Hindawi
Security and Communication Networks
Volume 2022, Article ID 6401316, 13 pages
https://doi.org/10.1155/2022/6401316

Research Article

Anomaly Detection of System Call Sequence Based on Dynamic Features and Relaxed-SVM

Xiaoyao Liao,[1] Changzhi Wang ,[1,2] and Wen Chen[1]

[1]*School of Cyber Science and Engineering, Sichuan University, Chengdu 610065, China*
[2]*Zhihuiyuntian Technology, Chengdu 610031, China*

Correspondence should be addressed to Changzhi Wang; wchzhi@gmail.com

Received 17 September 2021; Revised 29 November 2021; Accepted 21 December 2021; Published 1 February 2022

Academic Editor: Chunhua Su

The system call sequences of processes are important for host-based anomaly detection. However, the detection accuracy can be seriously degenerated by the subsequences which simultaneously appeared in the call sequences of both normal and abnormal processes. Furthermore, the detection may be obstructed especially when the normal/abnormal distributions of subsequences are extremely imbalanced along with many ambiguous samples. In the paper, the system call sequences are divided into weighted subsequences with fixed-length. Secondly, a suffix tree of each system call sequence is constructed to automatically extract the variable-length subsequence from the longest repeated substring of the tree. The frequencies of the fixed-and variable-length subsequences that appeared in each system call sequence constitute its feature vector. Finally, vectors are input into a cost-sensitive and relaxed support vector machine, in which the penalty-free slack of the relaxed SVM is split independently between the two classes with different weights. The experimental results on two public datasets ADFA-LD and UNM showed that the AUC of the proposed method can reach 99%, while the false alarm rate is only 2.4%.

1. Introduction

System calls provide interfaces between system functions and the user applications. And the call sequences can reflect the targets of process actions. Accordingly, the system call sequences stored in various auditing and logging systems are important intrusion detection objects [1, 2].

Usually, system call sequences are broken into subsequences and submitted to classification models [3]. Jewell and Beaver [4] proposed that the unique sequences of system calls are the basis for discriminating normal and abnormal behaviors of processes. Helman and Bhangoo [5] et al. defined the priority of system call sequence based on the probability of the occurrence of system calls to get typical features. Lee and Stolfo [6] performed machine learning tasks on operating system call sequences of normal and abnormal executions of the UNIX Sendmail program. Xie et al. [7–9] attempted to reduce the dimension of the frequency vectors of subsequences based on PCA to enhance the computational efficiency. Haider et al. [10] proposed to take both the rarest repeating subsequence and the most frequent repeating subsequence in the system call sequence as statistical features. Kan et al. [11] proposed a novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network (APSO-CNN). Bian et al. [12] extracted graph-based features from host authentication logs, which are then employed in the detection of APT targets in the network. Shin and Kim [13] preprocessed the collected data using n-gram to overcome the limitations of the sequence time delay embedding (STIDE) algorithm for host intrusion detection system (HIDS).

Recently inspired by linguistics mining in NLP (Nature language process), researchers started to analyze the semantic relations between subsequences. Forrest and Hofmeyr [1] defined the sequences as phrases composed of words (system calls), and then utilized artificial immune systems to classify the phrases. Creech and Hu [14] proposed to draw semantic features of the system call sequences (phrases) using context-free grammar and built an extreme

learning machine for the classifications. Liao and Vemuri [15] calculated the TF-IDF scores of system call sequences and input them into the K-NN model for abnormal detection. Zhang and Shen [16] built an improved TF-IDF model of subsequences, which takes both the time information and the correlation between the processes into consideration. Marteau [17] proposed a similarity measurement method to evaluate the similarity between symbolic sequences. Ambusaidi et al. [18] proposed a supervised feature selection algorithm, which is able to handle both linearly and nonlinearly related data features. Shams et al. [19] designed a new context-aware feature extraction method for convolutional neural network (CNN)-based multiclass intrusion detections. Subba [20] combined TF-IDF vectorizer and singular value decomposition (SVD) to design a novel HIDS framework for anomalous system processes detection.

However, it is difficult to select appropriate subsequences of system calls to discover the real purpose of the calling actions. Laszka et al. [21] proved that the optimal length of subsequence is highly dependent on data and applications and needs to be carefully fine-tuned. For example, if the subsequence is too short then we may get an incomplete calling trace. On the other hand, if the subsequence is too long, then the malicious calls are mixed with many normal system calls, and the extracted features may be disturbed. Finally, there are many normal call sequences while the abnormal calls are extremely few, and many unusual API sequences, along with incomplete sequences may have a strong influence on the classification model. How to deal with the imbalance and noise data is also a big challenge.

