Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 2, No. 1, March : 2023

INTRUSION DETECTION SYSTEM USING INTEGRATED DATA BALANCING TECHNIQUE WITH EXTREME LEARNING MACHINE

Kondru Mounika, Research Scholar, Department of Computer Science and Engineering, Adikavi Nannaya University, Rajamahendravaram, Andhra Pradesh, India*

P. Venkateswara Rao, Associate Professor, Department of Computer Science and Engineering, Adikavi Nannaya University, Rajamahendravaram, Andhra Pradesh, India

* Corresponding author's Email: kondrumounika25@gmail.com

Abstract

Intrusion Detection Systems (IDS) are crucial for maintaining the security and integrity of computer networks. The IDS can be classified into two types: anomaly-based IDS and signature-based IDS. Anomaly-based IDS detects attacks by monitoring the behavior of the network, while signature-based IDS detects attacks by matching patterns with known attack signatures. However, both types of IDS suffer from the problem of imbalanced datasets, where the number of normal instances significantly outweighs the number of anomalous instances. This problem can lead to low detection rates and high false positives. One way to address this problem is by using the Synthetic Minority Over-sampling Technique (SMOTE), a technique that creates synthetic samples of the minority class to balance the dataset. So, this work implemented SMOTE for balancing the NSL-KDD dataset. Then, Extreme Learning Machines (ELM) is trained with the SMOTE balanced features and classify the various types of intrusions from the dataset. The proposed SMOTE-ELM-IDS achieved accuracy as 95.22%, sensitivity as 95.72%, specificity as 80%, false alarm rate (FAR) as 20%. The simulation results show that the proposed SMOTE-ELM-IDS resulted in superior attack detection performance over state-of art methods.

Keywords: Intrusion Detection System, Synthetic Minority Over-sampling Technique, Extreme Learning Machines.

1. Introduction

Network intrusion detection is a security mechanism that has been developed in recent years to dynamically monitor, prevent, and defend against system intrusions. It mainly means that to find out whether the network system is attacked or violates the security policy by analyzing the information from several nodes of the network. Research on intrusion detection technologies at home and abroad has started since the 1980s and has now developed into an integral part of the network security architecture. Traditional machine learning methods have been widely used in network intrusion detection systems, such as Bayesian [1], support vector machines [2], decision tree [3], logistic regression [4], and so on. They all have achieved good results. However, these methods are not suitable for massive and high-dimensional data, and they cannot improve their own sensitivity to outliers and noise, resulting in the degradation of classification performance. At the same time, due to the continuous development of digital technology, network attack methods are becoming more and more diversified, and the traditional machine learning methods have been difficult to meet the needs of users. In recent years, deep learning techniques have been widely used in natural language processing, image recognition, and so on. It forms more abstract non-linear high-level representations by combining low-level features and then mines the input-output relationships between data, which has also achieved better results in the field of intrusion detection. Deep learning techniques commonly used in the field of intrusion detection include convolutional neural network (CNN) [6], recurrent neural network (RNN) [7], deep belief network, and so on. The literature converts the data traffic into individual pixel points in bytes to obtain the images generated by the traffic; then inputs the images into the convolutional neural network for convolution, pooling, and other operations; and finally obtains the classification results. The method achieves high accuracy in binary classification and multiclassification problems [8]. The literature uses the recognized KDD99 data set to conduct experiments, in which the long short-

term memory (LSTM) network is used to complete the selection of parameters and achieve more satisfactory experimental results. However, the method leads to a high false-alarm rate due to the improper selection of training parameters [9]. A hierarchical intrusion detection system based on spatial and temporal features is proposed in the literature. It first learns low-level spatial features of network traffic by deep convolutional neural networks and then acquires high-level temporal features by LSTM, but the method does not consider the problems of feature fusion and data imbalance [10]. The paper combines the features of WaveNet and bidirectional gated recurrent unit (BiGRU) for feature extraction and proposes an intrusion detection method that fuses WaveNet and BiGRU. The model of the paper can achieve better detection accuracy but does not consider the problem of sample imbalance [11]. Now the intrusion detection techniques have made great progress, but there are also the following problems. First, it faces the problem of feature redundancy; more feature dimensions will not only increase the training time of the model but also reduce the detection effect of the model. An intrusion detection method based on principal component analysis (PCA) and recurrent neural network is proposed in the literature. The principal component analysis method is used to reduce the dimension and noise of the data to find out the principal component feature subset with the maximum information. Finally, the processed data is trained for classification using a recurrent neural network and achieves high accuracy [12]. The literature proposes an intrusion detection method by combining the advantages of an autoencoder and residual network. The feature extraction is performed by reconstructing the network with an autoencoder, and then the designed residual network is trained with the extracted features. The experimental results are better in terms of accuracy, true rate, and false-alarm rate [13]. Secondly, it faces the problem of unbalanced samples of positive and negative classes in the data set used to evaluate the effects of the model. The literature uses an improved local adaptive synthetic minority oversampling technique for unbalanced traffic data to achieve abnormal traffic detection using RNN that has high detection accuracy for different types [14]. The novel contributions of this work are as follows.

- To perform the preprocessing operation, which maintains the uniform number of records with elimination of unknown data, missing data.
- To balance the dataset, SMOTE is adopted, which maintains the equal number of records in each class of NSL-KDD dataset.
- The model building and model training operation is carried out using ELM through SMOTE balanced features, which also performs the classification of various classes.

