Research Article

EFS-DNN: An Ensemble Feature Selection-Based Deep Learning Approach to Network Intrusion Detection System

Zehong Wang, Jianhua Liu, and Leyao Sun

Department of Computer Science and Engineering, Shaoxing University, Shaoxing 312000, China

Correspondence should be addressed to Jianhua Liu; ljh_541@163.com

Received 26 January 2022; Revised 3 March 2022; Accepted 10 March 2022; Published 5 April 2022

Academic Editor: Wenjuan Li

Copyright © 2022 Zehong Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In recent years, the scale of networks has substantially evolved due to the rapid development of infrastructures in real networks. Under the circumstances, intrusion detection systems (IDSs) have become the crucial tool to detect cyberattacks, malicious actions, and anomaly behaviors that threaten the credibility and integrity of information services in networks. Feature selection technologies are commonly applied in various intrusion detection algorithms owing to the potential of improving performance and speeding up decision-making. However, existing feature selection-based intrusion detection methods still suffer from high computational complexity or the lack of robustness. To mitigate these challenges, we propose a novel ensemble feature selection-based deep neural network (EFS-DNN) to detect attacks in networks with high-volume traffic data. In particular, we leverage light gradient boosting machine (LightGBM) as the base selector in the ensemble feature selection module to enhance the robustness of the selected optimal subset. Besides, we utilize a deep neural network with batch normalization and embedding technique as the classifier to improve the expressiveness. We conduct extensive experiments on three public datasets to demonstrate the superiority of the EFS-DNN compared with baselines.

1. Introduction

Owing to the rapid growth of network scale and the development of network infrastructure, the traffic flow in cyberspace is increasing at an alarming rate in recent years. Under the circumstance, cybersecurity issues have been paid much more attention in various networks, such as wireless networks [1–3], the Internet of things [4–6], and vehicle networks [7, 8]. To detect cyberattacks, intrusion detection system (IDS) [9–16] has become a common tool due to its serviceability and extendibility. However, the huge amount of information flow has posed a great challenge to the efficacy and efficiency of the traditional intrusion detection systems. How to handle the cyberattacks in networks with amounts of traffic flow is urgent needs to be solved.

To the best of our knowledge, the mainstream solutions on handling the high-volume data traffic in networks fall into three categories: (1) designing dedicated hardwares [17], (2) applying big data techniques [18], and (3) leveraging feature selection methods [10, 11, 19]. The first approach aims to design specialized hardwares to efficiently detect cyberattacks in real networks by reducing the time consumption in pattern matching. However, this is expensive to deploy and inflexible to extend in real networks. The second method utilizes big data techniques, such as Spark, to effectively handle large-scale network traffic data. Although a high volume of traffic flow is stored in distributed file systems, the approach inevitably sacrifices the real-time requirement in decision-making. The third approach applies feature selection before decision-making. These technologies usually drop redundant features to accelerate the decision-making phase and achieve robustness. Compared with hardware-based and big data-based methods, this can be applied to more scenarios due to the pluggability. Thus, in this study, we follow the third approach to enhance the performance of the model and speed up the decision-making on networks with large-scale traffic.

Feature selection is defined as the process of extracting the inputs from the original dataset to enhance the expressiveness of the particular classifier. In fact, feature selection techniques are leveraged in various regions because of the capability of eliminating redundant features, reducing
computational complexity, and alleviating the curse of dimensionality. The feature selection is roughly divided into three paradigms, i.e., wrapper, filter, and embedded.

The wrapper-based algorithms [10, 20, 21] apply a classifier to evaluate the performance of the selected subsets. The performance of the classifier indicates the quality of the feature combinations. In the wrapper-based models, the process of optimal feature combination search is combined with the training of the model. Although the procedure may yield a promising intrusion detection performance, the computational complexity in training will be high.

The filter-based feature selection approaches [1, 9, 22, 23] decouple the subset extraction and the training of the classification models. In other words, these methods select proper feature combinations in preprocessing phase and then leverage the selected features to train the classifiers. Compared with wrapper-based approaches, this is more computationally efficient. However, due to the irrelevance between the feature selection and the training process, these methods generally choose the suboptimal subset.

The embedded-based approaches can be identified as the incorporation of filter-based and wrapper-based methods. In particular, the weights or the importance scores of features are learned by the classifier while training. Existing embedded-based algorithms include decision tree-based methods, which utilize information gain to indicate the importance of features and penalty-based methods, e.g., Lasso. Although these methods are efficient, they cannot be flexibly deployed due to the integration between the optimization and the feature selection.

