Hindawi
Security and Communication Networks
Volume 2021, Article ID 2170788, 11 pages
https://doi.org/10.1155/2021/2170788

Research Article

A Novel Model for Anomaly Detection in Network Traffic Based on Support Vector Machine and Clustering

Qian Ma,[1,2] **Cong Sun**,[3] **and Baojiang Cui**[1,2]

[1]*School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China*
[2]*National Engineering Laboratory for Mobile Network Technologies, Beijing, China*
[3]*School of Science, Beijing University of Posts and Telecommunications, Beijing, China*

Correspondence should be addressed to Baojiang Cui; cuibj@bupt.edu.cn

Received 8 July 2021; Accepted 22 October 2021; Published 20 November 2021

Academic Editor: Marimuthu Karuppiah

New vulnerabilities and ever-evolving network attacks pose great threats to today's cyberspace security. Anomaly detection in network traffic is a promising and effective technique to enhance network security. In addition to traditional statistical analysis and rule-based detection techniques, machine learning models are introduced for intelligent detection of abnormal traffic data. In this paper, a novel model named SVM-C is proposed for the anomaly detection in network traffic. The URLs in the network traffic log are transformed into feature vectors via statistical laws and linear projection. The obtained feature vectors are fed into a support vector machine (SVM) classifier and classified as normal or abnormal. Based on the idea of SVM and clustering, we construct an optimization model to train the parameters of the feature extraction method and traffic classifier. Numerical tests indicate that the proposed model outperforms the state of the arts on all the tested datasets.

1. Introduction

With the rapid development of information technology and Internet, network security has become increasingly important. Anomaly detection in network traffic is an effective way to provide solid information for network security management and protect users' data and privacy. Through the analysis and study on the network traffic, the malicious behaviors in the network can be discovered as soon as possible, such as SQL injection attack, cross site scripting (XSS) attack, directory traversal attack, and other types of attack. Anomaly detection methods are required to enhance the adaptability and scalability because of the increasing volumes of traffic data. There are some inherent defects in traditional rule-based detection methods. For example, it is easy for attackers to bypass the predefined detection rules and new unknown attacks cannot be discovered via the rules based on existing attacks. Thus rule-based methods often suffer from high false positive rate. In essence, the anomaly detection in network traffic is a data classification problem. It aims to distinguish attack data from normal behaviors. Besides the traditional rule-based detection techniques [1, 2], considerable methods based on statistical theory [3, 4], information theory [5, 6], and machine learning [7, 8] are widely used in abnormal traffic detection problem. The machine learning based detection model is a promising method for intelligent anomaly detection in the large-scale and high-bandwidth network environment.

Researchers have studied multiple machine learning based detection methods [9]. Supervised learning models are commonly used in anomaly detection, where datasets labeled as normal or abnormal are used to train and test the model. Neural network [10], support vector machine [11], decision tree [12], Naive Bayes [13], and other supervised models are often used in traffic classification. Ensemble learning is a methodology that combines multiple basic models together and achieves better performance than single classifier. Therefore, random forest [14] and other ensemble algorithms have also been used for abnormal traffic detection. However, it is often difficult to obtain enough labeled data. Thus, unsupervised learning models have been adopted to find out the latent structure in data. The training and testing procedures of unsupervised detection models are based on unlabeled datasets. Clustering is a classic unsupervised model. The traffic is characterized and identified via selecting an appropriate distance metric [15]. Semisupervised detection models can be treated as combination of supervised and unsupervised methods, since they use labeled and unlabeled datasets simultaneously to build up the detection models. In [16], it shows that the classification accuracy is improved significantly via semisupervised detection models, spectral graph transducer, and Gaussian fields. Before training the machine learning-based detection models, feature selection and dimension reduction are two kinds of useful preprocessing methods to reduce the dataset dimension. Selecting a subset with small redundancy is helpful to improve the detection performance. Hybrid models consist of machine learning models and feature selection methods. In [17], the authors proposed an improved krill swarm algorithm based on linear nearest neighbor lasso step for feature selection in network intrusion detection. The authors in [18] applied an autoencoder module for dimension reduction of traffic feature. Then, the obtained compressed representations are fed into machine learning models for intrusion detection.

Generally, the anomaly detection method in network traffic follows the following steps. First, the traffic data are transformed into feature vectors via the feature extraction method. Then, the obtained feature vectors are used to train and test the traffic classification model. Finally, the new traffic data are classified as normal or abnormal via the trained classifier. An effective feature extraction method is helpful to improve the performance of the anomaly detection model. In the aforementioned works, the authors mainly adopt the hand-crafted feature extraction method. Since the feature set heavily relies on experts' domain knowledge, the trained classifiers have some disadvantages, such as poor adaptability for the datasets from different network environments. Therefore, it is critical to minimize the dependence of experts' knowledge in traffic feature extraction process. In fact, the network traffic can be treated as natural language. Therefore, researchers introduced natural language processing techniques to fully explore the semantic structure of traffic data. For example, the k-gram technique can be used to characterize the typical pattern of normal requests [19]. Any coming payload is labeled as abnormal, if it does not match the normal pattern [20].

In this paper, we propose a novel model called SVM-C to detect abnormal network traffic. The raw traffic data are transformed into fixed-length feature vectors via statistical laws and linear coding operation. Then, an optimization problem is constructed based on the basic idea of SVM and clustering. The parameters of SVM-C are trained via solving the optimization problem. The transformed vectors are classified via the SVM classifier. In summary, our contributions are summarized as follows:

(i) A new model SVM-C is proposed for anomaly detection in network traffic. The parameter training of SVM-C is accomplished via constructing an optimization problem.

(ii) We apply the block coordinate descent (BCD) and projected Barzilai–Borwein (PBB) method [21] to solve the proposed optimization problem.

(iii) The numerical results on all the tested datasets indicate that the proposed model outperforms the state of the arts.

The rest of the paper is organized as follows. In Section 2, related works of anomaly detection in network traffic are discussed. Section 3 describes the overall framework of the proposed model. In Section 4, the optimization problem and corresponding training algorithm of the proposed model are introduced. Section 5 shows the superior detection performances of the proposed model compared to the existing supervised machine learning models.

2. Related Work

The existing anomaly detection methods for network traffic are classified into two categories, including misuse-based and anomaly-based methods. The misuse-based [2, 22] detection methods are effective, but they cannot discover unseen attacks and suffer from high false negative rate. Anomaly-based methods are prevalent because they can discover unseen attacks. This paper mainly focuses on the machine learning-based anomaly detection method. Thus, we briefly survey the anomaly-based detection methods for network traffic. In general, anomaly-based network anomaly detection methods are classified into four categories [8], including classification-based, statistical theory-based, clustering-based, and information theory-based detection methods.

Statistical detection methods construct probabilistic models with training data for the purpose of tracking network behaviors. In [23], the authors used three IP flow features and four flow attributes to generate a network profile called digital signature of network segment, which contains a threshold for each dimension, respectively. Abnormal behaviors are detected according to the number of abnormal dimensions. The proposed method can only detect attacks that impact bits, packets, and flows. Principal component analysis (PCA) is known as a dimensionality reduction approach in data mining field. PCA is also a widely used statistical technique for anomaly detection in network traffic. Pascoal et al. [24] reduced the dimension of traffic feature space via a combination of robust feature selection based on mutual information metric and robust PCA. The proposed model is robust to outliers and obtains a robust feature subspace.

