \end{cases} \tag{10}$$

In (10), $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$, respectively, represent the state of the forward LSTM layer and the backward LSTM layer at time t, corresponding to the context feature of IoT network traffic. s_t is the word vector input at time t, and w_1 to w_6 represent the layer's weight parameters. Also, f and g are activation functions, including sigmoid and tanh, y_t is the final output result of the BiLSTM model. By this, we can effectively extract long-distance dependent features and solve the problem of local feature loss.

3.6.5. Attention Mechanism Layer. The attention mechanism aims to allocate limited attention resources to crucial information, thereby enhancing the relevance of feature vectors and output results, which is widely used in natural language processing. In this paper, the attention mechanism is introduced to strengthen the attention features of the neural network. These features are extracted from the previous knowledge graph and statistical analysis. Thus, the attention layer can increase the contribution of key features to the corpus and add weights to better distinguish between normal requests and abnormal requests. This paper mainly uses the basic form of attention mechanism to pay attention to the weight distribution of IoT network traffic. The attention mechanism will focus on the semantic features and

key features of different attack types. For example, in DoS attacks, the feature same_srv_rate and dst_host_same_srv_rate often appear. Our model will highlight their importance and ignore lesser features. The calculation process of this step is shown in equations (11) to (13).

$$v_t = \tanh\left(w_c \cdot y_t + b_c\right), \tag{11}$$

$$a_t = \text{soft}\max\left(w^T, v_t\right), \tag{12}$$

$$u = \sum_{t=1}^{n} a_t \cdot y_t. \tag{13}$$

Here, (11) generates the target attention weight, namely, v_t, which is a result of the nonlinear transformation of the activation function tanh, y_t is a vector output of the CNN-BiLSTM network, w_c is a parameter of training weight, and b_c is bias item. (12) uses the softmax function to calculate the importance of each component v_t, the probability vector of the weight is a_t, and w^T is the transposed matrix. Finally, the generated attention weight is matched to the corresponding output vector y_t in our model, and u represents the sentence vector of the weighted sum of the importance of y_t, as shown in (13).

3.6.6. Fully Connected Layer. The fully connected layer plays the role of classifier in the neural network model. It maps the learned distributed features representation to the sample label space. Finally, this paper designs a softmax classifier for obtaining the classification results of intrusion detection and complete IoT networks' intrusion detection. In a word, we have successfully constructed the attention-based CNN-BiLSTM model to distinguish different classes through the above six steps shown in Figure 3.

4. Evaluation

In this section, we evaluate the performance of the proposed system in intrusion detection and attack classification. Also, we analyze the results of feature extraction methods based on the knowledge graph and statistical analysis. Finally, we compare the proposed system with state-of-the-art systems to verify the effectiveness and robustness of intrusion detection.

4.1. Dataset. In this paper, we evaluate the proposed approach on the NSL-KDD dataset [42], which is a new revised version of the KDD Cup 99 that has been generated by IoT devices simulating real-time attacks, bench-marked datasets for IDS. We can get the NSL-KDD dataset through the link [45]. The NSL-KDD dataset is the traffic request generated by MIT Lincoln Laboratory using IoT devices to simulate attacks in the real-time environment. Thus, it is a relatively authoritative dataset. Table 4 gives details of the NSL-KDD dataset. There are 125973 training traffic samples and 22544 testing traffic samples in the dataset, involving four categories of attacks. DoS is a denial of service attack, Probe is a port listening or scanning attack, Remote to Local (R2L) is a remote to local attack, and User-to-Root (U2R) is an unauthorized and trying to gain superuser or root. In the NSL-KDD dataset, there are 39 subcategories of attack scattered in four categories (i.e., DoS, Probe, R2L, and U2R), among which 22 subcategories appear in the training set, and 17 more different attack subcategories (i.e., unknown attacks) exist in the testing set. By this, we can evaluate the performance of our model for unknown attack types.

Next, Table 5 is a detailed description of 41 features, which includes four categories: intrinsic features of TCP connections, content features of TCP connections, time-based network traffic statistics features, and host-based network traffic statistics features. In addition, the attribute features include three nonnumerical features (i.e., protocol_type, service, and flag) and 38 numeric features. In the previous data preprocessing section, we systematically explain how to clean and preprocess our dataset. This section mainly focuses on the experimental evaluation of the proposed model.

4.2. Evaluation Metrics. To evaluate the performance of the intrusion detection model for IoT networks, this paper calculates the precision, recall, F_1-score, accuracy, and false alarm rate (FAR). Table 6 shows the detailed confusion matrix. By comparing the actual network request label and the predicted network request label (including normal requests and attack requests), the results of the classification algorithm are evaluated.

As shown in Table 6, True Positive (TP) means that both the predicted results of the network request and the actual label are positive (i.e., attack label). True Negative (TN) implies that the predicted results of the network request and the actual label are negative (i.e., normal label). False Positive (FP) indicates that the predicted result is negative, but the actual label is positive. False Negative (FN) indicates that the predicted result is positive, but the actual label is negative.

The five metrics are calculated by the following equation:

$$\begin{aligned}
\text{Accuracy} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}},\\
\text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}},\\
\text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}},\\
F_1 - \text{score} &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}},\\
\text{FAR} &= \frac{\text{FP}}{\text{FP} + \text{TN}}.
\end{aligned} \tag{14}$$

4.3. Experimental Setup. We implement the intrusion detection model of IoT networks using TensorFlow and Keras, with the programming language Python (version 3.7). Also, all experiments are run in Windows 10 64-bit operating

Table 4: Distribution of the NSL-KDD dataset.

Category	Normal	Abnormal				Total
		DoS	Probe	R2L	U2R	
Train	67343	45927	11656	995	52	125973
Test	9711	7458	2421	2754	200	22544

Table 5: Feature list of the NSL-KDD dataset.

No.	Feature Category	Feature Name
1	Intrinsic feature	duration
2		protocol_type
3		Service
4		Flag
5		src_bytes
6		dst_bytes
7		land
8		wrong_fragment
9		Urgent
10	Content feature	hot
11		num_failed_logins
12		logged_in
13		num_compromised
14		root_shell
15		su_attempted
16		num_root
17		num_file_creations
18		num_shells
19		num_access_files
20		num_outbound_cmds
21		is_hot_login
22		is_guest_login
23	Time-based feature	count
24		srv_count
25		serror_rate
26		srv_serror_rate
27		rerror_rate
28		srv_rerror_rate
29		same_srv_rate
30		diff_srv_rate
31		srv_diff_host_rate
32	Host-based feature	dst_host_count
33		dst_host_srv_count
34		dst_host_same_srv_rate
35		dst_host_diff_srv_rate
36		dst_host_same_src_port_rate
37		dst_host_srv_diff_host_rate
38		dst_host_serror_rate
39		dst_host_srv_serror_rate
40		dst_host_rerror_rate
41		dst_host_srv_rerror_rate

system environment with Inter Core i7-8700K CPU, 64 GB memory, and GTX 1080Ti GPU.

To detect malicious requests in IoT networks, we construct an attention-based CNN-BiLSTM model in this paper. In terms of model parameter setting, the number of convolution kernels of the CNN model is 128. The activation function rectified linear unit (ReLU) is adopted for the convolutional and fully connected layers. At the same time, the number of neurons in both the forward and backward directions of the BiLSTM model is 128, and the optimization algorithm selects the Adam optimizer. Then, the initial learning rate is set to 0.001, which is a standard starting point for traditional deep learning. During the experiment, the loss functions are the binary cross-entropy for binary classification and the categorical cross-entropy for multiclass classification. The dropout mechanism is introduced to randomly sample data for training to prevent overfitting, and its keep_prob parameter is set to 0.4. For completeness, we leverage the sci-kit-learn library to construct the other machine learning classification models as baselines (e.g., logistic regression, support vector machine, random forest, etc.). Further, all experiments use the same environment and the NSL-KDD dataset for a fair comparison. We can not only complete the detection of malicious requests (for binary classification) but also identify the types of attacks (for multiclass classification).