Based on the discussion above, the filter-based feature selection technique is efficient in handling a large amount of traffic data compared with the other two methods, whereas robustness cannot be ensured. To overcome the challenge, recent works [24] apply ensemble feature selection to generate the optimal subset to improve the effectiveness and expressiveness. Ensemble feature selection is developed on the principle of ensemble learning. This is derived from the empirical results that the combination of multiple feature selectors generally yields better performance compared with the single selector. Owing to the capability of mitigating the volatility, ensemble feature selection has the potential of enhancing the robustness of filter-based feature selection algorithms.

In this study, we propose an ensemble feature selection-based deep neural network (EFS-DNN) to efficiently and effectively detect attacks in networks with high-volume traffic data. In particular, we propose a novel filter-based ensemble feature selection algorithm to improve the robustness of the selected subsets. We leverage light gradient boosting machine (LightGBM [25]), an ensemble algorithm of the decision tree, as the base selector to choose the optimal subset according to information gain. Afterward, we utilize a deep neural network (DNN) classifier with batch normalization and embedding technique to detect anomaly traffic on networks. Compared with the shallow classifiers, DNN captures the latent relations among features to provide a better expressiveness. The contribution of this study is summarized as follows:

(1) We propose a novel ensemble feature selection-based deep neural network (EFS-DNN) to efficiently detect intrusions in networks with a large amount of traffic flow by combining filter-based ensemble feature selection technique and deep neural network.

(2) We proposed a filter-based ensemble feature selection method to achieve robustness and decrease the variance. To further enhance the performance of feature selection, we leverage LightGBM as the base selector to extract the optimal subset.

(3) We conduct extensive experiments to evaluate the performance of EFS-DNN on three public datasets. Experimental results demonstrate that the model outperforms all baselines on binary-class classification and five-class classification tasks.

The study is organized as follows. Section 2 summarizes the related work about the feature selection-based and deep learning-based intrusion detection system. Section 3 demonstrates the structure of the proposed EFS-DNN model. In Section 4, we compare the EFS-DNN against state-of-the-art models in three public datasets. Finally, we conclude the paper and indicate the future research direction in Section 5.

2. Related Work

An intrusion detection system is used to detect anomaly traffic in networks. In recent year, owing to the capability to handle cyberattacks, an intrusion detection system has become the hottest research point in network security. Reference [26] proposes an ensemble learning strategy to adaptively choose the base classifiers, including SVM, KNN, and decision tree. EID3 [27] is the early attempt to effectively detect and defend attacks with unseen types in the real network based on ID3. MARK-ELM [28] combines multiple kernel boosting and the multiple classification reduced kernel ELM to detect multiple attack types with consistent results. HAST-IDS [29] 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 [30] 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 [12] 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 [31] 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 [32] 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 [33] 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 [20] proposes a network intrusion detection system by applying binary particle swarm optimization (PSO) to generate feature subsets and leveraging K-NN algorithm to perform classification. Chi-SVM [34] 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 the intrusion detection. PIO-IDS [10] 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 binary the continuous pigeon-inspired optimizer. SMOTE-CFS [19] 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 [21] proposes a heuristic algorithm to reduce dimensionality and choose the optimal subset on the basis of 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, the 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 [9] 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 [22] combines mutual information-based feature selection technique and machine learning algorithms to develop a hybrid intrusion detection model. In particular, they propose a voting algorithm with information gain to filter the original dataset. NDAE [35] proposes the nonsymmetric deep autoencoder layer to perform automatic feature engineering in an unsupervised approach. The learned feature hidden vectors are fed into a random forest model for classification. DBN-SVM [18] proposes a distributed approach for intrusion detection in large-scale networks based on Spark. The model is built on the combination of deep belief network that performs deep feature extraction and multilayer ensemble support vector machines to effectively detect anomaly behaviors. WFEU-FFDNN [1] combines the feature selection algorithm and deep learning. In particular, the extra tree-based feature selection unit is used to generate the reduced optimal feature vector, and the feed-forward deep neural network serves as the classifier to identify the malicious actions. AE-IDS [23] selects the optimal features based on random forest with information gain and AP cluster feature grouping algorithm to split multiple feature subsets. Then, the model leverages autoencoder to learn latent feature vector and the result is used in the decision-making phase. DL-SVM [36] employs Apache Spark as the data processing tool to efficiently detect intrusions in massive network traffic data and uses a stacked autoencoder network to generate latent features. Afterward, the extracted feature vectors are fed into multiple classifiers to perform decision-making. These filter-based methods choose the optimal subset through preprocessing, and then, the result is fed into the classifier to perform decision-making. Although these methods achieve computational efficiency, the selected feature combination suffers from volatility.