Information theory-based detection methods mainly use information-theoretic measures to explain the characteristics of network traffic features and identify specific distributions of anomalies. Amaral et al. [25] used the Tsallis entropy to detect anomalous traffic flow. The proposed model can be used for anomaly detection in different types of networks and detecting more inexpressive attacks than those detection methods based on volume analysis. In [26], long-term network anomalies were tracked, where the Kullback–Leibler divergence was used to measure the difference between global probability density functions for every two consecutive periods of time. This function produces a time series sequence to be analyzed and sets an adaptive threshold to identify abnormal changes in network. Bhuyan et al. [27] used the mutual information and generalized entropy-based feature selection technique to select a relevant nonredundant feature subset, which makes the anomaly detection process much more accurate and faster.

Clustering-based methods aim to group network data into several classes of similar data. The essential idea of clustering is to achieve a high intracluster similarity and a low intercluster similarity. Eskin et al. [15] applied standard clustering algorithm with unlabeled data and Euclidean distance metric to detect network intrusion. Dromard et al. [28] proposed an unsupervised anomaly detector based on a grid and incremental clustering algorithm called IDGCA and a discrete time sliding window. IDGCA is more efficient than classic clustering algorithms, due to its low system complexity and flexibility for real-time detection. Besides, clustering-based methods can also be used to reduce the redundancy in raw datasets. Perdisci et al. [29] applied a feature clustering algorithm to reduce the dimension of k-gram features.

Classification-based detection methods use normal traffic profile to build the classification knowledge base. The traffic data that deviate from the baseline profile are regarded as anomalous. Kim et al. [30] proposed a hybrid intrusion detection method that hierarchically integrates a misuse detection model and a classification based anomaly detection model in a decomposed structure. They first build a misuse detection model based on C4.5 decision tree algorithm. Then, the normal training data are decomposed into smaller subsets. Multiple one-class SVM models are created for the decomposed subsets, and the profiles of normal behaviors are built precisely. Different domain-specific techniques are commonly used in hybrid models, too. In [31], the authors adopted self-organized feature map to profile normal packets, passive TCP/IP fingerprinting to filter unknown packets, and the genetic algorithm to select appropriate packet fields. A dataset consisting of representative training samples is created via the combination of these techniques and then used as the input of a new classification model combining soft-margin SVM and one-class SVM.

3. SVM-C for Anomaly Detection in Network Traffic

In this section, a model named SVM-C is proposed for anomaly detection in network traffic. The overall framework of SVM-C is introduced, followed by the traffic feature extraction method and the training process of the detection model. Finally, the classification algorithm is described.

3.1. The Framework of SVM-C.

3.1. The Framework of SVM-C. The overall framework of the SVM-C model is shown in Figure 1. SVM-C has two main components: feature extraction and traffic classification. The first component, feature extraction, transforms raw URLs into feature vectors via a series of mapping rules and linear projection. The second component, traffic classification, trains a SVM model to classify the obtained feature vectors. The parameters of the two components are coupled in an optimization problem. The deployment procedure of SVM-C is summarized below.

(1) Traffic feature extraction (Section 3.2): the raw URLs are transformed into fixed-length feature vectors via a traffic feature extraction method based on the statistical laws and linear mapping.

(2) Anomaly detection model training (Section 3.3): the obtained feature vectors are taken as the input to the SVM classifier. The parameters of the feature extraction method and SVM model are solved via an optimization problem based on SVM and clustering.

(3) Traffic classification (Section 3.4): as depicted in Figure 1, new URLs are first transformed into feature vectors and then classified as abnormal or normal via the trained SVM classifier.

3.2. Traffic Feature Extraction. Traffic data can be regarded as short text. Therefore, natural language processing techniques can be used for the feature extraction of traffic data. Here the feature extraction method in [32] is applied, which is based on the statistical laws and k-gram technique. Before feature extraction, each URL is parsed into different segments, such as protocol, port, path, query, and so on. In this paper, we mainly focus on the path and query segment of each raw URL.

First, the malicious strings in raw URLs are extracted. Here ε percentage strings out of all are chosen for construction of a lexicon of malicious strings. Then, a set of mapping rules is defined based on the obtained lexicon, as illustrated in Figure 2. In the above two segments, two adjacent characters are mapped into a weight between 0 and 9 via the mapping rules. Larger weights indicate a higher probability of an abnormal URL. For instance, the parsed components of a URL "/data/cache/inc_catalog_base.inc" are transformed into a vector "332113252114411332 334114441144." Then, the obtained weight vectors are converted into a k-length feature vector via the k-gram technique. Denote the achieved k-length vector as $\mathbf{v}$. Details of the feature extraction method can be found in [32]. The impact of the parameter ε on the performance of the subsequent traffic classifier is analyzed in Section 5.

After the k-length feature vector is obtained, we apply a matrix-vector operation to produce local features around each character in the feature vector. The final obtained d-dimensional vector is

$$\mathbf{x} = \mathbf{A}\mathbf{v} + \mathbf{a}, \tag{1}$$

where $M = p - k + 1$ and $\mathbf{A} \in R^{d\times k}$ and $\mathbf{a} \in R^{d}$ are the weight matrix and the bias of the matrix-vector operation, respectively. In this way, a fixed-length feature vector is extracted for the corresponding raw URL. The matrix $\mathbf{A}$ and vector $\mathbf{a}$ are the parameters to be learned, and the detailed process will be shown in Section 4. The size of the sliding window k in the k-gram technique and length of final feature vector d are prefixed. The analysis of the parameters k and d will be shown in Section 5.

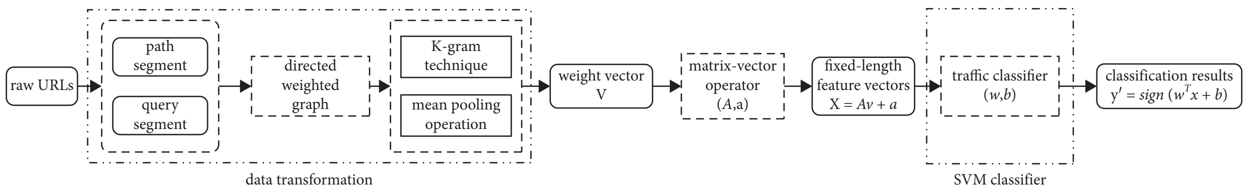

Figure 1: Overall structure of SVM-C.

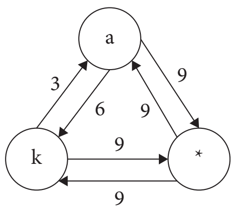

Figure 2: An illustration of the URL mapping rules.

