Development of an Early Warning System for Network Intrusion Detection Using Benford's Law Features

Liuying Sun[1], Anthony Ho[1,2], Zhe Xia[1(✉)], Jiageng Chen[3], and Mingwu Zhang[4]

[1] Department of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063, China
xiazhe@whut.edu.cn
[2] Department of Computer Science, University of Surrey, Guildford, Surrey GU2 7XH, UK
[3] Department of Computer Science and Technology, Central China Normal University, Wuhan 430079, China
[4] School of Computers, Hubei University of Technology, Wuhan 430068, China

Abstract. In order to ensure a high level of security in computer networks, it is important to prevent malicious behaviours from the intruders. However, high volumes of network traffic make it difficult for intrusion detection systems (IDSs) to separate abnormal network traffic from the normal ones. To alleviate this problem, a window-based feature extraction method using the Benford's law has been proposed in this paper. Our method employs six features of the divergence values, including the first digit and the first three digits of size difference between traffic flows. Experiments are performed and evaluated using the KDD99 dataset. To illustrate the advantages of our proposed method, three popular classifiers, Multi-Layer Perceptron (MLP), Support Vector Machine (SVM) and Naïve Bayes are analysed using different combinations of these six features as the input feature sets. The results demonstrated that the MLP classifier performs the best in classifying the normal, mixed and malicious windows by correctly classifying the normal and malicious windows. This is particularly useful to reduce the amount of network traffic that needs to be analysed. The only exception is the mixed window which contains both normal flows and attack flows, and it needs to be further analysed to distinguish normal flows from malicious ones. Our method is fast and can be used as an early warning system to trigger other more advanced IDSs to focus on the specific regions of the network traffic. The combined system, incorporating our method with a traditional IDS, can provide a lower FAR of 0.27% compared with 9.87% of the isolated IDS, along with no significant reduction of the detection performance. Moreover, the whole accuracy of the combined system achieves 92.09%.

W. Meng and S. Furnell (Eds.): SocialSec 2019, CCIS 1095, pp. 57–73, 2019.
https://doi.org/10.1007/978-981-15-0758-8_5

1 Introduction

The Internet has become an indispensable component of our daily life and it affects all aspects of people's lives. This has led to a great deal of attentions in the area of network security both in the industry and academia. This is because any malicious intrusion or attack against a network would cause serious consequences for an organization, potentially impacting their reputation and financial stability. Therefore, unauthorized access to communication or computer network must be detected, prevented and repelled as soon as possible. Intrusion Detection Systems (IDSs) can be used to prevent unauthorized access to network resources by monitoring and analysing the network traffic.

To distinguish the attacks from the normal network access, various artificial intelligence methods have been developed for solving problems that are related to the intrusion detection, and the following methods are most commonly used in the literature [1–8]: support vector machine (SVM), decision tree, genetic algorithm (GA), principal component analysis (PCA), Particle Swarm Optimization (PSO), K-nearest neighbours, Naïve Bayes networks, and Neural Networks such as Multi-Layer perceptron and Self-organizing map. Their detection targets vary from any class of anomalies to just a single class.

In general, IDSs deal with a tremendous amount of data which contains redundant and irrelevant features resulting in excessive training and predictive time. To minimize the computational costs and the number of data patterns need to be searched, various feature selection and extraction techniques have been developed. The feature selection techniques generally first rank the existing features according to their predictive significance, and then select the most meaningful feature subset from the original features. However, feature extraction techniques actually transform the features into linear combinations of the original attributes. There are various techniques that can be used for the feature extraction and selection. These include Genetic Algorithm (GA), Information Gain, correlation coefficient, Partial Least Square (PLS), and Kernel Principal Component Analysis (KPCA) [9–13].

In this paper, we propose a fast window-based feature extraction method based on the Benford's law to analyse and classify the KDD99 dataset. The proposed method is an extension of a previous work [14], and it can be used as an early warning system and incorporated into other IDSs. It composes of two phases. In the first phase, three machine learning techniques, Multi-Layer Perceptron (MLP), Support Vector Machine (SVM) and Naïve Bayes are implemented and compared to evaluate the performance of the features extracted over different training datasets reconstructed from the KDD99, and then the classifier and the subset of features that perform the best are selected. In the second phase, experiments are performed to compare the performance of when our proposed method is used as an early warning system incorporated into an IDS that was proposed in [15] with that of the IDS alone. The work in [15] has trained a classifier using 19 critical features chosen by the proposed feature reduction strategy called gradually feature removal method rather than 41 original features in KDD99, which resulted in contributing a more efficient intrusion system. The classifier used in the second phase is MLP. Classifiers used in these two phases are trained and designed independently, and the final results are aggregated from the individual ones in each model.

The remainder of this paper is organized as follows. Section 2 introduces the dataset used and Benford's law. Section 3 describes the proposed method. Experiments are conducted in Sect. 4. Results and analysis are presented in Sect. 5. Finally, we conclude and discuss some future works in Sect. 6.

2 Materials and Methods

2.1 Data Set

The experimental data used in our experiments is a benchmark database KDD99. Till now, KDD CUP'99 dataset is the only publicly available and widely used labelled dataset for IDS. The KDD99 dataset contains two types of data, training data and test data. Training data is consisted of seven weeks of network traffic, and the test data contains two weeks of network traffic. The network traffic contains network-based attacks inserted in the normal background data. The full dataset (18 MB, 743 MB Uncompressed) contains 22 types of attacks and is employed for the purpose of training. For testing, the "Corrected KDD" dataset containing 17 additional attacks is used. These attacks are grouped into four major categories: denial of service (DOS), unauthorized access from a remote machine (R2L), unauthorized access to local supervisor privileges (U2R), probing, surveillance and other probing (Probe) [16]. Since our focus is the TCP transmission, these data are therefore extracted from the KDD99 dataset for the experiments. The extracted data sets are composed of the training data and test data, containing 1,820,596 records and 119,357 records, respectively.