To overcome the issue of filter-based feature selection, SVM-IDS [24] leverages mutual information-based ensemble feature selection method to select the optimal feature combination of relevant and nonredundant features, as well as adopts SVM as the classifier to identify the category of traffic flow. However, owing to the usage of the shallow model (SVM), the classifier cannot capture the latent relationship between features.

Table 1 summarizes the surveyed paper and categorizes them according to the type of feature selectors and classifiers. Notice that EFS-DNN fills the gap of the combination of ensemble feature selection and deep neural network. To further enhance the robustness of intrusion detection and achieve the real-time requirement in networks, the proposed EFS-DNN adopts LightGBM-based ensemble feature selection to efficiently select the optimal subset and leverages the powerful deep neural network to learn the latent information from the selected features.

3. Methodology

The proposed EFS-DNN consists of two modules, i.e., ensemble feature selection module and DNN-based intrusion detection classifier. We leverage LightGBM as the base feature selector to perform ensemble feature selection and utilize a deep neural network with batch normalization and embedding technique as the classifier to perform intrusion detection. The overview structure of EFS-DNN is shown in Figure 1. The training procedure of EFS-DNN is demonstrated in Algorithm 1. For completeness, we present the acronyms employed following in Table 2.

3.1. Ensemble Feature Selection. Feature selection plays a vital role in machine learning tasks due to the capacity of alleviating the curse of dimensionality and improving robustness. In the intrusion detection domain, the filter-based feature selection technique is the mainstream, which decouples the process of feature selection and the training of the classifier. Although the strategy indeed saves time consumption in decision-making, the feature selection procedure is generally volatile. To overcome the weakness, we propose an ensemble feature selection algorithm to ensure the robustness of the selected optimal subset. We
Table 1: Summary of the surveyed paper. We categorize these models according to the types of feature selector and classifier.

| Methods           | Feature selector | Classifier               |
|-------------------|------------------|--------------------------|
|                   | Single           | Ensemble                 | Shallow (single or ensemble) | Deep |
| [26]              |                  |                          | ✓                         |
| EID3 [27]         |                  |                          | ✓                         |
| MARK-ELM [28]     |                  |                          | ✓                         |
| HAST-IDS [29]     |                  |                          | ✓                         |
| KitNet [30]       |                  |                          | ✓                         |
| SwiftIDS [12]     |                  |                          | ✓                         |
| H2ID [31]         |                  |                          | ✓                         |
| WIDMoDS [32]      |                  |                          | ✓                         |
| XGBoost-DNN [33]  |                  |                          | ✓                         |
| PSO-KNN [20]      | ✓                |                          | ✓                         |
| Chi-SVM [34]      | ✓                |                          | ✓                         |
| PIO-IDS [10]      | ✓                |                          | ✓                         |
| SMOTE-CFS [19]    | ✓                |                          | ✓                         |
| CFS-BA [21]       | ✓                |                          | ✓                         |
| LSSVM-IDS [9]     | ✓                |                          | ✓                         |
| [22]              |                  |                          | ✓                         |
| NDAE [35]         | ✓                |                          | ✓                         |
| DBN-SVM [18]      | ✓                |                          | ✓                         |
| WFEU-FFDNN [1]    | ✓                |                          | ✓                         |
| AE-IDS [23]       | ✓                |                          | ✓                         |
| DL-SVM [36]       |                  |                          | ✓                         |
| SVM-IDS [24]      | ✓                |                          | ✓                         |
| EFS-DNN           |                  |                          | ✓                         |

Figure 1: LightGBM-based ensemble feature selection. The feature selection algorithm consists of five steps, i.e., sample subsets, train selectors, get rankings, aggregate, and forward search.

Table 2: Acronyms employed in the paper.

| Acronym | Description                      |
|---------|----------------------------------|
| IDS     | Intrusion detection system       |
| DNN     | Deep neural network              |
| BN      | Batch normalization              |
| FC      | Fully connected layer            |
| LightGBM| Light gradient boosting machine   |
| GBDT    | Gradient boosting decision tree  |
| TPR     | True-positive rate               |
| FPR     | False-positive rate              |
| DoS     | Denial of service                |
| R2L     | Remote to local                  |
| U2R     | User to root                     |
utilize LightGBM as the base selector and accumulate feature importance derived from each LightGBM to generate the final feature importance.

3.2. LightGBM. LightGBM is an gradient boosting decision tree (GBDT) algorithm, which is applied in various fields. Given a training set \( \{(x_i, y_i)\}_{i=1}^{n} \), \( x_i \) is the attributes of sample \( i \) and \( y_i \) denotes the target on \( x_i \). LightGBM aims to divide all samples into categories guided by information gain. The loss function for classification is cross-entropy, as follows:

\[
L = -\frac{1}{n} \sum_{i=1}^{n} L_i = -\frac{1}{n} \sum_{i=1}^{n} -y_i \log(p_i),
\]

where \( n \) is the number of samples, \( m \) is the number of categories, \( y_i \) and \( p_i \) indicate ground truth and the predicted probability, respectively.

