Feature Selection Algorithm for Intrusions Detection System using Sequential Forward Search and Random Forest Classifier

Jinlee Lee[1], Dooho Park[2] and Changhoon Lee[3]
[1,3] Division of Computer Science and Engineering, Konkuk University
Seoul, Republic of Korea
[e-mail: {jini0910,chlee}@konkuk.ac.kr]
[2] Intelligent Service Development Team, XIIlab Co. Ltd
Seoul, Republic of Korea
[e-mail: pakddo@gmail.com]
*Corresponding author: Changhoon Lee

Received May 8, 2017; revised August 16, 2017; accepted August 29, 2017; published October 31, 2017

Abstract

Cyber attacks are evolving commensurate with recent developments in information security technology. Intrusion detection systems collect various types of data from computers and networks to detect security threats and analyze the attack information. The large amount of data examined make the large number of computations and low detection rates problematic. Feature selection is expected to improve the classification performance and provide faster and more cost-effective results. Despite the various feature selection studies conducted for intrusion detection systems, it is difficult to automate feature selection because it is based on the knowledge of security experts. This paper proposes a feature selection technique to overcome the performance problems of intrusion detection systems. Focusing on feature selection, the first phase of the proposed system aims at constructing a feature subset using a sequential forward floating search (SFFS) to downsize the dimension of the variables. The second phase constructs a classification model with the selected feature subset using a random forest classifier (RFC) and evaluates the classification accuracy. Experiments were conducted with the NSL-KDD dataset using SFFS-RF, and the results indicated that feature selection techniques are a necessary preprocessing step to improve the overall system performance in systems that handle large datasets. They also verified that SFFS-RF could be used for data classification. In conclusion, SFFS-RF could be the key to improving the classification model performance in machine learning.

Keywords: FeatureSelection, SFFS, RandomForest, IDS

A preliminary version of this paper was presented at ICONI 2016, December 16-19, Hong Kong, and was selected as an outstanding paper. This version includes a concrete analysis of algorithm and supporting experimental results on intrusion detection system data.

1. Introduction

AN intrusion detection system (IDS) is designed to protect internal resources from unauthorized users or services attempting to access internal networks or internal information. Recently, they have been used to prevent malicious code infections in applications and services, and to protect against attackers who want to gain access privileges such as in firewalls, antivirus software, and networks. To provide effective intrusion detection in systems, many types of studies have been conducted to generate rules based on the knowledge of security experts and analyze systems by applying machine-learning techniques. In an IDS, the data analyzed and rules applied are generally referred to as features. These features are extracted from a large amount of data and an analysis model is generated. Because each process performs operations based on a large amount of data, there are disadvantages, including features that do not affect the system or are redundant [1]. Using a feature selection technique for machine learning is a way to solve these problems. Feature selection (FS) is a necessary process when analyzing high-dimensional datasets. When dealing with high-level composite data, FS plays a key role in improving the efficiency and accuracy of the results in the field of big-data analysis. In data classification, feature selection can be defined as the problem of selecting the combination of features that maximizes the performance of the trained classifier prediction [2]. Among FS techniques, the sequential forward floating selection (SFFS) technique has the advantage of increasing the prediction performance of the classifier by eliminating any characteristic that degrades the performance.

However, there are disadvantages. When using a single classifier as the objective function in FS, an improvement in the classification performance is not significant. In addition, the performance differs with the classifier and dataset type and characteristics. Thus, an over-fitted learning result can be derived. In order to address this problem, we modified the SFFS algorithm using the random forest classifier. The random forest classifier (RFC), which has a high classification performance, creates multiple decision trees using different features and combines trees using ensemble techniques.

In this study, we proposed an optimization algorithm that combines SFFS techniques with an RFC for an IDS. We use the RFC to evaluate the prediction accuracy for FS. The proposed algorithm can take advantage of the complementary benefits of the FS method and RFC. This leads to the selection of the ideal set of the best features, which is expected to improve the performance of classification models.

This paper is organized as follows. Following the introduction to our work in section 1, section 2 introduces the IDS dataset and addresses the shortcomings of feature-selection techniques and the RFC. Section 3 focuses on the way the proposed algorithm will work. Section 4 describes experiments that were performed to verify the methodology and evaluates the results.

2. Related Work

This section describes the dataset for the intrusion detection system, and the principles of the FS technique and RFC. We identify the strengths and weaknesses of each technique.

2.1 Data of Intrusion Detection System

An IDS is used to detect and respond to an intrusion by monitoring the network. With the increases in the number and variety of attacks, it is beginning to include not only monitoring the network but also monitoring the computers where the malicious code is executed. Currently, an IDS inspects all of the data to detect a misuse or an intrusion. Because of the large amount of data that need to be examined, the analytical techniques used in an IDS are complex and difficult to achieve. As a result, the IDS needs to reduce the amount of data it handles. This is an important part of building a real-time detection system. Data reduction can be achieved through data filtering, FS, and so on. In general, the analysis standard used in an IDS is called a feature. Because the number of features increases exponentially with to the size of the system, the amount used in the operation of the IDS increases proportionally. In addition, not all features have a critical impact on the detection model, and some are also factors in performance degradation [3]. Therefore, if we can derive a small number of feature subsets that are essential for intrusion detection by applying FS techniques, it will be possible to detect intrusions more efficiently by reducing the amount of computation needed in the IDS.