Rest of the article is organized as follows: section 2 deals with the detailed analysis of related work. Section 3 provides the description of the proposed SMOTE-ELM-IDS methodology. Section 4 covers the simulation results of SMOTE-ELM-IDS with comparative analysis. Section 5 concludes the article.

2. Literature survey

An intrusion detection system is used to detect anomaly traffic in networks. In recent years, owing to the capability to handle cyberattacks, an intrusion detection system has become the hottest research point in network security. Reference [15] proposes an ensemble learning strategy to adaptively choose the base classifiers, including SVM, KNN, and decision tree. EID3 [16] is the early attempt to effectively detect and defend attacks with unseen types in the real network based on ID3. MARK-ELM [17] combines multiple kernels boosting and the multiple classification reduced kernel ELM to detect multiple attack types with consistent results. HAST-IDS [18] aims to cast off the explicit feature engineering and proposes a hierarchical learning strategy where the convolutional neural network is leveraged to learn low-level spatial features and LSTM to capture high-level temporal features. KitNet [19] proposes an ensemble autoencoder algorithm to detect anomaly traffic flow in networks without supervision, making sure the IDS can be deployed in various real-world networks. SwiftIDS [20] proposes a parallel intrusion detection mechanism by analyzing traffic flow arriving in various time windows. The light gradient boosting machine (LightGBM) is leveraged as the classifier to handle

massive traffic data. H2ID [21] presents a two-stage hierarchical hybrid intrusion detection algorithm by incorporating multimodel deep autoencoder and soft-output classifiers to detect attacks in IoT and preserve privacy. WIDMoDS [22] aims to customize multiple intrusion detection models for the property of networks. The model proposes a classifier selection strategy based on data classifier applicable indicators and utilizes weight voting to automatically customize the behavior of intrusion detection models. XGBoost-DNN [23] proposes an ensemble algorithm of different deep neural networks to achieve the robustness of intrusion detection. Then, an extra XGBoost is integrated to achieve higher accuracy. Although these methods generally achieve promising performance, they cannot be performed on large-scale networks, which contain a high volume of data traffic.

To mitigate the issue, various feature selection-based intrusion detection systems have been proposed to handle the large traffic flow and improve the robustness by dropping irrelevant and redundant features. PSO-KNN [24] proposes a network intrusion detection system by applying binary particle swarm optimization (PSO) to generate feature subsets and leveraging KNN algorithm to perform classification. Chi-SVM [25] utilizes the chi-square feature selection technique to reduce dimensionality and then applies multi-class SVM to classify different attacks. Based on the design, the model efficiently and accurately performs intrusion detection. PIO-IDS [26] proposes a wrapper-based feature selection algorithm that leverages pigeon-inspired optimizer (PIO) to select the optimal subset. To further improve the model performance, they propose an algorithm to binarize the continuous pigeon-inspired optimizer. SMOTE-CFS [27] employs imbalance correction and feature selection techniques to improve the data quality in intrusion detection. Besides, an ensemble learning strategy is further proposed to improve detection performance. CFS-BA [28] proposes a heuristic algorithm to reduce dimensionality and choose the optimal subset based on the correlation between attributes. Then, the subset is fed into an ensemble model, and the results are merged by voting strategy. These models generate selected feature subsets from the original dataset by applying wrapper-based algorithms. However, computational complexity will be the bottleneck, since the quality of the selected features is evaluated by the performance of the classifier. To reduce the time consumption, various filter-based feature selection algorithms have been proposed. LSSVM-IDS [29] proposes a feature selection algorithm based on mutual information to automatically select the optimal features to capture the linear and nonlinear dependence. Then, they perform least-squares support vector machine-based IDS to make the prediction. Reference [30] combines mutual information-based feature selection technique and machine learning algorithms to develop a hybrid intrusion detection model. They propose a voting algorithm with information gain to filter the original dataset.

3. Proposed Methodology

The new generation of IDSs increasingly demands automated and intelligent network intrusion detection strategies to handle threats caused by an increasing number of advanced attackers in the cyber environment. There have been high demands for autonomous agent-based IDS solutions that require as little human intervention as possible while being able to evolve and improve itself (e.g., by taking appropriate actions for a given environment), and to become more robust to potential threats that have not been seen before (e.g., zero-day attacks). To overcome these attacks, the SMOTE-ELM-IDS method is developed in this work. Figure 1 shows the block diagram of proposed SMOTE-ELM-IDS methodology. Initially, the NSL-KDD dataset is considered, which is an imbalanced and uncorrelated dataset. So, the dataset is preprocessing is performed, which identifies and eliminates the missing symbols, unknown data, and special characteristics. Moreover, the preprocessed dataset contains the uneven records for each class, i.e., probe, normal, R2L, U2R classes contains the different number of records. It causes class misbalancing issues and causes classification mismatch problems. So, SMOTE method is performed to maintain the uniform number of records in each class of NSL-KDD dataset. The SMOTE method applies edited nearest neighbourhood (ENN) on each class, count number of features of each class, finds class with minimum features, apply synthetic over sampling on

obtained class, and balances each feature (samples). Finally, the ELM model performs the training of SMOTE dataset. Here, ELM performs the multi-layer perception, dividing the dataset into groups, training of each subset, majority voting of each subset, which resulted in classification outcome.