In view of the above problems, not only the semantic information contained in short subsequence but also the representative features in variable-length subsequences are considered to generate combined features. Usually, system call sequences contain repeated subsequences, which are regarded as program-specific behavior patterns. And the length of repeated subsequences is different among call sequences. In order to generate variable-length repeating sub-sequences, a suffix tree is constructed for each system call sequence and the longest repeating substring is automatically extracted from the subtrees. Furthermore, to address the imbalance and noisy subsequences of normal and abnormal calls, the widely used relaxed support vector machine (RSVM) is improved by assigning different weights and free slack amounts to the positive and negative classes, which are scaled by the sizes of the two classes to reduce the influence of data imbalance and outliers.

2. Feature Extraction

In order to automatically extract the call subsequences and related features, which can reflect the real target of the system calls, we present a dynamic feature extraction method for system call sequences. As shown in Figure 1 the features are extracted from both fixed-length and variable-length subsequences. In the first step, the training sequences are split into subsequences by n-gram [22], and each subsequence is weighted by TF-IDF [23]. Then, the first n subsequences with big weight values are selected to composite a fixed-length subsequence set. In the second step, suffix trees are constructed for training sequences. The longest repeating substrings in the suffix trees are selected as variable-length subsequences. In the third step, both the fixed-length and variable-length subsequences are combined to get a corpus set. In step 4, the occurrence frequencies of the corpus subsequences in each system call sequences are counted to constitute feature vectors. Furthermore, in step 5, an autoencoder [24] is also utilized to reduce the vectors' dimension before being submitted to the classifiers in step 6.

2.1. Fixed-Length Subsequence Based on Semantics. The TF-IDF (term frequency-inverse document frequency) [23] is commonly utilized in text mining to evaluate the importance of every single word or phrase in a document. In this paper, the TF-IDF is employed to evaluate the system calls, which have been coded into word sequences. As described in [1], firstly, each system call is represented by a unique word (or number), accordingly, the call sequences become word sequences. As defined in (1), $\overrightarrow{seq_i}$ is the j^{th} element in the set.

Definition 1. Inverse ratio of sequence frequency. The inverse ratio of a sequence frequency is the IDF reverse file frequency defined in TF-IDF. As shown in equation (1), N is the number of training sequences, and n_{t_i} is the number of fixed-length subsequence t_i that appeared in the training set.

$$idf\left(t_i, \overrightarrow{seq_i}\right) = \frac{N}{n_{t_i}}. \tag{1}$$

Definition 2. Vocabulary frequency of single sequence. As shown in equation (2), fre_{ij} represents the frequency of the fixed-length subsequence t_i in a system call sequence $\overrightarrow{seq_i}$. And the vector Fre_j represents the frequency of all the fixed-length subsequences $t = \{t_1, t_2, \ldots t_m\}$ in the system call sequence $\overrightarrow{seq_i}$.

$$fre_{ij} = tf\left(t_i, \overrightarrow{seq_j}\right) = \frac{\text{the frequency of } t_i}{\text{the number of word in the } \overrightarrow{seq_j}}. \tag{2}$$

$$Fre_j = \left[fre_{1j}, fre_{2j}, \ldots fre_{mj}\right]. \tag{3}$$

Definition 3. Process behavior weight. It is the combination of inverse ratio and vocabulary frequency of every single subsequence, as shown in (4), to evaluate the importance of a single vocabulary (subsequence) t_i to one system call sequence $\overrightarrow{seq_j}$.

$$\text{weight}\left(t_i, \overrightarrow{seq_i}\right) = tf\left(t_i, \overrightarrow{seq_i}\right) \times \log\left(\frac{N}{n_{t_i}} + 0.01\right). \tag{4}$$

Definition 4. Fixed length corpus of system call sequences. The subsequences with the first three highest weights, calculated by (4), in a single process are included in the fixed-length sequence corpus. In equation (5), where t_{ji} represents