Table 1. Datasets

	Total	**Normal**	**Abnormal**
Training data	125,973	67,343	58,630
Test data	22,544	9,711	12,833

An IDS operates by analyzing data composed of packets collected from the network or logs of the system. Many studies have used KDD(Knowledge Discovery and Data Mining)'99 datasets to measure the performances of FS techniques for an IDS. The KDD'99 data consist of approximately 5 million pieces of training data and 300,000 pieces of test data, and the attack method is distinguished using four categories. However, the KDD'99 dataset is too large to analyze all the data, and there are many redundant pieces of data, which can cause the analysis result to be biased. Tavallaee et al. proposed the NSL_KDD dataset to solve the problems of the KDD'99 dataset [4]. By eliminating the redundancy of the data, the system prevented the bias caused by the high-frequency data. In addition, by limiting the amount of training and test data, the experiment made it possible to use the whole set [5]. **Table 1** lists the data type details for the NSL-KDD dataset, along with the number of individual instances and records in both the training and test sets. The training and test datasets include 41 features classified as normal traffic and specific attack types. All of the features are subdivided into four categories: basic features, time-based traffic features, content features, and host-based traffic features.

All categories are described below.

- Basic features: Protocol type, including all derived from the TCP / IP connection featur es such as service duration.
- Time-based Traffic Features: It is used to capture mature features over a 2 second time window (eg count, srv_count, serror_rate, etc.).

- Content Features: Use the domain feature to access the payload of the original TCP pac ket (for example, hot, num_root, is_guest_login, and so on).
- Host-based traffic feature: All attacks with a 2 second interval (eg, dst_host_count, dst _host_srv_count, etc.) with the same destination host as the current connection are acce ssed using this feature.

The classes or labels of the NSL KDD dataset are divided into four categories: denial of service (DoS), probe, remote to local (R2L), and user to root (U2R), representing the attack class and normal traffic. **Table 2** lists the details of the features. In this paper, we define the abnormal or normal detection problem as a binary classification problem. Furthermore, purpose on our suggestion is to determinate that the each feature of NSL-KDD for classification model is useful or not.

Table 2. The features of NSL-KDD datasets

No	Name	Type	No	Name	Type
1	Duration	continuous	22	is_guest_login	discrete
2	protocol_type	discrete	23	Count	continuous
3	Service	discrete	24	srv_count	continuous
4	Flag	discrete	25	serror_rate	continuous
5	src_bytes	continuous	26	srv_serror_rate	continuous
6	dst_bytes	continuous	27	rerror_rate	continuous
7	Land	discrete	28	srv_rerror_rate	continuous
8	wrong_fragment	continuous	29	same_srv_rate	continuous
9	Urgent	continuous	30	diff_srv_rate	continuous
10	Hot	continuous	31	srv_diff_host_rate	continuous
11	num_failed_logins	continuous	32	dst_host_count	continuous
12	logged_in	discrete	33	dst_host_srv_count	continuous
13	num_compromised	continuous	34	dst_host_same_srv_rate	continuous
14	root_shell	discrete	35	dst_host_diff_srv_rate	continuous
15	su_attempted	discrete	36	dst_host_same_src_port_rate	continuous
16	num_root	continuous	37	dst_host_srv_diff_host_rate	continuous
17	num_file_creations	continuous	38	dst_host_serror_rate	continuous
18	num_shells	continuous	39	dst_host_srv_serror_rate	continuous
19	num_access_files	continuous	40	dst_host_rerror_rate	continuous
20	num_outbound_cmds	continuous	41	dst_host_srv_rerror_rate	continuous
21	is_host_login	discrete	42	Label	discrete

2.2 Features Selection

The FS method is a technique used to enhance the performance by screening the characteristics used in machine learning. Using FS techniques reduces the number of features used for learning and selects high discrimination features in the FS process. In addition, FS helps to improve the accuracy by selecting the optimal features. It aims to reduce the amount of processing by reducing the number of features and improves the classification accuracy [6]. FS techniques can be divided into two types: the filter and wrapper types. In order to select features, filter type techniques only evaluate the individual characteristics. Wrapper type approaches use the classification performance of the classifiers as a numerical evaluation.

There are two typical methods: sequential forward selection (SFS) and SFFS. The SFS approach starts with an empty set of features. After each step, the classification accuracy for each feature is measured, and the most accurate features are added the feature set. This is repeated until all of the features are compared [7]. SFS shows better performance for a dataset with multiple features. The disadvantage of SFS is that an increase in the number of features increases the computational complexity and makes it more difficult to remove the added features. The SFFS approach is similar to SFS in that it starts with an empty set and selects the optimal feature in each sequential step. However, after adding a feature, it returns to the previous step to select the worst feature and removes it from the optimal set. This step can complement the disadvantages of SFS. By reducing the extraneous features, it has the benefit of reducing the generalization error of the model and increasing the computational efficiency. However, when it is used as a single classifier, SFFS cannot significantly improve the performance. Therefore, it needs to supplement other classification algorithms. In this study, we used the SFFS method to search for and select a specific set of features. Moreover, in order to overcome the limitations of the FS method, we combined SFFS with the RFC.

From the perspective of IDS, studies are being conducted to apply FS to improve the detection performance in the field. Not all of the features included in the data collected by the IDS contribute to the improvement of the classification model. There are features that have redundancy or noise. The FS process improves the classification performance by eliminating nonsensical features from the data and reducing noise. In addition, the selection process helps distinguish the features that affect the analysis results, which assists in interpreting the results. In particular, because the feature dimension of the data is reduced, rather than using all the data, an improved detection model can be constructed. Zaman [11] proposed an FS method using the support vector machine (SVM) approach for IDSs. They applied the forward selection ranking and backward elimination ranking algorithms in the feature search process and evaluated feature weights using the enhanced support vector decision function. They performed experiments based on the KDD'99 Cup data and proved that their method was superior to other FS techniques. However, because their experiment used only 6,000 data points from the KDD'99 Cup data, which include about 4 million instances, their method had the disadvantage that the accuracy of the detection model could change when a large amount of data is processed. To address this problem, we used the entire NSL-KDD dataset.