3.1 NSL-KDD dataset preprocessing

The NSL-KDD dataset is a widely used benchmark dataset for intrusion detection systems. The dataset contains a set of network traffic features extracted from a simulated environment, with each data point labeled as either normal or belonging to one of several attack categories. The following are the typical preprocessing steps for the NSL-KDD dataset:

Data cleaning: Check for and remove any missing values or duplicates in the dataset.

Data normalization: Normalize the input features to a common scale to avoid any bias in the learning algorithm towards features with larger values. Common normalization techniques include min-max scaling and z-score scaling.

Label encoding: Convert the categorical labels (e.g., attack categories) into numerical values. This can be done using techniques such as one-hot encoding or label encoding.

Train-test split: Split the dataset into training and testing subsets. The training subset is used to train the intrusion detection model, while the testing subset is used to evaluate its performance. The split can be done randomly or using a specific strategy, such as stratified sampling to ensure that the proportions of the different attack categories are preserved in both subsets.

Save preprocessed data: Save the preprocessed dataset as a new file in a format that can be easily loaded into the intrusion detection algorithm, such as a CSV file or a binary file. It is also important to carefully evaluate the performance of the intrusion detection model on the testing subset to ensure that it is generalizable to new data and not overfitting to the training data.

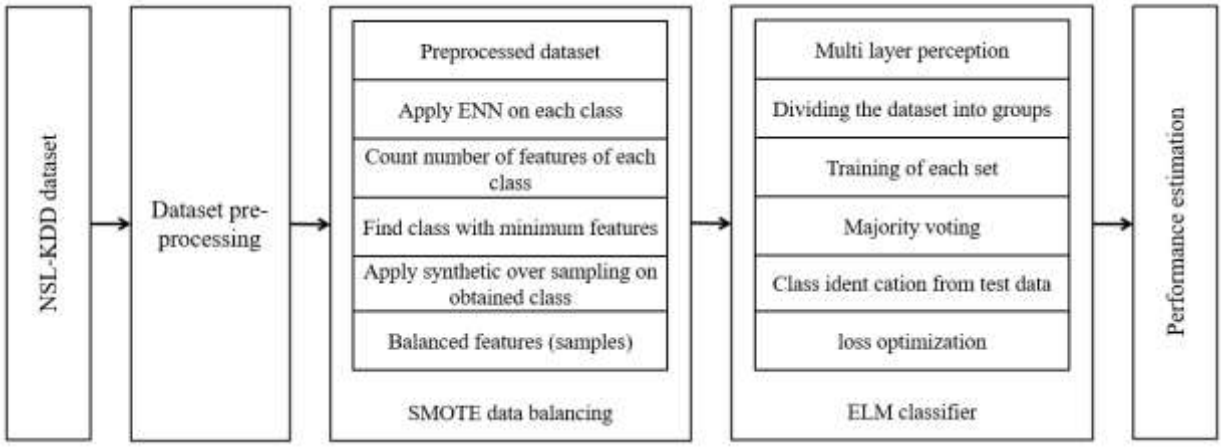

Figure 1. Proposed SMOTE-ELM-IDS methodology.

3.2 SMOTE

SMOTE is a popular algorithm used for addressing class imbalance problems in machine learning. The algorithm works by generating synthetic samples for the minority class, which helps to balance the class distribution and improve the performance of the classifier. The machine learning classifiers performance is degraded by uneven sizes of the dataset. Table 1 provides the SMOTE algorithm and Figure 2 shows the block diagram of SMOTE. Usually, the DoS, Probe, R2L, U2R attacks and normal classes of NSL-KDD dataset contains different number of records, which are unbalanced. So, the GBDT classifier will neglects the minority class with lesser records, and assigns maximum probabilities (priorities) to majority class with higher records. This can result in improper classification and misprediction. So, SMOTE data balancing technique applied on pre-processed NSL-KDD dataset. Further, the SMOTE is a type of over sampling technique, which always increases the number of records from minority class. Here, the oversampling operation creates the duplicate samples with nearest possible combinations. Figure 2 shows the block diagram of SMOTE. The SMOTE

algorithm calculates the total number of records in dataset, and also calculates the number of records for each class. Then, average records value will be estimated. Then, randomization concept is adopted by the SMOTE to generate the ENN for each record. Consider the dataset with X minority class samples. Then, $y_1, y_2, \cdots, y_N$ represents the N number of samples presented in each minority class. So, there must be more than N-number of KNN samples, i.e., $K > N$. Then, the random samples were inserted into dataset by adopting the correlation between records. The random correlation factor p_i is constructing the interpolation operation.

$$p_i = X + rand(0,1) \times (y_i - X),\ i = 1,2,\cdots,N \quad (1)$$

Here, a random integer is assumed in the range of [0,1]. The y_i denotes the nearest neighbor for sample i the X represents data records presented in each minority class.