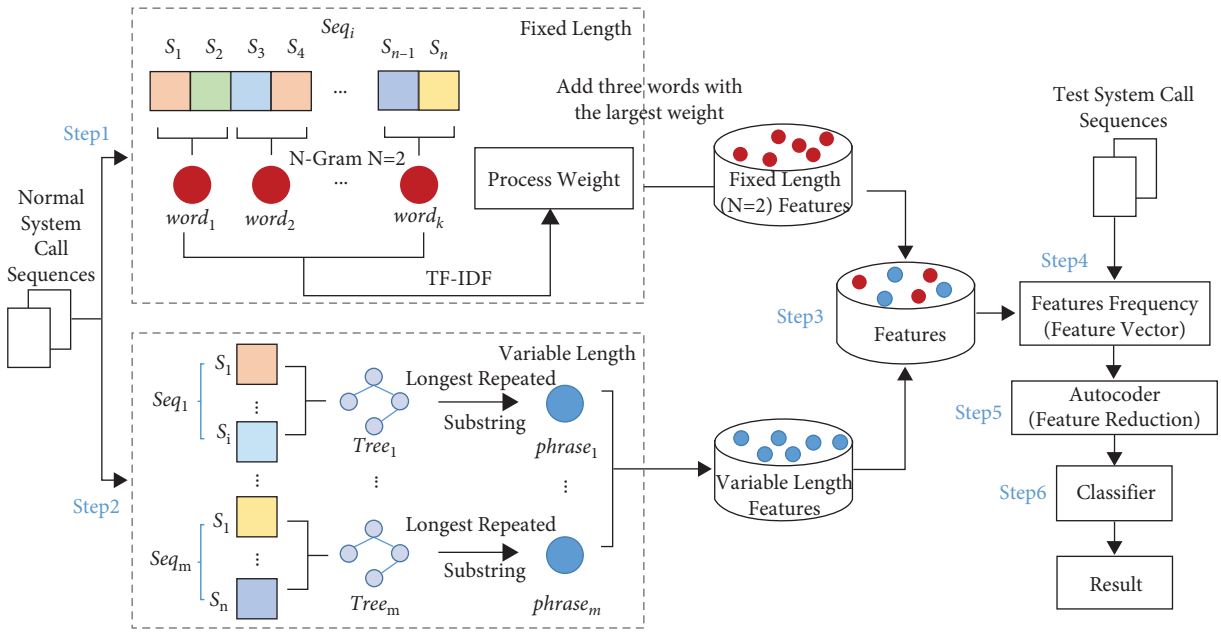

Figure 1: Dynamic feature extraction.

the i^{th} fixed-length subsequence in the system call sequence $\overrightarrow{seq_j}$.

$$\text{corpus}_{\text{fixed}} = [t_{11}, t_{12}, t_{13}, \ldots, t_{n1}, t_{n2}, t_{n3}]. \quad (5)$$

2.2. Variable-Length Subsequence. Usually, the repeated subsequence in a system call sequence can reflect the behavior patterns of processes, which are much useful for abnormal detection. In the paper, these sequence fragments are defined as the representative features with variable lengths.

2.2.1. Constructing Suffix Tree. Ukkonen [25] is a commonly used suffix tree construction algorithm based on path compression and suffix chain. In the paper, Ukkonen is utilized to construct suffix trees from single system call sequences. Let a sequence seq = "6, 4, 1, 4, 1, 4, 3." Firstly, the suffix tree for initial string S = "6", c = null (S is the current string for constructing suffix tree, p = "1,4" is the next character) is shown in Figure 2(a); and then, S = "6," c = "4," so all possible suffixes of $S + c$ are seq_1 = "4," seq_2 = "6, 4," as shown in Figure 2(b); finally, S = "6, 4," c = "1," so all possible suffixes of $S + c$ are seq_1 = "1," seq_2 = "4, 1," and seq_3 = "6, 4, 1" as shown in Figure 2(c). In the same way, the result of the suffix tree for seq = "6, 4, 1, 4, 1, 4, 3" is shown in Figure 3.

2.2.2. Search the Longest Repeating Substring. After the establishment of a suffix tree, the longest repeating substring p_k of the tree is selected to represent the behavior patterns of a system call sequence. As shown in Figure 4, for seq = "6, 4, 1, 4, 1, 4, 3" the deepest nonleaf node is node "4" and the longest repeating substring is p = "1, 4," which is incorporated into the variable-length sequence set $\text{corpus}_{\text{variable}}$.

$$\text{corpus}_{\text{variable}} = [p_1, p_2, \ldots, p_n]. \quad (6)$$

2.2.3. Segmenting Long System Call Sequence. To alleviate the effect of long system call sequences on the suffix trees' generation efficiency, the long sequences are segmented into subsequences. As shown in Figure 5, for the system call sequence $seq_i = \{s_1, s_2, \ldots, s_{500}, \ldots, s_{len}\}$ with length $len > 500$. The sequence seq_i is divided into subsequence $\{seq_{i1}, seq_{i2}, \ldots, seq_{ij}\}$. Let seq_{ij} represents the j^{th} subsequence of seq_i. The suffix trees $\{Tree_{i1}, Tree_{i2}, \ldots, Tree_{ij}\}$ are constructed for each subsequence $\{seq_{i1}, seq_{i2}, \ldots, seq_{ij}\}$.

2.2.4. Generating Corpus. Both the fixed-length subsequence set $\text{corpus}_{\text{fixed}}$ and the variable-length subsequence set $\text{corpus}_{\text{variable}}$ constitute a combination set Corpus to represent the behavior patterns of the system calls.

$$\text{Corpus} = \text{corpus}_{\text{fixed}} \cup \text{corpus}_{\text{variable}}. \quad (7)$$

2.2.5. Feature Extraction. Firstly, the frequency of the subsequences defined in (8) that appeared in each system call sequence is counted by AC automaton (Aho–Corasick automaton) [24]. Let $fre_{t_{ji}}$ represents the occurrence frequency of fixed-length subsequence t_i in a system call sequence seq_j. $fre_{t_{ji}}$ represents the frequency of a variable-length subsequence p_i in seq_j. The frequency vectors of all the call sequences constitute a feature matrix.