2.3 Random Forest Classifier

The RFC is a multiple decision tree classifier that uses an ensemble method [8]. It creates small subsets of datasets (bootstrap) from the overall dataset, along with numerous decision trees based on each sample. The first phase begins with learning data, as defined in formulas (1)–(3).

$$X = (x_1, y_1), \cdots, (x_N, y_N) \tag{1}$$

$$x_i = \left[x_i^1, \cdots, x_i^D\right]^T \tag{2}$$

$$y_i \in 1, \cdots, K \tag{3}$$

Here, D indicates the depth of the tree and X is made up of x_i and y_i, where $x_i$is a feature and y_i is a class. After selecting one dimension of x_i, the maximum and minimum values of the data are determined, and a threshold is set arbitrarily within the min–max range. This threshold is used to divide the data and calculate the impurity based on the data being distributed.

There are several ways to calculate the impurity. Well-known methods include the Gini coefficient and entropy. The formulas for obtaining the impurity are shown in (4) and (5).

$$im(T) = -\sum_{i=1}^{M} P(w_i|T) log_2 P(w_i|T) \tag{4}$$

$$m(T) = 1 - \sum_{i=1}^{M} P(w_i|T)^2 = \sum_{i \neq j} P(w_i|T)P(w_i|T) \tag{5}$$

Here, w_i represents the i-th variant among the types of M variants, and $P(w_i|T)$ is the probability at node T. The entropy has the lowest level of impurity when it has a value of zero, and the impurity has a minimum value when the probability has a value of one. The value of the gain is computed as shown in (6).

$$\Delta im(T) = im(T) - \frac{|X_L|}{|X_T|} im(T_L) - \frac{|X_R|}{|X_T|} im(T_R) \tag{6}$$

A larger gain value indicates better classification. Next, the threshold value is stored, and the dimension at the root node is selected. The obtained gain value is divided at each node. Node splitting is repeated in the tree until the depth of the tree is maximized. In the resultant tree, the final node is called the leaf node. The number of trees is determined by the user. The dimensions and threshold are established at random; therefore, there is a small chance that the new tree could be in the same form as the existing tree. Inputs of small amounts of data will eventually reach the leaf node of each tree, and the probability value of the leaf node becomes the probability value of the tree. To obtain the learning process results, the RFC creates several trees, as previously mentioned, and aggregates their probabilities. This is a key idea of the RFC, and is called bagging (bootstrap aggregating). As previously explained, the RFC determines the overall classification accuracy after generating a large number of decision-making models. Because it has the ensemble step, the RFC can avoid over-fitting [9].

On the other hand, the RFC has the disadvantage that a selected feature group can affect its classification accuracy, which will vary with different training data. In addition, because of the hierarchical structure, the decision tree (DT) could easily be over-fitted to a specific input dataset. The DT could also select unnecessary features. Because of its decreased generality, the RFC cannot efficiently deal with the new data. Thus, its generality needs to be ensured. In particular, the classification result of the RFC is difficult to understand intuitively. The relationships between all of the features are important to improve the accuracy of general classifiers. Moreover, it is difficult to interpret the method of creating a prediction model [10].

To address this problem, this study ensured the generality of the RFC using an FS based on SFFS.

3. SFFS-RF Feature Selection Algorithm

3.1 SFFS-Random Forest Complementary Strategy

IDSs need to accurately detect large amounts of data with many features. Therefore, the performance of the threat detection model greatly affects the detection result processing. In order to improve the performance of the threat detection model, an important role is played by FS techniques to distinguish features that affect the detection results from features that interfere with the detection. The key part of FS is the step of generating the appropriate feature subset and the evaluation step of comparing the feature subset to the previously generated subset. This requires an evaluation metric that determines the suitability of a feature subset from the perspective of a particular data-mining task such as classification [11].

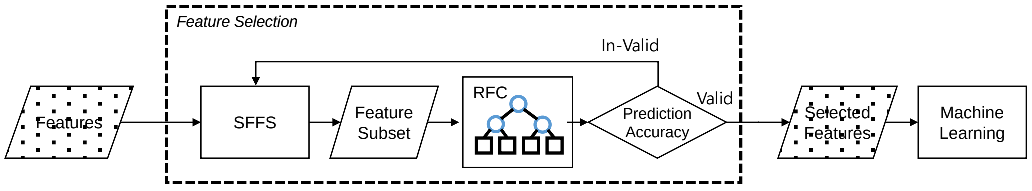

Fig. 1. Overview of SFFS-RF

In this study, the SFFS-RF FS technique was applied as the FS method for IDS based on machine learning. **Fig. 1** shows the SFFS-RF FS process. The FS process consists of a search strategy for generating a new feature subset, an evaluation method for evaluating the generated feature subset, and a stop criterion for terminating the FS process. SFFS-RF uses an RFC to generate a feature subset using a sequential floating forward search algorithm and measure the performance of each subset. The SFFS-RF FS method is a method for searching all the features and has the advantage of accurately determining the importance of the features. This makes it possible to eliminate unnecessary and duplicate features. The features contributing to the classification accuracy are expected to be capable of excluding unnecessary features. In particular, it is suitable for IDS because it has good performance when there are many data and classification objects.