Besides, the optimization goal of each decision tree in the LightGBM is to minimize the following loss function \( L(y, f(x)) \):

\[
f_t(x) = \arg\min_{f_t} L(t) = \arg\min_{f_t} L(y, \hat{y}_{t-1} + f_t(x)) + \Omega(f_t),
\]

where \( t \) is the iteration number and \( \hat{y}_{t-1} \) is the predicted label in \( t-1 \)th iteration. To overcome the overfitting, we introduce L2 regularization, denoted as \( \Omega(f_t) \).

Note that LightGBM is the ensemble of decision trees, which implies that the feature importance is evaluated according to information gain. Actually, in the training process of LightGBM, we can measure the feature importance by counting the split number of features in each decision tree, as shown below:

\[
\text{Imp}_f = \sum_{i=1}^{n} \text{Imp}_{f,i},
\]

where \( \text{Imp}_{f,i} \) is the feature importance for feature \( f \) in decision tree \( i \), \( n \) is the number of decision trees, and \( \text{Imp}_f \) denotes the accumulated feature importance on feature \( f \) for the LightGBM.

3.3. Feature Selection Framework. Figure 1 illustrates the framework of the proposed LightGBM-based ensemble feature selection algorithm. The framework consists of five steps, i.e., sampling subsets, training base selectors, getting rankings, aggregating rankings, and performing forward search. We will explain these steps in detail as follows.

Firstly, we perform bootstrapping on the original dataset multiple times at a certain sampling ratio to generate subsets, which contain different samples and data distribution. Then, we train base selectors (i.e., LightGBMs) on each sampled subset. Owing to the diversity inherent in different subsets, the training result varies across different base selectors. Therefore, the feature importance derived from each base selector will also be different. In other words, the ranking of feature importance will vary across every base selector. Afterward, we accumulate the feature importance derived from each base selector to obtain the final feature importance. Finally, we apply forward search to select the optimal feature combination.

In the procedure of forward search, we first sort the feature importance in descending order to ensure that the most significant feature is posed in the first position. The sort procedure is necessary since the final feature importance is unordered, which deter the following operations. Then, we accumulate the feature importance from the first one (the most important one) to the last one (the most trivial one). When the accumulated value is greater than a predefined threshold, the forward search process stops. At the time, we combine these features as the optimal subset.

Note that the proposed ensemble feature selection algorithm shares some similarities with bootstrapping-based feature selection techniques, such as the well-known random forest. Compared with these methods, the main difference in the proposed algorithm is the expressiveness of the base selector. In random forest, the feature importance is derived from a single decision tree. In the proposed algorithm, the feature importance is the accumulation of the results from all decision trees in LightGBM, leading to a better generalization and robustness. These properties also ensure that the proposed algorithm can converge faster than random forest.

From the perspective of ensemble learning, when the number of base selectors is large enough, the gap between these two algorithms might be close. However, in the real scenarios, e.g., high-speed networks [12], we should limit the number of base selectors to achieve the trade-off between efficiency and accuracy. In these conditions, the proposed algorithm can achieve better performance with a small number of base selectors.

3.4. Intrusion Detection Classifier. In this section, we present the structure of the proposed DNN classifier. We utilize batch normalization to rescale the range of inputs, improving the robustness of training. Besides, we leverage the embedding technique instead of one-hot encoding to map categorical features to dense vectors, avoiding the sparsity brought by the one hot. The structure of the DNN classifier is illustrated in Figure 2. Empirically, we set the number of layers as 3.

3.5. Batch Normalization. Batch normalization is a universal technique in training neural networks. The technique works since it rescales the inputs in a batch to the learnable scale, which indeed facilitates the procedure of stochastic gradient descend. When all inputs are independent identically distributed in a given scale, the model can converge to the optimal point easily. In batch normalization, we apply Z score to process the input data. Z score normalization rescales the inputs in the normal distribution, as follows:

\[
X' = \alpha \times \frac{X - E(X)}{\sqrt{\text{Std}(X)} + \epsilon} + \beta,
\]

where \( E(X) \) and \( \text{Std}(X) \) are the expectation and standard deviation of \( X \), and \( \alpha, \beta, \epsilon \) are three learnable parameters, which controls the scale and the shift.
3.6. Embedding. In this study, the embedding technique is used to map the representation of categorical features from high-dimensional vector space to low-dimensional vector space. The intuitive preprocessing of categorical features is to encode them into one-hot space. However, the simple approach will lead to the high sparsity of the feature space, especially when the number of values is large. Besides, in the one-hot space, we cannot identify the similarity among values since the distance between all value pairs is the same. Thus, we apply embedding to project the representation in one-hot space into dense vector space. In particular, the more similar the values in semantic space, the closer the distance between them in dense feature space.