3.3. Anomaly Detection Model Training. After feature extraction, the obtained feature vectors are then fed into the traffic classifier. In the proposed model, the SVM model is used as the classifier because of its significant performance of binary classification and mathematical formulation for good interpretability. The idea of SVM is a hyperplane classifier, which has maximum functional margin to the nearest training data point of any class. Suppose we have N training data points $\{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_N}, y_N)\}$, where $\mathbf{x_i} \in R^d$ and $y_i \in \{1, -1\}$. The hyperplane plane is denoted as $\mathbf{w}^T\mathbf{x} + b = 0$, where $\mathbf{w}$ is the weight vector and b is the bias. A new data point $\mathbf{x}$ is classified as

$$f(x) = \operatorname{sign}\left(\mathbf{w}^T\mathbf{x} + b\right) = \operatorname{sign}\left(\sum_{i=1}^{N} \alpha_i y_i (\mathbf{x} \bullet \mathbf{x_i}) + b\right), \quad (2)$$

where α_i is the Lagrange multiplier of its dual problem and $\operatorname{sign}(\cdot)$ is the signum function. Details of SVM can be found in [33]. In the proposed model, the parameters of the feature extraction method and traffic classifier are optimized together in an optimization problem, which will be described in Section 4.

3.4. Traffic Classification. According to the classification rule of SVM, the detection rule of SVM-C is set as $p(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$. $p > 0$ and $p < 0$ indicate normality and abnormality, respectively. As for a new URL $\overline{x}$, it is first transformed into a d-length feature vector $\mathbf{x}$, following the feature extraction method in Section 3.2. Denote its label as y. Then, it is classified as $y = 1$ (normal) or $y = -1$ (abnormal) via the trained SVM classifier. The classification algorithm of SVM-C is shown in Algorithm 1.

4. The Optimization of SVM-C

In order to obtain the parameters of the feature extraction method and traffic classifier in the proposed SVM-C, we construct an optimization model based on the idea of SVM and clustering. In this section, the constructed optimization problem and corresponding training algorithm are described.

4.1. Notation. Lowercase and uppercase boldface represent vectors and matrices, respectively. $\mathbf{I}_d$ represents the $d \times d$ identity matrix. $|S|$ is the number of elements in the set S. $\lambda_{\min}(\mathbf{A})$ means the minimum eigenvalue of $\mathbf{A}$. $\operatorname{Abs}(c)$ denotes the absolute value of a scalar c. $\bullet$ represents the dot product. $\otimes$ represents the Kronecker product. $\mathbf{B}^{1/2}$ is defined as $\mathbf{B}^{1/2} = \mathbf{U}\Sigma^{1/2}\mathbf{U}^T$, where $\mathbf{B} = \mathbf{U}\Sigma\mathbf{U}^T$ is a real definite matrix, $\mathbf{U}$ is an orthogonal matrix, and Σ is a diagonal matrix.

4.2. The Optimization Problem. We construct an optimization problem via combining the idea of SVM and clustering. The parameters of SVM-C are trained by solving the optimization problem. Its objective function consists of two parts. One aim is to obtain the parameters $(\mathbf{A}, \mathbf{a})$, which minimizes the sum of the square of the distance between the same type of points to their center point. In SVM-C, we simply let the center point be the mean of the homogeneous data points. The other part is the same as the standard SVM model, which intends to find the hyperplane classifier $(\mathbf{w}, \mathrm{b})$.

The original labeled dataset $\{\overline{x}_\mathbf{l}, y_l\}_{l=1}^{N}$ is denoted by $\overline{D}$, where $y_l \in \{1, -1\}$ indicates the label of data instance $\overline{x}_\mathbf{l}$. In this paper, we assign label 1 to normal URLs and -1 to abnormal ones. Through the feature extraction method in Section 3.2, each URL $\overline{x}_\mathbf{l}$ is transformed into a k-length vector $\mathbf{v_l}$. The obtained dataset is denoted as $D = \{\mathbf{x_l}, y_l\}_{l=1}^{N}$, where $\mathbf{x_l} = \mathbf{A}\mathbf{v_l} + \mathbf{a}$. Then, the dataset D is fed into the SVM model. The corresponding optimization problem is formulated as follows:

$$\min_{\mathbf{w},b,\mathbf{A},\mathbf{a}} \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{p\in S_1} \left\|\mathbf{x_p} - \mathbf{x}_1^*\right\|_2^2 + \sum_{j\in S_2} \left\|\mathbf{x_j} - \mathbf{x}_2^*\right\|_2^2, \quad (3a)$$

$$\text{s.t.} \quad \begin{array}{l} y_i\left(\mathbf{w}^T\mathbf{x_i} + b\right) \geq 1, \\ \mathbf{x_i} = \mathbf{A}\mathbf{v_i} + \mathbf{a}, \quad \forall i \in S_1 \cup S_2, \end{array} \quad (3b)$$

where $\mathbf{w}$ and b are parameters in the SVM model, $\mathbf{A}$ and $\mathbf{a}$ are parameters defined in Section 3.2, $S_1 = \{i y_i = 1\}$, $S_2 = \left\{j y_j = -1\right\}$, $\mathbf{x}_\mathbf{m}^* = \mathbf{A}\mathbf{v}_\mathbf{m}^* + \mathbf{a}$ is the mean of the points in the same class, and $\mathbf{v}_\mathbf{m}^* = 1/|S_m|\sum_{j\in S_m}\mathbf{v_j}$ $(m = 1, 2)$.

4.3. The Training Method. Problem (1) is nonconvex, which is difficult to solve optimally. The block coordinate descent method is applied, and subproblems for variables $(\mathbf{w}, \mathrm{b})$ and

```
Input: $\overline{x}$, A, a, w, b
Output: y
Step 1. Transform $\overline{x}$ into a feature vector $\mathbf{x}$ according to the data transformation method in Section 3.2.
Step 2. Calculate $\mathbf{p} = \mathbf{w}^{\mathbf{T}}\mathbf{x} + \mathbf{b}$.
Step 3. If $\mathbf{p} > \mathbf{0}$, let $\mathbf{y} = \mathbf{1}$ and $\overline{x}$ is labeled as a normal URL. Otherwise, let $\mathbf{y} = -\mathbf{1}$ and $\overline{x}$ is labeled as an abnormal URL.
```

Algorithm 1: The classification algorithm for SVM-C.

($\mathbf{A}$, $\mathbf{a}$) are solved alternatively in each iteration. We also introduce the idea of the soft-margin SVM [33], where in the t th iteration, the right-hand side of constraint (3b) is replaced by η^{t-1} and $0 < \eta < 1$ is a prefixed parameter. The analysis of the parameter η will be shown in Section 5.

4.3.1. The Subproblem of ($\mathbf{w}$, $\mathbf{b}$*).* When ($\mathbf{A}$, $\mathbf{a}$) is fixed, the subproblem for ($\mathbf{w}$, b) becomes

$$\begin{aligned} &\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2, \\ &s.t. \quad y_i\left(\mathbf{w}^T\mathbf{x_i} + b\right) \geq \eta^{t-1}, \quad \forall i \in S_1 \cup S_2, \end{aligned} \tag{4}$$

where $\mathbf{x_i} = \mathbf{Av_i} + \mathbf{a}$. Subproblem (4) is just a standard SVM problem. It is solved via the dual method. According to the analysis of the SVM problem [33], the dual problem of (4) is

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\left(\mathbf{x_i}\bullet\mathbf{x_j}\right)\eta^{t-1}\sum_{i=1}^{N}\alpha_i, \\ &s.t. \quad \alpha_i \geq 0, \quad \forall i \in S_1 \cup S_2, \\ &\qquad \sum_{i=1}^{N}\alpha_i y_i = 0, \end{aligned} \tag{5}$$

where α_i represents the Lagrange multipliers of (4). To solve problem (3a), we apply the Courant penalty function [34] and eliminate constraint (5) by penalizing it to the objective function. Then, the corresponding problem becomes

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{H}_0\boldsymbol{\alpha} - \eta^{t-1}\mathbf{g}^T\boldsymbol{\alpha} + \frac{\sigma}{2}\|\boldsymbol{\alpha}^T\mathbf{y}\|^2, \\ &s.t. \quad \alpha_i \geq 0, \quad \forall i \in S_1 \bigcup S_2, \end{aligned} \tag{6}$$

where $\alpha = (\alpha_1, \ldots, \alpha_N)^T$, $\mathbf{H}_0 = [h_{ij}]_{N\times N}$, $h_{ij} = y_i y_j (\mathbf{x_i}\bullet\mathbf{x_j})$, $\mathbf{g} = (1, 1, \ldots, 1) \in R^N$, and $\mathbf{y} = (y_1, \ldots, y_N)^T$.

