A hybrid intrusion detection system with K-means and CNN+LSTM

Haifeng Lv[1], Yong Ding[2,*]

[1]Guangxi Key Laboratory of Machine Vision and Intelligent Control, WuZhou University, Wuzhou, China
[2]Guangxi Key Laboratory of Cryptography and Information Security, School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin, China,

Abstract

Intrusion detection system (IDS) plays an important role as it provides an efficient mechanism to prevent or mitigate cyberattacks. With the recent advancement of artificial intelligence (AI), there have been many deep learning methods for intrusion anomaly detection to improve network security. In this research, we present a novel hybrid framework called KCLSTM, combining the K-means clustering algorithm with convolutional neural network (CNN) and long short-term memory (LSTM) architecture for the binary classification of intrusion detection systems. Extensive experiments are conducted to evaluate the performance of the proposed model on the well-known NSL-KDD dataset in terms of accuracy, precision, recall, F1-score, detection rate (DR), and false alarm rate (FAR). The results are compared with traditional machine learning approaches and deep learning methods. The proposed model demonstrates superior performance in terms of accuracy, DR, and F1-score, showcasing its effectiveness in identifying network intrusions accurately while minimizing false positives.

1. Introduction

The digital landscape has significantly paved the way for innovative advancements, transforming communication methods across various technical domains [1]. This growing reliance on internet services across all aspects of life, including military cybersecurity, e-commerce, education, and business, has become a major concern due to the potential intrusion threats that jeopardize the security of individuals and sensitive organizations [2]. Notably, recent security breaches in companies like Yahoo, Exactis, British Airways, and Under Armour have prompted industry professionals and academia to embrace cybersecurity as a captivating research area aimed at safeguarding critical information from devastating cyberattacks. Traditionally, the "first line of defense" has relied on measures such as encryption, antivirus software, decryption, access control, and firewalls to detect intrusions and protect networks from malicious cyberattacks [3]. However, these conventional security technologies often prove inadequate and sometimes fail to shield networks against emerging intrusion techniques [4]. Consequently, addressing these challenges necessitates the adoption of defense-in-depth strategies [5]. In response, researchers have designed efficient intrusion detection systems (IDS).

*Corresponding author. Email: stone_dingy@126.com

IDS monitor networks for malicious activities and provide protection against various cyberattacks that affect the availability, integrity, and confidentiality (AIC) of the network. IDSs encompass different configurations and types, including those that record information, notify security administrators of abnormal activity, and generate reports [6]. IDS software can be installed in different ways depending on the type and source of data analyzed, such as network-based IDS (NIDS), host-based IDS (HIDS), and hybrid or distributed IDS. These software systems detect threats using single or combined methods, including anomaly-based and signature-based approaches [7]. Anomaly-based detection has the advantage of identifying zero-day attacks that may go unnoticed in signature-based methods, but it tends to generate more false positives [8]. On the other hand, signature-based detection is effective for known threats, while anomaly-based detection excels in identifying new threats. In the case of anomaly-based detection, instead of relying on static databases, it examines typical activities and maintains a log of normal data flow patterns. Alerts are generated for activities that deviate from these patterns [9]. Anomaly-based classification methods can be categorized into machine learning (ML)-based, statistical-based, and knowledge-based approaches [10]. ML-based methods employ data mining techniques to automatically develop a model from labeled regular datasets. One restriction is the need for computational resources and time during the initial model deployment, but subsequent analysis is often efficient [11]. In order to improve IDS performance, researchers have investigated the use of Deep Learning-based approaches, a specific branch of artificial intelligence.

In deep learning-based IDS, utilizing advanced and updated datasets is preferable to enhance the effectiveness of intrusion detection. The most recent datasets created especially for intrusion detection, which include a wide range of attack types and features, are frequently used to train these systems. The accuracy of intrusion detection achieved by deep learning architectures tends to increase with more features in the dataset [12]. Deep learning techniques automatically decrease the attribute vector's size to the ideal amount of needed attributes, negating the necessity for attribute extraction or selection procedures [13]. However, in some studies, attributes were selected using ML models before applying deep learning techniques to achieve improved performance [14]. The outcomes of the model suggested using the NSL-KDD dataset [15] were further examined, which frequently used by scholars. The NLS-KDD dataset, derived from the original KDD Cup 1999 dataset [16], represents a comprehensive collection of network traffic data with labeled instances of normal and anomalous behavior. However, the dataset poses challenges due to its class imbalance, high dimensionality, and complex patterns inherent in real-world network traffic.

To address these challenges, the proposed hybrid model leverages the K-means clustering algorithm to create clusters of network traffic instances, enabling a more nuanced analysis of the data. By distinguishing between normal and anomalous instances at the cluster level, the system can effectively identify outliers and potential intrusions. To capture both spatial and temporal dependencies in the network traffic data, a hybrid deep learning architecture combining CNN and LSTM models is introduced. The CNN component extracts spatial features from the network traffic samples, allowing the system to identify local patterns and anomalies. The LSTM component, on the other hand, captures temporal patterns and long-term dependencies, enabling the system to analyze the sequential behavior of the network traffic. This fusion of spatial and temporal analysis enhances the system's ability to detect sophisticated attacks that may involve both localized and sequential anomalies.

The primary objective of this research is to evaluate the effectiveness of the proposed hybrid model on the NLS-KDD dataset. Extensive experiments are conducted to assess the system's performance in terms of accuracy, DR, and F1-score, comparing it with common machine learning approaches. The results indicate the superiority of the hybrid model in accurately identifying network intrusions while minimizing false positives, thus showcasing its potential for practical deployment in real-world security scenarios.

The contributions of this paper lie in the development of the novel hybrid KCLSTM model is summarized as follows:

(1)The frequency of cyber-attacks on industrial network traffic has been increasing steadily, resulting in both monetary and non-monetary losses. As a result, it is crucial to identify these attacks and secure the system. This study implements intrusion detection systems using K-means, CNN, and LSTM models.

(2)Moreover, the proposed hybrid approach for IDS underwent testing using the NSL-KDD dataset, which is known for its large volume of data and high number of attributes. To assess the performance of our proposed model, we implemented a binary classification process on current datasets, resulting in a more dependable and accessible procedure.

(3)Lastly, a comparison was made between our proposed KCLSTM model and those presented in previous studies. The accuracy for intrusion detection achieved by our proposed model were significantly higher than the results obtained in previous studies. These findings confirm that the hybrid KCLTSM model enhances the performance of IDS.

The remainder of this paper is organized as follows: Section 2 provides a comprehensive review of related work in intrusion detection and the application of clustering and deep learning techniques. Section 3 details the methodology of the proposed hybrid KCLSTM model, including data preprocessing, K-means clustering, and the CNN+LSTM architecture. Section 4 presents the experimental setup and analyzes the performance of the system on the NLS-KDD dataset. Finally, Section 5 concludes the paper by summarizing the findings, discussing the implications of the research, and suggesting avenues for future work.