Table 6: Confusion matrix.

Predicted label	Actual label	
	Normal	Attack
Normal	True Negative (TN)	False Negative (FN)
Attack	False Positive (FP)	True Positive (TP)

Moreover, we implement two feature extraction algorithms based on knowledge graph and statistical analysis in Python, as shown in the previous section. Then, these semantic feature pairs and key features will be visually displayed by the Gephi tool and some Python libraries (e.g., matplotlib, seaborn, and pyecharts). In particular, these key features will add to the weight parameters of the IDS neural network model proposed in this paper. Finally, to better verify the effectiveness and robustness of the model, the final experimental result is the average of 10 network traffic classification results. By this, we can reduce the noise influence of a possible abnormal result through multiple experiments.

4.4. Analysis of Proposed Feature Extraction. We implement a feature extraction algorithm based on knowledge graphs in this section and conduct detailed experimental evaluations. We use the previous Algorithm 1 to perform entity extraction and relationship extraction on the training set of the NSL-KDD dataset and generate the corresponding feature knowledge graphs of four attack types as shown in Figure 4. The knowledge graph is scattered from the center to the surroundings, and the feature relationship pairs are represented by two tuples. Note that the thicker the lines between the features that are more closely related, and the more features that appear, the greater their size, and the features are clustered in different colors. For example, Figure 4(a) is the knowledge graph of the DoS attack type. It can be found that the feature two tuples with close semantic relationship are <same_srv_rate, dst_host_same_srv_rate>, <dst_host_diff_srv_rate, same_srv_rate>, <tcp, same_srv_rate>, <dst_host_count255, same_srv_rate>, etc.

In addition, by comparing each subgraph, we can find that the semantic relationship of different attack types is

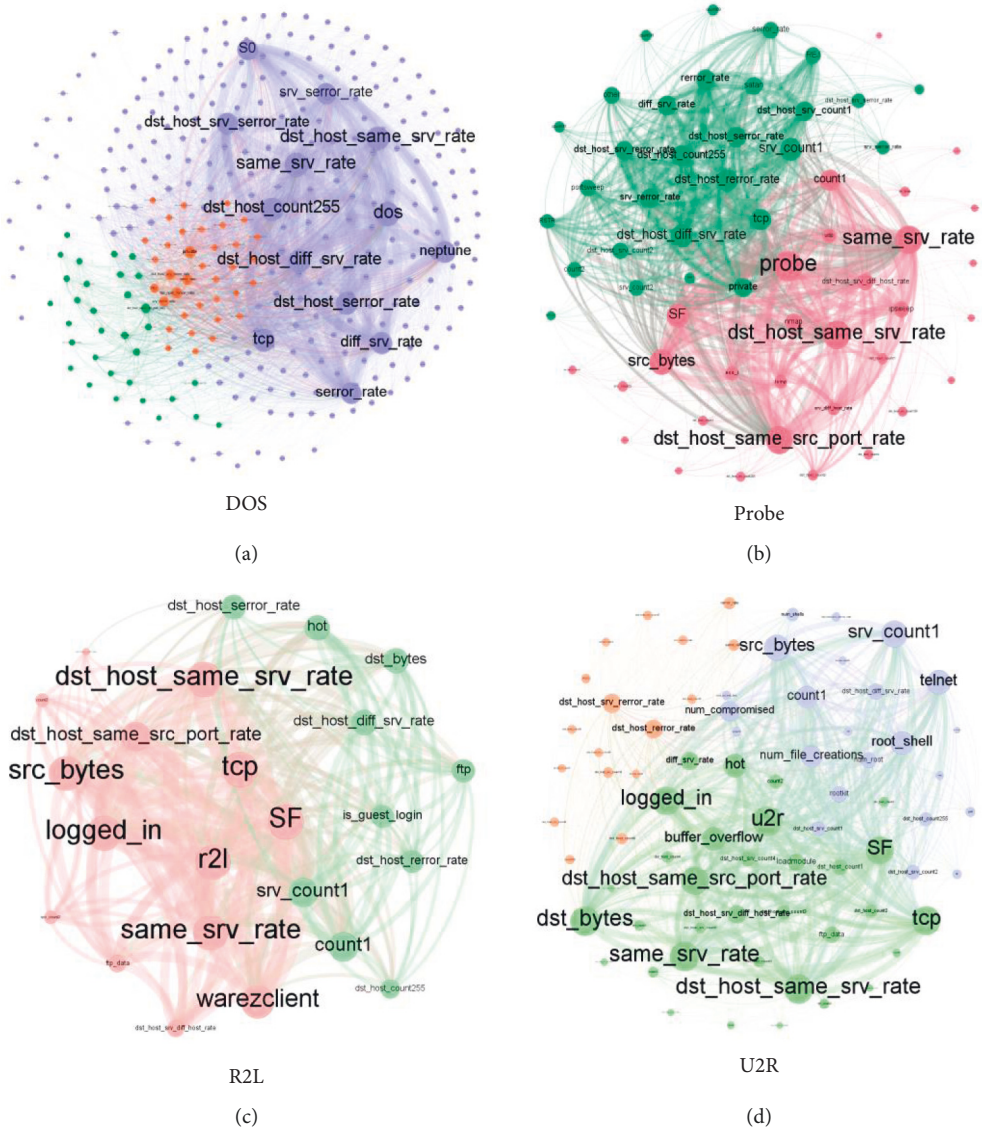


Figure 4: The semantic feature relationships of four attack types are extracted by a knowledge graph. (a) DoS. (b) Probe. (c) R2L. (d) U2R.

different. Among them, the closely related feature pairs in the DoS attack are concentrated in the host-based feature; the closely related feature pairs in the Probe attack are concentrated in the two major categories of time-based features and host-based features; the closely related feature pairs in the R2L attack cover four categories, namely, intrinsic features (e.g., SF), content features (e.g., logged_in), time-based features (e.g., same_srv_rate), and host-based features (e.g., dst_host_same_srv_rate); U2R attacks also cover four categories of features.

At the same time, we construct a knowledge graph for the training set of normal network requests, as shown in Figure 5 in the appendix. Table 7 shows the critical feature pairs of normal and abnormal requests (covering five labels). Each classification label displays the top 8 unique results represented by triples <src_feature, dst_feature, weight>, corresponding to the source position of the feature, the destination position of the feature, and semantic relation weight. Each weight adopts Min-Max normalization processing.

This section implements the feature extraction algorithm based on statistical analysis. We select features on the different classification labels of the four feature categories (i.e., normal, DoS, Probe, R2L, and U2L). Each category extracts two essential features. The results are shown in Table 8 in the appendix, represented by the two-tuple < feature, diff>, corresponding to the key features and the average difference of this feature's category. The calculation process is shown in the previous equation (4).