When the penalty factor σ approaches infinity, problem (6) is equivalent to problem (5). Thus, we solve (6) iteratively. First, we set σ as a small value, such as $\sigma = 1$. In each inner iteration, we solve (6) and update σ as follows:

$$\sigma := \begin{cases} 10\sigma, & \text{if } \left\|u_{q+1}^T y\right\|^2 > \dfrac{\left\|u_q^T y\right\|^2}{4}, \\ \sigma, & \text{otherwise,} \end{cases} \tag{7}$$

where $\mathbf{u_q}$ is the feasible point of problem (6) obtained in q th inner iteration.

The projected Barzilai–Borwein (PBB) method [21] is applied to solve problem (6), which is introduced in Section 4.3.3. In practice, problem (6) is an ill-conditioned problem. We add a regulation item $1/2\mu\|\alpha\|$ to avoid calculation difficulties, where $\mu = c * \mathrm{abs}\left(\lambda_{\min}(\mathbf{H}_0)\right)$.

Suppose the optimal solution to problem (6) is $\alpha^* = (\alpha_1^*, \ldots, \alpha_N^*)^T$. According to the analysis of the SVM problem [33], $\mathbf{w} = \sum_{i=1}^N \alpha_i^* y_i \mathbf{x_i}$ and $b = -(\mathbf{w}\cdot\sum_{i=1}^N \alpha_i^*\mathbf{x_i})/2\sum_{j\in S_1}\alpha_j^*$.

4.3.2. The Subproblem of ($\mathbf{A}$, $\mathbf{a}$*).* When ($\mathbf{w}$, b) is fixed, the subproblem for ($\mathbf{A}$, $\mathbf{a}$) becomes

$$\begin{aligned} &\min_{\mathbf{A},\mathbf{a}} \quad \frac{1}{|S_1|}\sum_{p\in S_1}\left\|\mathbf{x_p} - \mathbf{x}_1^*\right\|_2^2 + \frac{1}{|S_2|}\sum_{j\in S_2}\left\|\mathbf{x_j} - \mathbf{x}_2^*\right\|_2^2, \\ &s.t. \quad y_i\left(\mathbf{w}^T\mathbf{x_i} + b\right) \geq \eta^{t-1}, \\ &\qquad \mathbf{x_i} = \mathbf{Av_i} + \mathbf{a}, \quad \forall i \in S_1 \cup S_2. \end{aligned} \tag{8}$$

By equivalent transformation, the problem becomes

$$\begin{aligned} &\min_{\mathbf{A},\mathbf{a}} \quad \frac{1}{2}\mathrm{trace}\left(\mathbf{AHA}^{\mathrm{T}}\right), \\ &s.t. \quad y_i\left(\mathbf{w}^T(\mathbf{Av_i} + \mathbf{a}) + b\right) \geq \eta^{t-1}, \quad \forall i \in S_1 \cup S_2, \end{aligned} \tag{9}$$

where $\mathbf{H} = \sum_{i\in S_1}(\mathbf{v_i} - \mathbf{v}_1^*)(\mathbf{v_i} - \mathbf{v}_1^*)^T + \sum_{i\in S_2}(\mathbf{v}_j - \mathbf{v}_2^*)(\mathbf{v}_j - \mathbf{v}_2^*)^T$. Further, $\mathbf{A}$ is reshaped into a vector $\mathbf{z} = (\mathbf{a}_1^T, \mathbf{a}_2^T, \ldots, \mathbf{a_k}^T)^T$, where $\mathbf{a_i}$ is the i th column of $\mathbf{A}$. The obtained problem is

$$\begin{aligned} &\min_{\mathbf{z},\mathbf{a}} \quad \frac{1}{2}\mathbf{z}^{\mathrm{T}}\widetilde{H}\mathbf{z}, \\ &s.t. \quad y_i\left(\mathbf{z}^T\mathbf{t_i} + \left(\mathbf{w}^T\mathbf{a} + b\right)\right) \geq \eta^{t-1}, \quad \forall i \in S_1 \cup S_2, \end{aligned} \tag{10}$$

where $\widetilde{\mathbf{H}} = \mathbf{H}\otimes\mathbf{I_d}$, $\mathbf{t_i} = \mathbf{v_i}\otimes\mathbf{w}$. Since $\widetilde{\mathbf{H}}$ is positive definite, we can introduce $\widetilde{\mathbf{z}} = \mathbf{H}^{1/2}\mathbf{z}$, and problem (10) is equivalent to

$$\begin{aligned} &\min_{\widetilde{z},\mathbf{a}} \quad \frac{1}{2}\|\widetilde{\mathbf{z}}\|^2, \\ &s.t. \quad y_i\left(\mathbf{z}^T\widetilde{\mathbf{t}}_\mathbf{i} + \left(\mathbf{w}^T\mathbf{a} + b\right)\right) \geq \eta^{t-1}, \quad \forall i \in S_1 \cup S_2, \end{aligned} \tag{11}$$

where $\widetilde{\mathbf{t}}_\mathbf{i} = \widetilde{\mathbf{H}}^{-1/2}\mathbf{t_i}$.

Problem (11) has precisely the same mathematical form as problem (4). The same method is applied to solve (11) and further obtain ($\mathbf{A}$, $\mathbf{a}$).

4.3.3. The Projected Barzilai–Borwein (PBB) Method. The PBB method [21] is an efficient algorithm to solve the large-scale box-constrained quadratic programming (BQP) problem (12).

$$\begin{aligned} \min_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{G}\mathbf{x} - \mathbf{f}^{\mathrm{T}}\mathbf{x}, \\ s.t. \quad & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \end{aligned} \tag{12}$$

where $\mathbf{G} \in R^{n*n}$ is an symmetric matrix and $\mathbf{b}$, $\mathbf{l}$, and $\mathbf{u}$ are vectors in R^n. Since subproblem (6) is a BQP problem, we apply the PBB method to solve it.

The basic idea of the PBB method is to project the current point which is generated from gradient decent to the feasible set of (12). Its algorithm framework is shown in Algorithm 2. Here the operation $P(\mathbf{x}) = \mathrm{mid}(\mathbf{l}, \mathbf{u}, \mathbf{x})$ means the median of $\mathbf{l}$, $\mathbf{u}$, and $\mathbf{x}$, and $\nabla q(\mathbf{x_j})$ is defined by

$$\left[\nabla q(\mathbf{x_j})\right]_i = \begin{cases} (\mathbf{g_j})_i, & \text{if } x_i \in (l_i, u_i), \\ \min\{(\mathbf{g_j})_i, 0\}, & \text{if } x_i = l_i, \\ \max\{(\mathbf{g_j})_i, 0\}, & \text{if } x_i = u_i, \end{cases} \tag{13}$$

where $\mathbf{g_j} = \mathbf{G}\mathbf{x_j} - \mathbf{f}$.