2. Related work

This section provides a summary of IDS generated through machine and deep learning methods, along with the results obtained from related studies. Furthermore, an evaluation is conducted on the various models and architectures implemented previously for intrusion detection. This section furnishes a summary of IDS generated through machine and deep learning methods, as well as the outcomes of related researches. Additionally, we evaluate several methods that were formerly implemented for intrusion detection. The NSL-KDD dataset is constructed by selecting a subset of the KDD Cup 1999 dataset and pre-processing it to remove duplicate records, correct labeling errors, and balance the class distribution. The dataset consists of network traffic instances, each characterized by a set of features that capture various aspects of network communication. These features are categorized into three types: basic features, content-based features, and traffic features.

A trustworthy intrusion detection model was created by Zhou et al. [17] using the CFS-BA approach and an ensemble classifier. Their suggested solution uses an ensemble classifier made up of C4.5, Random Forest (RF), and Penalized Attribute Forest (Forest PA) to identify the most pertinent features based on feature correlation. The final classification is then made using a voting method. The experiment used the NSL-KDD dataset, and their model produced accuracy and DR of 87.37% and 87.4%, respectively.

In a different study [18], a paradigm for preventing ransomware attacks on devices in Industrial Internet of Things (IIoT) networks was put out. There are two sections to the model. Data is cleaned up using an auto-encoder in the first section to ensure better representation. The second part enables deep neural network-based intrusion detection and identification. The suggested model was examined individually using the NSL-KDD, ISOT, and X-IIoTID datasets. The findings of the study show that the suggested methodology successfully attained a high detection rate for targeted ransomware on devices in IIoT networks.

An IDS for intrusion detection utilizing the genetic algorithm for attribute selection was suggested in another work [19]. The fitness function for the genetic algorithm was the Random Forest model. Classification techniques such Extra Trees, Naive Bayes, Random Forest, Linear Regression, Extreme Gradient Boosting, and Decision Tree were used to detect intrusions. Ten feature vectors were produced for binary classification and seven for multi-class classification by the genetic algorithm-random forest model. The implementation produced an area under the curve of 0.98 and a test accuracy of 87.61% for binary classification using the UNSW-NB15 dataset. These results were produced by a genetic algorithm-random forest model with sixteen features.

An IoT-based intrusion detection system with heuristic feature selection was proposed by Liu et al. [20]. The particle swarm optimization technique employed in the research was based on the Light Gradient Boosting Machine (LightGBM) algorithm, while the Support Vector Machine was used to classify incursions. The UNSW-NB15 dataset was utilized to test the suggested models, and the accuracy value and false alarm rate performance metrics were employed. The accuracy rate of the suggested model was 86.68%, and the false alarm rate was 10.62%. These findings suggest that, in comparison to other studies in the literature, the proposed model had a greater false alarm rate for binary classification. The model was not used by the researchers in the multi-class classification procedure.

For massive data systems utilized in the industrial sector, Zhou et al. [21] suggested an intrusion detection system based on Variational Long Short-Term Memory (VLSTM). For feature extraction from the complicated and big dataset, the model rebuilt features and underwent a selection procedure among the new features produced using an autoencoder. This implementation employed the UNSW-NB15 dataset, and the performance of the suggested model was assessed using metrics for precision, recall, false alarm rate, F1-Score, and the area under the curve. The area under the curve, recall, precision, and F1-Score values for the VLSTM approach were 0.895, 97.8%, 86.0%, and 90.7% respectively. These values exceed those that were stated in various studies from the literature. However, the researchers acknowledged that uneven distributions of attack types in the dataset used in this study could be addressed in future implementations to obtain better results.

A proposed IDS based on an Extreme Learning Machine was made by Gao et al. [22]. The Extreme Learning Machine was presented with the chosen features for classification using the model's adaptive principal component. A multi-class classification method was carried out for both the KDD and UNSW-NB15 datasets in order to test the suggested model on them. The performance metric was the accuracy rate as determined by the test data. In comparison to earlier research, the suggested model outperformed them with accuracy rates of 81.22% for the NSL-KDD dataset and 70.51% for the UNSW-NB15 dataset. To improve the accuracy rate in actual industrial control systems, the authors highlighted the need for more research.

Researchers suggested an IDS employing Deep Neural Networks (DNNs) in a different study [23] to quickly identify new threat types and work with different platforms. Six distinct datasets were used to assess the performance of the proposed model, including Kyoto, CICIDS 2017, KDD-Cup99, NSL-KDD, UNSW-NB15, and WSN-DS. Using the updated NSL-KDD dataset as a benchmark, the deep neural network scored an F1-Score of 79.7%, a recall of 96.3%, a precision of 68.0%, and an accuracy of 78.9% in binary classification. In contrast to binary classification, the deep neural network performed worse in multi-class classification, with F1-Score of 76.5%, accuracy of 78.5%, precision of 81.0%, and recall of 78.5%.

A novel intrusion detection method was suggested by Mushtaq et al. [24] using a hybrid architecture that combines a deep auto-encoder (AE) with the LSTM and the BiLSTM for categorization into normal and anomalous data. On the well-known dataset NSL-KDD, the proposed model is assessed in terms of error indices such as precision, recall, F-score, accuracy, DR, and FAR. In comparison to existing

deep and shallow machine learning techniques, including other recently disclosed methods, the results show that the proposed AE-LSTM performs noticeably better with less prediction error. On the NSL-KDD dataset, AE-LSTM displays classification accuracy of 89% with DR of 89.84% and FAR of 11%, illustrating the improved performance of the suggested model over current state-of-the-art methodologies.

Liu et al. [25] proposes a hybrid IDS that combines machine learning and deep learning algorithms to classify intrusion events. The model uses k-means and random forest algorithms for binary classification, which are parallelized on the Spark platform to improve the speed of data preprocessing and training. The normal and abnormal events are collected by the distributed storage system, HDFS, and sent directly to the driver side after binary classifications. In the third stage, the intrusion detection data stored on the driver side is used for classification. The model also divides aberrant events into various attack types using CNN, LSTM, and other deep learning methods. The unbalanced dataset is addressed via adaptive synthetic sampling (ADASYN). With the use of the NSL-KDD and CIS-IDS2017 datasets, the performance of the suggested model is assessed. According to the experimental findings, the suggested model has a higher True Positive Rate (TPR) for the majority of attack events, a quicker data preprocessing rate, and perhaps a shorter training period.