2.2 Benford's Law

Benford's law is an empirical law that states the probability distribution of the leading digits for naturally occurring sets of numbers. According to Benford's law, its significant digits for a collection of numbers are not uniformly distributed. Instead, they obey the distribution shown in Eq. (1). It was first proposed by Newcomb [17] in 1881. In 1938, Frank Benford, whom this law was named after, re-discovered this phenomenon by testing it on data from 20 different domains including the sizes of populations, the surface areas of rivers, physical constants, molecular weights and so on [18]. In 1995, Hill provided a statistical interpretation of this law and also generalized it for all significant digits in [19].

$$P_d = \log_{10}\left(1 + \frac{1}{d}\right) \tag{1}$$

Where P_d denotes the probability of d, d = 1,2,3,…,9.

Although Benford's law has been applied in various fields for a long time, such as forensic accounting, auditing, nature science and image forensics [20–23], its use in network security has only been investigated until recently. Nevertheless, Benford's law has been demonstrated to be very effective and reliable in anomaly detection [24–26].

3 Our Proposed Method

3.1 Construction of Feature Sets

KDD 99 consists of 41 features excluding the target-class label. In our recent research [14], src_bytes, dst_bytes and bytes which are the sum of src_bytes and dst_bytes have been proved to be effective parameters of the network flows that could be used for Benford's law in distinguishing the normal network flows from the malicious ones. Thus, new features are reconstructed based on these three basic features.

Since the first-digit law is a distribution, a window-based method proposed in [28] is employed to collect sufficient samples for a given feature. This is then used to construct an observed distribution which can then be compared with the target distribution for deviation detection. To compare the conformity of first-digit frequencies with the logarithmic distribution in different windows, we use chi-square goodness-of-fit statistics test [29]. The chi-square divergence is given as in Eq. (2).

$$\chi^2 = \sum_{d=1}^{9} \frac{\left(\hat{P}_d - P_d\right)^2}{P_d} \tag{2}$$

Where P_d is the expected frequency of digit, d is the first digit according to the logarithmic distribution and $\hat{P}_d$ is the actual observed frequency in the data set. In all cases, the lower the discrepancy measure, the higher the similarity will be obtained between the data set and the distribution.

In this paper, the Benford's law divergence value is taken as a new feature of the window. Firstly, the following features, src_bytes, dst_bytes and bytes of each flow, are extracted and then divided into sets of consecutive windows. Window size is chosen as W = 2000, which is a reasonable choice according to [14]. The difference is calculated between every two contiguous flows in each window. The frequency of the first-digit in each window is then computed. Finally, the results are compared with the expected logarithmic distribution by applying the chi-square measure. Thus, three features are constructed and they are denoted as src_bytes_chi, dst_bytes_chi and bytes_chi, respectively. Since the results have been shown previously that the Benford's Law can be very effective in distinguishing between normal and malicious network flows [24–26], especially when using multiple digits of the Benford's Law [14], its first three digits are also calculated, resulting in three features, src_bytes_chi_3, dst_bytes_chi_3 and bytes_chi_3.

To evaluate the performance of features extracted, the six features are divided into three feature sets. Different feature sets are used to construct different data sets. Detailed properties of each feature set have been tabulated in Table 1.

3.2 ClassLabelling of Instances

KDD99 has labels at the flow level, but our features works at the flow window level, so flow labels need to be converted to flow window labels. Depending on whether the windows contain attack flows or not, they are classified into three groups: normal,

Table 1. Constructed feature sets and the corresponding features

Feature set	Feature	Dimension
F_set1	src_bytes_chi, dst_bytes_chi, bytes_chi	3
F_set2	src_bytes_chi_3, dst_bytes_chi_3 and bytes_chi_3	3
F_set3	src_bytes_chi, dst_bytes_chi, bytes_chi src_bytes_chi_3, dst_bytes_chi_3, bytes_chi_3	6

mixed, and malicious. The normal group, as the name implies, contains all TCP flows that are normal. The windows labeled mixed contain both normal and attack flows. A window is labeled malicious if all TCP flows in this window are malicious.

3.3 Construction of Datasets

As already mentioned, the traffic flows in KDD99, which is used as the base dataset, are divided into consecutive windows using a window-based method. Firstly, non-overlapping is used with a window size W = 2000, resulting 317 normal windows, 83 mixed windows, 535 malicious windows and a total of 935. In order to obtain sufficient instances, an overlapping window is then applied with sliding window steps of 1000 and 100, respectively.

Similarly, non-overlapping strategy is adopted to reconstruct the test dataset from the original test data set "Corrected KDD". However, the new test dataset contains only one normal window. To increase the number of normal windows, all normal traffic flows in "Corrected KDD" are extracted, and non-overlapping is then re-applied to the normal traffic flows. Thus, the final test dataset consists of 23 normal windows, 49 mixed windows and 9 malicious windows. The results of the constructed datasets are illustrated in Table 2.