To further evaluate the effect of the feature extraction algorithm based on statistical analysis, we use the power-law distribution function to analyze the NSL-KDD dataset and extract six representative features (i.e., duration, num_compromised, num_root, src_bytes, dst_bytes, and srv_count). The experimental results are shown in Figure 6. The experimental results show that the power-law

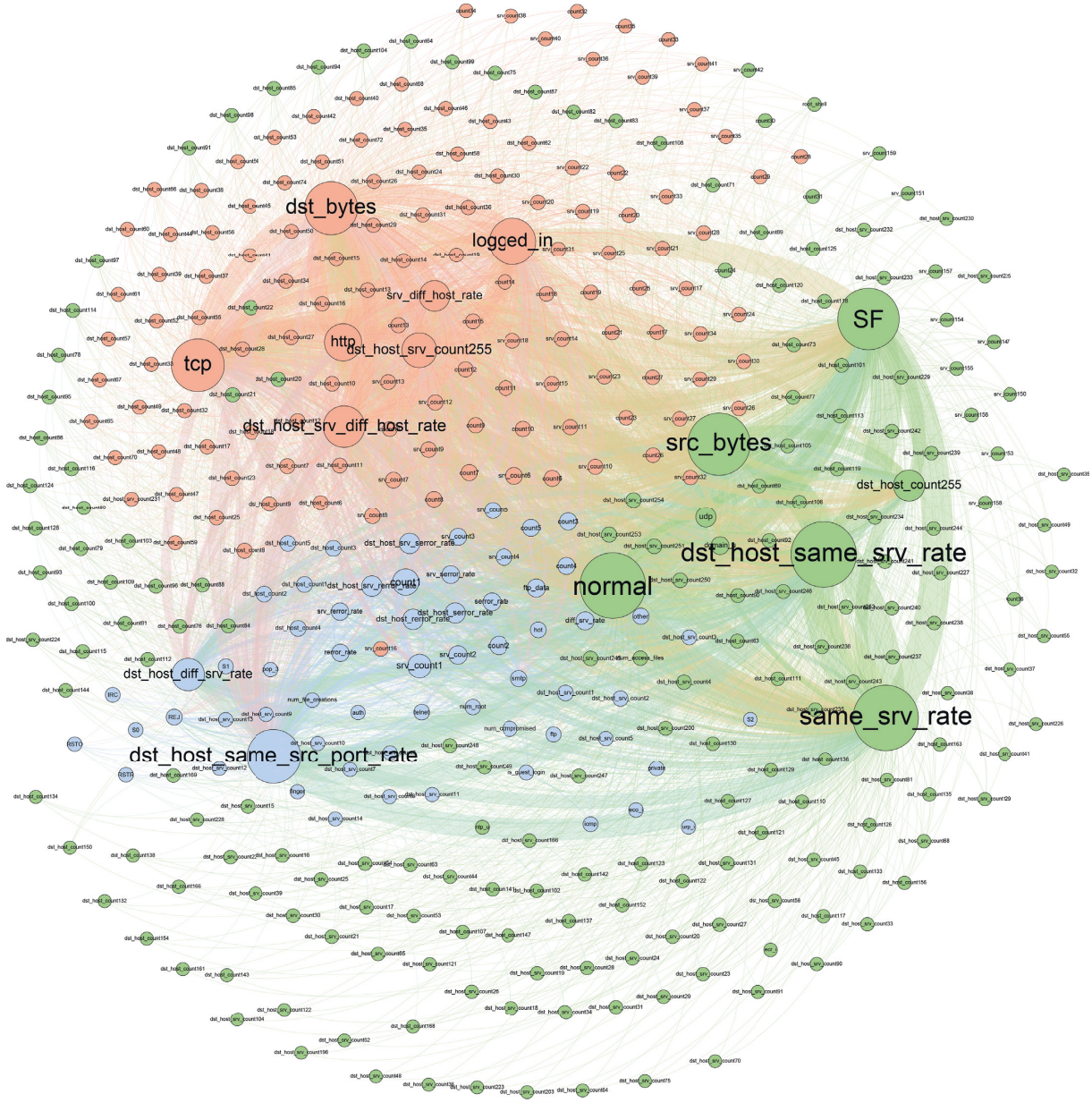


Figure 5: The semantic feature relationships of normal requests are extracted by a knowledge graph.

distribution of features is generally consistent with the features extracted in this section, and some features are affected by unbalanced data distribution. However, the traffic features of the whole NSL-KDD dataset basically conform to the law of human network attack activities, and the attacks are mainly caused by human factors. So far, we effectively verified that the feature extraction algorithm proposed in this paper is very important to the identification of abnormal requests and the detection of attack types.

Finally, we implement feature fusion and feature alignment algorithms according to Algorithm 2. The fusion features are shown in Table 9. Among them, $f^{(KG)}$ represents the experimental results of the knowledge graph-based feature extraction algorithm, $f^{(SA)}$ represents the experimental results of the statistical analysis-based feature extraction algorithm, and $f^{(KG,SA)}$ represents the features recognized by two algorithms. Also, f_i represents the i-th feature, with a total of 41 features. Through Algorithm 2, we perform feature fusion and alignment for normal, DoS, Probe, R2L, and U2L, and the number of key features extracted for each type is 12, 14, 15, 13, and 14, respectively. Then, these features will be added to the initial weight of the subsequent IDS system based on deep learning. Therefore, since the semantic relationship between features and the statistical distribution of features are considered in our system, IDS combining the two views can better detect the IoT malicious traffic and identify different types of network attack requests.

Table 7: Key semantic features extracted by the knowledge graph.

Label	Feature category
Normal	①<same_srv_rate, dst_host_same_srv_rate,0.980> ②<src_bytes, same_srv_rate,0.950> ③<SF,src_bytes,0.940> ④<src_bytes, dst_bytes,0.830> ⑤<tcp,same_srv_rate,0.800> ⑥<logged_in,same_srv_rate,0.710> ⑦<same_srv_rate, dst_host_same_src_port_rate,0.670> ⑧<http, same_srv_rate,0.570>
DoS	①<same_srv_rate, dst_host_same_srv_rate,0.970> ②<dst_host_diff_srv_rate, same_srv_rate,0.950> ③<tcp,same_srv_rate,0.920> ④<dst_host_count255,same_srv_rate,0.920> ⑤<tcp,diff_srv_rate,0.910> ⑥<dst_host_serror_rate, dst_host_diff_srv_rate,0.780> ⑦<tcp,dst_host_srv_serror_rate,0.770> ⑧<serror_rate, dst_host_serror_rate,0.770>
Probe	①<same_srv_rate, dst_host_same_src_port_rate,0.950> ②<same_srv_rate, dst_host_same_srv_rate,0.810> ③<src_bytes, same_srv_rate,0.700> ④<dst_host_diff_srv_rate, dst_host_rerror_rate,0.640> ⑤<count, dst_host_same_src_port_rate,0.610> ⑥<SF,src_bytes,0.600> ⑦<dst_host_count255,dst_host_diff_srv_rate,0.580> ⑧<rerror_rate, dst_host_rerror_rate,0.550>
R2L	①<tcp,same_srv_rate,1.000> ②<same_srv_rate, dst_host_same_srv_rate,0.990> ③<src_bytes,tcp,0.970> ④<SF,same_srv_rate,0.940> ⑤<logged_in,SF,0.910> ⑥<dst_host_same_src_port_rate, dst_host_same_srv_rate,0.740> ⑦<count, same_srv_rate,0.710> ⑧<srv_count, count,0.700>
U2L	①<SF,same_srv_rate,0.980> ②<dst_bytes, same_srv_rate,0.961> ③<tcp,dst_bytes,0.941> ④<dst_host_same_srv_rate, same_srv_rate,0.922> ⑤<logged_in,dst_bytes,0.882> ⑥<srv_count, same_srv_rate,0.824> ⑦<dst_host_same_src_port_rate, dst_host_same_srv_rate,0.824> ⑧<src_bytes, dst_bytes,0.725>