Another a study [26] proposed a novel 5-layer Autoencoder (AE)-based model for network anomaly detection tasks. The paper emphasizes the importance of network anomaly detection as an effective mechanism to block or stop cyberattacks. The authors extensively investigate several performance indicators of an AE model to understand the critical impacts of the core set of important performance indicators and the detection accuracy. The proposed model utilizes a new data pre-processing methodology that removes the most affected outliers from the input samples to reduce model bias caused by data imbalance across different data types in the feature set. The paper also discusses the use of different reconstruction loss functions and their sensitivity to the detection accuracy. The proposed model outperforms other similar methods, achieving the highest accuracy and F1-Score of 90.61% and 92.26%, respectively, in detecting the NSL-KDD dataset.

Vinayakumar et al. [27] discussed the development of an intelligent IDS using deep learning techniques. The paper evaluated the performance of various machine learning algorithms on publicly available benchmark malware datasets, such as NLS-KDD and proposed an extremely scalable and hybrid DNNs architecture named scale-hybrid-IDS-AlertNet, which can be applied in real-time to efficiently monitor network traffic and host-level events to preventatively detect potential threats. Overall, this article advances the field of cyber security by offering a thorough assessment of DNNs and other machine learning classifiers on several benchmark malware datasets that are made publically available.

Patil et al. [28] explores the use of majority voting and feature selection techniques to improve detection accuracy. Similarly, Venkateswaran et al. [29] focuses on neuro deep learning methods for wireless intrusion detection system that distinguishes the attacks in MANETs. Additionally, Singh et al. [30] demonstrates the application of machine learning in social media analysis. Other notable works include [31-34] provide valuable insights into various aspects of IDS and related fields. You et al. [31] introduced a Two-Layer Evolutionary Framework (TLEF) for t-closeness anonymization, balancing data utility and privacy using evolutionary algorithms. Yin et al. [32] proposed a framework using heterogeneous graphs to predict software vulnerabilities' exploitability, enhancing vulnerability prioritization through integrated data sources. Ge et al. [33-34] explored evolutionary approaches for database partitioning and data publishing privacy. In [33], they optimized database partitioning to balance privacy and utility, allowing real-time adjustments. In [34], they introduced a cooperative coevolutionary framework to address data publishing privacy and transparency, promoting collaboration among entities to protect sensitive information while maintaining transparency. Their coevolutionary approach ensures the system evolves to meet dynamic privacy and transparency needs.

Papalkar et al. [35-38] explored various techniques to enhance attack detection capabilities in IoT and cloud computing environments across a series of studies. They proposed a hybrid CNN approach specifically designed for detecting unknown attacks in edge-based IoT networks [35]. Additionally, they analyzed various defense techniques against DDoS attacks in IoT environments, providing a comprehensive overview of existing strategies and highlighting the need for more robust and adaptive solutions [36]. To optimize the efficiency of DDoS attack detection, they developed an optimized feature selection method to guide lightweight machine learning models in cloud computing environments, aiming to enhance detection efficiency while minimizing computational overhead [37]. They also discussed various deep learning techniques for detecting unknown attacks, emphasizing the potential of deep learning models in identifying novel and sophisticated threats that traditional methods may miss [38]. Hamadouche et al. [39] combined lexical, host, and content-based features to detect phishing websites using machine learning models. It highlights the importance of multi-feature integration for improving the accuracy of phishing detection systems. The above works highlight the need for innovative solutions like the proposed KCLSTM model.

3. Approaches

3.1 Motivation

The increasing complexity and frequency of cyberattacks necessitate advanced IDS capable of accurately identifying malicious activities. Traditional IDS methods often struggle with high-dimensional data and class imbalance issues, which can lead to high false alarm rates and missed detections. To address these challenges, we propose a hybrid framework, KCLSTM, that combines K-means clustering with CNN and LSTM architectures. This approach leverages the strengths of clustering and deep learning to enhance the detection of sophisticated network intrusions, making it a robust solution for modern cybersecurity threats.

Figure.1 shows the intrusion detection framework proposed in this paper which consists of two stages. The first stage processed the original intrusion detection dataset, containing the deletion of invalid data, the digitization of categorical features and the normalization of digital features. The second stage is to classify abnormal and normal network traffic based on a hybrid model with K-means and CNN+LSTM. The model framework is shown in Figure 1.
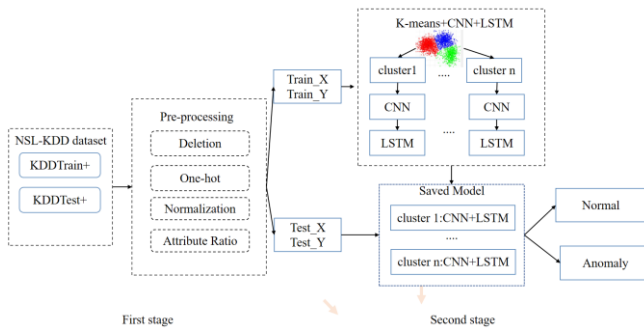

Figure 1. Intrusion detection framework of the proposed KCLSTM model

3.2. K-means Algorithm

A well-liked unsupervised machine learning approach [40] called "K-means clustering" is used to group data points into k clusters according to how similar they are to one another in the feature space. The approach aims to reduce the sum of squared distances between the data points and the cluster centroids that are allocated to them. The cluster assignments and centroids don't change during the convergence phase of the iterative process. Given a dataset of n data points $\{x_1, x_2,..., x_n\}$, the algorithm aims to group the data points into k clusters $\{c_1, c_2,..., c_k\}$, where each cluster contains a set of data points and a centroid μ_k. The objective function to be minimized is the sum of squared distances between each data point x_i and its assigned centroid μ_j

$$J = \sum_{i=1}^{n} \left\| x_i - \mu_j \right\|^2, \qquad (1)$$

where $\|.\|$ denotes the Euclidean distance.

The algorithm seeks to minimize this objective function by iteratively updating the assignment of each data point to its nearest centroid and recalculating the centroid of each cluster. The algorithm works as follows:

(1) Initialize k cluster centroids randomly.

(2) Assign each data point to the nearest cluster centroid based on the Euclidean distance between the data point and each centroid.

(3) Recalculate the centroid of each cluster by taking the mean of all the data points assigned to that cluster.

(4) Repeat steps (2) and (3) until convergence is reached.

3.3. Convolutional Neural Network

Convolutional Neural Network (CNN) [41] is a type of deep learning model that has been widely used in image and anomaly detection analysis tasks. CNN consist of multiple layers, including several convolutional layers and pooling layers, which are designed to extract and process features from the input attribute.