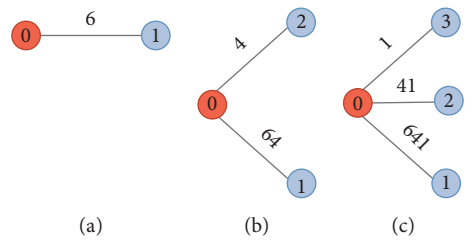

Figure 2: Constructing suffix tree.

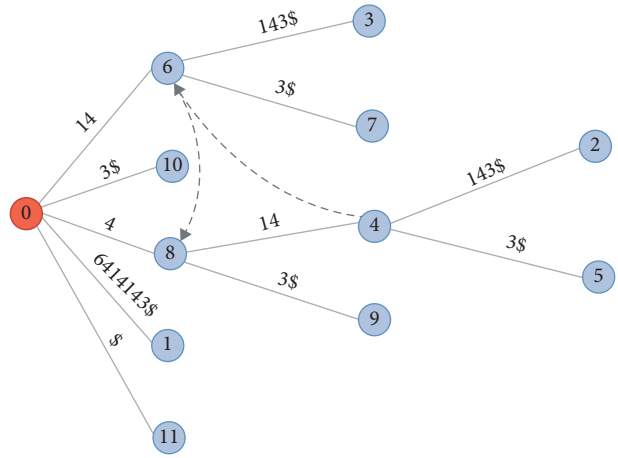

Figure 3: Constructed suffix tree.

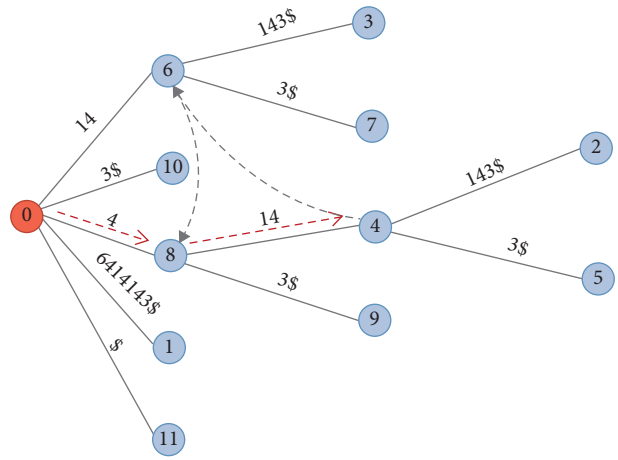

Figure 4: The longest repeating substring.

$$Vec = \begin{bmatrix} fre_{t_{11}} & \cdots & fre_{t_{1k}} & fre_{p_{11}} & \cdots & fre_{p_{1m}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ fre_{t_{j1}} & \cdots & fre_{t_{jk}} & fre_{p_{j1}} & \cdots & fre_{p_{jm}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ fre_{t_{n1}} & \cdots & fre_{t_{nk}} & fre_{p_{n1}} & \cdots & fre_{p_{nm}} \end{bmatrix}. \quad (8)$$

With the increasing of system call sequences, the number of subsequences in corpus also increases dramatically. In order to control the dimension of the feature vectors and fascinate mining of the potential features in the matrix Vec, in (8), the autoencoder [26] is utilized to reduce the dimension of Vec. Finally, Vec is submitted to weighted relaxed support vector machines described in the next section.

3. Weighted Relaxed Support Vector Machines

The widely used RSVM [27] is an extension of SVM-L2 with an additional penalty-free slack variable for each sample, which allows influential support vectors to be relaxed, such that a restricted amount of penalty-free slack is used to relax support vectors and push them towards their respective classes. In the paper, based on WRSVM [28], we modify

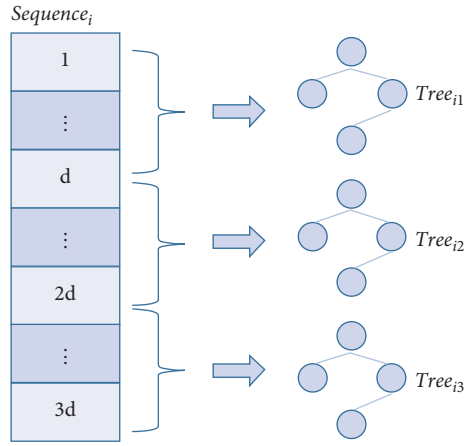

Figure 5: The suffix tree for substrings.

RSVM by assigning different weights and free slack amounts to the two classes, normal and abnormal call sequences, which are scaled by the positive and negative class sizes with different penalty control factors, C_1 and C_2.