Table 2. The list of constructed datasets with their corresponding composition information

Dataset name	Feature	Normal	Mixed	Malicious	Total
Trainset data 1	F_set1	317	83	535	935
Trainset data 1-1	F_set1	637	167	1065	1869
Trainset data 1-2	F_set1	6374	1638	10674	18686
Trainset data 2	F_set2	317	83	535	935
Trainset data 2-1	F_set2	637	167	1065	1869
Trainset data 2-2	F_set2	6374	1638	10674	18686
Trainset data 3	F_set3	317	83	535	935
Trainset data 3-1	F_set3	637	167	1065	1869
Trainset data 3-2	F_set3	6374	1638	10674	18686
Test data 1	F_set1	23	49	9	81
Test data 2	F_set2	23	49	9	81
Test data 3	F_set3	23	49	9	81

3.4 Machine Learning Algorithms Applied to Intrusion Detection

3.4.1 Multilayer Perceptron

A neural network (NN) consists of information processing units which can mimic the neurons of human brain [30]. It works by accepting input data, extracting rules and then making decisions. Multilayer perceptron (MLP) is a feed-forward type of NN, which is usually trained with the standard back propagation algorithm. A MLP network consists of an input layer, one or more hidden layers, and an output layer of calculation nodes. The layers are fully connected from one layer to the next one. Each node in hidden layer and output layer has a nonlinear activation function, which enables MLP to distinguish data that is not linearly separable. MLPs are very powerful pattern classifiers that they can approximate virtually any input-output map with one or two hidden layers [31].

3.4.2 Support Vector Machine

SVM (Support Vector Machine) is based on statistical learning theory by finding an optimal hyperplane in the feature space that separates input dataset with maximum margin [31]. It uses just a portion of the data called the support vectors that represent the training data to train a model. Although linear SVM classifiers are efficient and work well in many cases, it often happens that many datasets are not even close to being linearly separable. To solve this problem, a kernel trick is applied, which implicitly maps the inputs into high-dimensional feature spaces. There are three major kernel functions, Sigmoid kernel, Polynomial kernel and Gaussian kernel which are used to build SVM classifier. In addition to binary classification, SVMs can also handle multiclass problems by reducing the multi-class task to several binary problems via One-vs-One or One-vs-All strategy. One-vs-One multiclass SVMs are used for the experiments in this paper.

3.4.3 Naive Bayes

A Naïve Bayes classifier is a simplified probabilistic classification model based on applying Bayes's theorem, which estimates the conditional and prior probabilities to generate a learning model, with a naive independence assumption between the features [27]. It assumes that all features are conditionally independent given class. Despite the fact that the strong independence assumption is generally poor and rarely true in real-world applications, the naive Bayes classifiers perform surprisingly well in practice and can be as effective as other more sophisticated classifiers. However, for intrusion detection, the strong independent relation assumption may result in lower attack detection accuracy when the features are correlated [34].

4 Our Proposed Method

The experiments consist of two phases: In the first phase, the windows passing through are classified into three classes, normal, mixed and malicious, which can be used as an early warning system. In the second phase, the network flows in mixed windows are further classified into normal and attack using the IDS proposed in [15]. And then, we

show the performance that our method is used as an early warning system combined with the IDS. To show the advantages of our proposed method, the performances of whole test data set detected by the single IDS are compared with the results of the combined system.

4.1 Phase-1: Multiclass Classification

To perform multi-class classification, the models are first trained using different training sets (Table 2). Each training set is evaluated with the corresponding test dataset using three different machine learning algorithms, MLP, SVM, and Naïve Bayes.

The structure of MLP is composed of 3 layers, input, hidden and output layer. This structure is commonly chosen as a basic structure for many applications such as image processing. Whereas the number of units at the input layer is equal to the number of selected features, the output layer consists of three softmax output neurons which is the equal number of categories to be classified to make the network's prediction. The number of units at the hidden layer is set up based on the rules-of-thumb proposed in [32]. Hyper-parameters which show the best performance are selected through trial and error. ReLU function is used as a neuron function and MLP is learned using back propagation.

The test dataset is then predicted to evaluate the performance of the trained models. The trained models are divided into three groups based on the feature sets selected. In each group, comparisons are performed to select the best performing model. Finally, the comparison among the selected best models is made to determine the feature set that performs best.

4.2 Phase-2: Anomaly Detection in Mixed Window

The main purpose of this phase is to further classify the traffic flows in mixed window from phase-1 into corresponding two classes, i.e., normal or anomalies. The classification returns back to flow level, and so does the features. The features used in this phase are 19 original features in KDD99. For explicit information of the 19 features, refer to [15].

In this phase, 10% of the KDD99 dataset is used as the training data. After training the model, two test experiments are conducted using two different test datasets. One consists of TCP traffic flows in the "mixed" windows classified by the best model chosen in phase-1. The other is the whole test dataset of the group the best model belongs to, and it is converted to flow level.

5 Results and Discussion

For MLP, Keras backend theano is used. SVM and Naïve Bayes classifiers are constructed using a machine learning package called scikit-learn. The experiments are conducted with Windows 10 as the test bed operating system on Intel i7-6700 HQ CPU @ 2.60 GHz processor, 8 GB of RAM. All the codes are written in python.

5.1 Phase-1

5.1.1 Classification Using MLP

Classifier is evaluated with a 10-fold cross validation, which is a standard technique for estimating the performance of a classifier. The results of three groups are listed in Tables 3, 4 and 5. It can be observed that the gap between the training accuracy and validation accuracy narrows as the volume of training data sets grows, which means a lower probability of overfitting. The best model in each group is chosen based on two factors: validation accuracy and training time. Validation accuracy shows the generalization ability of a model, to some extent, the higher the better. On the contrary, the lower the training time, the less the system overhead. To summarize, the best models are those trained by Trainset data 1, Trainset data 2-1 and Trainset data 3, as can be seen from Table 3, 4 and 5, respectively.