FS methods enhance the classification performance by screening the characteristics used in machine learning. Using the FS techniques, SFFS reduces the unnecessary features; however, it creates a single model with a subset of features. When used as a single classifier, the classification performance decreases. Therefore, a performance improvement can be expected when using multiple classifiers to compensate for the SFFS techniques. Because the RFC creates derivative trees with randomly selected features, it has the characteristic of multiple classifiers. However, because all of the features are used for learning, it has the disadvantage that the learning process may include redundant features or unrelated characteristics. The two techniques have limitations when used individually; however, they could complement each other. Therefore, in this paper, we propose the use of an SFFS-RF algorithm.

3.2 SFFS-RF

The method proposed in this paper consists of a combination of the FS method and RFC. The SFFS-based FS method is used with the RFC machine learning algorithm. **Fig. 2** shows the overall procedure of the algorithm proposed in this paper.

The FS process consists of a search strategy for generating a new feature subset, an evaluation method for evaluating the generated feature subset, and a stop criterion for terminating the FS process. SFFS-RF uses an RFC to generate a feature subset using a sequential floating forward search algorithm and measure the performance of each subset. The SFFS-RF FS method is used to search all the features and has the advantage of accurately determining the importance of each feature. This makes it possible to eliminate unnecessary and duplicate features. The features contributing to the classification accuracy are expected to be capable of excluding unnecessary features.

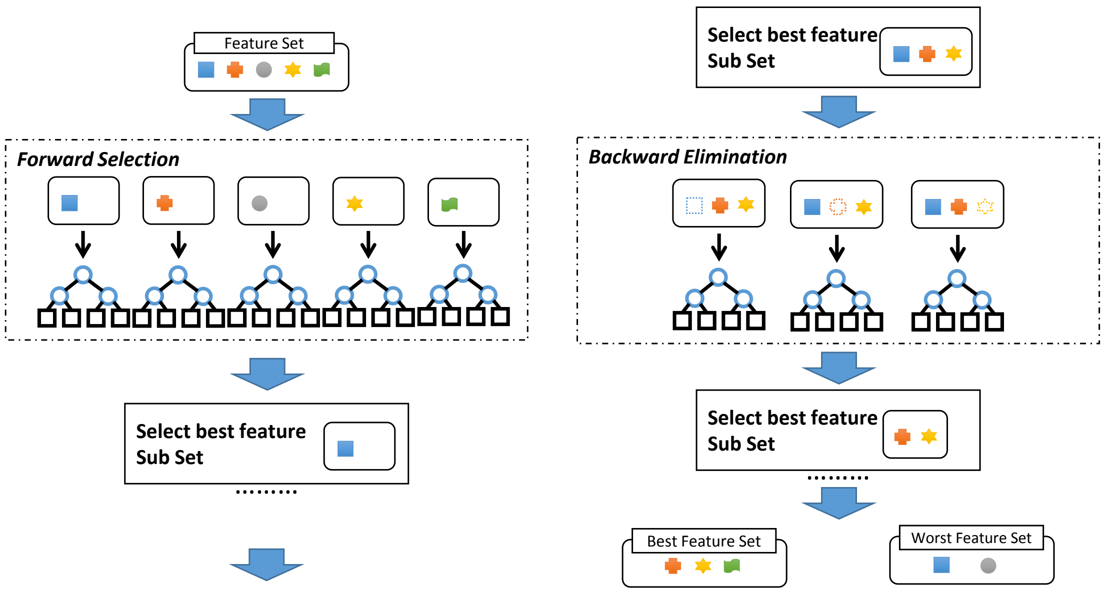
Feature Set
Forward Selection
Select best feature Sub Set
Select best feature Sub Set
Backward Elimination
Select best feature Sub Set
Best Feature Set
Worst Feature Set

Fig. 2. SFFS-RF process

```
Input:
All features {F} := (f_1, f_2 ...)
Optimal features without redundant and irrelevant features {F_O}
A training set {S} := (x_1, y_1), ...., (x_n, y_n) and number of trees in forest T

Code:
function featureOptimization(){
    initialize{F_O} = empty;
    for i = 1 to number of{F} //Loop SFFS
            //get best accuracy with f_i
        ACCURACY = randomForest({F_O} + f_i)
        if (ACCURACY > MAX_ACCURACY)
            MAX_ACCURACY = ACCURACY
            {F_O} = {F_O} + f_i
        endif

        for k = 1 to number of{F_o} // SFFS Backtrack
            //get best accuracy without f_k
            ACCURACY = randomForest({F_O} - f_k)
            if (ACCURACY > MAX_ACCURACY)
                MAX_ACCURACY = ACCURACY
                {F_O} = {F_O} - f_k
            endif
        endfor
    endfor
    return{F_O}
};
```

Fig. 3. Pseudo Code I

In the first step, using SFFS-RF, we eliminate redundant or duplicate features and extract the key features; this process involves selecting a subset of highly rated features. In this process, the RFC creates a predictive model using each feature and repeated training. When a new feature is added, the prediction model is used to determine whether the classification performance is improved or not. **Fig. 3** and **Fig. 4** illustrate the operation of the SFFS-RF algorithm.