The enhanced weighted relaxed support vector machines (EWR-SVM) model is given in equation (10), which is an extension of the well-known SVM formulation RSVM. However, the constructed model is different from RSVM in such a way that, it differentiates between positive and negative classes, and considers different weights inversely proportional to the class sizes.

$$\min_{\omega,b,\xi,v} \frac{1}{2}\langle \omega \bullet \omega \rangle + \frac{C_1}{2n^+}\xi_i^2 - \frac{C_2}{2n^-}\sum_{i\in I^-}\xi_i^2. \tag{9}$$

$$\begin{aligned} s.t.\ & \langle \omega \bullet \omega \rangle + b \geq 1 - \xi_i - v_i, \forall i \in I^+ \ -\langle \omega \bullet \omega \rangle - b \geq 1 - \xi_i - v_i, \forall i \in I^-, \\ & \sum_{i\in I^+} v_i \leq n^+\gamma, \\ & \sum_{i\in I^-} v_i \leq n^-\gamma, \\ & v_i \geq 0, \end{aligned} \tag{10}$$

where n^+ and n^- are the sizes of the majority (normal) and minority (abnormal) class, I^+ and I^-, respectively. Free slack is denoted by the variable v_i in the constraints for the i^{th} sample. Due to imbalance, we provide separate amounts of total free slack for the normal and the abnormal classes in the constraints and v_i is parameterized by γ, which is the free slack provided per sample.

Then calculate the first order partial derivative of the Lagrangian function of equation (10) with respect to the related Lagrangian multipliers, we can get the Wolfe dual of equations (10) and (12) can be efficiently solved using the sequential minimal optimization (SMO) algorithm, and the dot products $\langle x_i \cdot x_j \rangle$ in equation (12) can be replaced by a kernel $k(x_i, x_j)$ for nonlinear classification.

$$\min_{\alpha,\beta^+,\beta^-} \sum_{i\in I}\alpha_i - \frac{1}{2}\sum_{i\in I}\sum_{j\in I} yy\alpha_i\alpha_j\langle x_i \cdot x_j \rangle - \frac{n^+}{2C_1}\sum_{i\in I^+}\alpha_i^2 - \frac{n^-}{2C_2}\sum_{i\in I^-}\alpha_i^2 - n^+\gamma\beta^+ - n^-\gamma\beta^-, \tag{11}$$

$$\begin{aligned} s.t.\ & -\sum_{i\in I^+}\alpha_i + \sum_{i\in I^-}\alpha_i = 0, \\ & 0 \leq \alpha_i \leq \beta^+ \quad \forall i \in I^+, \\ & 0 \leq \alpha_i \leq \beta^- \quad \forall i \in I^-. \end{aligned} \tag{12}$$

4. Experiment

4.1. Datasets Description. In this section, a group of comparisons is carried out to test the performance of the proposed method based on two public system call sequence datasets ADFA-LD and UNM. The ADFA-LD [29] was released by the Australian defense force academy. It contains thousands of system call traces collected from modern Linux local servers. The UNM [30] was released by the University of New Mexico, which contains system call sequences of normal applications in the Linux system and sequences from attacking progress. The detailed descriptions of the two datasets are shown in Tables 1–3. And we can see that the UNM is an obviously imbalanced dataset.

Firstly, the sequence features are extracted by counting the number of the appearance of the fixed-length and variable-length subsequences in each system call sequence, and the abstracted features are verified by mathematical tests including information gain and Mann–Whitney U [31] test in the attachment. And then the features are input into classification models including the proposed EWR-SVM, naive Bayes, logistic regression, random forest, and gradient descent, and the performance is also compared with some traditional methods [8, 10, 32].

The results are evaluated by the index of AUC, $F1$-Score, false alarm rate, and ROC, which is also known as the receptivity curve. The ROC curve takes both the false positive rate and the true positive rate into consideration. It represents a curve drawn by the subjects under specific stimulus conditions due to different results obtained in different conditions. And AUC is the area under the ROC curve.

$$F_1 = \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}},$$
$$FA = \frac{\text{FP}}{\text{FP} + \text{TP}}. \tag{13}$$

4.2. Experimental Result

4.2.1. Experimental Result. The experimental data set ADFA consists of 833 normal system call sequences and 719 attack system call sequences. The experimental data set UNM consists of 6 normal sample files and 16 attack sample files. The system call sequences' features are extracted from the data set ADFA-LD and UNM datasets. Fixed length features are extracted from system call sequences by a sliding window with length 2. Variable-length features are extracted from system call sequences by a suffix tree. The anomaly detection results based on the classifiers, including EWR-SVM, naive Bayes, logistic regression, random forest, and gradient descent tree are shown in Table 4 and Figures 6 and 7.

Obviously, the performance of EWR-SVM is better than the others on the two datasets. The results demonstrated that the features extracted by our method contain useful information to reflect the behavior patterns behind the call sequences. And the features are useful for model recognition of the abnormal program behaviors. In order to further demonstrate the effectiveness of our method, a feature validation is carried out on the datasets ADFA-LD and UNM.

Table 1: Detail of UNM and ADFA-LD dataset.

Sequence length	ADFA-LD	UNM
Min	75	7
Max	4494	183 018
Mean	462	9031
Deviation	522	25 111

4.2.2. Feature Validation Verification Experiment. The occurrence frequencies of fixed- and variable-length subsequence in ADFA-LD and UNM are shown in Figures 8–15. From the figures, we can see that the distributions of the word (call) frequency are different between the normal and attacking traces (system call sequences). The information gain and Mann–Whitney U test is carried out to demonstrate the effectiveness of the extracted features.