Table 3. Performance comparisons among training data sets composed of F_set1

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 1	93.48%	92.62% (2.28%)	17.682498 s
Trainset data 1-1	92.51%	92.29% (2.56%)	32.524289 s
Trainset data 1-2	92.93%	93.20% (0.79%)	150.547682 s

Table 4. Performance comparisons among training data sets composed of F_set2

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 2	89.09%	90.79% (4.99%)	20.983321 s
Trainset data 2-1	92.88%	92.03% (2.45%)	14.954141 s
Trainset data 2-2	92.86%	92.80% (0.72%)	173.994690 s

Table 5. Performance comparisons among training data sets composed of F_set3

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 3	93.69%	93.58% (1.91%)	15.837295 s
Trainset data 3-1	93.37%	92.88% (2.46%)	17.778418 s
Trainset data 3-2	93.59%	93.53% (0.51%)	236.564110 s

To evaluate the models obtained, data sets listed in Table 2 are tested by the corresponding trained models. Specifically, models which are trained separately by Trainset data 1, Trainset data 2-1 and Trainset data 3, are tested on Test data 1, Test data 2 and Test data 3, respectively. Figure 1(a)–(c) illustrates the classification results of test data sets via confusion matrices.

Confusion Matrixes show that all the normal and malicious windows in Test data 1 and 3 are correctly classified. In other words, the detection rate of both the normal and malicious is 1.0. Furthermore, classification results of normal and malicious among the three test sets have minor differences. The main difference lies in the classification of

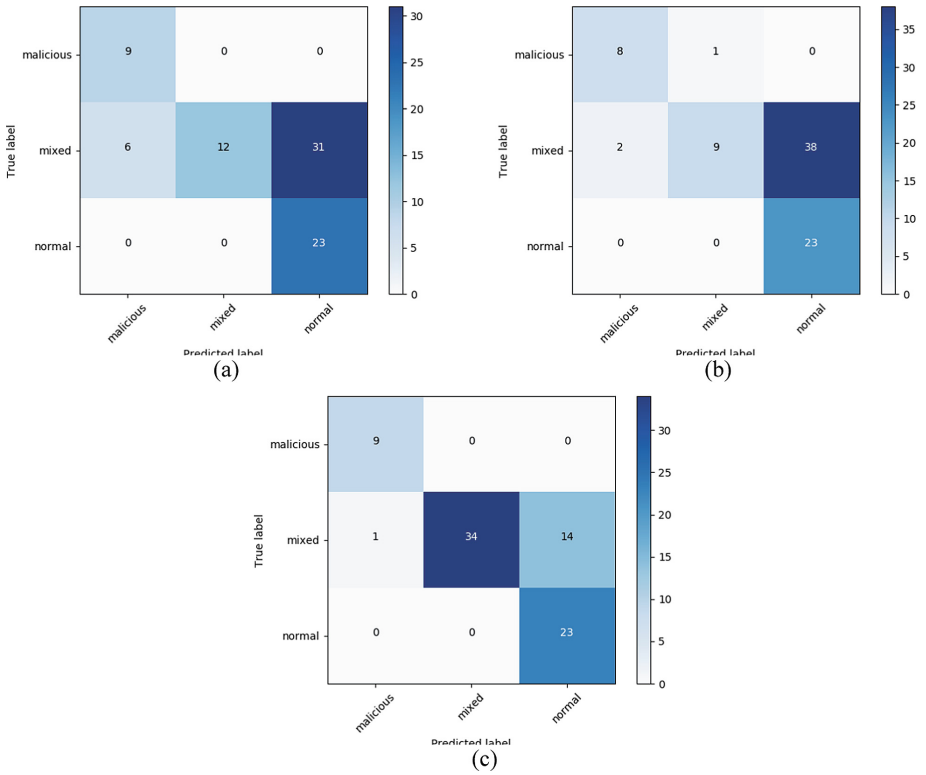

Fig. 1. Confusion matrixes for window classification using MLP: (a) Trainset data 1 with Test data 1; (b) Trainset data 2-1 with Test data 2; (c) Trainset data 3 with Test data 3

mixed windows. Trainset data 2 performs the worst with only 18% of mixed windows correctly classified, while the Trainset data 3 performs the best with 69% of mixed windows correctly classified. Nevertheless, the average classification accuracy of Test data 3, which achieved the best result, is as low as 81.48%. There is a gap of 12.1% between validation accuracy and test accuracy.

The main reason is that χ^2 divergence, the feature of the mixed window is sensitive to the number of attack flows in this window. Figure 2 shows the distributions of normal TCP traffic flows in mixed windows in Trainset data 3 and Test data 3, respectively. In Fig. 2, the percentage of normal flows on the total 2000 flows in a window is shown horizontally, and the count of windows corresponding to x-axis is shown vertically. As one can see, the majority of the normal TCP traffic flows in mixed windows in Trainset data 3 account for 90% to 100%, i.e. most of these mixed windows consist of more than 1800 normal TCP flows and less than 200 attack TCP flows. However, Test data 3 shows just the opposite. The difference of normal flow number between training set and test set results in different distributions of χ^2 divergence value. That is, the value range regarding the same feature of mixed windows in training set and test set is different, as can be seen in Fig. 3, resulting in incomplete