The convolutional layer is the most important layer in CNN. It applies a set of learnable filters (also called kernels or weights) to the input attribute to produce a set of output feature maps. The filters are typically small in size (e.g.,3×3 or 5×5) and slide over the input attribute in a systematic way, computing dot products between the filter weights and the corresponding values in the input attribute. The output feature maps capture different types of local input attribute patterns, such as edges, corners, and blobs, and they can be interpreted as a set of high-level features that represent the input attribute. The output feature maps are typically passed through an activation function (e.g., ReLU) to introduce non-linearity and enhance the discriminative power of the features. The mathematical formula for the convolutional layer can be expressed in Eq.2. The x_i^L represents the attribute i map of convolution layer L, while σ denotes the activation function. k_i refers to the input attribute set of layer (L-1), w_{ji}^L represents the connection weight between attribute i of convolution layer L and attribute j of layer (L-1), and b_j^L is the deviation in the related layer.

$$x_i^L = \sigma \left(\sum_{i \in k_i} x_j^{L-1} * w_{ji}^L + b_j^L \right), \qquad (2)$$

After the convolution layer comes the pooling layer. The pooling layer's goal is to minimize the attribute map's size. By doing this procedure, it is ensured that key qualities are identified, the complexity of the data is reduced, and the

network's resistance to environmental changes is increased. The pooling layer can be shown as shown in Eq. 3.

$$x_i^L = \sigma\left(\beta_i^L c\left(x_i^{L-1} + b_j^L\right)\right), \tag{3}$$

The sub-sampling function is denoted by the symbol c in this example, while the weighting matrix is denoted by the symbol β. A well-liked unsupervised machine learning approach called "K-means clustering" is used to group data points into k clusters according to how similar they are to one another in the feature space. The approach aims to reduce the sum of squared distances between the data points and the cluster centroids that are allocated to them. The cluster assignments and centroids don't change during the convergence phase of the iterative process.

3.4. Long Short-Term Memory

Long Short-Term Memory (LSTM) network is a type of recurrent neural network (RNN) that is designed to handle the vanishing gradient problem and capture long-term dependencies in sequential data. Architecture of LSTM is graphically shown in Figure. 2. LSTM consist of a memory cell and three gates: forget gate, input gate, and output gate, which control the flow of information in and out of the cell.
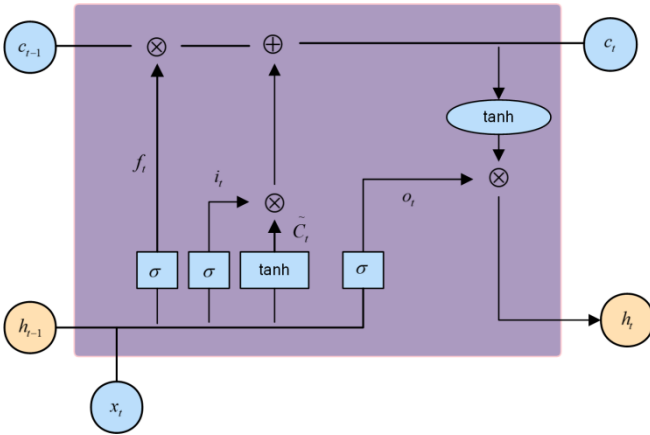

Figure 2. Architecture of LSTM

The forget gate is used to selectively discard information from the memory cell that is no longer relevant. It takes as input the previous hidden state and the current input, and produces a forget vector that indicates which information to keep and which to discard from the cell state. The mathematical formula for the forget gate can be expressed in Eq.4.

$$f_t = \sigma\left(\mathbf{W}_f * [h_{t-1}, x_t] + b_f\right), \tag{4}$$

where σ is the sigmoid activation function, $\mathbf{W}_f$ is the weight matrix, h_{t-1} is the previous hidden state, x_t is the current input, and b_f is the bias vector.

The input gate is used to selectively update the memory cell with new information from the current input. It takes as input the previous hidden state and the current input, and produces an input vector that indicates which information to add to the cell state. The mathematical formula for the input gate can be expressed in Eq.5 and Eq.6.

$$i_t = \sigma\left(\mathbf{W}_i * [h_{t-1}, x_t] + b_i\right), \tag{5}$$

$$\tilde{C}_t = tanh\left(\mathbf{W}_C * [h_{t-1}, x_t] + b_C\right), \tag{6}$$

where i_t is the input gate vector, $\tilde{C}_t$ is the temporary cell state vector, σ is the sigmoid activation function, $\mathbf{W}_i$ and $\mathbf{W}_C$ are the weight matrices, b_i and b_C are the bias vector.

The cell state is the internal memory of the LSTM network. It stores the information that is relevant for the current task and is updated by the input and forget gates. Eq. 7 is a mathematical expression that may be used to represent the cell state.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t, \tag{7}$$

where C_t is the current cell state, C_{t-1} is the previous cell state, f_t is the forget gate vector, i_t is the input gate vector, and $\tilde{C}_t$ is the temporary cell state vector.

The output gate is used to selectively output information from the memory cell. It takes as input the previous hidden state and the current input, and produces an output vector that indicates which information to output from the cell state. The mathematical formula for the output gate can be expressed in Eq.8 and Eq.9.

$$o_t = \sigma\left(\mathbf{W}_o * [h_{t-1}, x_t] + b_o\right), \tag{8}$$

$$h_t = o_t * tanh\left(C_t\right), \tag{9}$$

where o_t is the output gate vector, h_t is the current hidden state, σ is the sigmoid activation function, $\mathbf{W}_o$ and b_o are weight and bias matrices, respectively.

3.5. The Proposed KCLSTM Model :K-means+CNN+LSTM

The proposed hybrid KCLSTM model used the NSL-KDD dataset to identify network abnormalities by combining the K-means clustering algoritrhm with CNN and LSTM networks. Figure. 1 presents the conceptual diagram of the developed technique. The latter subsections included descriptions of the specifics.

3.5.1 Dataset Description
The NSL-KDD dataset was used to evaluate the validity of the proposed model in this study and was divided into two subsets: KDDTrain+ and KDDTest+. We employed KDDTrain+ and KDDTest+ for training and testing the KCLSTM model, respectively. To focus on the main performance metrics, we classified the traffic samples in both subsets into two categories: normal and anomaly. As shown in Table 1, the KDDTrain+ dataset contains a total of 125,973 records, with 67,343 labeled as normal and 58,630 labeled as anomaly. Similarly, the KDDTest+ dataset has 22,544 records, with 9,711 labeled as normal

and 12,833 labeled as anomaly. The NSL-KDD dataset comprises one label and 41 features.

Table 1. Normal and anomaly records distribution on the NSL-KDD dataset.

NSL-KDD	Total	Normal	Anomaly
KDDTrain+	125973	67343	58630
KDDTest+	22544	9711	12833

The NSL-KDD dataset has four classes of features, including basic features, Content features, Time based traffic features and Connection based features, as shown in Figure 3. The features with character values cannot be used directly as input, instead numeric features must be selected. Several features in the NSL-KDD dataset have big deviations. The classification results tend to be affected by the features with big deviations. Thus, it is necessary to digitize and normalize the input sample dataset. The NSL-KDD dataset consists of network traffic instances, each characterized by a set of features that capture various aspects of network communication. These features are categorized into four types: basic features, content-based features, time based traffic features and connection based traffic features, as shown Figure.3 .