4.5. Comparison of Different Intrusion Detection System Models. This paper proposes an enhanced intrusion detection system model based on deep learning and a knowledge graph. To verify the effectiveness of our system, the experimental evaluation is compared with the traditional machine learning models and the existing deep learning models. Detailed comparative experimental results are shown in Table 10. The precision, recall, and F_1-score of the proposed system are 0.9035, 0.9107, and 0.9071, respectively, superior to state-of-the-art systems. Moreover, the F_1-score of our system is 10.29% higher than that of the best machine learning algorithm (random forest) and 5.46% higher than that of the compared deep learning algorithm (CNN-BiLSTM-attention). Therefore, our system improves the semantic relationship between traffic features through knowledge graphs, improves the importance of crucial features through statistical analysis, and adds weight to the deep learning model to improve the accuracy of IDS for IoT networks. Meanwhile, the CNN-BiLSTM-attention model constructed in this paper can effectively capture long-distance dependent information, and the attention mechanism can highlight some features. These factors can highlight the contribution of our model to intrusion detection.

In addition, the detection time of IoT network traffic is also an important indicator for evaluating IDS, which can effectively measure the time cost and algorithm complexity of a model. Column 6 of Table 10 gives the detection times for different systems in seconds. The detection time of the system in this paper for the network traffic of the test set is 22.17 seconds, and it can effectively detect 1016 network requests per second. The entire detection time efficiency is in the upper-middle range. Compared with the improvement of the F1-score, the time cost is within an acceptable range, only slightly higher than other models, and does not show an exponential increase.

Table 8: Key features extracted by the statistical analysis.

Label	Intrinsic feature	Content feature
Normal	<dst_bytes,0.1187>	<logged_in,0.3149>
	<src_bytes,0.0971>	<is_guest_login,0.0036>
DoS	<dst_bytes,0.1379>	<logged_in,0.3749>
	<duration,0.0284>	<is_guest_login,0.0094>
Probe	<dst_bytes,0.1435>	<logged_in,0.3886>
	<src_bytes,0.1082>	<is_guest_login,0.0093>
R2L	<src_bytes,0.1625>	<logged_in,0.5179>
	<duration,0.0736>	<is_guest_login,0.3052>
U2L	<duration,0.2563>	<logged_in,0.4889>
	<dst_bytes,0.2138>	<num_shells,0.0671>
Label	Time-based Feature	Host-based Feature
Normal	<same_srv_rate,0.3085>	<dst_host_srv_count,0.2927>
	<serror_rate,0.2711>	<dst_host_same_srv_rate,0.2907>
DoS	<serror_rate,0.4690>	<dst_host_srv_serror_rate,0.4659>
	<srv_serror_rate,0.4642>	<dst_host_serror_rate,0.4634>
Probe	<srv_rerror_rate,0.3233>	<dst_host_same_src_port_rate,0.5034>
	<rerror_rate,0.3173>	<dst_host_srv_rerror_rate,0.3208>
R2L	<same_srv_rate,0.3358>	<dst_host_same_src_port_rate,0.4485>
	<serror_rate,0.2726>	<dst_host_count,0.3651>
U2L	<srv_serror_rate,0.2825>	<dst_host_count,0.5270>
	<same_srv_rate,0.2706>	<dst_host_srv_count,0.4147>

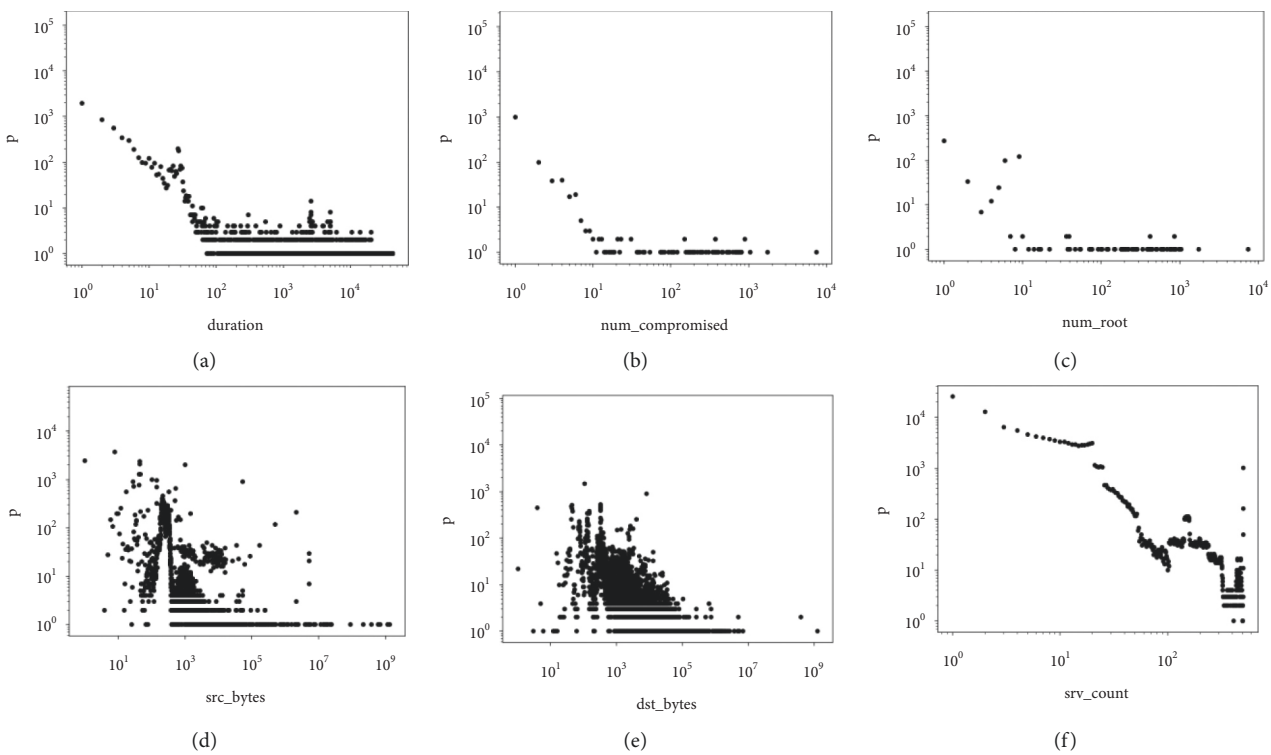


Figure 6: Experimental results of power-law distribution with different features. (a) duration. (b) num_compromised. (c) num_root. (d) src_bytes. (e) dst_byte.s (f) srv_count.

Note that the comparison IDS in this paper is due to differences in datasets, and some of the code is not open source. Therefore, the comparison systems combine the methods and ideas of the references and the typical intrusion detection methods to reconstruct the code, such as SVM [13], KNN [14], DT [15], RF [16], LSTM [32], and BiLSTM [33] in Table 10. Finally, we reproduce these baseline methods and conduct a detailed comparative analysis of our dataset.

In order to further measure the detection effect of the proposed IDS model and evaluate its identification results of malicious network requests and normal network requests,

Table 9: Feature fusion results for the five types of requests on the NSL-KDD dataset.