4.4. The Algorithm Framework of SVM-C. In summary, the complete training algorithm for the proposed model SVM-C is presented in Algorithm 3.

5. Performance Evaluation

In this section, the following aspects are analyzed experimentally.

(1) The numerical performance of the proposed model compared with benchmark models.

(2) How the main parameters influence the performance of the proposed model.

5.1. Experimental Setup. The performances of the proposed model are evaluated on three different datasets.

(1) Dataset 1 [35]: the first dataset is provided by a well-known Chinese Internet company, which specializes in cyberspace security and captures web logs of up to 2 TB every day. 70,000 original URL requests are used for testing.

(2) Dataset 2: the second dataset is collected from our campus network traffic and consists of 8,000 original URL requests.

(3) Dataset 3 [36]: the third dataset is from a project on GitHub. It consists of 50,000 raw URLs.

In the above three datasets, the attack data mainly include SQL injection attack, cross site scripting (XSS) attack, directory traversal attack, and other types of attacks.

The performances of the proposed model are evaluated in the following two aspects. The traffic feature extraction method based on statistical laws and linear projection is compared with the hand-crafted feature extraction method designed for dataset 1 [35]. Moreover, we choose Naive Bayes (NB) [37], linear SVM [33], and multilayer perceptron (MLP) [38] as benchmarks to evaluate the performance of the proposed classification model on different datasets.

The proposed model is evaluated via the hold-out method. It is a standard technique to estimate the model performances. The entire dataset is partitioned into two subsets. SVM-C is trained on one of them. Then, its classification performance is evaluated on the other subset. This process is repeated for several times, and the mean value of each index is eventually returned as the evaluation result of the hold-out method.

To quantify the performances of the proposed model and other compared models, the standard measurements are used: overall accuracy (acc), precision (p), false positive rate (fpr), and F1 score (f1). Here, the positive and negative instances refer to the abnormal and normal URLs, respectively. The four evaluation indexes are defined as follows in terms of confusion matrix (Table 1).

Accuracy: $\mathrm{acc} = (\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN})$. It indicates the percentage of the correctly classified instances over total instances.

Precision: $p = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$. It indicates the percentage of the correctly classified positive instances over total instances which are classified as positive.

False positive rate: $\mathrm{fpr} = \mathrm{FN}/(\mathrm{TP} + \mathrm{FN})$. It indicates the percentage of the misclassified positive instances over total positive instances.

F1 score: $f1 = 2 * p * r/(p + r)$. It is the harmonic average of precision and recall rate, where recall rate is defined as $r = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$.

5.2. Experimental Results. The proposed model is evaluated on three datasets mentioned in Section 5.1. The main parameters of SVM-C are set as $\varepsilon = 50\%$, $d = 30$, and $\eta = 0.9$ for three datasets. For dataset 1 and dataset 3, the value of k is set as 15, and for dataset 2, it is 9. First, the feature extraction method with linear projection of SVM-C is compared with the hand-crafted feature extraction method designed for dataset 1 [35]. In [35], each raw URL in transformed into a 22-dimensional feature vector. In the numerical tests, the numerical vectors obtained from different data transformation methods are fed into the benchmark classifiers mentioned in Section 5.1. The average accuracy, F1 score, precision, and false positive rate are shown in Figures 3–6. We observe the superior performances of the feature extraction method of SVM-C over the hand-crafted feature extraction method for all the classification models. Furthermore, it greatly reduces the human intervention during the feature extraction process.

```
Input: G, f, l, u, $x_0$, $\gamma$ (used for terminating the algorithm)
Output: optimal solution $x^*$ of the BQP problem (12)
Step 1. Let $i = 0$.
Step 2. Calculate $x_{i+1} = P(x_i - s_i g_i)$, where $s_i$ is the alternative BB step size, and $g_i = Gx_i - f$.
Step 3. If $\|\nabla q(x_{i+1})\|^2 \leq 10^{-5}\gamma$, output $x_{i+1}$ as the optimal solution $x^*$ and terminate the algorithm. Otherwise, go to Step 4.
Step 4. Let $i = i + 1$ and return to Step 2.
```

Algorithm 2: PBB method [21].

```
Input: $\overline{D} = \{\overline{x}_l, y_l\}_{l=1}^{N}$, $k$, d, the maximum number of iterations $i_{max}$, and initial $(A, a)$
Output: $(w, b)$, $(A, a)$
Step 1. Let $i = 1$.
Step 2. Apply the feature extraction method in Section 3.2 to obtain the dataset $D = \{(x_l, y_l)\}_{l=1}^{N}$ and divide it into the training set and
    test set.
Step 3. Solve subproblem (2) using the PBB method and obtain $(w, b)$. Compute the accuracy on the current testing set.
Step 4. Solve subproblem (6) using the PBB method and obtain $(A, a)$.
Step 5. If $i = i_{max}$, return $(w, b)$ and $(A, a)$ corresponding to the maximum accuracy and terminate the algorithm. Otherwise, let
    $i = i + 1$ and return to Step 2.
```

Algorithm 3: The training algorithm for SVM-C.

Table 1: Confusion matrix.

Actual label	Predicted label	
	Positive	Negative
Positive	True positive (TP)	False negative (FN)
Negative	False positive (FP)	True negative (TN)

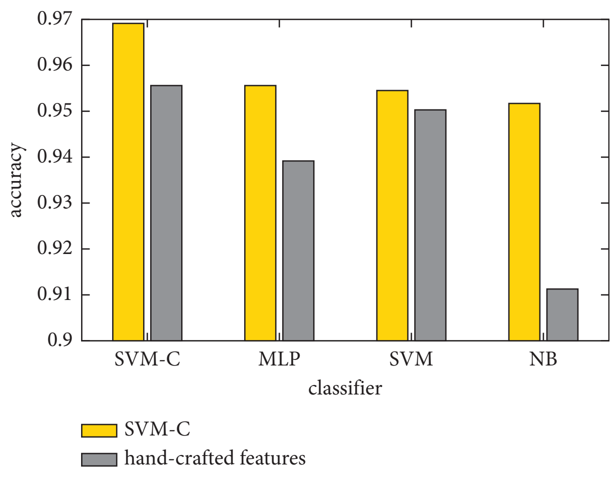

Figure 3: Accuracy of different algorithms.