After obtaining a set of characteristics, the final classification model is generated in the second phase; this process involves learning with selected features.

```
function randomForest($\{F_O\}$) {
        for $j = 1$ to $T$
            $\{S'\}$ = bootstrap sample from $\{S\}$
            $\{F_O'\}$ = random feature from $\{F_O\}$
            $C_i$ = buildDecisionTree($\{S'\}, \{F_O'\}$)
            $C^*(x) = C^*(x) + C_i$
        end for

        $C^*(x) = \underset{y \in Y}{\arg\max} \sum i : C_i(x) = y$
    return $C^*(x)$ //ACCURACY of TREES
};
```

```
function buildDecisionTree($\{S'\}, \{F_O'\}$){
    Create treeNode $\mathcal{L}_t$ for $\{S'\}, \{F_O'\}$
    if the stopping criterion is met for LEARNED TREE then
        LEARNED TREE = t
    else
        Find the split on $\mathcal{L}_t$ that maximizes impurity decrease
        $S^* = \underset{s \in \varrho}{\arg\max} \, \Delta i(s, t)$
        Partition $\mathcal{L}_t$ into $\mathcal{L}_{tL} \cup \mathcal{L}_{tR}$ according to $S^*$
            $t_L$ = buildDecisionTree($\mathcal{L}_L$)
            $t_R$ = buildDecisionTree($\mathcal{L}_R$)
    end if
    return LEARNED TREE
};
```

Fig. 4. Pseudo Code II

Using the proposed FS algorithm, we include features that contribute to the classification accuracy and exclude unnecessary features. Moreover, because it can clearly identify the characteristics that affect the results, SFFS-RF can solve the problem that has been shown to be a disadvantage of the RFC. This should make it possible to find classification models with better performances.

4. Experiments and Verifications

4.1 Experimental Data and Environments

This section describes the data used for the performance evaluation and evaluates the performance of the FS technique for the proposed IDS.

The purpose of this experiment was to understand the effect of the proposed FS technique on the accuracy improvement of the IDS. Java and machine learning were used to implement the experimental program, and the detection steps were performed using WEKA, an open source machine-learning tool. **Table 3** lists the experimental environment information.

Table 3. Experimental Environment

	Description
CPU	Intel i7-4790 3.6Ghz
RAM	32GB
OS	Windows 10
DATA	NSL-KDD[4]

In order to verify the superiority of the FS technique for the proposed IDS, the detection rate, error rate, and time required to select the features from the training data were compared with those for another FS technique.

The other FS techniques were techniques that combined the feature subset search technique and feature evaluation index, as described below.

Feature subset generation technique:

- GeneticSearch : Feature subset search using genetic algorithm
- BestFirst : Greedy hill-climbing search using backtracking

Feature subset evaluation technique:

- CfsSubsetEval[12] : Evaluate feature subset based on correlation relation coefficient
- ConsistencySubsetEval[13] : Measuring and evaluating the level of consistency for the entire class of feature subsets

1) Comparison of General Algorithms for FS by Number of Features

The features extracted using the proposed SFFS-RF FS technique were selected from 10 features that accounted for approximately 25% of the total features. **Table 4** lists the features derived using the FS techniques.

Table 4. – Feature selection result on the NSL-KDD dataset

Method		**No. of Features**	**Selected Features**
Search	**Evaluator**		
BestFirst	CfsSubsetEval	8	{4, 5, 6, 12, 26, 29, 30, 37}
SFFS	RF	10	{5, 3, 38, 6, 40, 39, 1, 4, 29, 32}
GeneticSearch	ConsistencySubsetEval	19	{1, 3, 5, 6, 7, 8, 9, 13, 17, 18, 21, 23, 27, 29, 34, 36, 37, 38, 40}

2) Comparison of General Algorithms for FS by Classification Accuracy

In order to verify the performance, experiments were conducted to learn the C4.5 classifier based on the feature set derived by the FS technique and evaluate the result. The true positive (TP), false negative (FN), true negative (TN), and false positive (FP) values were used to evaluate the performance of the classifier. In addition, a 10-fold cross validation method was used to minimize the influence of the data on the experimental results. Generally, a higher performance for the classification model was associated with a higher probability of a high detection rate. The detection rate can also be expressed as a reproducible rate (recall). This means that recall is the actual attack rate among the detection results of the malicious data. The higher the value, the more the malicious data can be correctly classified. The rate of detection refers to the ratio of the total number of counts and the number of counts in the total number of units, and also expresses the accuracy[14]. In the case of detecting malicious data from malicious systems, the detection of false alarms is difficult to determine. Thus, the test uses false positives to evaluate the performance of this criterion. The formulas for the detection, accuracy, and opacity of the detector are as follows.

$$\text{Detection Rate} = \frac{TP}{TP + FN} \tag{1}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$\text{False Positive Rate} = \frac{FP}{FP + TN} \tag{3}$$

In order to test the performance of the objective detection model, we measured the experimental results for the learning data used in the learning process and the experimental data of the untrained test data. **Table 5** lists the classification accuracy results.

Table 5. Performance Comparison of Feature Selection Algorithms

FS Algorithm	**Training Dataset**			**Test Dataset**		
	DR(%)	**FP(%)**	**Time(s)**	**DR(%)**	**FP(%)**	**Time(s)**
BestFirst+ CfsSubsetEval	99.3	0.7	0.14	75.6	19.3	0.15
SFFS-RF	99.9	0.1	0.15	84.4	12.5	0.21
GeneticSearch+ ConsistencySubsetEval	99.6	0.4	0.67	80.5	15.4	0.34
All Feature	99.6	0.4	0.84	81.1	15.0	1.3

The detection rate of the detection model-based overall features before applying the FS technique is 81.1%, but the detection rate of 84.4% shows an improvement of approximately 3.3% after applying the proposed FS technique. The measurement results for the detection rate show that the classification performance of the IDS is improved by the FS step. In particular, the FS scheme proposed in SFFS-RF shows the highest detection rate (DR) and FPs compared to the other techniques. Thus, the FS algorithm proposed for SFFS-RF showed good results in terms of a lower computation cost and higher classification results than the other FS techniques.