Label	Number	Feature fusion and feature alignment
Normal	12	$f_2^{(KG)}, f_3^{(KG)}, f_4^{(KG)}, f_5^{(KG,SA)}, f_6^{(KG,SA)}, f_{12}^{(KG,SA)}, f_{22}^{(SA)}, f_{25}^{(SA)}, f_{29}^{(KG,SA)}, f_{33}^{(SA)}, f_{34}^{(KG,SA)}, f_{36}^{(KG)}$
DoS	14	$f_1^{(SA)}, f_2^{(KG)}, f_6^{(SA)}, f_{12}^{(SA)}, f_{22}^{(SA)}, f_{25}^{(KG)}, f_{26}^{(SA)}, f_{29}^{(KG,SA)}, f_{30}^{(KG)}, f_{32}^{(KG)}, f_{34}^{(KG)}, f_{35}^{(KG)}, f_{38}^{(KG,SA)}, f_{39}^{(KG,SA)}$
Probe	15	$f_4^{(KG)}, f_5^{(KG,SA)}, f_6^{(SA)}, f_{12}^{(SA)}, f_{22}^{(SA)}, f_{23}^{(KG)}, f_{27}^{(KG,SA)}, f_{28}^{(SA)}, f_{29}^{(KG)}, f_{32}^{(KG)}, f_{34}^{(KG)}, f_{35}^{(KG)}, f_{36}^{(KG,SA)}, f_{40}^{(KG)}, f_{41}^{(SA)}$
R2L	13	$f_1^{(SA)}, f_2^{(KG)}, f_4^{(KG)}, f_5^{(KG,SA)}, f_{12}^{(KG,SA)}, f_{22}^{(SA)}, f_{23}^{(KG)}, f_{24}^{(KG)}, f_{25}^{(SA)}, f_{29}^{(KG,SA)}, f_{32}^{(SA)}, f_{34}^{(KG)}, f_{36}^{(KG,SA)}$
U2L	14	$f_1^{(SA)}, f_2^{(KG)}, f_4^{(KG)}, f_5^{(KG)}, f_6^{(KG,SA)}, f_{12}^{(KG,SA)}, f_{18}^{(SA)}, f_{24}^{(KG)}, f_{26}^{(SA)}, f_{29}^{(KG,SA)}, f_{32}^{(SA)}, f_{33}^{(SA)}, f_{34}^{(KG)}, f_{36}^{(KG)}$

Table 10: Performance comparison of various IoT intrusion detection models.

Type	Model	Precision	Recall	F_1-score	Detection Time
ML-based IDS	KNN	0.7997	0.7940	0.7968	23.40
	LR	0.7739	0.7578	0.7658	5.97
	DT	0.7726	0.7613	0.7669	5.44
	SVM	0.7741	0.7600	0.7670	13.26
	RF	0.8131	0.7956	0.8043	10.99
	AdaBoost	0.8065	0.7874	0.7969	18.40
DL-based IDS	CNN	0.8294	0.8228	0.8261	7.82
	TextCNN	0.8417	0.8391	0.8404	19.40
	LSTM	0.8265	0.8179	0.8222	13.93
	BiLSTM	0.8500	0.8504	0.8502	14.15
	BiGRU	0.8271	0.8183	0.8227	11.46
	CNN-BiLSTM-Att	0.8520	0.8530	0.8525	19.34
	Our System	0.9035	0.9107	0.9071	22.17

we compared the accuracy and FAR of different IDS models. The results are shown in Figure 7, and the accuracy of our system is 0.9001, and the FAR is 0.0120, both of which are better than the experimental results of other models. The accuracy is 14.71% higher than the average of the six machine learning models and 8.50% higher than the average of the other six deep learning models. On the other hand, the FAR is 4.61% lower than the average of the six machine learning models and 2.45% lower than the average of the other six deep learning models. The experiment result indicates that our system can identify more malicious requests and discover semantic feature relationships between normal and abnormal requests, that is, the feature relationship pairs that often appear at the same time, including < same_srv_rate, dst_host_same_srv_rate>, <dst_host_diff_srv_rate, same_srv_rate>, <same_srv_rate, and dst_host_same_src_port_rate>.

This paper implements various malicious request attack identification experiments on the proposed model. The experimental results of the five labels are shown in Table 11. Among them, the recognition performance of normal request, DoS attack, and Probe attack is better, and their F_1-score is 0.9107, 0.9273, and 0.8849, respectively, indicating that our system can effectively extract semantic relations and key features. Although the experimental results of the U2R type are poor due to small samples, our system yet detects 12 attack requests, reflecting that our system has better robustness and accuracy in identifying unknown attacks.

To better evaluate the performance of the attention-based CNN-BiLSTM intrusion detection system and compare the ability of IDSs to identify malicious requests with different thresholds, this paper selects the random forest algorithm (a better machine learning model) and the CNN-BiLSTM-Att algorithm (a better deep learning model) to compare the ROC curve with the IDS system. Figure 8 shows the final renderings. It can be seen that the AUC area corresponding to the ROC curve of our system is the largest, which proves that our IDS has the best comprehensive performance and can obtain higher TPR and lower FPR.

Moreover, this paper compares different intrusion detection systems to identify malicious attack types (i.e., multiclassification tasks), and the experimental results are shown in Figure 9. The abscissa is various systems, and the ordinate is the F_1-score corresponding to the detection results of different systems. Note that we only select the random forest model, the best machine learning algorithm in our experiment. In Figure 9, whether the system in this paper detects normal network requests or recognizes malicious request types, its F_1-score is better than other systems, and it can effectively identify unknown attack types. Among them, the F_1-score of the normal type is 13.37% higher than the average value of other comparison systems; the DoS attack type is 7.25% higher than the average value of the other comparison systems; the Probe attack type is 15.68% higher than the average value of the other comparison systems; the R2L attack type is higher than the average value. The average value of other comparison systems is 43.12% higher; the U2R attack type is 10.67% higher than the average value of other comparison systems. In short, this experiment further proves the contribution of the IDS in this paper. The two feature extraction algorithms proposed in this paper (i.e., knowledge graph-based feature extraction algorithm and statistical analysis-based feature extraction algorithm) can effectively mine the semantic relationship between features and improve the performance of the IDS for IoT networks.

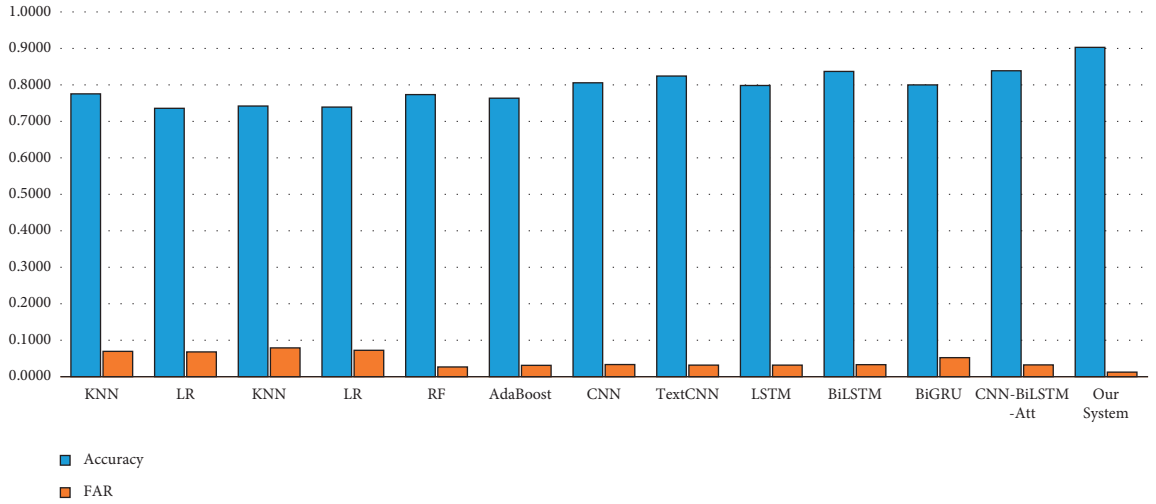


Figure 7: Comparison of accuracy and FAR of different IoT intrusion detection models.