The information gain of both the selected fixed-length sequences and the left ones are shown in Table 5. The table showed that the information gain of the selected fixed-length sequence feature is obviously higher than that of the others. The results demonstrated that fixed-length sequences with the high TF-IDF weight are useful, which can be utilized to extract the behavior features of system call sequences.

According to the statistical results, we found that the frequency of the sub-sequences of normal sequences and attacking sequences does not follow the normal distribution in two datasets, so a nonparametric method is employed to conduct Mann–Whitney U test [31]. The following hypotheses are proposed:

H_0: there is no significant difference between the subsequence frequency of the normal sequence and the sub-sequence frequency of the abnormal sequence

H_1: opposite hypothesis of H_0

The results of the Mann–Whitney U test is shown in Table 6. According to the law of statistics, we know that when $P - \text{value} > 0.05$, the original hypothesis H_0 is true, while $P - value < 0.05$, the original hypothesis H_0 is rejected, and the hypothesis H_1 is true. From Table 6, we can see that the appearing frequency of fixed-length subsequence of normal sequences in ADFA-LD and UNM datasets is significantly different from that of the abnormal sequences. Similarly, the frequency of variable-length subsequences of normal sequences in ADFA and UNM datasets is also obviously different from that of the abnormal ones.

4.2.3. Comparison with Traditional Methods. In this section, we compared the EWR-SVM with other abnormal traces detection methods [8, 10, 32], and the results are shown in Table 7. From the table, we can see that our method is obviously superior to others. In [8], system call sequences are classified according to the appeared abnormal frequencies in the tested traces. In their method, the optimal window width

Table 2: The information for the ADFA-LD dataset.

Data type	Data size	Label
Training	833	Normal
Hydra-FTP	162	Attack
Hydra-SSH	148	Attack
Adduser	91	Attack
Java-meterpreter	125	Attack
Meterpreter	75	Attack
Webshell	118	Attack

Table 3: UNM dataset.

Data type	Sample files	Num
Normal	bounce.int, bounce1.int,plus.int, queue.int, sendmail.int, sendmail1.int	6
Code intrusion	decode-280.int, decode-314.int	2
Circular forwarding error	fwd-loops-1.int, fwd-loops-2.int, fwd-loops-3.int, fwd-loops-4.int, fwd-loops-5.int	5
forwarding error	sscp-10 763.int, sscp-10 814.int, sscp-10 801.int	3
syslogd	syslog-local-1.int, syslog-local-2.int, syslog-remote-1.int, syslog-remote-2.int	4
Failed invasion	cert-sm5x-1.int, cert-sm565a-1.int	2

Table 4: ADFA-LD and UNM dataset, the results for different models.

Dataset	Algorithm	AUC	$F1$-score	False alarm rate
ADFA-LD	EWR-SVM	99%	0.93	2.4%
	Naive Bayes	94%	0.90	8%
	Logistic regression	96%	0.94	3%
	Random forest	98%	0.92	7%
	GBDT	98%	0.94	4%
UNM	EWR-SVM	97%	0.83	0%
	Naive Bayes	89%	0.36	0%
	Logistic regression	95%	0.72	10%
	Random forest	97%	0.83	0%
	GBDT	97%	0.8	0%

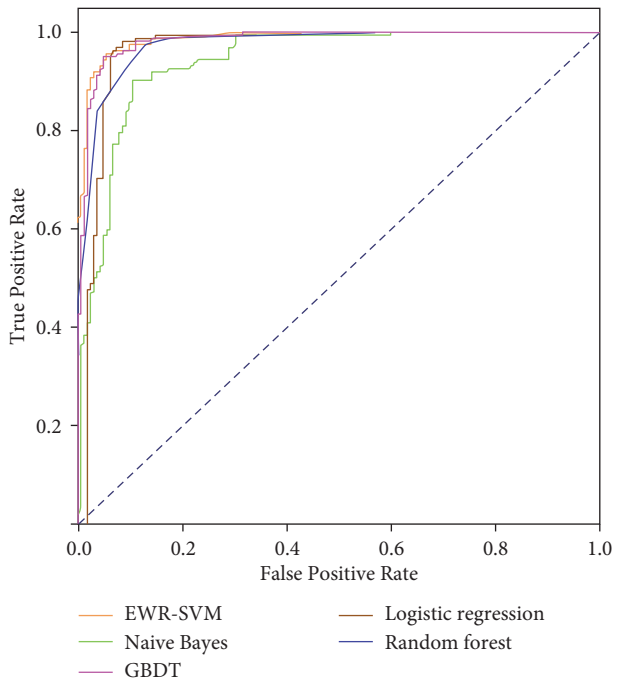

Figure 6: ADFA-LD dataset: Area under the ROC curve (AUC) for different models.