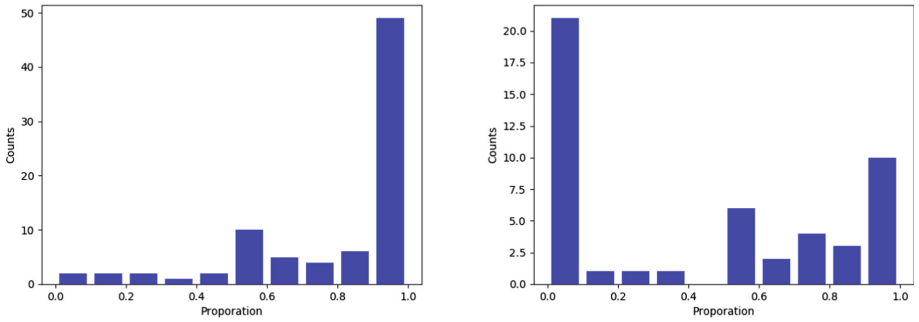

Fig. 2. The distribution of normal flows in training and test dataset: (a) Trainset data 3; (b) Test data 3

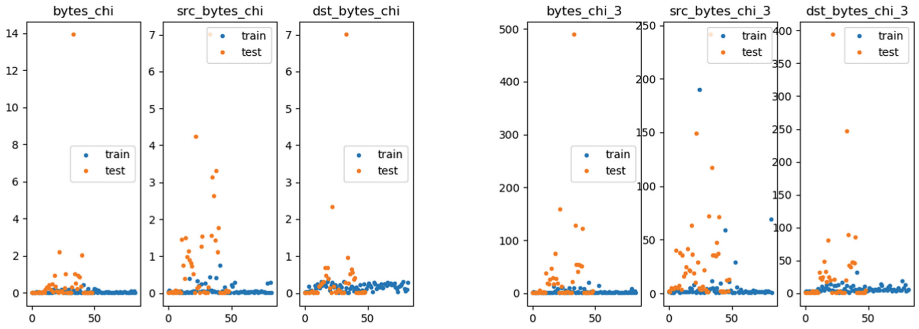

Fig. 3. The value of different features composing the mixed windows in training samples and test samples

training sets for mixed windows. Because the lower the attack flow number, the smaller difference between mixed window and normal window. Similarly, the higher the attack flow number, the smaller difference between mixed window and malicious window. Therefore, these characteristics make it difficult to distinguish mixed windows from normal and malicious ones.

Figure 4 shows the distribution of normal flows in mixed windows which were classified as normal windows. As one can see, seven of 14 misclassified mixed windows are those in which normal TCP traffic flow account for 90% to 100%. The windows that were misclassified as malicious consist of 1999 attack TCP flows and only one normal flow.

5.1.2 Classification Using SVM

For the SVM, the classifier is evaluated with 5-fold cross validation. The results achieved for these three groups are listed in Tables 6, 7 and 8. Based on the two factors

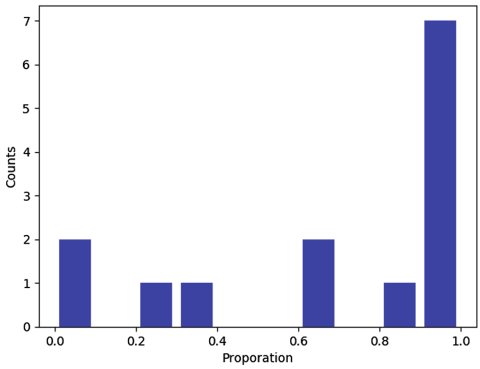

Fig. 4. The distribution of normal flows in misclassified mixed windows of Test data3

Table 6. Performance comparisons among training data sets composed of F_set1

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 1	93.9%	93.40%	0.009569 s
Trainset data 1-1	93.63%	93.48%	0.036576 s
Trainset data 1-2	95.11%	94.70%	8.50749 s

Table 7. Performance comparisons among training data sets composed of F_set2

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 2	94.65%	93.83%	0.026069 s
Trainset data 2-1	94.44%	93.69%	0.064608 s
Trainset data 2-2	95.75%	94.44%	1212.171683 s

Table 8. Performance comparisons among training data sets composed of F_set3

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 3	94.55%	94.04%	0.021469 s
Trainset data 3-1	94.06%	93.8%	0.031255 s
Trainset data 3-2	96.91%	95.51%	294.210544 s

mentioned above in terms of accuracy and time efficiency, SVM classifiers trained by Trainset data 1-1, Trainset data 2-1 and Trainset data 3-1 are then chosen. This is because SVM classifiers achieve higher training accuracy and lower training time for generating a model as compared with MLP classifiers. Confusion matrices of classification results using selected SVM classifiers are depicted in Fig. 5(a)–(c).

From Fig. 5, it can be seen, classification accuracies of the three classifiers are approximately the same, with 66.67% having 416 support vectors, 62.96% having 290 support vectors and 64.20% having 312 support vectors, respectively. This indicates

that the difference among feature sets in SVM does not cause a significant difference in classification results as it did in MLP. However, since nearly half of the mixed windows are misclassified as normal, the difficulty of classification in SVM still lies in the mixed windows, which is similar to the case in MLP. In addition, as shown in Fig. 5, one of nine malicious windows is classified as being mixed. Since mixed windows will be further classified, windows that are mistakenly classified as mixed theoretically do not affect the final result of anomaly detection in phase-2. However, in this case, the classification process in phase-1 will be meaningless. In addition, Fig. 5 (b) shows that one in 23 normal windows is misclassified as malicious, which may result in a tendency of higher false alarm rate that needs to be avoided.