The basic features include attributes like the duration of the connection, the protocol type (e.g., TCP, UDP), the service associated with the connection (e.g., ftp, http), and the source and destination IP addresses and port numbers. The content-based features are derived from the packet payload and include attributes like the number of failed login attempts, the number of root accesses, and the number of file creations. These features provide insights into the actual content of the network communication. The traffic features capture statistical information about the network traffic, such as the number of connections to the same host in the past two seconds or the number of connections from the same service in the past two seconds.
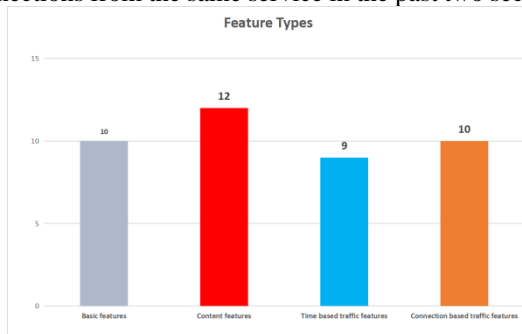

Figure.3 Types of features in the NSL-KDD dataset

3.5.2 Data Preprocessing
The presence of character values in some of the features in the NSL-KDD dataset renders them unsuitable for direct

use as input. Therefore, it is necessary to select only numeric features for this purpose. Additionally, certain features in the dataset exhibit significant deviations, which can adversely impact the classification results. To address this issue, it is essential to preprocess the input sample dataset.

Deletion. The deletion step involves removing any irrelevant or redundant features from the NSL-KDD dataset. This process helps reduce the dimensionality of the dataset and eliminate noise or irrelevant information that may hinder the anomaly detection process. Features that do not contribute significantly to the task at hand are removed, improving computational efficiency and reducing the risk of overfitting, such as the "num_outbound_cmds" attribute .

Digitization. To digitize categorical features in the NSL-KDD dataset, we utilized one-hot encoding. This approach was applied to four categorical features, namely flag, service, and protocol_type. For instance, the protocol_type feature comprised the values ICMP, UDP, and TCP, which were converted into three 3-dimensional binary vectors: [1,0,0], [0,1,0], and [0,0,1], respectively. Thus, the single feature 'protocol_type' was transformed into three features through one-hot encoding. In this experiment, the service feature was converted into digital values ranging from 1 to 70 with a step size of 1, while the flag feature was transformed into digital values within the range of [1,11] using a step size of 1. Subsequently, these values were further processed using the one-hot method.

Normalization. It involves scaling the values of the dataset's numerical features to a standardized range. Normalization prevents certain features from dominating the learning process due to differences in their magnitudes or ranges. It ensures fair comparisons between features during the training and inference stages. Common normalization techniques include Min-Max scaling, where the feature values are linearly transformed to a specified range (e.g., 0 to 1), or Z-score normalization, which standardizes the features by subtracting the mean and dividing by the standard deviation. In this study, we employed min-max normalization to map all characteristic values to the range [0,1]. This normalization technique uses the formula Eq.10, where $x_{m,n}$ denotes the n-th feature of the m-th data, $x_{max,n}$ represents the maximum value of the n-th feature, and $x_{min,n}$ represents the minimum value of the n-th feature.

$$x_{m,n}^{*} = \frac{x_{m,n} - x_{min,n}}{x_{max,n} - x_{min,n}} \ , \qquad (10)$$

Attribute Ratio-Based Feature Selection. After normalization, Attribute Ratio (AR) [42]-based feature selection is applied to the preprocessed NSL-KDD dataset. This process aims to reduce the dimensionality of the dataset by selecting the most relevant and informative features. Attribute ratio-based feature selection considers the information gain and correlation between features and the target variable (i.e., the class labels) to determine their significance. Features with low information gain or low

correlation with the target variable are discarded, as they are considered less informative for the anomaly detection task. This step reduces the number of features, improves computational efficiency, and focuses on the most informative attributes that contribute significantly to the detection of network anomalies. In this paper, We calculate the AR from numeric and binary type using attribute average and frequency for each class. AR may be determined using Equation 11.

$$AR(i) = MAX(CR(j)), \tag{11}$$

Where Class Ratio (CR) is the attribute ratio of each class for attribute i. There are two ways to calculate CR depending on the type of attributes. For numeric attributes, the CR is calculated as the ratio of the sum of attribute i for each class to the total sum of attribute i, the formula is expressed in Eq.12.

$$CR(j) = \frac{AVG(C(j))}{AVG(total)}, \tag{12}$$

For binary attributes, the CR is calculated as the ratio of the number of times attribute i appears in each class to the total number of times attribute i appears in all classes, the formula is expressed in Eq.13.

$$CR(j) = \frac{Frequency(1)}{Frequency(0)}, \tag{13}$$

The CR is used to calculate the Attribute Ratio (AR) for each feature, which is then used to rank the features for feature selection.

3.4.3 Classification based on KCLSTM model

The second stage of our approach involves the use of the proposed KCLSTM model to rapidly classify anomaly events while ensuring the accuracy of the intrusion detection system (IDS). It is crucial to minimize the false negative rate of the IDS and accurately distinguish between normal and anomalous events to ensure the effectiveness of the boundary protection. Therefore, this phase focuses on accurately separating normal and anomalous events to the best of our ability. The validity of the data is a critical factor that determines the accuracy of the IDS. Additionally, Algorithm 1 presents the KCLSTM model's pseudocode.

Algorithm 1: KCLSTM

Input: KDDTrain, KDDTest+, Hyper-Parameters: k, optimizer, learning_rate, filter_size, poolsize, hidden_size, batch_size, epochs, loss_func, and et.al.

Output: predicted labels of the KDDTest+.

```
1.begin:
2.  Pre-processing(KDDTrain, KDDTest+)
3.  kmeans=KMeans(n_clusters=k).fit(Train_X)
4.  kmeans.transform(Test_X)
5.  pred_labels=[]
6.  For cluster in [0...k] do
7.      kcmodel = Sequential()
8.          kcmodel.add(Conv1D,MaxPooling1D,Conv1D,MaxPooling1D,
Dropout)
9.      kcmodel.add(LSTM(hidden_size), Dropout)
10.     kcmodel.add(Dense(hidden_size/2), Dense(1), Dropout)
11.     train_x, train_y, test_x = filter(Train_X, Train_Y, Test_X)[cluster]
12.     kcmodel.compile(loss_func, optimizer)
13.     kcmodel.fit(train_x, train_y, epochs, batch_size)
14.     pred_labels.append(kcmodel.predict(test_x))
15.  Return pred_labels
```

---

4. Experiments

The assessment measures used in our experiment and result analysis are provided in this section.