3.7. Deep Neural Network. After the optimal subset is selected, we leverage deep neural network (DNN) to perform classification on the subset. Compared with shallow models, such as SVM and KNN, deep neural network learns the latent hidden state among features. The training of DNN is composed of two phases, namely forward propagation and back propagation. In the classification task, the model classifies the category of inputs and computes the loss value through predefined loss function in forward propagation. Afterward, in back propagation, we use the optimizer to update the parameters in the model by applying stochastic gradient descend to reduce the loss value.

The DNN model consists of three layers, i.e., input layer, hidden layer, and output layer. The number of neurons in the input layer is the same as that for input features. The hidden layer consists of multiple layers, where the neuron number in each layer is generally different. The outputs of the previous layer are projected to a novel vector space by applying the nonlinear transformation, as follows:

$$x_{i+1} = \text{ReLU} \left( X_i \cdot W_i + b_i \right),$$  \hspace{1cm} (5)

where the subscript indicates the layer index, $W$ is the parametric weight matrix, $b$ is the bias, and ReLU introduces the nonlinear transformation.

The number of neurons in the output layer depends on the number of categories. In binary-class classification task, the number of neuron in the output layer is 2; in five-class task, the neuron number is 5. The activation function in the final layer is Softmax ($\cdot$), which maps the results of the output layer into probability. After that, we use the Adam optimizer [37] to update the parameters in the model to narrow the gap between the ground truth and the predicted value. The cross-entropy loss is defined as follows:

$$L = -\frac{1}{n} \sum_{i=1}^{n} y_i \cdot \log(\hat{y}_i),$$  \hspace{1cm} (6)

where the $n$ is the number of instances, $y_i$ is the ground truth, and $\hat{y}_i$ is the predicted value.

4. Training

Algorithm 1 demonstrates the steps involved in the training procedure of the proposed EFS-DNN classifier for network intrusion detection. Firstly, we perform min-max normalization on the original dataset to rescale all values to a given range. The preprocessing phase will facilitate the following feature selection and the training of intrusion detection classifier. Then, we leverage the LightGBM-based ensemble feature selection algorithm to compute feature importance scores and select the optimal subset based on that. Note that the feature selection threshold is a hyper-parameter, controlling the number of the selected features. After that, we construct the intrusion detection classifier based on deep neural network with batch normalization and embedding technique. Finally, we utilize the Adam to optimize the parameters in the defined intrusion detection classifier by minimizing the cross-entropy loss between the ground truth and the predicted value.

5. Experiments

5.1. Datasets. We use three public benchmark datasets to evaluate the performance of the proposed EFS-DNN model,
shown in Tables 3 and 4. Note that the data distribution and the attack types are different across these three datasets.

**KDD 99** [38] is collected from the simulated US Air Force LAN, which consists of 9-week network connection traffic data. The original training set of KDD 99 contains \(\approx 5,000,000\) pieces of labeled samples with 41 features. The label on each instance is either “Normal” or “Attack,” where the “Attack” is further divided into four attack types, i.e., DoS, probe, R2L, and U2R. We use KDD 99-10% without redundant instances as the dataset, which contains \(\approx 140,000\) examples.

**NSL-KDD** [39] improves the KDD 99 by solving the inherent problems in KDD 99, e.g., redundancy and data imbalance. The attack types in NSL-KDD are the same as those in KDD 99, whereas the data distribution is different. The NSL-KDD dataset is composed of \(\approx 120,000\) labeled samples.

**UNSW-NB15** [40] is a model dataset generated by the IXIA PerfectStorm tool. Compared with KDD 99 and NSL-KDD, UNSW-NB15 contains more modern attack types (9 types) and more user behaviors. The attributes on the dataset are categorized into five classes, i.e., basic features, flow features, content features, time features, and additional generated features.