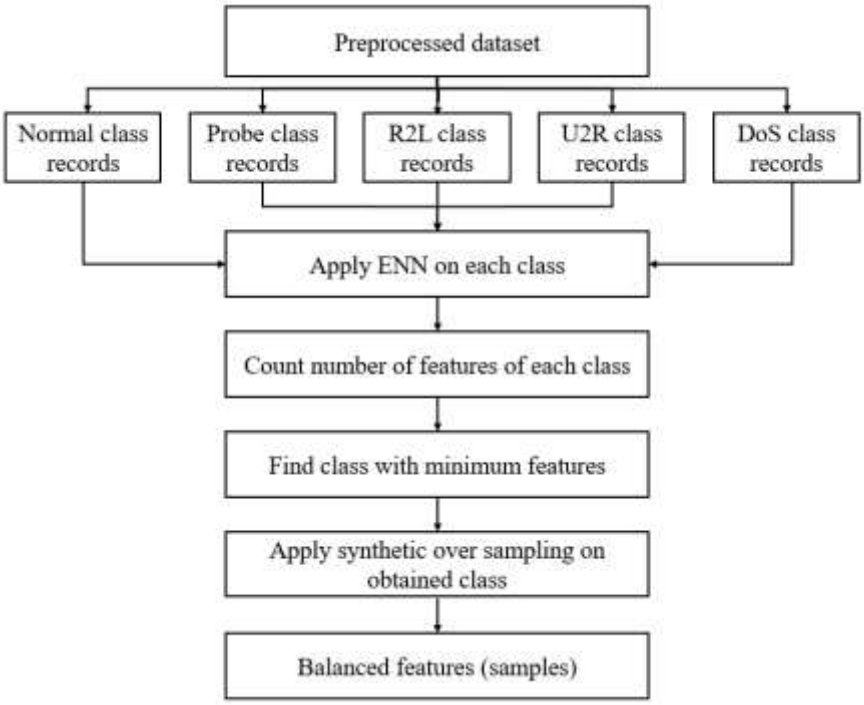

Figure 2. Block diagram of SMOTE.

Table 1. SMOTE algorithm.

Input: Preprocessed NSL-KDD dataset **Output:** SMOTE balanced features
Step 1: Select a minority class sample from the dataset. **Step 2:** Find the k nearest neighbors for this sample in the feature space. **Step 3:** Choose one of these neighbors at random. **Step 4:** Generate a synthetic sample by taking a linear combination of the original sample and the chosen neighbor. **Step 5:** Repeat steps 1-4 until the desired level of oversampling has been achieved.

The key parameter in the SMOTE algorithm is k, which determines the number of neighbors to consider when generating synthetic samples. Typically, k is set to 5 or 10. The distribution of the synthetic samples can be characterized by the following parameters:

Density: The density of the synthetic samples is higher in regions where the minority class is densely populated.

Spread: The spread of the synthetic samples depends on the number of nearest neighbors k. Larger values of k lead to more spread out synthetic samples, while smaller values of k lead to more localized synthetic samples.

Bias: The synthetic samples are biased towards the minority class samples that are closer to the majority class samples. This bias can be reduced by increasing the number of nearest neighbors k.

Overlapping: The synthetic samples can overlap with the majority class samples, which can reduce the effectiveness of the SMOTE algorithm. This problem can be mitigated by using other algorithms, such as ADASYN, which generate synthetic samples in a more adaptive manner.

Finally, SMOTE is a powerful algorithm for addressing class imbalance problems in machine learning. The algorithm generates synthetic samples by taking linear combinations of the minority class samples and their nearest neighbors. The distribution of the synthetic samples depends on the density, spread, bias, and overlapping of the original data. Careful selection of the parameters, such as the number of nearest neighbors, is necessary to ensure effective oversampling.

3.3 ELM

Given a dataset consisting of N input-output pairs $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where xi is an input vector of dimension D and yi is a target output, the ELM algorithm aims to learn a function $f(x)$ that maps inputs to outputs. Table 2 provides algorithmic steps of the ELM. Figure 3 shows the ELM block diagram. Specifically, ELM learns a single-hidden layer feedforward neural network (SLFN) of the form:

$$f(x) = g(w_2^T h(x) + b2) \quad (2)$$

where g is an activation function (usually a sigmoid or radial basis function), h(x) is a vector of hidden layer activations computed as:

$$h(x) = [g(w_1^T x1 + b1), g(w_1^T x2 + b1), \ldots, g(w_1^T xN + b1)] \quad (3)$$

Here, $w1$ and $b1$ are the input-to-hidden layer weights and biases, respectively, and $w2$ and $b2$ are the output layer weights and biases, respectively. The goal of the ELM algorithm is to find the optimal values of w1, b1, w2, and b2 that minimize the following regularized least square's objective function:

$$J = ||Y - Hw2||\text{^}2 + \lambda\, ||w2||\text{^}2 \quad (4)$$

where Y is the N-by-1 target output matrix, H is the N-by-M hidden layer activation matrix, $w2$ is the M-by-1 output layer weight vector, and λ is a regularization parameter. Note that H can be written as the following matrix product:

$$H = G(X) * W1 \quad (5)$$

where X is the N-by-D input matrix, W1 is the D-by-M input-to-hidden layer weight matrix, and G(X) is the N-by-M matrix whose (i,j)th element is given by $g(w_1^T x_i + b1_j)$. To obtain the optimal solution for w2, we take the partial derivative of J with respect to w2 and set it to zero:

$$\frac{dJ}{dw2} = -2H^T(Y - Hw2) + 2\lambda w2 = 0 \quad (6)$$

This gives us the optimal solution for $w2$:

$$w2 = (H^T H + \lambda I)^{-1} H^T Y \quad (7)$$

where I is the M-by-M identity matrix. Once we have obtained the optimal solution for w2, we can use it to make predictions for new inputs x by computing the corresponding hidden layer activations h(x) and applying the output layer weights:

$$f(x) = g(w2^T h(x) + b2) \quad (8)$$

One important aspect of ELM is that the input-to-hidden layer weights w1 and biases b1 are randomly generated and fixed before training the output layer weights w2. This means that ELM is a single-pass algorithm, and the computational cost of training the hidden layer is much lower than that of traditional neural networks, which use iterative optimization methods to learn all weights. So, ELM learns a SLFN by randomly generating input-to-hidden layer weights and biases, computing the hidden layer activations, and solving for the output layer weights using a closed-form solution. The regularization

parameter λ controls the trade-off between fitting the data and preventing overfitting. The ELM is a single-pass algorithm, which means that steps 1-3 are performed only once, and the output layer weights are computed using a closed-form solution. The computational cost of ELM is therefore much lower than that of traditional neural networks, which require iterative optimization methods to learn all weights.

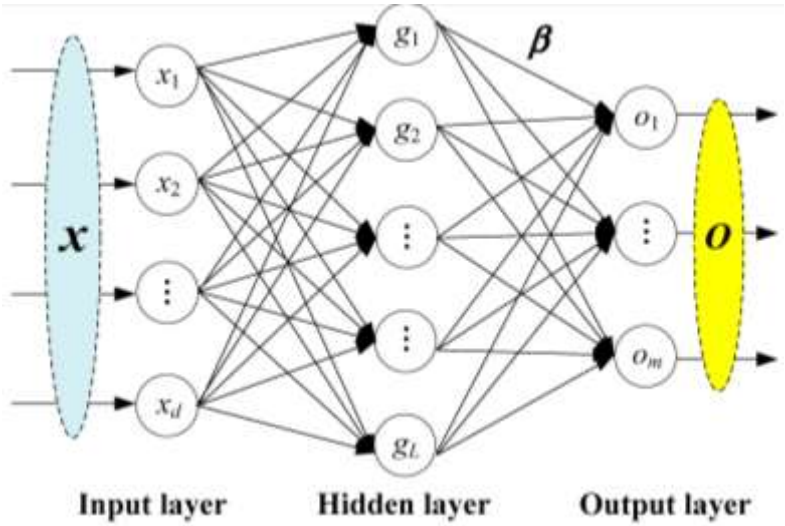

Figure 3. ELM operational diagram.

Table 2. ELM algorithmic steps

Input: RFES selected features **Output:** Classified outcome.
Step 1: Initialize the input-to-hidden layer weights w1 and biases b1 randomly. The input-to-hidden layer weights can be generated from a uniform distribution or a Gaussian distribution. **Step 2:** Compute the hidden layer activations h(x) for each input vector x in the dataset. This can be done by applying the activation function g to the weighted sum of the input vector and the corresponding hidden layer weights and bias using equation (4). **Step 3:** Construct the hidden layer activation matrix H by stacking the hidden layer activations for all input vectors in the dataset using equation (6). **Step 4:** Compute the output layer weights w2 using the Moore-Penrose pseudoinverse of H using equation (8). **Step 5:** Repeat steps 2-4 until a stopping criterion is met, finally the SoftMax classifier presented in output layer classifies the various classes from NSL-KDD dataset of RFES selected features.

4. Results and discussion

The proposed method is evaluated on the NSL-KDD dataset, which is a widely used dataset for evaluating intrusion detection systems. The dataset contains 41 features and 23 types of attacks. The dataset is preprocessed by removing redundant features and scaling the features to a common range. The dataset is then split into training and testing sets in a 70:30 ratio. SMOTE is applied to the training set to balance the dataset, resulting in a new training set that has an equal number of normal and anomalous instances. ELM is trained on the balanced training set, and the performance of the trained model is evaluated on the testing set.

4.1 Dataset

The NSL-KDD dataset is a benchmark dataset for intrusion detection systems (IDSs) developed by the University of New Brunswick in Canada. It is an improved version of the original KDD Cup 99 dataset, which was widely used for evaluating IDSs but was criticized for its limitations, including the use of

an outdated set of attacks and the lack of a representative sample of normal traffic. The NSL-KDD dataset was created to address these limitations and provide a more comprehensive and realistic dataset for evaluating IDSs. It consists of a set of network traffic data collected from a local area network (LAN) and simulated attacks generated using a variety of tools and techniques. The dataset includes several types of attacks, such as DoS, R2L, U2R, and probing attacks, and normal traffic. The NSL-KDD dataset is divided into three subsets: training, testing, and validation. The training subset is used for training the IDS, the validation subset is used for optimizing the parameters of the IDS, and the testing subset is used for evaluating the performance of the IDS. The NSL-KDD dataset has become a popular benchmark dataset for evaluating IDSs, and many researchers have used it to develop and evaluate new IDSs and intrusion detection techniques. However, it should be noted that the NSL-KDD dataset is not without its own limitations, and its use for evaluating IDSs should be done with caution. The NSL-KDD dataset has a total of 42 columns (or features) that describe different aspects of the network traffic data. These columns can be grouped into four categories:

Basic features (9 columns): These columns provide basic information about each network connection, such as the protocol type, the service used, and the source and destination IP addresses and port numbers.