4.1. Evaluation Metrics

To evaluate the performance of our proposed framework, we utilize evaluation metrics such as accuracy, precision, recall, F1 score, detection rate, and false alarm rate. We label normal samples as class 0 and abnormal samples as class 1. The results are categorized into the following four variables: True Positive (TP), which represents correctly predicted cases for class 1; True Negative (TN), which denotes correctly predicted cases for class 0; False Positive (FP), which represents class 0 cases that are incorrectly predicted as class 1; and False Negative (FN), which denotes class 1 cases that are misjudged as class 0. The mathematical expressions for all evaluation metrics are presented in Table 2.

Table 2. Metrics for evaluation and the formulas underlying them

Metric	Symbol	Mathematical formulae
Precision	P	P=TP/(TP+FP)
Recall	R	R=TP/(TP+FN)
F1-Score	F1	F1 = 2 * (P* R)/(P+R)
Accuracy	ACC	ACC= (TP+TN) /(TP+TN+FP+FN)*100
Detection rate	DR	DR = TP/(TP+FN)
False alarm rate	FAR	FAR = FP/(TN+FP)

4.2 Hyperparameter settings for the KCLSTM model

The proposed KCLSTM model consists of a CNN with two convolutional layers, each followed by a ReLU activation function and max-pooling. The convolutional layers use output filter sizes of 64, 128, respectively. The LSTM component includes two layers with 128 and 64 units, respectively. Hyperparameter tuning was performed using grid search, optimizing parameters such as learning rate, batch size, and dropout rate. This process significantly improved the model's performance, achieving higher accuracy and lower false alarm rates. Hyperparameters were set for the proposed model to achieve optimal results. Table 3 presents the hyper-parameters obtained from the development experiment using the NSK-KDD dataset.

Table 3. Hyper-parameters for KCLSTM model

parameters	settings	parameter description
k	8	The number of K-means clusters
initSteps	25	The number of steps for k-means
maxIter	100	The maximum number of iterations
ar	0.005	Feature selection by Attribute Ratio. If feature importance >ar,then selects this feature
num_lstm	1	Number of lstm layers
num_maxpool	2	Number of maxpooling layers
num_convo	2	Number of convolutional layers
kernel_size	3	specifying the length of the convolution window
filters_size	64,128	the number of output filters in the convolution
hidden_size	128	dimensionality of the output space in LSTM
learning rate	0.001	KCLSTM model learning rate
dropout	0.2	the fraction of neurons dropped out
es_train	0.5	threshold for judgment as anomaly in train dataset
es_test	0.022	threshold for judgment as anomaly in test dataset
optimizer	adam	String (name of optimizer) or optimizer instance

4.3 The performance indices of KCLSTM model

A graphic representation of the proposed KCLSTM performance for several performance indices is shown in Fig. 4. Fig. 4 indicates KCLSTM outperform kmeans-lstm with regard to accuracy (0.933), precision (0.942), recall (0.932), DR (0.932), and F-score (0.937), while accuracy (0.895), recall (0.906), and F-score (0.904) are less for kmeans-lstm.
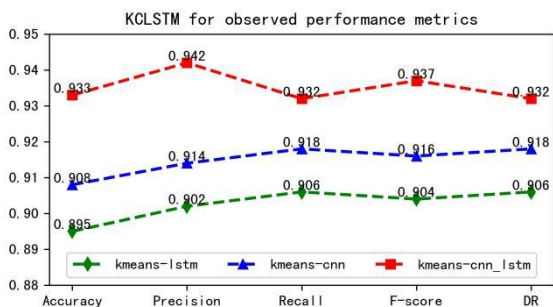

Figure.4 KCLSTM for observed performance metrics

4.3.1 Accuracy and loss curve for KCLSTM model

Figure 5 displays the accuracy and loss plot of the proposed model. The accuracy plot shows that the model was properly trained because the curves eventually reach saturation. The model showed comparable performance on both datasets and did not overfit the training set. As more epochs are added, the loss plot demonstrates that the model performs similarly on both the training and test datasets. It is a sign to stop training if the parallel curves start to diverge.
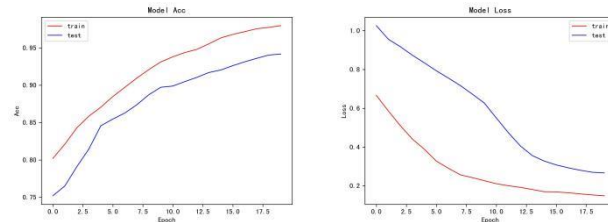

Figure.5 Accuracy and loss plots for train and test data

4.3.2 ROC and PR curve analysis of the KCLSTM

The ROC curve is a performance measurement tool that plots a model's true positive rate on the y-axis and false positive rate on the x-axis in various threshold settings [43]. The ROC and PR curves of the proposed models are shown in Figure 6. From Figure 6, it can be observed that (a) and (b) represent the ROC and PR of the proposed model without k-means, and (c) and (d) show corresponding curves with k-means. It is evident from the ROC curve that cnn_lstm exhibits an AUC of 91.76, but with k-means, the proposed model's AUC substantially improves to 95.64. Likewise, for the PR curve, the proposed model's PR-AUC without k-means is 93.56, but with k-means, it improves to 95.99. Given the dataset's imbalanced nature, the PR curve is a standard metric for evaluating the proposed model's performance. Additionally, the proposed KCLSTM exhibits an excellent PR-AUC of 95.99, indicating its excellent classification ability.
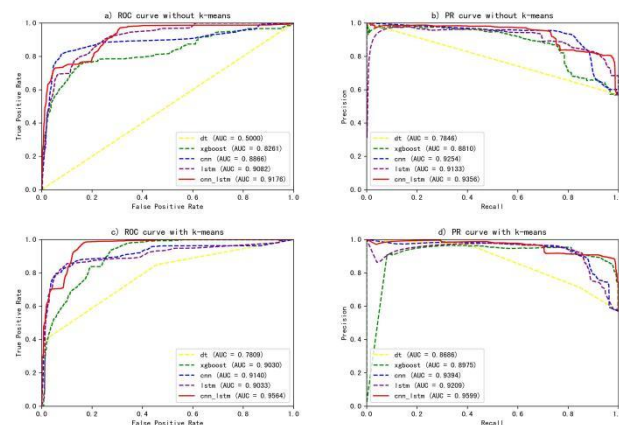
Figure.6 (a)ROC-AUC woutk (b) PR-AUC woutk (c) ROC-AUC withk (d) PR-AUC withk

4.3.3 Performance comparison with K-means assisted baseline models

The output of baseline models using k-means on the chosen feature set is shown in Table 4. Table 4 shows that Decision tree exhibited the lowest performance for all performance indices, including recall 83.6%, accuracy 83.3%, precision 82.7% and F-score 83.1%. Among the other baseline models, CNN demonstrated good performance, with an recall of 91.8%, F-score of 91.6%, accuracy of 90.8%, DR of 91.8%, and precision of 91.4%. In contrast, the proposed KCLSTM achieved a DR of 93.2%, precision of 94.2%, accuracy of 93.3%, F-score of 93.7%, and recall of 93.2%. The results indicate that the proposed KCLSTM outperforms other baseline models significantly.