5.2. Evaluation Metrics. We use accuracy, TPR, FPR, and F1 score as the indicators to evaluate the performance of all baselines in the paper. The accuracy is used to evaluate the overall correctness of the intrusion detection on both normal traffic flow and anomaly traffic flow. TPR and FPR measure the performance of classification on malware traffic detection and normal traffic detection, respectively. F1 score is the harmonic mean of precision and recall, which tests the overall performance of the intrusion detection models. TP is the number of instances correctly classified as positive, TN is the number of instances correctly classified as negative, FP is the number of instances incorrectly classified as positive, and FN is the number of instances incorrectly classified as negative.

| Attack type | KDD 99 | NSL-KDD |
|-------------|--------|---------|
| Normal      | 87832  | 67343   |
| DoS         | 54572  | 45927   |
| Probe       | 2131   | 11656   |
| U2R         | 999    | 995     |
| R2L         | 52     | 52      |

Table 3: Statistics of KDD 99 and NSL-KDD.
Table 4: Statistics of UNSW-NB15.

| Attack type | Train Size | Distribution (%) | Test Size | Distribution (%) |
|-------------|------------|------------------|-----------|------------------|
| Normal      | 56000      | 31.94            | 37000     | 44.94            |
| Generic     | 40000      | 22.81            | 18871     | 22.92            |
| Exploits    | 33393      | 19.04            | 11132     | 13.52            |
| Fuzzers     | 18184      | 10.37            | 6062      | 7.36             |
| DoS         | 12264      | 6.99             | 4089      | 4.97             |
| Reconnaissance | 10491     | 5.98             | 3496      | 4.25             |
| Analysis    | 2000       | 1.14             | 677       | 0.82             |
| Backdoor    | 1746       | 1.00             | 583       | 0.71             |
| Shellcode   | 1133       | 0.65             | 378       | 0.46             |
| Worms       | 130        | 0.07             | 44        | 0.05             |

5.3. Comparison Experiments. In this section, we compare the proposed EFS-DNN against existing intrusion detection models on three public datasets. We conduct binary-class classification on these three datasets to evaluate the superiority of EFS-DNN on detecting anomaly traffic in networks. Then, we conduct five-class classification on KDD 99 dataset to demonstrate the capability of detecting certain types of attacks of the proposed model. For fair comparison, we report the average score over 10 random seeds.

Table 5 shows the performance of EFS-DNN against various baselines on KDD 99 in terms of accuracy, TPR, and FPR. The EFS-DNN outperforms all baselines even though they use more features to predict the category of the instances. The observation demonstrates that by applying ensemble feature selection and deep neural network, the proposed EFS-DNN obtains performance gain and drops redundant features. Compared with SVM-IDS, an ensemble feature selection-based model, EFS-DNN selects fewer features and achieves better accuracy. This is because EFS-DNN leverages the powerful LightGBM as the base selector and utilizes the expressive deep neural network as the classifier, whereas SVM-IDS ensembles three information gain-based algorithms to select features and uses SVM, a shallow model, as the classifier.

Table 6 shows the experimental results for binary-class classification task on UNSW-NB15. The results presented show that the EFS-DNN outperforms baselines in all indicators but TPR. Note that EFS-DNN achieves the highest F1 score and the lowest FPR compared with other baselines. This suggests that the model effectively detects modern attacks in the low alarm rate, showing the potential of deployment in real networks.

5.4. Hyper-Parameter Sensitivity. In this section, we evaluate the impact of the ensemble feature selection module in the EFS-DNN. Figure 3 illustrates the accuracy of EFS-DNN on three public datasets with different feature selection thresholds. The color of the lines indicates the feature.
Table 7: Binary-class classification performance on UNSW-NB15 in terms of TPR (%), FPR (%), Acc (%), and F1 score (%). EFS-DNN outperforms all baselines when the number of selected features is 21.

| Method                          | TPR  | FPR  | Acc  | F1 score | # Features |
|---------------------------------|------|------|------|----------|------------|
| Rule-based [43]                 | 88.91| 12.52| 88.28| 87.01    | 13         |
| GWO [11]                        | 93.80| 20.95| 85.68| 85.48    | 20         |
| PSO and GWO [11]               | 80.83| 23.27| 78.58| 77.25    | 12         |
| FFA and GWO [11]               | 94.15| 24.13| 83.99| 83.93    | 15         |
| FFDNN [1]                      | n/a  | n/a  | 87.10| n/a      | 21         |
| EFS-DNN                        | 87.54| 12.46| 88.34| 88.14    | 15         |

Table 8: Five-class classification results on KDD 99 in terms of Acc (%). EFS-DNN outperforms all baselines.

| Method                          | Normal | DoS   | Probe | R2L   | U2R   |
|---------------------------------|--------|-------|-------|-------|-------|
| Flexible neural tree [44]       | 99.19  | 98.75 | 98.39 | 99.09 | 99.70 |
| Radial SVM [45]                 | n/a    | 98.94 | 97.11 | 97.78 | 97.80 |
| IPSO-RBF [46]                   | n/a    | 93.95 | 95.80 | 96.02 | 95.20 |
| LSSVM-IDS + FMIFS [9]           | 99.79  | 99.86 | 99.91 | 99.92 | 99.97 |
| KNN + PSO [20]                  | 99.57  | 99.91 | 94.41 | 99.73 | 99.77 |
| SVM-IDS [24]                    | 98.95  | 98.99 | 95.36 | 98.76 | 99.01 |
| EFS-DNN                         | 99.82  | 99.96 | 99.95 | 99.90 | 99.97 |