Next, the performance of the proposed classification model is evaluated on different datasets. We apply some classical methods to do the classification, including Naive Bayes (NB), SVM, and multilayer perceptron (MLP). The raw URLs are converted into fixed-length feature vectors via the feature extraction method of SVM-C. The obtained numerical vectors are used as input into the benchmark classifiers. The numerical results on different datasets are shown in Figures 7–9, respectively. We observe that SVM-C performs the best on dataset 1 and dataset 2, with a distinct improvement compared to other classifiers. On dataset 3, the performances of SVM-C and SVM are similar, while SVM is slightly better than SVM-C. On all the three datasets, the proposed SVM-C achieves more than 93% accuracy, precision, and F1 score and lower than 5% false positive rate. The test results

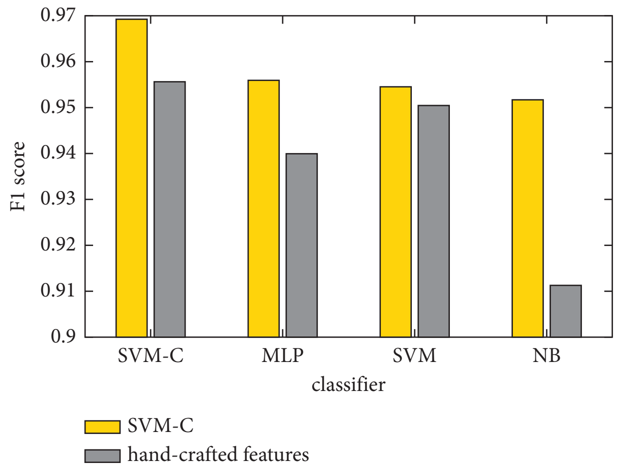

Figure 4: F1 score of different algorithms.

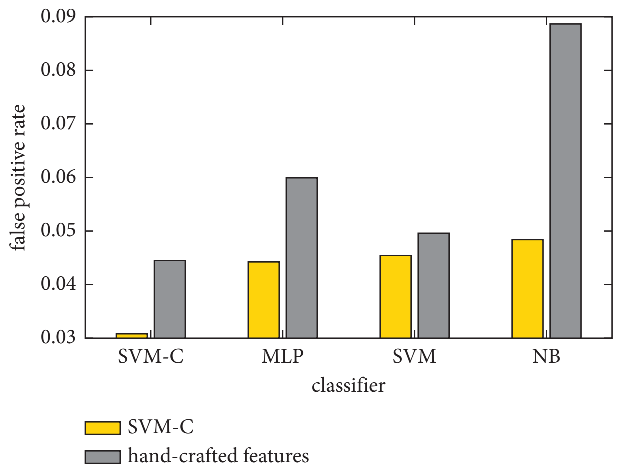

Figure 5: False positive rate of different algorithms.

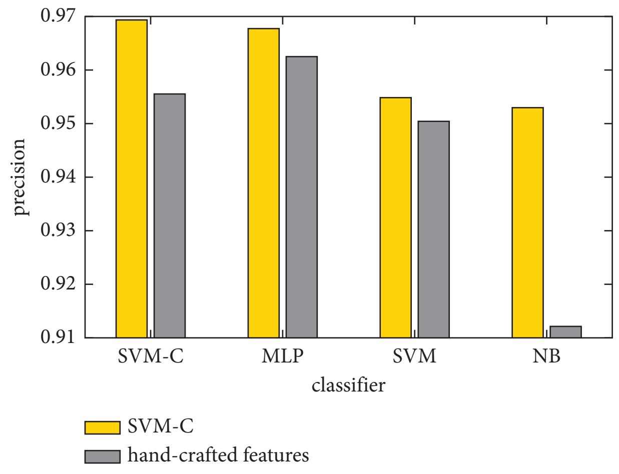

Figure 6: Precision of different algorithms.

show that SVM-C is robust to different datasets and outperforms the compared classification methods generally.

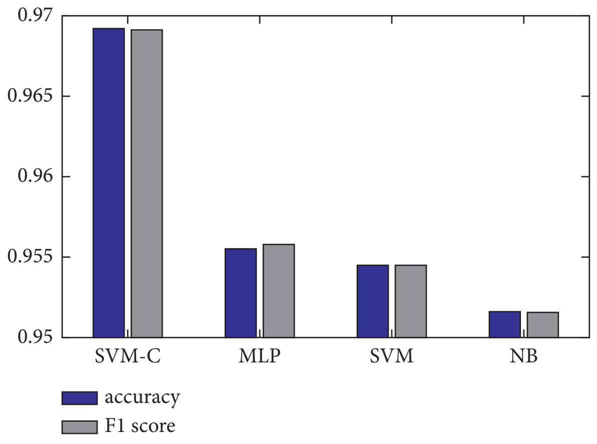

Figure 7: Accuracy and F1 score of different classifiers on dataset 1.

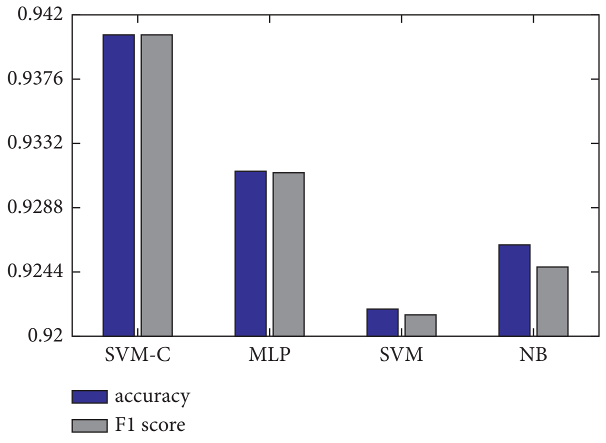

Figure 8: Accuracy and F1 score of different classifiers on dataset 2.

5.3. Parametric Analysis. There are several parameters in SVM-C: the percentage of chosen malicious keywords ε, the size of the sliding window in the k-gram technique k, the length of final feature vectors d, and the parameter of soft-margin SVM η. Here dataset 1 is used for analysis.

In the traffic feature extraction method in [32], a certain ε percentage of typical keywords in abnormal URLs is chosen to construct a lexicon of malicious words. Figure 10 shows the overall accuracy of the proposed model varying ε from 20% to 80%. Here we fix $d = 30$, $k = 15$, and $\eta = 0.9$. As ε increases, the accuracy begins to increase and reaches its maximum at $\varepsilon = 50\%$. Then, it reduces with the increment of ε. Large ε is helpful to characterize abnormal URLs. But too large ε may impair the detection performance. Selecting an appropriate value of ε is to balance the false positive rate and false negative rate. According to Figure 10, we use $\varepsilon = 50\%$ in the numerical tests.

The other important parameter in the traffic feature extraction method is the sliding window size k in the k-gram technique. The selection of k value reflects the effectiveness of local feature extraction. Figure 11 shows the classification performances of the proposed model under different sizes of sliding window. In the numerical test, we

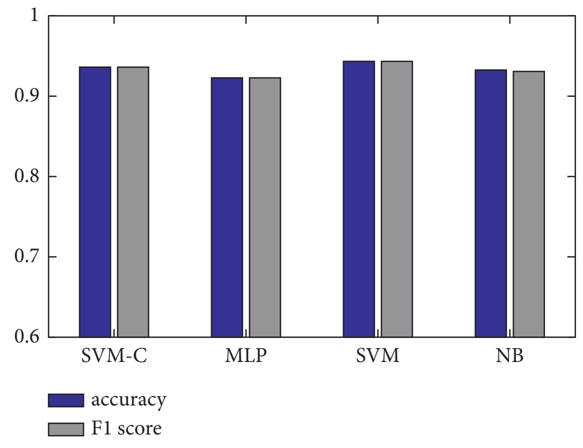

Figure 9: Accuracy and F1 score of different classifiers on dataset 3.