Table 4. Performance comparison with K-means assisted baseline models

Model	Accuracy	Precision	Recall	F1-score	DR
Decision tree	0.803	0.827	0.836	0.831	0.836
XGBOOST	0.852	0.864	0.874	0.869	0.874
CNN	0.908	0.914	0.918	0.916	0.918
LSTM	0.895	0.902	0.906	0.904	0.906
Ours	**0.933**	**0.942**	**0.932**	**0.937**	**0.932**

4.4 Comparison With Other Models

In this experiment, we trained the proposed model on the NSL-KDD dataset using KDDTrain+ and tested it using KDDTest+. The results of our comparison between the proposed KCLSTM model and other models are shown in Table 5. It is clear from the table that the proposed KCLSTM outperforms all previously published approaches in terms of accuracy. Even while the proposed model has a larger FAR than the AE-LSTM [24], it is more accurate and has a higher DR by 4.28%. KMRF [25] exhibits the best performance among current baseline models, with an accuracy of 92.89%, DR of 98.57%, and FAR of 14.6%. However, the proposed KCLSTM achieves an accuracy of 93.3%, with DR of 93.2% and FAR of 12%. EM-FS [45] performs noticeably better in terms of FAR, however it has substantially worse accuracy than our model. RNN is utilized for intrusion detection in RNN-IDS [48], and it achieves an accuracy of 83.28%, which is lower than our model. Similar to DST-TL [49], which employs a self-taught learning-based IDS and a sparse auto-encoder, it has produced promising results on KDDTest+. Its accuracy and detection rate are lower than those of KCLSTM, though. Bi-LSTM with an attention mechanism was used by BAT-MC [50], another IDS based on deep learning architecture, to achieve an accuracy of 84.25% with 122 features. This is less accurate than our model with fewer features.

Table 5. Performance comparison of the proposed KCLSTM with other models

Model	Feature selection	Classifier	Features	ACC (%)	DR (%)	FAR (%)
FSSL[44]	clustering	FSSL	41	84.12	N/A	N/A
EM-FS[45]	IGR	Bagging(C4.5)	32	84.25	N/A	**2.79**
FSSL-EL[46]	PCA	Ensemble	20	84.54	N/A	5.31
TSE-IDS[47]	hybrid	Two-stage ensemble	37	85.79	86.8	11.7
RNN-IDS[48]	N/A	RNN	122	83.28	N/A	N/A
DST-TL[49]	N/A	Sparse auto-encoder	122	84.60	86	14
BAT-MC[50]	N/A	BiLSTM with attention	122	84.25	N/A	N/A
AELSTM[24]	AE	LSTM	30	89	88	11
KMRF[25]	hybrid	Two-stage ensemble	N/A	92.89	**98.57**	14.6
KCLSTM	hybrid	Two-stage ensemble	77	**93.3**	93.2	12.23

5. Conclusion and future work

This paper presents a hybrid intrusion detection model based on K-means clustering, CNN, and LSTM to address the time-consuming and low-efficiency bottlenecks of IDS. Through comprehensive analysis and evaluation, we demonstrate that integrating these three components yields promising results in accurately detecting network anomalies. The proposed hybrid categorization model was evaluated on the NSL-KDD dataset, and the experimental results indicate that the model can successfully detect anomalous events with a high accuracy of 93.3%, a F1-score of 93.7%, and a DR of 93.2% for both anomalous and normal events. Compared with other intrusion detection models, our proposed model achieves a better detection rate for anomaly events, indicating that the vast majority of anomaly events can be correctly identified. The proposed model also offers advantages in terms of faster data preprocessing.

For limitations and future work, while the KCLSTM model demonstrates promising results, it has certain

limitations. The model's performance may vary with different datasets, and its effectiveness in real-time scenarios needs further evaluation. Future research could focus on optimizing the model for specific network environments and attack patterns, as well as exploring its deployment in live network settings.

Acknowledgements

References

[1] Gauthama Raman M R, Somu N, Jagarapu S, et al. An efficient intrusion detection technique based on support vector machine and improved binary gravitational search algorithm[J]. Artificial Intelligence Review, 2020, 53: 3255-3286.

[2] Zhang J, Ling Y, Fu X, et al. Model of the intrusion detection system based on the integration of spatial-temporal features[J]. Computers & Security, 2020, 89: 101681.

[3] Manzoor I, Kumar N. A feature reduced intrusion detection system using ANN classifier[J]. Expert Systems with Applications, 2017, 88: 249-257.

[4] Wang W, Liu J, Pitsilis G, et al. Abstracting massive data for lightweight intrusion detection in computer networks[J]. Information Sciences, 2018, 433: 417-430.

[5] Marin G A. Network security basics[J]. IEEE security & privacy, 2005, 3(6): 68-72.

[6] Jabez J, Muthukumar B. Intrusion Detection System (IDS): Anomaly detection using outlier detection approach[J]. Procedia Computer Science, 2015, 48: 338-346.

[7] Depren O, Topallar M, Anarim E, et al. An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks[J]. Expert systems with Applications, 2005, 29(4): 713-722.

[8] Gyanchandani M, Rana J L, Yadav R N. Taxonomy of anomaly based intrusion detection system: a review[J]. International Journal of Scientific and Research Publications, 2012, 2(12): 1-13.

[9] Jyothsna V, Prasad R, Prasad K M. A review of anomaly based intrusion detection systems[J]. International Journal of Computer Applications, 2011, 28(7): 26-35.

[10] Wagh S K, Pachghare V K, Kolhe S R. Survey on intrusion detection system using machine learning techniques[J]. International Journal of Computer Applications, 2013, 78(16): 30-37.

[11] Liao H J, Lin C H R, Lin Y C, et al. Intrusion detection system: A comprehensive review[J]. Journal of Network and Computer Applications, 2013, 36(1): 16-24.

[12] Avci İ, Özarpa C. Machine learning applications and security analysis in smart cities[M]//Machine Learning for Smart Environments/Cities: An IoT Approach. Cham: Springer International Publishing, 2022: 183-197.

[13] Zhang P, Wang C, Jiang C, et al. Deep reinforcement learning assisted federated learning algorithm for data management of IIoT[J]. IEEE Transactions on Industrial Informatics, 2021, 17(12): 8475-8484.

[14] Vallathan G, John A, Thirumalai C, et al. Suspicious activity detection using deep learning in secure assisted living IoT environments[J]. The Journal of Supercomputing, 2021, 77: 3242-3260.

[15] Serinelli B M, Collen A, Nijdam N A. Training guidance with kdd cup 1999 and nsl-kdd data sets of anidinr: Anomaly-based network intrusion detection system[J]. Procedia Computer Science, 2020, 175: 560-565.