Table 11: The results of five labels by our intrusion detection system.

Label	Precision	Recall	F_1-score
Normal	0.8541	0.9754	0.9107
DoS	0.9021	0.9540	0.9273
Probe	0.8742	0.8959	0.8849
R2L	0.9060	0.3500	0.5050
U2R	0.5455	0.0600	0.1081

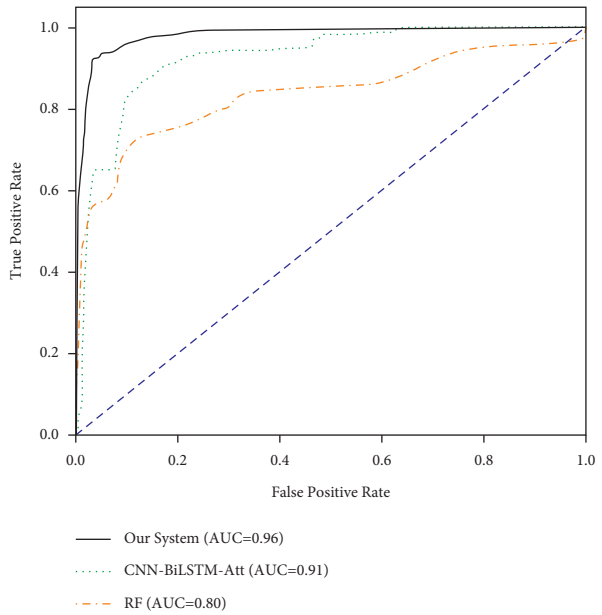


Figure 8: Comparison of ROC curves of different systems.

Finally, to prove that our system has a good detection effect on unknown attacks, this paper compares the results of different models against unknown types of attacks. In the testing set of the NSL-KDD dataset, 17 subcategories belong to unknown attacks. Among them, DoS includes 4 types of unknown attacks (i.e., apache2, mailbomb, processtable, and udpstorm) and 1717 traffic samples; Probe includes 2 types of unknown attacks (i.e., mscan and saint), a total of 1315 traffic samples; R2L includes 7 types of unknown attacks (i.e., named, sendmail, snmpgetattack, snmpguess, worm,

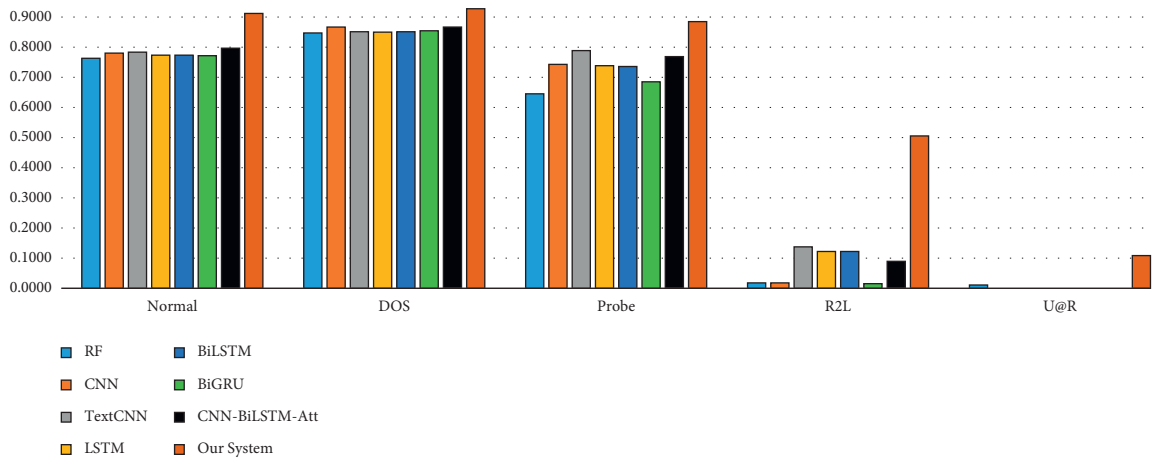
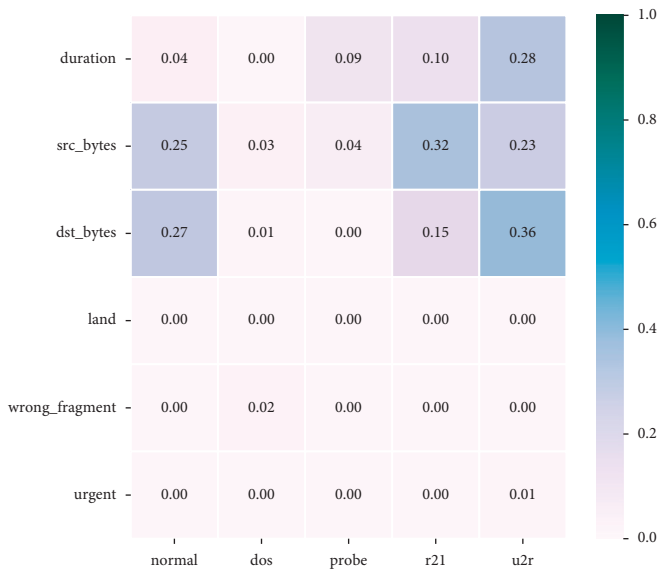


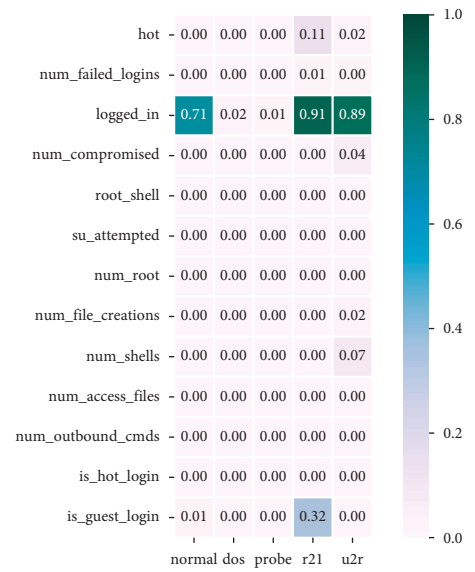
Figure 9: Experimental results of different systems to identify malicious attack types.

Table 12: Performance comparison of different systems for unknown types.

Type	Model	F_1-score	Ranking of various labels of detection performance
ML-based IDS	KNN	0.6201	Probe > DoS > U2R > R2L
	LR	0.5701	Probe > DoS > U2R > R2L
	DT	0.5804	DoS > Probe > U2R > R2L
	SVM	0.5865	Probe > DoS > U2R > R2L
	RF	0.6329	Probe > DoS > U2R > R2L
	AdaBoost	0.5676	Probe > DoS > U2R > R2L
DL-based IDS	CNN	0.7096	Probe > DoS > U2R > R2L
	TextCNN	0.7566	Probe > DoS > U2R > R2L
	LSTM	0.7169	Probe > DoS > U2R > R2L
	BiLSTM	0.8033	Probe > DoS > U2R > R2L
	BiGRU	0.7579	Probe > DoS > U2R > R2L
	CNN-BiLSTM-Att	0.8146	Probe > DoS > U2R > R2L
	Our System	0.8783	Probe > DoS > U2R > R2L



(a)



(b)

Figure 10: Continued.