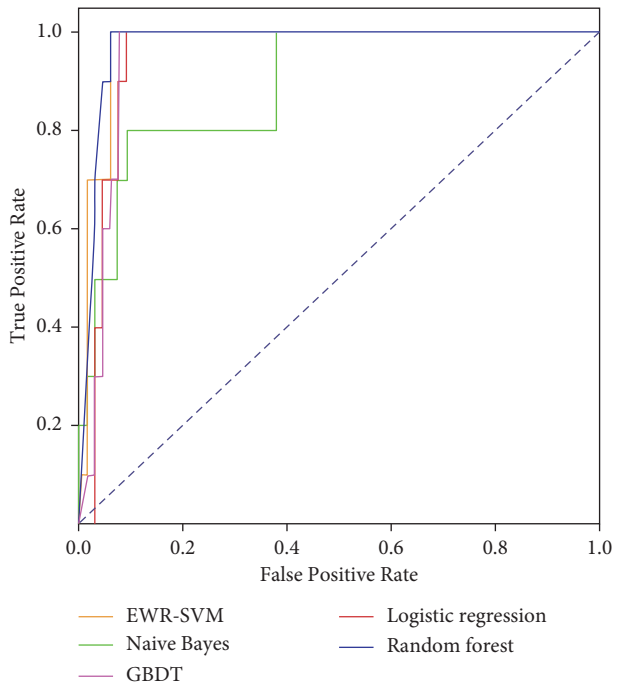

Figure 7: UNM dataset: Area under the ROC curve (AUC) for different models.

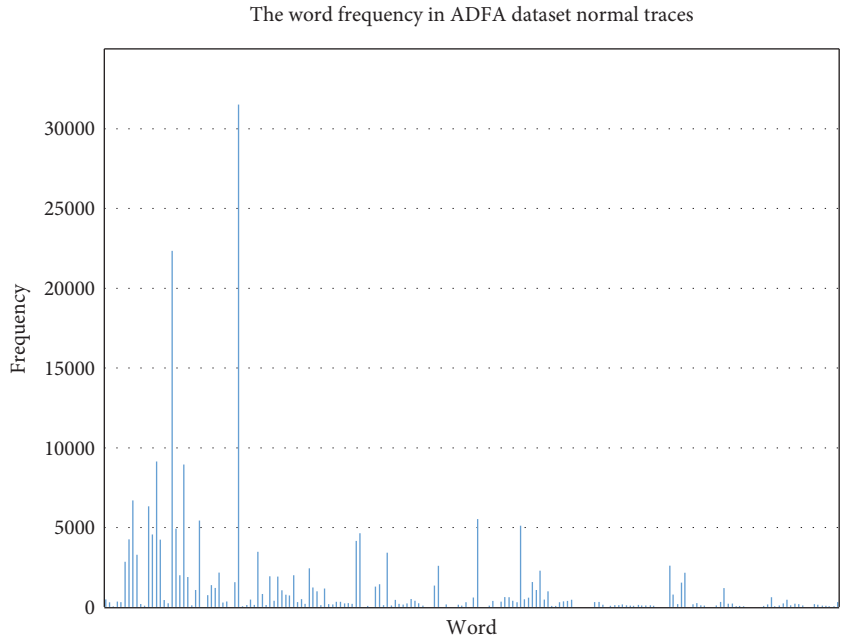

Figure 8: The word frequency in ADFA normal traces.

is set by empirical test. However, the optimal length of the subsequence varies among different datasets, resulting in unstable performance. Haider [10] proposed to utilize the rarest repeating subsequence, the most frequently repeated subsequence, along with the maximum and minimum system calls in a single sequence to extract the trace features. However, the correlative contextual semantic information in the sequence is ignored. Anandapriya and Lakshmanan [32] brought the conception of contextual semantic information into trace detection with fixed-length windows. The results in Table 7 showed that the AUC of the proposed EWR-SVM is much larger than the others, while its false-positive rate is lower than the comparison targets. Thatis because our method considers not only semantic information of the sequences but also the behavior patterns of processes behind the calls. Furthermore, we proposed to assign different weights and free slack amounts to the two classes (normal and attacking traces), which are scaled by the normal and

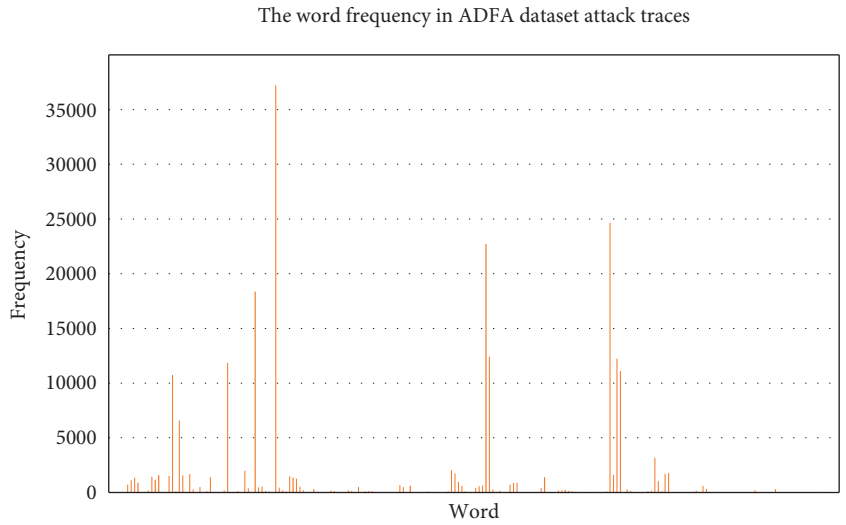

Figure 9: The word frequency in ADFA attacking traces.