Table 9: Five-class classification results in terms of TPR (%) and FPR (%) on KDD 99 dataset.

| Method                          | DoS   | Probe | R2L   | U2R   |
|---------------------------------|-------|-------|-------|-------|
| ID3 [27]                        | 99.90 | 0.03  | 99.70 | 0.55  | 93.50 | 0.98  | 49.10 | 0.15 |
| MARK-LEM [28]                   | 99.96 | 0.02  | 97.30 | 0.03  | 94.94 | 0.05  | 62.87 | 0.01 |
| HAST-IDS [29]                   | 99.10 | 0.02  | 83.35 | 0.01  | 74.19 | 0.02  | 64.25 | 0.02 |
| SwiftIDS [12]                   | 99.96 | 0.02  | 98.20 | 0.00  | 89.40 | 0.01  | 57.10 | 0.01 |
| EFS-DNN                         | 99.97 | 0.02  | 98.79 | 0.00  | 98.84 | 0.09  | 66.67 | 0.01 |

Figure 3: Continued.
Figure 3: Binary-class classification performance of EFS-DNN in terms of accuracy on three benchmark datasets under different feature selection thresholds and different sample ratios. (a) KDD 99. (b) NSL-KDD. (c) UNSW-NB15.

Figure 4: Continued.
Figure 4: Continued.
selection threshold. For these three datasets, the EFS-DNN obtains the trade-off between the accuracy and the number of selected features when the threshold is 0.8. Besides, we note that the sample ratio of the base selector is crucial for the performance of EFS-DNN. The best sample ratio is 0.6 for KDD 99 and NSL-KDD and 0.8 for UNSW-NB15.

Figures 3(a) and 3(b) illustrate the binary-class classification results in terms of accuracy on KDD 99 and NSL-KDD under different feature selection thresholds. Note that when the sample ratio in feature selection is set 1.0, the model fails to achieve the best results. The observation proves the feasibility of the generalization of the ensemble principle in the filter-based feature selection technique. Figure 3(c) shows the experimental results on UNSW-NB15 in terms of accuracy. Compared with KDD 99 and NSL-KDD, UNSW-NB15 contains more types of attacks and more features. When the feature selection threshold is 0.6 or 0.7, the EFS-DNN achieves the best performance with the 1.0 sample ratio. When the threshold is 0.8 or 0.9, the best performance appears when the sample ratio is 0.8. This is because when the number of selected features is too small, they cannot express the complicated attributes of traffic flow.

To evaluate the influence of ensemble feature selection on multi-class tasks, we conduct five-class classification task on NSL-KDD with four indicators, i.e., accuracy, TPR, FPR, and F1 score. The reason why we do not choose KDD 99 is the imbalanced data distribution. Figure 4 shows the experimental results of EFS-DNN under different feature selection thresholds. From Figure 4(a), we observe that the accuracy of the model on normal, DoS, and probe increases with the raise of threshold, whereas the accuracy on R2L and U2R remains at a high level. The reason is as follows: (1) more features provide more useful information to identify the types of traffic data, and (2) the accuracy fails to indicate the performance of models when the data distribution is imbalanced. Figures 4(b) and 4(c) illustrate the performance of the EFS-DNN on TPR and FPR. Unlike accuracy, we observe a clear performance gain on all traffic types with the increase in the number of selected features. When the threshold is set at 0.8, the model achieves a desirable performance. At the threshold, the low FPR denotes the low power dissipation and the high TPR indicates the high accuracy of intrusion detection. Figure 4(d) shows the experimental results on NSL-KDD in terms of F1 score. The metric is used to evaluate the overall performance of the model. As we discussed before, the model achieves the best trade-off between the performance and the number of selected features when the threshold is 0.8.