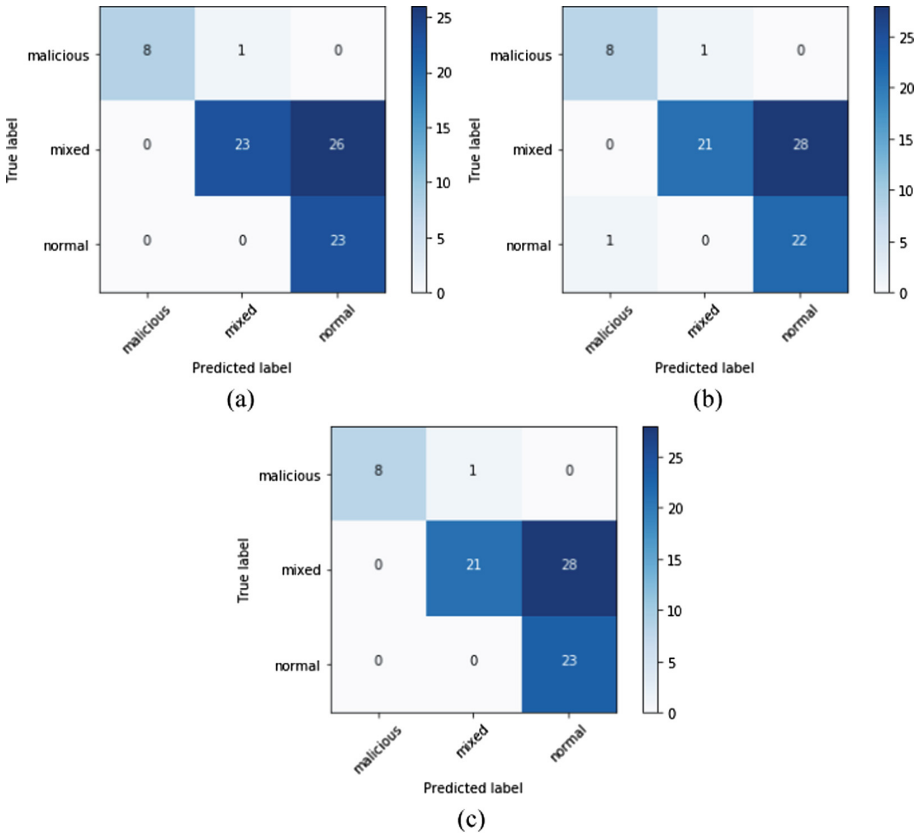

Fig. 5. Confusion matrixes for window classification using SVM: (a) Trainset data 1-1 with Test data 1; (b)Trainset data 2-1 with Test data 2; (c) Trainset data 3-1 with Test data 3

5.1.3 Classification Using Naïve Bayes

Tables 9, 10 and 11 illustrate the corresponding results of Naïve Bayes classifiers. The classification performance has been evaluated using a 10-fold cross validation procedure. As shown in Table 9, classifiers using F_set1 as the feature vector achieve the

best performance. While Table 10 shows that F_set2 is not a good feature vector as the Naïve Bayes classifiers performed poorly in the learning process. In other words, features in F_set2, src_bytes_chi_3, dst_bytes_chi_3 and bytes_chi_3 are not sufficiently distinctive enough to use as features for Naïve Bayes classifiers. In addition, F_set3, combination of F_set1 and F_set2, did not show any advantages of using all these six features. In Tables 9, 10 and 11, the highest validation accuracy rates achieved were approximately 93.36% for Trainset data 1, 34.03% for Trainset data 2-2 and 91.37% for Trainset data 3-2, respectively. As such, considering accuracy and time efficiency, Naïve Bayes classifiers trained by Trainset data 1, Trainset data 2-2 and Trainset data 3-2 are chosen. The classification results of the chosen features are shown in Fig. 6(a)–(c) via confusion matrices.

Table 9. Performance comparisons among training data sets composed of F_set1

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 1	93.37%	93.36%	0.003971 s
Trainset data 1-1	91.17%	90.53%	0 s
Trainset data 1-2	91.44%	91.22%	0.019967 s

Table 10. Performance comparisons among training data sets composed of F_set2

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 2	8.45%	10.39%	0 s
Trainset data 2-1	34.08%	33.43%	0 s
Trainset data 2-2	34.42%	34.03%	0.019979 s

Table 11. Performance comparisons among training data sets composed of F_set3

Dataset name	Training accuracy	Validation accuracy	Training time
Trainset data 3	67.59%	68.47%	0.004002 s
Trainset data 3-1	91.06%	90.53%	0 s
Trainset data 3-2	91.64%	91.37%	0.01994 s

Figure 6(a)–(c) show that Trainset data 1 with test data 1, Trainset data 2-2 with test data 2 and Trainset data 3-2 with test data 3 achieving classification results of approximately 41.98%, 38.27%and 46.91%, respectively. Figure 6(a) shows that all malicious and normal windows are correctly classified. However, only 4.08% mixed window are detected. Figure 6(b) shows that the majority of the samples are classified as normal. Since eight of nine malicious windows are misclassified as normal, the classifier trained by Trainset data 2-2 is too insensitive to detect abnormal activities. Figure 6(c) shows similar results to Fig. 6(a). However, there is a slight improvement in distinguishing mixed windows from malicious windows.