3) Comparison with Other Algorithms of FS for NSL-KDD

In addition, we compared the existing features of the FS-based IDS using the NSL-KDD data to evaluate the performance of the IDS when applying the FS method proposed in this paper. Because the experimental conditions of the present study may be different from those of other studies, the training time (train time) and test time (test time) were excluded from the comparison. Experiments were conducted to compare the results when the same data were used in different ways. **Table 6** lists the results of these comparative experiments.

Table 6. Comparison Results Based on NSL-KDD Dataset
(n/a Means No Available Results)

Feature Selection Algorithm	No. of Features	DR (%)	FP (%)	Accuracy (%)
SFFS-RF	10	99.9	0.1	99.89
Eid [15]	20	n/a	n/a	99.20
S. Mukherjee [16]	24	n/a	n/a	97.78
H. F. Eid [17]	17	n/a	n/a	99.10
E. de la Hoz [18]	10	n/a	n/a	88.30
Abd-Eldayem [19]	13	99.03	1.0	99.38

It can be seen that the proposed method has a higher DR and fewer FPs with fewer features than the existing method.

5. Additional Experiments

5.1 Experiment Settings

The purposes of this experiment were divided into three main categories. The first was demonstrating that the proposed random forest-based optimization algorithm was applicable to various fields. The second was to verify the improvement in the accuracy of the generated classification. Finally, we validated the performance of SFFS-RF.

Datasets for experiments were gathered from the UCI Machine Learning Repository [20]. We chose six kinds of data and utilized them as experimental input. **Table 7** summarizes the input data.

To minimize the impact of over-fitting, we used the 10-fold cross validation method. All of the data were divided into 10 sets, and each dataset was alternately used for learning and evaluation. This ensured that the data could be used equally.

The number of features that can be selected by the RFC is not limited, and it can grow up to 100 trees. The prototype was implemented using the Java language. The machine learning and accuracy detection components were implemented using the open source machine learning tool WEKA.

Table 7. Data Sets

Category	Title of Data Set	No. of sample	No. of classes	No. of features
Economics	Adult	48,842	2	14
Economics	Credit	30,000	2	24
Text	Letter	20,000	26	16
Physical	Sonar	208	2	60
Physical	Wine	178	3	13
Audio	Waveform	5,000	3	40

5.2 Experiment Results

The experimental results were split into three categories: the number of features, classification accuracy, and memory usage.

1) Number of Features

The first experiment was a comparison of the number of features derived from the initial data and the number of optimization algorithms proposed. The results of the tests are listed in **Table 8** and reflect the variation in the number of features produced by the total number of features and the number of characteristics derived using the SFFS algorithm.

Table 8. Selected features of data set

Data name	No. of features	No. of SFFS-RF features	Selected features
Adult	14	4	{11, 12, 5, 8}
Credit	24	3	{7, 12, 4}
Letter	16	12	{13, 11, 15, 8, 9, 12, 10, 16, 14, 7, 6, 4}
Sonar	60	17	{12, 16, 26, 54, 31, 58, 37, 19, 9, 45, 18, 47, 20, 17, 30, 14, 40, 53}
Wine	13	7	{8, 11, 2, 6, 4, 13, 3}
Waveform	40	15	{13, 17, 11, 16, 6, 15, 10, 12, 5, 9, 7, 8, 3, 2, 20}

Credit card data are the most feature-reduced data, showing a total reduction of 88%. Data containing many features that act as factors reduce the accuracy of the classification model. In addition, data that include many features and samples, such as adult and sonar data, show a large reduction. This means that when using a large amount of multiple-feature data, unnecessary features are being learned in the process of creating a classification model. Therefore, it is possible to eliminate a significant number of features from the data. Thus, the derived optimal set of features was advantageous because the insignificant features were excluded, and the classification accuracy was improved.

2) Classification Accuracy

In the second experiment, the classification accuracy was compared when generating a classification model. Comparisons were made between a set of optimized features and the total feature set. The accuracy of the classification model was measured in terms of the TP, TN, FP, and FN, and the values are shown based on the reference value of 100% accuracy. The accuracy was calculated as shown in formula (7).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{7}$$

Table 9 lists the results of the comparative classification accuracy experiments.

Table 9. Accuracy of experiment data

Data name	Total features	SFFS-RF features
Adult	85.07%	85.91%
Credit	81.88%	82.05%

Letter	96.30%	96.38%
Sonar	81.25%	92.31%
Wine	98.31%	98.88%
Waveform	85.16%	85.52%

Reducing the features improved the classification accuracy, which was the opposite of what we expected. **Fig. 5** shows the improved classification accuracy. The dataset with the greatest improvement is the sonar data; the classification accuracy was improved by 11.05%. This was because reducing the number of unnecessary features improved the classification accuracy.