Content features (23 columns): These columns describe the content of each network connection, such as the number of failed login attempts, the number of bytes sent and received, and the duration of the connection.

Traffic features (7 columns): These columns provide information about the traffic pattern of each network connection, such as the number of connections to the same host in the past two seconds and the number of connections to different hosts in the past two seconds.

Attack type (3 columns): These columns specify whether each network connection is a normal connection or an attack, and if it is an attack, the type of attack it is (DoS, R2L, U2R, or probing).

4.2 Performance evaluation

Table 3 provides the detailed description of performance evaluation of proposed SMOTE-ELM-IDS. Here, the DoS, Probe, R2L, U2R attacks classes, and normal class performance is measures. Further, the overall performance is measured by taking the average of all classes.

Table 3. Performance evaluation of proposed SMOTE-ELM-IDS

Class	Accuracy	Sensitivity	Specificity	FAR
DoS	99.5187	99.256	94.3782	5.6218
Probe	99.6936	98.3148	93.8671	6.1329
R2L	99.7781	94.3787	75.1278	24.8722
U2R	88.978	100	43.7819	56.2181
normal	88.174	86.6643	92.845	7.155
Overall	95.22848	95.72276	80	20

Table 4 compares the performance comparison of various methods for DoS class of NSL-KDD dataset. Here, the proposed SMOTE-ELM-IDS method has increased accuracy by 2.76%, sensitivity 3.66%, specificity by 0.77%, and FAR by 23.55% as compared to the MARK-ELM [17]. Then, the proposed SMOTE-ELM-IDS method has increased accuracy by 3.41%, sensitivity 6.00%, specificity by 1.77%, and FAR by 415.76% as compared to the SWIFTIDS [20]. Later, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.59%, sensitivity 5.76%, specificity by 1.34%, and FAR by 65.83% as compared to the CHI-SVM [25]. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 3.87%, sensitivity 2.88%, specificity by 0.62%, and FAR by 20.89% as compared to LSSVM-IDS [29].

Table 4. DoS class performance comparison of various methods.

Method	Accuracy	Sensitivity	Specificity	FAR
MARK-ELM [17]	96.84	95.75	93.65	4.55

SWIFTIDS [20]	96.23	93.63	92.73	1.09
CHI-SVM [25]	95.15	93.85	93.13	3.39
LSSVM-IDS [29]	95.81	96.47	93.79	4.65
Proposed SMOTE-ELM-IDS	99.5187	99.256	94.3782	5.6218

Table 5 compares the performance comparison of various methods for probe class of NSL-KDD dataset. Here, the proposed SMOTE-ELM-IDS method has increased accuracy by 5.09%, sensitivity 2.32%, specificity by 3.29%, and FAR by 22.90% as compared to the MARK-ELM [17]. Then, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.56%, sensitivity 5.82%, specificity by 1.66%, and FAR by 411.07% as compared to the SWIFTIDS [20]. Later, the proposed SMOTE-ELM-IDS method has increased accuracy by 7.77%, sensitivity 5.89%, specificity by 1.47%, and FAR by 22.65% as compared to the CHI-SVM [25]. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 2.92%, sensitivity 2.30%, specificity by 3.90%, and FAR by 190.65% as compared to LSSVM-IDS [29].

Table 5. Probe class performance comparison of various methods.

Method	**Accuracy**	**Sensitivity**	**Specificity**	**FAR**
MARK-ELM [17]	94.86	96.08	90.87	4.99
SWIFTIDS [20]	95.34	92.90	92.33	1.20
CHI-SVM [25]	92.50	92.84	92.50	5.00
LSSVM-IDS [29]	96.86	96.10	90.34	2.11
Proposed SMOTE-ELM-IDS	99.6936	98.3148	93.8671	6.1329

Table 6 compares the performance comparison of various methods for R2L class of NSL-KDD dataset. Here, the proposed SMOTE-ELM-IDS method has increased accuracy by 3.11%, sensitivity 2.35%, specificity by 6.89%, and FAR by 17.87% as compared to the MARK-ELM [17]. Then, the proposed SMOTE-ELM-IDS method has increased accuracy by 5.67%, sensitivity 1.66%, specificity by 2.35%, and FAR by 8.14% as compared to the SWIFTIDS [20]. Later, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.39%, sensitivity 2.41%, specificity by 5.39%, and FAR by 17.71% as compared to the CHI-SVM [25]. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 6.66%, sensitivity 1.58%, specificity by 1.96%, and FAR by 22.28% as compared to LSSVM-IDS [29].

Table 6. R2L class performance comparison of various methods.