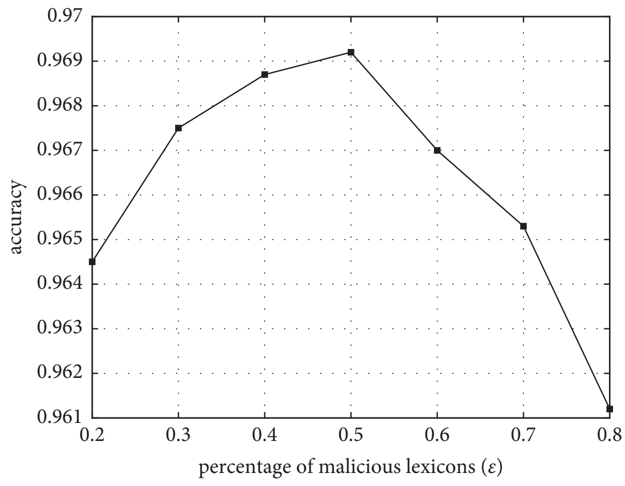

Figure 10: Accuracy of different percentages of chosen malicious strings.

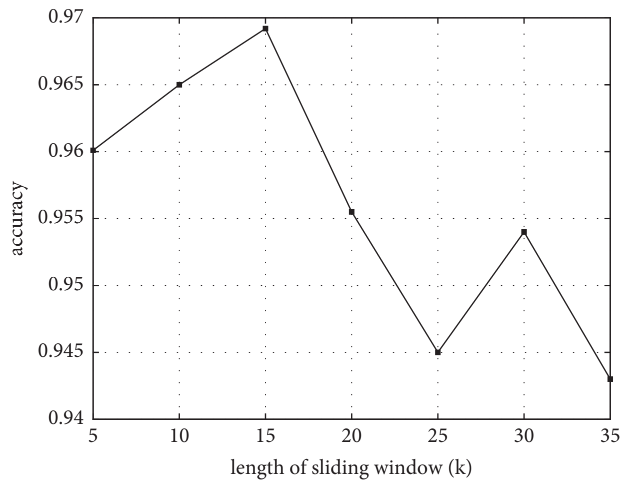

Figure 11: Accuracy of different sliding window sizes.

fix other parameters as $d = 30$, $\varepsilon = 50\%$, and $\eta = 0.9$. The accuracy reaches its peak at $k = 15$ and then begins to reduce. Thus, we use $k = 15$ for our experiments.

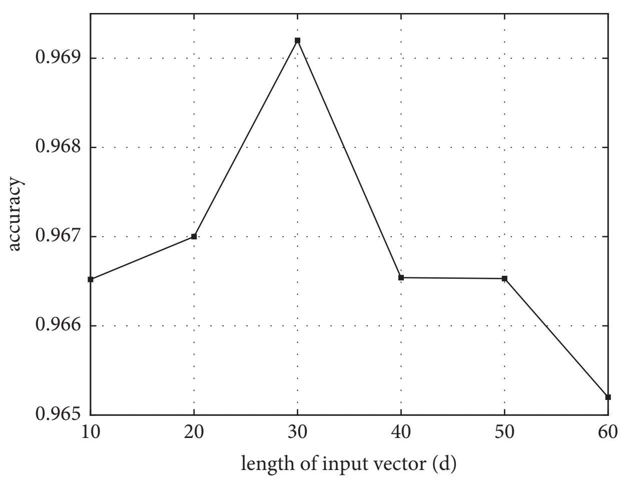

Figure 12: Accuracy of different input vector lengths.

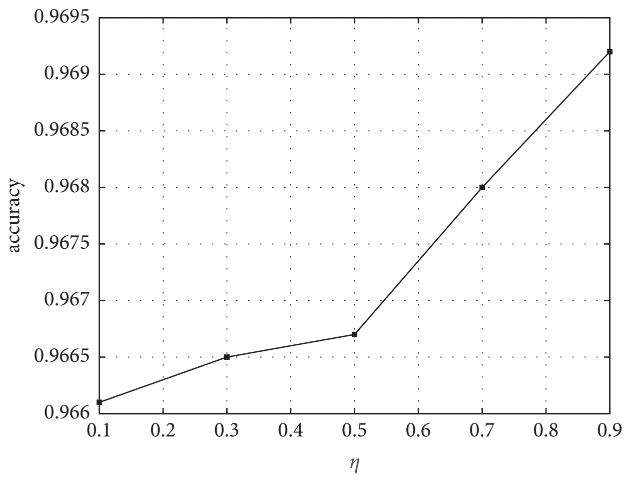

Figure 13: Accuracy of different η.

In the SVM-C model, an original URL is converted into a d-dimensional feature vector via a matrix-vector operation $(\mathbf{A}, \mathbf{a})$. The parameter d determines the dimension of extracted feature vectors and is related to the computational complexity of the method. Figure 12 displays the overall accuracy of the proposed model under different choices of d. Here we fix $k = 15$, $\varepsilon = 50\%$, and $\eta = 0.9$. As the length d is changed in range of 10 to 60, it reaches the highest accuracy at $d = 30$. Therefore, we use $d = 30$ in the experiments.

The right-hand side of constraint (3b) indicates the lower bound of the distance between data instances and hyperplane $(\mathbf{w}, b)$. In practice, some data instances do not satisfy (3b) and such distance is lower than 1. To address this problem, we replace it with powers of a prefixed parameter $\eta \in (0, 1)$ and change it in each inner iteration. In the numerical tests, we fix $k = 15$, $d = 30$, and $\varepsilon = 50\%$. Figure 13 shows the overall accuracy of the proposed model under different η. We change η from 0.1 to 0.9 with a step size of 0.2. The accuracy of SVM-C increases along with the value of η. It reaches the highest accuracy at $\eta = 0.9$. The result reveals that larger η makes SVM-C more flexible and represents higher classification confidence. Thus, we use $\eta = 0.9$ in the numerical tests.

Based on the experimental results, we use $k = 15$, $d = 30$, $\varepsilon = 50\%$, and $\eta = 0.9$ for dataset 1. At this point, the detection accuracy, precision, and F1 score reach more than 96%, while the false positive rate is lower than 5%.

6. Conclusion

In this paper, a novel model called SVM-C was proposed for anomaly detection in network traffic. First, the traffic feature extraction was accomplished based on statistical laws and linear projection. Then, we constructed an optimization problem to obtain the parameters of the proposed model. Finally, the network traffic was classified via SVM classifier. The optimization problem was solved via the BCD method. In the training process, the optimization problem was divided into two subproblems. Each subproblem was solved via the Courant penalty function technique and the PBB method. The numerical results indicated that the proposed model outperformed the benchmark models in terms of accuracy, F1 score, false positive rate, and precision and was robust to different datasets. Furthermore, four main parameters of SVM-C were also explored: the percentage of chosen malicious keywords, the size of sliding window in traffic extraction, the length of final feature vectors, and the parameter in the constraint of the proposed optimization problem.

Data Availability

The dataset 1 and dataset 2 are not public. The dataset 3 is available at https://github.com/exp-db/AI-Driven-WAF.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) (nos. 11771056, 12171051, 11871115, 61941114, 61872836, and 61802025).

References

[1] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection models," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy (Cat. No. 99CB36344)*, pp. 120–132, IEEE, Oakland, CA, USA, May 1999.