[16] Tavallaee M, Bagheri E, Lu W, et al. A detailed analysis of the KDD CUP 99 data set[C]//2009 IEEE symposium on computational intelligence for security and defense applications. Ieee, 2009: 1-6.

[17] Zhou Y, Cheng G, Jiang S, et al. Building an efficient intrusion detection system based on feature selection and ensemble classifier[J]. Computer networks, 2020, 174: 107247.

[18] Al-Hawawreh M, Sitnikova E, Aboutorab N. Asynchronous peer-to-peer federated capability-based targeted ransomware detection model for industrial IoT[J]. IEEE Access, 2021, 9: 148738-148755.

[19] Kasongo S M. An advanced intrusion detection system for IIoT based on GA and tree based algorithms[J]. IEEE Access, 2021, 9: 113199-113212.

[20] Liu J, Yang D, Lian M, et al. Research on intrusion detection based on particle swarm optimization in IoT[J]. IEEE Access, 2021, 9: 38254-38268.

[21] Zhou X, Hu Y, Liang W, et al. Variational LSTM enhanced anomaly detection for industrial big data[J]. IEEE Transactions on Industrial Informatics, 2020, 17(5): 3469-3477.

[22] Gao J, Chai S, Zhang B, et al. Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis[J]. Energies, 2019, 12(7): 1223.

[23] Vinayakumar R, Alazab M, Soman K P, et al. Deep learning approach for intelligent intrusion detection system[J]. Ieee Access, 2019, 7: 41525-41550.

[24] Mushtaq E, Zameer A, Umer M, et al. A two-stage intrusion detection system with auto-encoder and LSTMs[J]. Applied Soft Computing, 2022, 121: 108768.

[25] Liu C, Gu Z, Wang J. A hybrid intrusion detection system based on scalable K-Means+ random forest and deep learning[J]. IEEE Access, 2021, 9: 75729-75740.

[26] Xu W, Jang-Jaccard J, Singh A, et al. Improving performance of autoencoder-based network anomaly detection on nsl-kdd dataset[J]. IEEE Access, 2021, 9: 140136-140146.

[27] Vinayakumar R, Alazab M, Soman K P, et al. Deep learning approach for intelligent intrusion detection system[J]. Ieee Access, 2019, 7: 41525-41550.

[28] Patil D R, Pattewar T M. Majority voting and feature selection based network intrusion detection system[J]. EAI Endorsed Transactions on Scalable Information Systems, 2022, 9(6).

[29] Venkateswaran N, Prabaharan S P. An efficient neuro deep learning intrusion detection system for mobile adhoc networks[J]. EAI Endorsed Transactions on Scalable Information Systems, 2022, 9(6).

[30] Singh R, Subramani S, Du J, et al. Antisocial Behavior Identification from Twitter Feeds Using Traditional Machine Learning Algorithms and Deep Learning[J]. EAI Endorsed Transactions on Scalable Information Systems, 2023, 10(4).

[31] You M, Ge Y F, Wang K, et al. TLEF: Two-Layer Evolutionary Framework for t-Closeness Anonymization[C]//International Conference on Web Information Systems Engineering. Singapore: Springer Nature Singapore, 2023: 235-244.

[32] Yin J, Chen G, Hong W, et al. Empowering Vulnerability Prioritization: A Heterogeneous Graph-Driven Framework for Exploitability Prediction[C]//International Conference on Web Information Systems Engineering. Singapore: Springer Nature Singapore, 2023: 289-299.

[33] Ge Y F, Wang H, Bertino E, et al. Evolutionary dynamic database partitioning optimization for privacy and utility[J]. IEEE Transactions on Dependable and Secure Computing, 2023.

[34] Ge Y F, Bertino E, Wang H, et al. Distributed cooperative coevolution of data publishing privacy and transparency[J]. ACM Transactions on Knowledge Discovery from Data, 2023, 18(1): 1-23.

[35] Papalkar R R, Alvi A S. A Hybrid CNN Approach for Unknown Attack Detection in Edge-Based IoT Networks[J]. EAI Endorsed Transactions on Scalable Information Systems, 2024.

[36] Papalkar R R, Alvi A S. Analysis of defense techniques for DDos attacks in IoT–A review[J]. ECS Transactions, 2022, 107(1): 3061.

[37] Papalkar R R, Alvi A S, Ali S, et al. An optimized feature selection guided light-weight machine learning models for DDoS attacks detection in cloud computing[M]//Artificial Intelligence, Blockchain, Computing and Security Volume 1. CRC Press, 2023: 975-982.

[38] Papalkar R R, Alvi A S. Review of unknown attack detection with deep learning techniques[M]//Artificial Intelligence, Blockchain, Computing and Security Volume 1. CRC Press, 2023: 989-997.

[39] Hamadouche S, Boudraa O, Gasmi M. Combining Lexical, Host, and Content-based features for Phishing Websites detection using Machine Learning Models[J]. EAI Endorsed Transactions on Scalable Information Systems, 2024.

[40] Hartigan J A, Wong M A. Algorithm AS 136: A k-means clustering algorithm[J]. Journal of the royal statistical society. series c (applied statistics), 1979, 28(1): 100-108.

[41] Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks[J]. Pattern recognition, 2018, 77: 354-377.

[42] Chae H, Choi S H. Feature selection for efficient intrusion detection using attribute ratio[J]. Int. J. Comput. Commun, 2014, 8: 134-139.

[43] Shaukat K, Luo S, Varadharajan V, et al. Performance comparison and current challenges of using machine learning techniques in cybersecurity[J]. Energies, 2020, 13(10): 2509.

[44] Ashfaq R A R, Wang X Z, Huang J Z, et al. Fuzziness based semi-supervised learning approach for intrusion detection system[J]. Information sciences, 2017, 378: 484-497.

[45] Pham N T, Foo E, Suriadi S, et al. Improving performance of intrusion detection system using ensemble methods and feature selection[C]//Proceedings of the Australasian computer science week multiconference. 2018: 1-6.

[46] Gao Y, Liu Y, Jin Y, et al. A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system[J]. IEEE Access, 2018, 6: 50927-50938.

[47] Tama B A, Comuzzi M, Rhee K H. TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system[J]. IEEE access, 2019, 7: 94497-94507.

[48] Yin C, Zhu Y, Fei J, et al. A deep learning approach for intrusion detection using recurrent neural networks[J]. Ieee Access, 2017, 5: 21954-21961.

[49] Qureshi A S, Khan A, Shamim N, et al. Intrusion detection using deep sparse auto-encoder and self-taught learning[J]. Neural Computing and Applications, 2020, 32(8): 3135-3147.

[50] Su T, Sun H, Zhu J, et al. BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset[J]. IEEE Access, 2020, 8: 29575-29585.