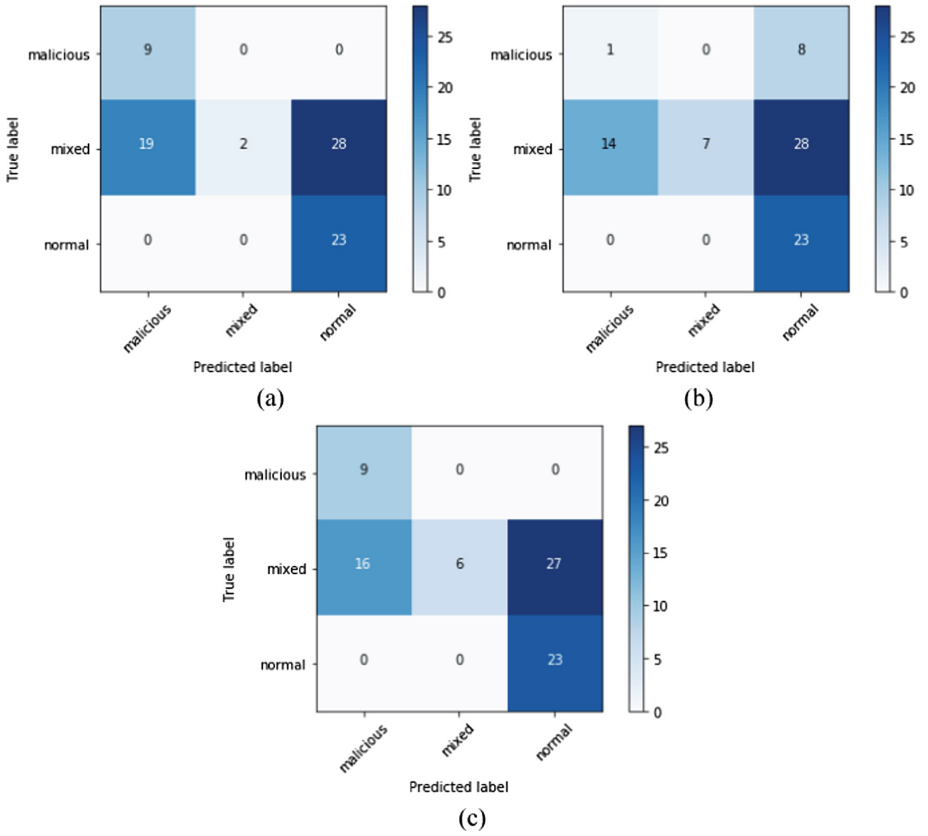

Fig. 6. Confusion matrixes for window classification using Naïve Bayes: (a) Trainset data 1 with Test data 1; (b) Trainset data 2-2 with Test data 2; (c) Trainset data 3-2 with Test data 3

From the above analysis, MLP classifier trained by all six features achieved the highest accuracy with 81.48% when compared to SVM with 66.67% and Naïve Bayes with 46.91%. Thus, the 34 mixed windows detected by the MLP classifier will be further analyzed and then classified in phase-2.

5.2 Phase-2

Phase-2 further classifies the mixed window recognized in phase-1 into normal and malicious traffic flows. Since the samples classified in phase-1 are all windows containing 2000 traffic flows, windows classified as mixed need further classification to obtain a better performance in accuracy of the whole dataset. Furthermore, the superiority of the combined model compared to a single model will be evaluated in detail via five metrics: accuracy, precision, recall, f1-score and false alarm rate. They are commonly defined as follows:

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

$$\text{Precision} = \frac{TP}{FP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1} - \text{score} = \frac{2TP}{2TP + FP + FN}$$

$$\text{False alarm rate} = \frac{FP}{TP + FP}$$

Where TP, FP, FN and TN are the number of True Positives, False Positives, False Negatives and True Negative, respectively.

The reason for measuring the false alarm rate is that the cost incurred when IDS misclassifies a normal flow as malicious could be very high. In addition, it is well known that the false alarm occurs frequently in IDS experiments, so that it should be considered when developing IDSs [33]. The time incurred for processing will also be evaluated. The results obtained with respect to the evaluation metrics are provided in Table 12.

Table 12. Performance comparisons between proposed combined system and single IDS

Approach	Accuracy	Precision	Recall	F1-score	False alarm rate	Testing time (sec)
combined IDS	92.09%	99.73%	87.48%	93.2%	0.27%	0.34
single IDS	93%	90.13%	97.83%	93.82%	9.87%	0.80

Table 12 shows that the accuracy and F1-score of proposed method combining an IDS is slightly lower as compared to the IDS alone, but it outperforms with a much lower rate of only 0.27% FAR. Besides, the proposed approach consumes less time for training and testing.

6 Conclusion

This paper presented an application of Benford's law for anomaly-based network flow IDS based on six new features as an early warning system for the detection of malicious attacks. The six features used were src_bytes_chi, dst_bytes_chi, bytes_chi, src_bytes_chi_3, dst_bytes_chi_3 and bytes_chi_3. Based on these six features, three feature sets were selected to train the classifiers. The MLP classifier trained by all six features was shown to perform the best in accurately distinguishing and classifying the

normal or malicious windows from the mixed windows. The mixed windows were further classified by combining our proposed method with existing IDSs to detect the malicious network flows. The experimental results showed that the proposed IDS improved the performance by reducing the computational complexity and decreasing the FAR by approximately 9.6%. Our future work will focus on finding other optimum subsets of features for the Benford's law and the window size, in order to further improve the overall performance of the proposed system.

Data Availability

The data used to support the findings of this study are available at http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China (Grant No. 61702212, 61672010) and the Natural Science Foundation of Hubei Province (Grant No. 2017CFB303).