Method	**Accuracy**	**Sensitivity**	**Specificity**	**FAR**
MARK-ELM [17]	96.76	92.21	70.28	21.10
SWIFTIDS [20]	94.42	92.83	73.40	23.00
CHI-SVM [25]	95.58	92.15	71.28	21.13
LSSVM-IDS [29]	93.54	92.91	73.68	20.34
Proposed SMOTE-ELM-IDS	99.7781	94.3787	75.1278	24.8722

Table 7 compares the performance comparison of various methods for U2R class of NSL-KDD dataset. Here, the proposed SMOTE-ELM-IDS method has increased accuracy by 1.74%, sensitivity 2.64%, specificity by 4.39%, and FAR by 0.96% as compared to the MARK-ELM [17]. Then, the proposed SMOTE-ELM-IDS method has increased accuracy by 5.29%, sensitivity 3.65%, specificity by 8.42%, and FAR by 3.39% as compared to the SWIFTIDS [20]. Later, the proposed SMOTE-ELM-IDS method has increased accuracy by 3.64%, sensitivity 2.84%, specificity by 4.44%, and FAR by 7.75% as compared to the CHI-SVM [25]. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 1.24%, sensitivity 4.96%, specificity by 5.39%, and FAR by 6.11% as compared to LSSVM-IDS [29].

Table 7. U2R class performance comparison of various methods.

Method	Accuracy	Sensitivity	Specificity	FAR
MARK-ELM [17]	87.45	97.42	41.94	55.68
SWIFTIDS [20]	84.50	96.47	40.38	54.37
CHI-SVM [25]	85.85	97.23	41.92	52.17
LSSVM-IDS [29]	87.88	95.27	41.54	52.98
Proposed SMOTE-ELM-IDS	88.978	100	43.7819	56.2181

Table 8 compares the performance comparison of various methods for normal class of NSL-KDD dataset. Here, the proposed SMOTE-ELM-IDS method has increased accuracy by 1.79%, sensitivity 4.40%, specificity by 2.57%, and FAR by 102.11% as compared to the MARK-ELM [17]. Then, the proposed SMOTE-ELM-IDS method has increased accuracy by 1.62%, sensitivity 5.81%, specificity by 1.62%, and FAR by 83.93% as compared to the SWIFTIDS [20]. Later, the proposed SMOTE-ELM-IDS method has increased accuracy by 3.63%, sensitivity 5.77%, specificity by 1.92%, and FAR by 29.85% as compared to the CHI-SVM [25]. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 0.86%, sensitivity 4.85%, specificity by 1.19%, and FAR by 43.38% as compared to LSSVM-IDS [29].

Table 9 compares the performance comparison of various methods for overall classes of NSL-KDD dataset. Here, the average of five individual classes are considered. Here, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.08%, sensitivity 4.92%, specificity by 2.78%, and FAR by 5.48% as compared to the MARK-ELM [17]. Then, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.17%, sensitivity 1.09%, specificity by 6.24%, and FAR by 25.94% as compared to the SWIFTIDS [20]. Later, the proposed SMOTE-ELM-IDS method has increased accuracy by 0.34%, sensitivity 4.04%, specificity by 8.91%, and FAR by 15.27% as compared to the CHI-SVM [25]. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.70%, sensitivity 2.57%, specificity by 7.93%, and FAR by 16.61% as compared to LSSVM-IDS [29].

Table 8. Normal class performance comparison of various methods.

Method	Accuracy	Sensitivity	Specificity	FAR
MARK-ELM [17]	86.62	83.01	90.51	3.54
SWIFTIDS [20]	86.76	81.90	91.36	3.89
CHI-SVM [25]	85.08	81.93	91.09	5.51
LSSVM-IDS [29]	87.42	82.65	91.75	4.99
Proposed SMOTE-ELM-IDS	88.174	86.6643	92.845	7.155

Table 9. Overall performance comparison of various methods.

Method	Accuracy	Sensitivity	Specificity	FAR
MARK-ELM [17]	91.49	91.23	77.83	18.96
SWIFTIDS [20]	91.41	94.69	75.30	15.88
CHI-SVM [25]	94.90	92.00	73.45	17.35
LSSVM-IDS [29]	90.95	93.32	74.12	17.15
Proposed SMOTE-ELM-IDS	95.22848	95.72276	80	20

5. Conclusion

This work is focused on implementation of SMOTE-ELM-IDS methodology for multi class intrusion detection. The NSL-KDD dataset is characterized by its imbalance and lack of correlation, which is examined at the beginning of the process. Hence, the dataset goes through preprocessing, which detects

and gets rid of any missing symbols, unknown data, and special features. In addition, the preprocessed dataset has an unequal number of records for each class. This means that the probe, normal, R2L, and U2R classes all have a different total number of records. It leads to difficulties with the classes not being evenly distributed and with the classifications not being a good fit. So. The SMOTE approach is used so that the NSL-KDD dataset may continue to have a same number of entries in each category. The SMOTE technique starts by applying modified ENN to each class. Then, it counts the number of features that each class has, identifies the class that possesses the fewest features, applies synthetic over sampling to the class that was acquired, and then balances each feature. In conclusion, the training of the SMOTE dataset is carried out via the ELM model. At this stage, ELM is responsible for carrying out the multi-layer perception, which involves partitioning the dataset into groups, training each subset, and doing majority voting on each subset. This ultimately leads to the classification results. Finally, the proposed SMOTE-ELM-IDS method has increased accuracy by 4.70%, sensitivity 2.57%, specificity by 7.93%, and FAR by 16.61% as compared to existing methods. Further, this work can be extended with other recursive feature extraction and feature selection with data balancing methods for improved performance.