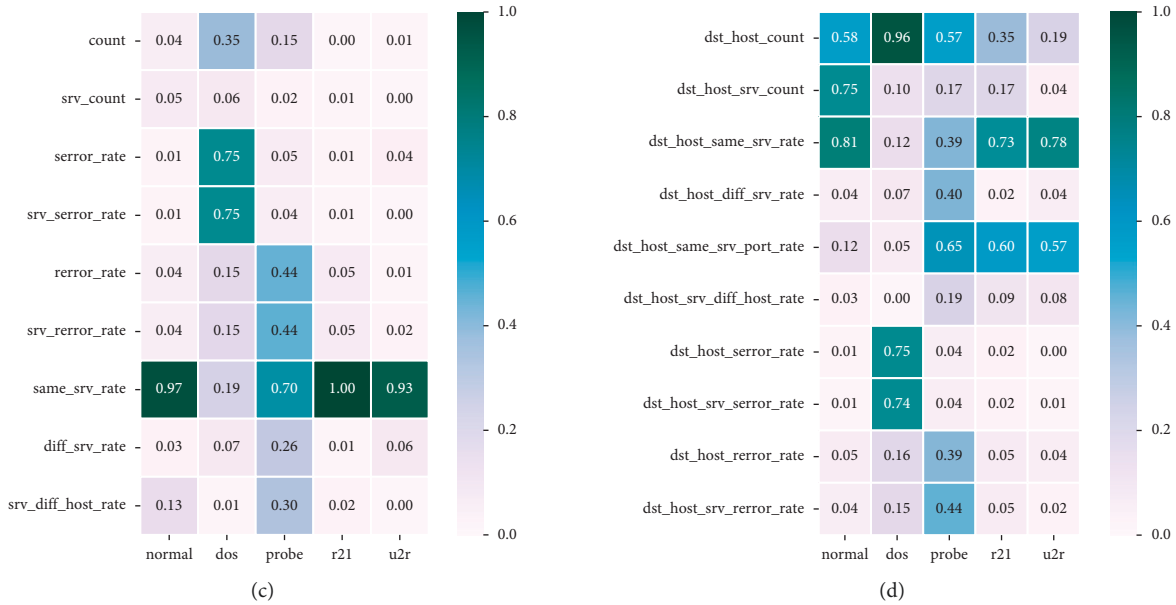


FIGURE 10: Heat map distribution of different traffic features. (a) Intrinsic feature. (b) Content feature. (c) Time-based feature. (d) Host-based feature.

xlock, and xsnoop) and 555 traffic samples; U2R includes 4 types of unknown attacks (i.e., httptunnel, ps, sqlattack, and xterm) and 163 traffic samples. The results of the whole experiment are shown in Table 12.

Among them, the F_1-score of the proposed IDS in this paper is 0.8783, which is higher than other IDSs based on machine learning and deep learning. This F_1-score of our IDS is 25.82%, 30.82%, 29.79%, 29.18%, 24.54%, and 31.07% higher than KNN, LR, DT, SVM, RF, and AdaBoost. Also, this value is higher than CNN, TextCNN, LSTM, BiLSTM, BiGRU, and CNN-BiLSTM-Att out 16.87%, 12.17%, 16.14%, 7.50%, 12.04%, and 6.37%. This experiment shows that the system in this paper can better detect unknown attacks, and its improvement is better than the previous Table 10. In addition, this paper compares the F_1-score of different categories and puts the obtained ranking results in the fourth column of Table 12. The experimental results show that, except for the DT algorithm, other systems' detection and ranking results are Probe, DoS, U2R, and R2L. In short, our IDS has a good performance in both semantic feature extraction and malicious detection of unknown attacks.

5. Conclusions

This paper proposed and implemented an enhanced intrusion detection system based on a knowledge graph and CNN-BiLSTM-attention. Our IDS combines knowledge graph-based feature extraction and statistical analysis-based feature extraction, which can effectively extract the contextual semantic relationship and crucial features of IoT network traffic. The model has better accuracy and robustness. In particular, the proposed system extracts the key features of normal and abnormal requests (including DoS, Probe, R2L, and U2R attack types) in detail, ensuring robust detection and identifying the attack types (including unknown attacks) of network requests in real-time. We demonstrated that the feature extraction algorithm based on knowledge graph and multiviews fusion could accurately extract key traffic features and have certain interpretability. Extensive experiments showed that our IDS could effectively detect various attacks on IoT networks. It achieved a precision of 90.35%, a recall of 91.07%, and an F_1-score of 90.71%, which outperformed state-of-the-art systems. Moreover, the F_1-score of our system is 10.29% higher than that of the best machine learning algorithm (random forest) and 5.46% higher than that of the compared deep learning algorithm (CNN-BiLSTM-attention). The accuracy of our system is 0.9001, which is 14.71% higher than the average of the six machine learning models and 8.50% higher than the average of the other six deep learning models. In particular, our IDS can identify unknown attack types with small samples, the recognition performance of DoS attack and Probe attack is better than other systems, and their F_1-score is 0.9273 and 0.8849. In short, our system can detect various stealthy attack types (including DoS, Probe, R2L, and U2L) and extract semantic relationships among features.

In the future, on the one hand, we will construct a network intrusion detection system based on knowledge graphs and deep learning for larger-scale network traffic, which can realize real-time monitoring of malicious traffic on enterprise IoT networks. On the other hand, we will combine the advantages of graph neural network and provenance graph to optimize our neural network model in this paper, thereby building a more robust intrusion detection system to identify encrypted or obfuscated malicious network traffic.

Appendix

Figure 5 is a knowledge graph, which is constructed by the training set of normal network requests. Among them, the closely related feature relationship pairs mainly cover features such as src_bytes, same_srv_rate, dst_host_same_srv_rate, and SF.

The key features extracted based on statistical analysis are shown in Table 8. Each category extracts two essential features, and each feature is represented by a two-tuple < feature, diff>, corresponding to the key features and the average difference of this feature's category. The calculation process is shown in the previous equation (4). For example, the time-based feature of the type of DoS attack is error_rate, and the average difference is 0.4690.

In particular, the authors calculate the average value of different (numerical) features. The heat map distribution is shown in Figure 10. The authors can see the key features of different attack types. The statistical analysis results of the whole distribution are basically consistent with those in Table 8. For example, src_bytes, logged_in, same_srv_rate, and dst_host_same_srv_rate features play an important role in detecting different types of network requests.

Data Availability

The dataset used in this study is free and publicly available on the Internet. We can get the NSL-KDD dataset through the following link: https://www.unb.ca/cic/datasets/nsl.html.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was in part supported by the National Natural Science Foundation of China under Grant Nos. 62172308, U1626107, 61972297, and 62172144.

References

[1] M. A. Al-Garadi, A. Mohamed, A. K. 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. 22, no. 3, pp. 1646–1685, 2020.

[2] 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.

[3] Wikipedia. 2021. Stuxnet, 2021, https://en.wikipedia.org/wiki/Stuxnet.

[4] Wikipedia, *Ukraine Power Grid Hack*, 2021, https://en.wikipedia.org/wiki/Ukraine_power_grid_hack.

[5] Wikipedia, *2020 United States Federal Government Data Breach*, 2021, https://en.wikipedia.org/wiki/2020_United_States_federal_government_data_breach.

[6] Wikipedia, *Colonial Pipeline Ransomware Attack*, 2021, https://en.wikipedia.org/wiki/Colonial_Pipeline_ransomware_attack.