References

1. Khan, L., Awad, M., Thuraisingham, B.: A new intrusion detection system using support vector machines and hierarchical clustering. VLDB J. **16**(4), 507–521 (2007)
2. Amor, N.B., Benferhat, S., Elouedi, Z.: Naive Bayes vs decision trees in intrusion detection systems. In: Proceedings of the 2004 ACM Symposium on Applied Computing, pp. 420–424. ACM (2004)
3. Shafi, K., Abbass, H.A.: An adaptive genetic-based signature learning system for intrusion detection. Expert Syst. Appl. **36**(10), 12036–12043 (2009)
4. Wang, W., Battiti, R.: Identifying intrusions in computer networks with principal component analysis. In: International Conference on Availability, Reliability and Security, pp. 270–279. IEEE (2006)
5. Kennedy, J.: Particle swarm optimization. In: Encyclopedia of Machine Learning, pp. 760–766. Springer, Heidelberg (2011)
6. Li, Y., Guo, L.: An active learning based TCM-KNN algorithm for supervised network intrusion detection. Comput. Secur. **26**(7–8), 459–467 (2007)
7. Moradi, M., Zulkernine, M.: A neural network based system for intrusion detection and classification of attacks. In: Proceedings of the IEEE International Conference on Advances in Intelligent Systems-Theory and Applications, pp. 15–18 (2004)
8. Labib, K., Vemuri, R.: NSOM: a real-time network-based intrusion detection system using self-organizing maps. Netw. Secur., 1–6 (2002)
9. Aslahi-Shahri, B.M., Rahmani, R., Chizari, M., et al.: A hybrid method consisting of GA and SVM for intrusion detection system. Neural Comput. Appl. **27**(6), 1–8 (2016)
10. Kayacik, H.G., Zincir-Heywood, A.N., Heywood, M.I.: Selecting features for intrusion detection: a feature relevance analysis on KDD 99 intrusion detection datasets. In: Proceedings of the Third Annual Conference on Privacy, Security and Trust (2005)
11. Parsazad, S., Saboori, E., Allahyar, A.: Fast feature reduction in intrusion detection datasets. In: 2012 Proceedings of the 35th International Convention MIPRO, pp. 1023–1029. IEEE (2012)
12. Gan, X.S., Duanmu, J.S., Wang, J.F., et al.: Anomaly intrusion detection based on PLS feature extraction and core vector machine. Knowl.-Based Syst. **40**(1), 1–6 (2013)

13. Kuang, F., Zhang, S., Jin, Z., et al.: A novel SVM by combining kernel principal component analysis and improved chaotic particle swarm optimization for intrusion detection. Soft. Comput. **19**(5), 1187–1199 (2015)
14. Sun, L., Anthony, T.S.H., Xia, Z., et al.: Detection and classification of malicious patterns in network traffic using Benford's law. In: 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), pp. 864–872. IEEE (2017)
15. Li, Y., Xia, J., Zhang, S., et al.: An efficient intrusion detection system based on support vector machines and gradually features removal method. Expert Syst. Appl. **39**(1), 424–430 (2012)
16. Elkan, C.: Results of the KDD'99 classifier learning. ACM SIGKDD Explor. Newslett. **1**(2), 63–64 (2000)
17. Newcomb, S.: Note on the frequency of use of the different digits in natural numbers. Am. J. Math. **4**(1), 39–40 (1881)
18. Benford, F.: The law of anomalous numbers. Proc. Am. Philos. Soc., 551–572 (1938)
19. Hill, T.P.: A statistical derivation of the significant-digit law. Stat. Sci. **10**, 354–363 (1995)
20. Nigrini, M.: Benford's Law: Applications for Forensic Accounting, Auditing, and Fraud Detection. Wiley, Hoboken (2012)
21. Durtschi, C., Hillison, W., Pacini, C.: The effective use of Benford's law to assist in detecting fraud in accounting data. J. Forensic Account. **5**(1), 17–34 (2004)
22. Fu, D., Shi, Y.Q., Su, W.: A generalized Benford's law for JPEG coefficients and its applications in image forensics. In: Security, Steganography, and Watermarking of Multimedia Contents IX. International Society for Optics and Photonics, vol. 6505, p. 65051L (2007)
23. Sambridge, M., Tkalčić, H., Jackson, A.: Benford's law in the natural sciences. Geophys. Res. Lett. **37**(22) (2010)
24. Arshadi, L., Jahangir, A.H.: An empirical study on TCP flow interarrival time distribution for normal and anomalous traffic. Int. J. Commun. Syst. **30**(1), e2881 (2017)
25. Asadi, A.N.: An approach for detecting anomalies by assessing the inter-arrival time of UDP packets and flows using Benford's law. In: 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), pp. 257–262. IEEE (2015)
26. Iorliam, A., Tirunagari, S., Ho, A.T.S., et al.: "Flow size difference" can make a difference: detecting malicious TCP network flows based on Benford's law. arXiv preprint arXiv:1609.04214 (2016)
27. Lewis, D.D.: Naive (Bayes) at forty: the independence assumption in information retrieval. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 4–15. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026666
28. Sperotto, A., Pras, A.: Flow-based intrusion detection. In: 2011 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 958–963. IEEE (2011)
29. Plackett, R.L.: Karl Pearson and the chi-squared test. Int. Stat. Rev., 59–72 (1983)
30. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, Upper Saddle River (1994)
31. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Heidelberg (2013)
32. Panchal, G., Ganatra, A., Kosta, Y.P., et al.: Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers. Int. J. Comput. Theory Eng. **3**(2), 332–337 (2011)
33. Ghorbani, A.A., Lu, W., Tavallaee, M.: Network Intrusion Detection and Prevention: Concepts and Techniques. Springer, Heidelberg (2009). https://doi.org/10.1007/978-0-387-88771-5
34. Ibrahim, H.E., Badr, S.M., Shaheen, M.A.: Adaptive layered approach using machine learning techniques with gain ratio for intrusion detection systems. arXiv preprint arXiv:1210.7650 (2012)