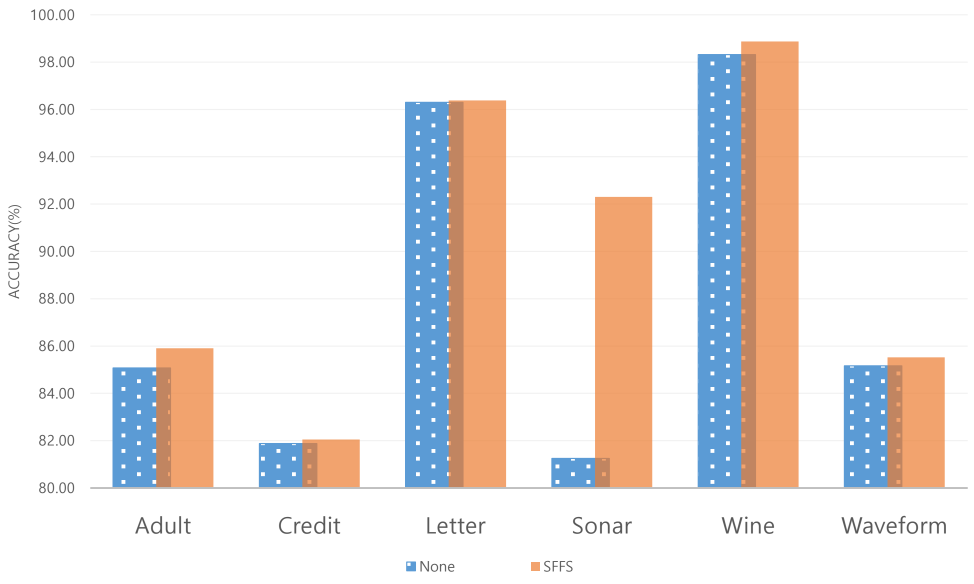

Fig. 5. Accuracy Experiment Result

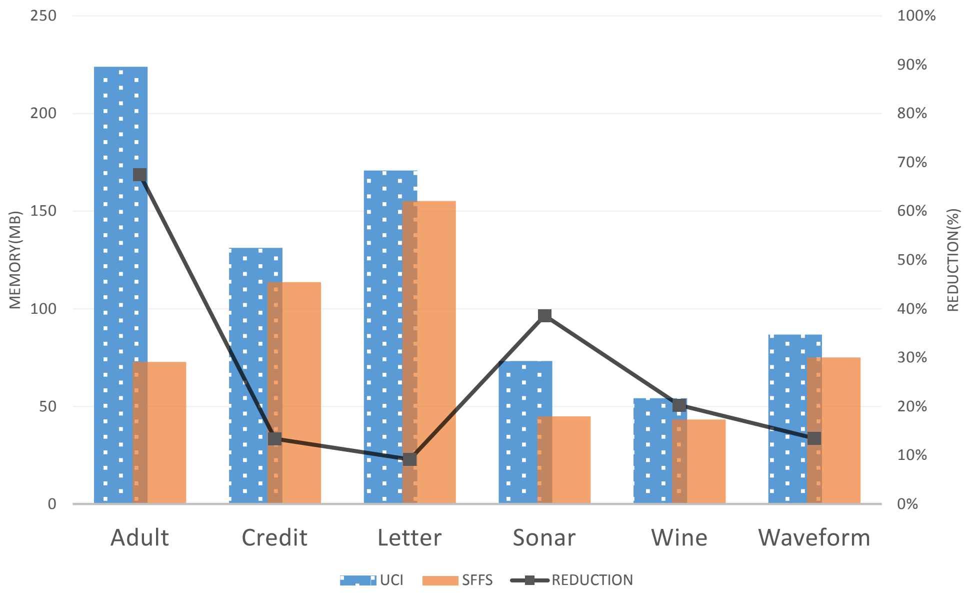

Fig. 6. Result of Memory Consumption Experiment

3) Memory Usage Measurement

The third experiment was a memory resource utilization experiment. It verified the utilization of system resources. Two cases were measured. The first involved generating the classification model based on the initial feature set. The second involved constructing a classification model with a set of features derived from the optimization algorithm. The results of the experiment are shown in **Fig. 6**. In the graphical representation of the results, the y axis on the left shows the amount of memory (MB), and the y axis on the right indicates the memory usage reduction ratio (%). The adult data show the most significant memory usage reduction (approximately 68%). One problem was that as the number of features and samples increased, the required system resources also increased. This problem could be solved through SFFS-RF by reducing the use of system resources.

Therefore, these results verified that the selected feature set obtained via the proposed FS algorithm improved both the classification accuracy and memory usage.

6. Conclusions

This paper proposed an FS technique for IDS to reduce the FPs of existing IDSs and overcome performance problems. We described an FS algorithm based on RFC and SFFS to reduce system resource usage and improve the classification accuracy. Experiments were performed on the NSL_KDD data to show the superiority of the SFFS-RF FS technique. The experimental results showed that a detection model with a 0.4% false rate could be generated. In addition, we demonstrated that the number of features decreased to six categories of data. As the classification performance of the model improved, the memory usage was reduced. Thus, SFFS-RF could have a positive impact on the classification accuracy and improve system resource management. In this way, it is possible to shorten the learning time and detection time by selectively learning the features that affect intrusion detection. The detection model generated based on the FS technique could be used as a base model for a lightweight IDS.

In the future, we will research a suitable weight extraction technique for the IDS and a technique that can accurately reflect the importance of each feature in the weight. Moreover, we will study the use of the IDS architecture as a real-time system in parallel and decentralized applications.