6. Conclusion

An intrusion detection system aims to detect potential attacks in networks. With the development of network scale, existing methods cannot satisfy the real-time requirement owing to the large volume of traffic flow in networks. Some researchers leverage feature selection technologies to mitigate the issue by reducing the time consumption in the decision-making phase. However, these feature selection-based methods still suffer from the lack of robustness or the high computational complexity. To overcome the challenges, in this study, we propose a novel ensemble feature selection-based intrusion detection system algorithm named EFS-DNN to efficiently detect the anomaly behaviors in the networks. The proposed EFS-DNN consists of two modules, i.e., ensemble feature selector and classifier. The ensemble feature selection algorithm leverages LightGBM as the base selector to improve the robustness of the selected subset. The classifier applies a deep neural network to capture the latent
relations among the selected features to enhance the efficacy. We conduct extensive experiments on three public datasets, i.e., KDD 99, NSL-KDD, and UNSW-NB15, to evaluate the performance of the EFS-DNN in terms of accuracy, TPR, FPR, and F1 score. The experimental results show that the ensemble feature selection technique has the capability of improving the stability and robustness of the selected feature combination.

The limitations of the study are as follows: (1) we only utilize LightGBM as the base selector, ignoring the other feature selection techniques, and (2) we apply deep learning to perform intrusion detection, which lacks explainability. Therefore, in the future, we will attempt to study the potential of heterogeneous ensemble feature selection in intrusion detection systems to further improve expressiveness and leverage XAI techniques [47, 48] to improve the interpretability and explainability of the IDS algorithms.

Data Availability

The data and material are available at https://github.com/Dece-brove/EFS-DNN.git.

Conflicts of Interest

The authors report no conflicts of interest. The authors alone are responsible for the content and writing of this article.

Acknowledgments

This work was supported by the Joint Funds of the Zhejiang Provincial Natural Science Foundation of China under grant no. LZY21F020001.

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, fia 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. 89, 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. Choraş 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,” in International Conference on Network and System Security, pp. 277–291, Springer, New York, NY, USA, 2020.

[20] A. R. Syarif and W. Gata, “Intrusion detection system using hybrid binary PSO and K-nearest neighborhood algorithm,” 2017 11th International Conference on Information & Communication Technology and System (ICTS), 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," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc, 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, "MARKELM: 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. Doitschin, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," 2022, http://arxiv.org/abs/1802.09089.

[31] G. Bovenzi, G. Aceto, D. Ciouonzo, V. Persico, and A. Pescape, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *Proceedings of the GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pp. 1–7, IEEE, Taipei, Taiwan, December 2020.

[32] T. Feng and M. Dou, "A weighted intrusion detection model of dynamic selection," *Applied Intelligence*, vol. 51, no. 7, pp. 4860–4873, 2021.

[33] S. T. Ikram, A. K. Cherukuri, B. Poorva et al., "Anomaly detection using XBoost ensemble of deep neural network models," *Cybernetics and Information Technologies*, vol. 21, no. 3, pp. 175–188, 2021.

[34] I. Sumaiya Thaseen and C. Aswani Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class SVM," *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 4, pp. 462–472, 2017.

[35] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[36] S. N. Mighan and M. Kahani, "A novel scalable intrusion detection system based on deep learning," *International Journal of Information Security*, vol. 20, no. 3, pp. 387–403, 2021.

[37] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2022, http://arxiv.org/abs/1412.6980.

[38] S. J. Stolfo, W. Wei Fan, W. Wenke Lee, A. Prodromidis, and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: results from the JAM project," *Proceedings DARPA Information Survivability Conference and Exposition. DIS-CEX’00*, vol. 2, pp. 130–144, 2000.

[39] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6, IEEE, Ottawa, ON, Canada, July, 2009.

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

[41] P. Gogoi, M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Packet and flow based network intrusion dataset," in *Communications in Computer and Information Science*, M. Parashar, D. Kaushik, O. F. Rana, R. Samtaney, Y. Yang, and A. Zomaya, Eds., pp. 322–334, Springer, Berlin Heidelberg, 2012.

[42] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[43] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, "An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset," *Cluster Computing*, vol. 23, no. 2, pp. 1397–1418, 2020.

[44] Y. Chen, A. Abraham, and B. Yang, "Feature selection and classification using flexible neural tree," *Neurocomputing*, vol. 70, no. 1, pp. 305–313, 2006.

[45] A. M. Chandrasekhar and K. Raghuveer, "An effective technique for intrusion detection using neuro-fuzzy and radial SVM classifier," in *Computer Networks & Communications (NetCom)*, N. Chaki, N. Meghanathan, and D. Nagamalai, Eds., pp. 499–507, Springer, New York, NY, USA, 2013.

[46] W. Xing-zhu, "Network intrusion prediction model based on RBF features classification," *International Journal of Security and Its Applications*, vol. 10, no. 4, pp. 241–248, 2016.

[47] A. Adadi and M. Berrada, "Peeking inside the black-box: a survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[48] A. Nascita, A. Montieri, G. Aceto, D. Ciouonzo, V. Persico, and A. Pescape, "XAI meets mobile traffic classification: understanding and improving multimodal deep learning architectures," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4225–4246, 2021.