[7] T. Shekari, C. Bayens, M. Cohen, L. Graber, and amd R. Beyah, "RFDIDS: radio frequency-based distributed intrusion detection system for the power grid," *Proceedings of the 26th Network and Distributed System Security Symposium*, San Diego CA, USA, 2019.

[8] Y. Luo, Y. Xiao, L. Cheng, G. Peng, and D. Yao, "Deep learning-based anomaly detection in cyber-physical systems: progress and opportunities," *ACM Computing Surveys*, vol. 54, no. 5, pp. 1–36, 2021.

[9] A. Alsaheel, Y. Nan, S. Ma et al., "ATLAS: a sequence-based learning approach for attack investigation," *Proceedings of the 30th USENIX Security Symposium*, pp. 3005–3022, August 2021.

[10] Y. Tang, Y. Wang, H. Li, and X. Li, "To cloud or not to cloud: an on-line scheduler for dynamic privacy-protection of deep learning workload on edge devices," *CCF Transactions on High Performance Computing*, vol. 3, no. 1, pp. 85–100, 2021.

[11] Y. Duan, X. Li, J. Wang, and H. Yin, "DeepBinDiff: learning program-wide code representations for binary diffing," *Proceedings of the 27th Network and Distributed System Security Symposium*, San Diego CA, USA, 2020.

[12] W. Meng, W. Li, L. Jiang, K.-K. R. Choo, and C. Su, "Practical bayesian poisoning attacks on challenge-based collaborative intrusion detection networks," *Lecture Notes in Computer Science*, pp. 493–511, Luxembourg, 2019.

[13] P. Hadem, D. K. Saikia, and S. Moulik, "An SDN-based intrusion detection system using SVM with selective logging for IP traceback," *Computer Networks*, vol. 191, Article ID 108015, 2021.

[14] W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "CANN: an intrusion detection system based on combining cluster centers and nearest neighbors," *Knowledge-Based Systems*, vol. 78, pp. 13–21, 2015.

[15] A. J. Malik and F. A. Khan, "A hybrid technique using binary particle swarm optimization and decision tree pruning for network intrusion detection," *Cluster Computing*, vol. 21, no. 1, pp. 667–680, 2018.

[16] S. Masarat, S. Sharifian, and H. Taheri, "Modified parallel random forest for intrusion detection systems," *The Journal of Supercomputing*, vol. 72, no. 6, pp. 2235–2258, 2016.

[17] M. Mazini, B. Shirazi, and I. Mahdavi, "Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms," *Journal of King Saud University - Computer and Information Sciences*, vol. 31, no. 4, pp. 541–553, 2019.

[18] C. Liu, Z. Gu, and J. Wang, "A hybrid intrusion detection system based on scalable k-means+ random forest and deep learning," *IEEE Access*, vol. 9, pp. 75729–75740, 2021.

[19] S. Ahn, H. Yi, Y. Lee, W. R. Ha, G. Kim, and Y. Paek, "Hawkware: network intrusion detection based on behavior analysis with ANNs on an IoT device," *Proceedings of the 57th Design Automation Conference*, pp. 1–6, San Francisco, CA, USA, 2020.

[20] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986–2998, 2016.

[21] C. Fu, Q. Li, M. Shen, and K. Xu, "Realtime robust malicious traffic detection via frequency domain analysis," in *Proceedings of the 27th ACM SIGSAC Conference on Computer and Communications Security*, pp. 3431–3446, Virtual Event, Korea, 2021.

[22] J. B. D. Caberera, B. Ravichandran, and R. K. Mehra, "Statistical traffic modeling for network intrusion detection,"

Proceedings of the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 466–473, San Francisco, California, USA, 2000.

[23] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, 1987.

[24] T. F. Lunt and R. Jaganna, "A prototype real-time intrusion-detection expert system," *Proceedings of the 9th IEEE Symposium on Security and Privacy*, pp. 59–66, Oakland, California, USA, 1988.

[25] K. Borders, J. Springer, and M. Burnside, "Chimera: a declarative language for streaming network traffic analysis," in *Proceedings of the 21st USENIX Security Symposium*, pp. 365–379, Bellevue, WA, USA, 2012.

[26] H. Li, H. Hu, G. Gu, G. Ahn, and F. Zhang, "vNIDS: towards elastic security with safe and efficient virtualization of network intrusion detection systems," *Proceedings of the 25th ACM SIGSAC Conference on Computer and Communications Security*, pp. 17–34, Toronto, ON, Canada, 2018.

[27] H. Haugerud, H. N. Tran, N. Aitsaadi, and A. Yazidi, "A dynamic and scalable parallel Network Intrusion Detection System using intelligent rule ordering and Network Function Virtualization," *Future Generation Computer Systems*, vol. 124, pp. 254–267, 2021.

[28] W. Li, W. Meng, and M. H. Au, "Enhancing collaborative intrusion detection via disagreement-based semi-supervised learning in IoT environments," *Journal of Network and Computer Applications*, vol. 161, Article ID 102631, 2020.

[29] R. Bitton and A. Shabtai, "A machine learning-based intrusion detection system for securing remote desktop connections to electronic flight bag servers," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1164–1181, 2021.

[30] X. Li, M. Zhu, L. T. Yang et al., "Sustainable ensemble learning driving intrusion detection model," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1591–1604, 2021.

[31] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[32] N. Gupta, V. Jindal, and P. Bedi, "LIO-IDS: handling class imbalance using LSTM and improved one-vs-one technique in intrusion detection system," *Computer Networks*, vol. 192, Article ID 108076, 2021.

[33] Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, "A bidirectional LSTM deep learning approach for intrusion detection," *Expert Systems with Applications*, vol. 185, pp. 1–12, 2021.

[34] P. Sun, P. Liu, Q. Li et al., "DL-IDS: extracting features using CNN-LSTM hybrid network for intrusion detection system," *Security and Communication Networks*, vol. 2020, pp. 1–11, Article ID 8890306, 2020.

[35] D. Li, L. Deng, M. Lee, and H. Wang, "IoT data feature extraction and intrusion detection system for smart cities based on deep migration learning," *International Journal of Information Management*, vol. 49, pp. 533–545, 2019.

[36] N. Balakrishnan, A. Rajendran, D. Pelusi, and V. Ponnusamy, "Deep Belief Network enhanced intrusion detection system to prevent security breach in the Internet of Things," *Internet of Things*, vol. 14, Article ID 100112, 2021.

[37] M. Mahdavisharif, S. Jamali, and R. Fotohi, "Big data-aware intrusion detection system in communication networks: a deep learning approach," *Journal of Grid Computing*, vol. 19, no. 4, pp. 1–28, 2021.

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

[39] M. A. Ferrag, L. A. 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.

[40] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *Proceedings of the 25th Network and Distributed System Security Symposium*, San Diego CA, USA, 2020.

[41] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "A hierarchical hybrid intrusion detection approach in IoT scenarios," *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pp. 1–7, 2020.

[42] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 53–58, 2009.

[43] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: mobile encrypted traffic classification using multimodal deep learning," *Computer Networks*, vol. 165, Article ID 106944, 2019.

[44] Z. Bu, B. Zhou, P. Cheng, K. Zhang, and Z.-H. Ling, "Encrypted network traffic classification using deep and parallel network-in-network models," *IEEE Access*, vol. 8, pp. 132950–132959, 2020.

[45] UNB NSL-KDD Datasets: 2020, https://www.unb.ca/cic/datasets/nsl.html.