References

[1] C. Yin, L. Ma, L. Feng, Z. Yin and J. Wang, "A Feature Selection Algorithm towards Efficient Intrusion Detection," *International Journal of Multimedia and Ubiquitous Engineering*, vol.10, no.11, pp.253-264, 2015. Article (CrossRef Link)

[2] S. Y. Ohn, S. D. Chi, and M. Y. Han, "Feature Selection for Classification of Mass Spectrometric Proteomic Data Using Random Forest," *The Korea Society For Simulation(KSS)*, Vol.22, No.4, pp.139-147, 2013. Article (CrossRef Link)

[3] W. Lee and S. Oh, "Efficient Feature Selection Based Near Real-Time Hybrid Intrusion Detection System," *KIPS Tr. Comp. and Comm. Sys.*, vol.5, no.12, pp.471-480, Dec. 2016. Article (CrossRef Link)

[4] NSL-KDD Dataset [Internet], http://www.unb.ca/research/iscx/dataset/iscx-NSL-KDD-dataset.html.

[5] M. Tavallaee, E. Bagheri, W. Lu, and A.-A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," *Computational Intelligence for Security and Defense Applications, CISDA 2009. IEEE Symposium on. IEEE*, pp.1-6, 2009. Article (CrossRef Link)

[6] L. C. Molina, L. Belanche, and A. Nebot, "Feature selection algorithms: a survey and experimental evaluation," *in Data Mining, ICDM 2003. Proceedings. 2002 IEEE International Conference on. IEEE*. pp.306-313, 2002. Article (CrossRef Link)

[7] G. CHANDRASHEKAR, F. SAHIN, "A survey on feature selection methods," *Computers & Electrical Engineering*, Vol.40, No.1, pp.16-28, 2014. Article (CrossRef Link)

[8] L. Breiman, "Random forests," *Machine learning*, Vol.45, No.1, pp.5-32, 2001. Article (CrossRef Link)

[9] F. Baumann, A. Ehlers, K. Vogt, and B. Rosenhahn, "Cascaded Random Forest for Fast Object Detection," *Scandinavian Conference on Image Analysis*, Springer Berlin Heidelberg , pp. 131-142, 2013. Article (CrossRef Link)

[10] Y. Mishina, R. Murata, Y. Yamauchi, T. Yamashita, and H. Fujiyoshi, "Boosted random forest," *IEICE TRANSACTIONS on Information and Systems*, Vol.98, No.9, pp.1630-1636, 2015. Article (CrossRef Link)

[11] Ian H. Witten, Eibe Frank and Mark A. Hall, "Data Mining. 3rd," Trans. Lee. S. H, acorn, 2014.

[12] M. A. Hall, "Correlation-based Feature Subset Selection for Machine Learning," *doctoral dissertation*, The University of Waikato, Canada, 1999.

[13] H. Liu and R. Setiono, "A probabilistic approach to feature selection - A filter solution," in *Proc. of 13th International Conference on Machine Learning*, pp.319-327, 1996.

[14] Kakavand, M., Mustapha, N., Mustapha, A., and Abdullah, M. T., "Effective Dimensionality Reduction of Payload-Based Anomaly Detection in TMAD Model for HTTP Payload," K*SII Transactions on Internet and Information Systems*, Vol. 10, No.8, pp.3884-3910, 2016 Article (CrossRef Link)

[15] Eid, H. F., Salama, M. A, Hassanien, A. E., and Kim, T. H, "Bi-layer behavioral-based feature selection approach for network intrusion classification," *International Conference on Security Technology*, Springer Berlin Heidelberg, vol. 259, pp.195-203, 2011. Article (CrossRef Link)

[16] S. Mukherjee and N. Sharma, "Intrusion detection using naive Bayes classifier with feature reduction," *Procedia Technology*, vol.4, pp.119-128, 2012. Article (CrossRef Link)

[17] H. F. Eid, A. E. Hassanien, T.-h. Kim, and S. Banerjee, "Linear correlation-based feature selection for network intrusion detection model," *Advances in Security of Information and Communication Networks*, Springer Berlin Heidelberg, vol.381, pp.240-248, 2013. Article (CrossRef Link)

[18] E. de la Hoz, A. Ortiz, J. Ortega, and E. de la Hoz, "Network anomaly classification by support vector classifiers ensemble and non-linear projection techniques," *International Conference on Hybrid Artificial Intelligence Systems*, Springer Berlin Heidelberg, vol.8073, pp.103-111, 2013. Article (CrossRef Link)

[19] Abd-Eldayem and Mohamed M, "A proposed HTTP service based IDS," *Egyptian Informatics Journal*, vol.15, no.1, 13-24, 2014. Article (CrossRef Link)

[20] A. Frank and A. Asuncion, "UCI machine learning repository," 2010, http://archive.ics.uci.edu/ml

Jinlee Lee received the M.S. degree in Education from Dankook University, Korea in 2007 and the Ph.D. degree in Computer Engineering from Konkuk University, Korea in 2017. Her research interests include Security, anti-phishing, and Machine learning.

.

Dooho Park received the B.S. degree from Konkuk University, Korea in 2012 and the M.S. degree in Computer Engineering from Konkuk University, Korea in 2014. He is working for XIIlab ltd., Seoul, Korea, from 2016 to now. His research interests are in Machine Learning System and Artificial Intelligence.

Changhoon Lee received the B.S. degree from Yonsei University, Korea in 1975, and the M.S. degree in Computer Science from KAIST (Korea Advanced Institute of Science and Technology), Korea in 1977 and the Ph.D. degree in Computer Science from KAIST, Korea in 1993. He joined the faculty of Konkuk University in 1980, where he was a Professor of Electronic and Computer Engineering until 2017. His research interest lies in Artificial Intelligent, Security.