References

[1] S. M. Kasongo and Y. Sun, "A deep learning method with wrapper-based feature extraction for wireless intrusion detection system," Computers & Security, vol. 92, Article ID 101752, 2020.
[2] M. Safaldin, M. Otair, and L. Abualigah, "Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 2, pp. 1559–1576, 2021.
[3] V. Singh, G. Sharma, R. C. Poonia, N. K. Trivedi, and L. Raja, "Source redundancy management and host intrusion detection in wireless sensor networks," Recent Advances in Computer Science and Communications, vol. 14, no. 1, pp. 43–47, 2021.
[4] G. Thamilarasu and S. Chawla, "Towards deep-learning-driven intrusion detection for the internet of things," Sensors, vol. 19, no. 9, p. 1977, 2019.
[5] S. Smys, A. Basar, and H. Wang, "Hybrid intrusion detection system for internet of things (IoT)," Journal of ISMAC, vol. 2, no. 04, pp. 190–199, 2020.
[6] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Towards a deep learning-driven intrusion detection approach for Internet of Things," Computer Networks, vol. 186, Article ID 107784, 2021.
[7] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," Vehicular Communications, vol. 21, Article ID 100198, 2020.
[8] A. Derhab, M. Belaoued, I. Mohiuddin, F. Kurniawan, and M. K. Khan, "Histogram-based intrusion detection and filtering framework for secure and safe in-vehicle networks," IEEE Transactions on Intelligent Transportation Systems, vol. 23, 2021.
[9] 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.
[10] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," Expert Systems with Applications, vol. 148, Article ID 113249, 2020.
[11] O. Almomani, "A feature selection model for network intrusion detection system based on PSO, gwo, ffa and ga algorithms," Symmetry, vol. 12, no. 6, p. 1046, 2020.
[12] D. Jin, Y. Lu, J. Qin, Z. Cheng, and Z. Mao, "SwiftIDS: real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism," Computers & Security, vol. 97, Article ID 101984, 2020.

[13] M. A. Umar, C. Zhanfang, and Y. Liu, "A Hybrid Intrusion Detection with Decision Tree for Feature Selection," 2020, https://arxiv.org/ftp/arxiv/papers/2009/2009.13067.pdf.

[14] B. S. Bhati, G. Chugh, F Al-Turjman, and N. S. Bhati, "An improved ensemble based intrusion detection technique using XGBoost," Transactions on Emerging Telecommunications Technologies, vol. 32, no. 6, p. e4076, 2021.

[15] M. Choraś and M. Pawlicki, "Intrusion detection approach based on optimised artificial neural network," Neurocomputing, vol. 452, pp. 705–715, 2021.

[16] W. Liang, L. Xiao, K. Zhang, M. Tang, D. He, and K. C. Li, "Data Fusion Approach for Collaborative Anomaly Intrusion Detection in Blockchain-Based Systems," IEEE Internet of Things Journal, 2021.

[17] E. Papadogiannaki and S. Ioannidis, "Acceleration of intrusion detection in encrypted network traffic using heterogeneous hardware," Sensors, vol. 21, no. 4, p. 1140, 2021.

[18] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, "Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using Spark," IEEE Access, vol. 6, Article ID 59657, 2018.

[19] A. H. Engly, A. R. Larsen, and W. Meng, "Evaluation of anomaly-based intrusion detection with combined imbalance correction and feature selection," International Conference on Network and System Security, Springer, New York, NY, USA, pp. 277–291, 2020.

[20] A. R. Syarif and W. Gata, "Intrusion detection system using hybrid binary PSO and K-nearest neighborhood algorithm," in Proceedings of the 2017 11th International Conference on Information & Communication Technology and System (ICTS), pp. 181–186, IEEE, Surabaya, Indonesia, October, 2017.

[21] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," Computer Networks, vol. 174, Article ID 107247, 2020.

[22] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," Journal of Computational Science, vol. 25, pp. 152–160, 2018.

[23] X. Li, W. Chen, Q. Zhang, and L. Wu, "Building Auto-Encoder Intrusion Detection System based on random forest feature selection," Computers & Security, vol. 95, Article ID 101851, 2020.

[24] A. K. Shukla and P. Singh, "Building an effective approach toward intrusion detection using ensemble feature selection," International Journal of Information Security and Privacy, vol. 13, no. 3, pp. 31–47, 2019.

[25] G. Ke, Q. Meng, and T. Finley, "LightGBM: a highly efficient gradient boosting decision tree," Advances in Neural Information Processing Systems, Curran Associates, Inc., vol. 30, 2017.

[26] Y. Meng and L.-F. Kwok, "Enhancing false alarm reduction using voted ensemble selection in intrusion detection," International Journal of Computational Intelligence Systems, vol. 6, no. 4, p. 626, 2013.

[27] V. Jaiganesh, S. Mangayarkarasi, and P. Sumathi, "An efficient algorithm for network intrusion detection system," International Journal of Computer Application, vol. 90, no. 12, pp. 12–16, 2014.

[28] J. M. Fossaceca, T. A. Mazzuchi, and S. Sarkani, "MARK-ELM: application of a novel multiple kernel learning framework for improving the robustness of network intrusion detection," Expert Systems with Applications, vol. 42, no. 8, pp. 4062–4080, 2015.

[29] W. Wang, Y. Sheng, J. Wang et al., "HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," IEEE Access, vol. 6, pp. 1792–1806, 2018.

[30] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," 2022, http://arxiv.org/abs/1802.09089.