[2] M. V. Mahoney and P. K. Chan, "Learning rules for anomaly detection of hostile network traffic," in *Proceedings of the Third IEEE International Conference on Data Mining*, pp. 601–604, IEEE, Leipzig, Germany, July 2003.

[3] C. Krügel, T. Toth, and E. Kirda, "Service specific anomaly detection for network intrusion detection," in *Proceedings of the 2002 ACM Symposium on Applied Computing*, pp. 201–208, Montreal, Canada, May 2002.

[4] E. Eskin, *Anomaly Detection over Noisy Data Using Learned Probability Distributions*, Citeseer, Princeton, New Jersey, USA, 2000.

[5] W. Lee and D. Xiang, "Information-theoretic measures for anomaly detection," in *Proceedings of the 2001 IEEE Symposium on Security and Privacy, S&P 2001*, pp. 130–143, IEEE, Philadelphia, PA, USA, November 2000.

[6] M. A. Ambusaidi, Z. Tan, X. He, P. Nanda, L. F. Lu, and A. Jamdagni, "Intrusion detection method based on nonlinear correlation measure," *International Journal of Internet Protocol Technology*, vol. 8, no. 2-3, pp. 77–86, 2014.

[7] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: a review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11994–12000, 2009.

[8] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[9] C. Sinclair, L. Pierce, and S. Matzner, "An application of machine learning to network intrusion detection," in *Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC'99)*, pp. 371–377, IEEE, Phoenix, ARI, USA, December 1999.

[10] Z. Wang, X. Ren, S. Li, B. Wang, J. Zhang, and T. Yang, "A malicious URL detection model based on convolutional neural network," *Security and Communication Networks*, vol. 2021, 2021.

[11] S. Mukkamala, G. Janoski, and A. Sung, "ntrusion detection using neural networks and support vector machines," in *Proceedings of the 2002 International Joint Conference on Neural Networks*, vol. 2, pp. 1702–1707, Honolulu, Hawaii, May 2002.

[12] G. Stein, B. Chen, A. S. Wu, and K. A. Hua, "Decision tree classifier for network intrusion detection with GA-based feature selection," *In Proceedings of the 43rd Annual Southeast Regional Conference*, vol. 2, pp. 136–141, 2005.

[13] K. M. Al-Gethami, M. T. Al-Akhras, and M. Alawairdhi, "Empirical evaluation of noise influence on supervised machine learning algorithms using intrusion detection datasets," *Security and Communication Networks*, vol. 2021, Article ID 8836057, 28 pages, 2021.

[14] M. Panda, A. Abraham, and M. R. Patra, "A hybrid intelligent approach for network intrusion detection," *Procedia Engineering*, vol. 30, pp. 1–9, 2012.

[15] L. Portnoy, *Intrusion Detection with Unlabeled Data Using Clustering*, Ph.D. dissertation, Columbia University, New York, NY, USA, 2000.

[16] C. Chen, Y. Gong, and Y. Tian, "Semi-supervised learning methods for network intrusion detection," in *Proceedings of the 2008 IEEE International Conference on Systems, Man and Cybernetics*, pp. 2603–2608, IEEE, Singapore, October 2008.

[17] X. Li, P. Yi, W. Wei, Y. Jiang, and L. Tian, "LNNLS-KH: A feature selection method for network intrusion detection," *Security and Communication Networks*, vol. 2021, Article ID 8830431, 22 pages, 2021.

[18] C. Zhang, Y. Chen, Y. Meng et al., "A novel framework design of network intrusion detection based on machine learning techniques," *Security and Communication Networks*, vol. 2021, Article ID 6610675, 15 pages, 2021.

[19] H. Bai, G. Liu, W. Liu, Y. Quan, and S. Huang, "N-gram, semantic-based neural network for mobile malware network traffic detection," *Security and Communication Networks*, vol. 2021, Article ID 5599556, 17 pages, 2021.

[20] K. Wang and S. J. Stolfo, "Anomalous payload-based network intrusion detection," in *Proceedings of the International Workshop on Recent Advances in Intrusion Detection*, pp. 203–222, Springer, Antipolis, France, September 2004.

[21] Y.-H. Dai and R. Fletcher, "Projected barzilai-borwein methods for large-scale box-constrained quadratic programming," *Numerische Mathematik*, vol. 100, no. 1, pp. 21–47, 2005.

[22] Y. Peng, "Research of network intrusion detection system based on snort and ntop," in *Proceedings of the 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 2764–2768, IEEE, Chongqing, China, May 2012.

[23] M. V. O. de Assis, J. J. P. C. Rodrigues, and M. L. Proenca Jr, "A seven-dimensional flow analysis to help autonomous network management," *Information Sciences*, vol. 278, pp. 900–913, 2014.

[24] C. Pascoal, M. R. De Oliveira, R. Valadas, P. Filzmoser, P. Salvador, and A. Pacheco, "Robust feature selection and robust PCA for internet traffic anomaly detection," in *Proceedings of the 2012 Proceedings IEEE Infocom*, pp. 1755–1763, IEEE, Orlando, FL, USA, March 2012.

[25] A. A. Amaral, L. d. S. Mendes, B. B. Zarpelão, and M. L. P. Junior, "Deep IP flow inspection to detect beyond network anomalies," *Computer Communications*, vol. 98, pp. 80–96, 2017.

[26] M. Xie, J. Hu, S. Guo, and A. Y. Zomaya, "Distributed segment-based anomaly detection with Kullback–Leibler divergence in wireless sensor networks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 101–110, 2016.

[27] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "A multistep outlier-based anomaly detection approach to network-wide traffic," *Information Sciences*, vol. 348, pp. 243–271, 2016.

[28] J. Dromard, G. Roudière, and P. Owezarski, "Online and scalable unsupervised network anomaly detection method," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 34–47, 2016.

[29] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, "Mcpad: McPAD: a multiple classifier system for accurate payload-based anomaly detection," *Computer Networks*, vol. 53, no. 6, pp. 864–881, 2009.

[30] 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.

[31] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Information Sciences*, vol. 177, no. 18, pp. 3799–3821, 2007.

[32] Q. Ma, C. Sun, B. Cui, and X. Jin, "A novel model for anomaly detection in network traffic based on kernel support vector machine," *Computers & Security*, vol. 104, 2021.

[33] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[34] W. Sun and Y. Yuan, *Optimization Theory and Methods: Nonlinear Programming*, Springer Science & Business Media, Berlin, Germany, 2006.

[35] J. Yang, P. Yang, X. Jin, and Q. Ma, "Multi-classification for malicious url based on improved semi-supervised algorithm," in *Proceedings of the IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, vol. 1, pp. 143–150, IEEE, Guangzhou, China, July 2017.

[36] Github, "Artificial intelligence-driven web firewall," 2017, https://github.com/exp-db/AI-Driven-WAF.

[37] D. D. Lewis, "Naive (Bayes) at forty: the independence assumption in information retrieval," in *Proceedings of the European Conference on Machine Learning*, pp. 4–15, Springer, Chemnitz, Germany, April 1998.

[38] H. Ramchoun, M. Amine, J. Idrissi, Y. Ghanou, and M. Ettaouil, "Multilayer perceptron: architecture optimization and training," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 1, p. 26, 2016.