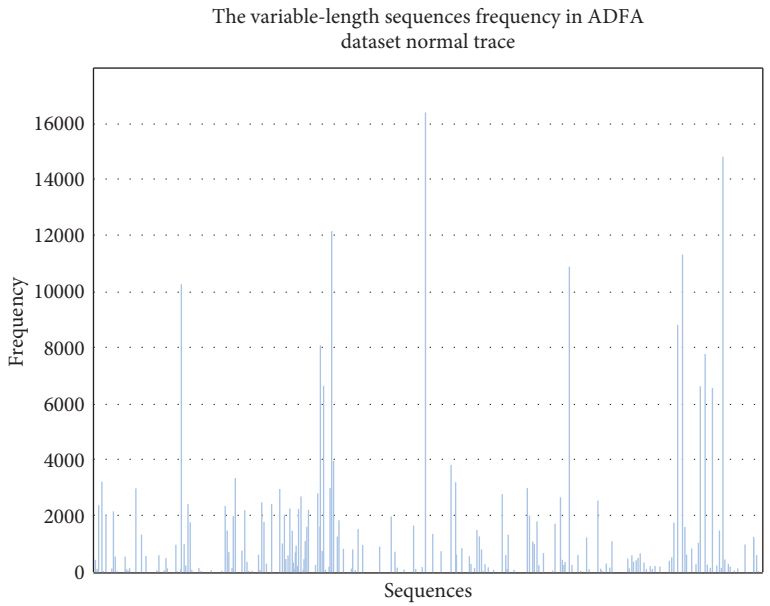

Figure 10: The variable-length sequences frequency in ADFA normal traces.

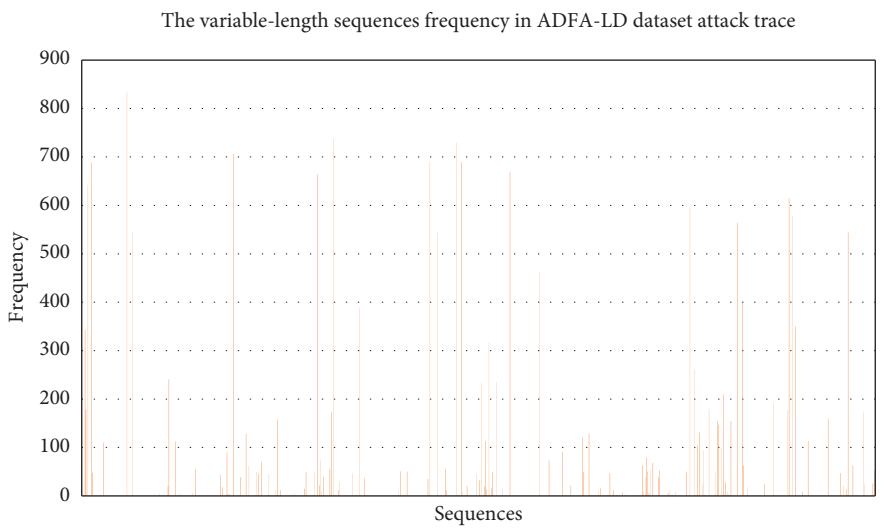

Figure 11: The variable-length sequences frequency in ADFA attacking traces.

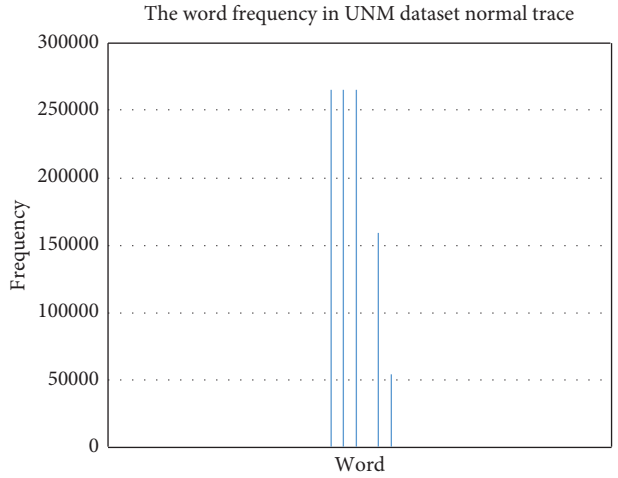

Figure 12: The word frequency in UNM normal traces.

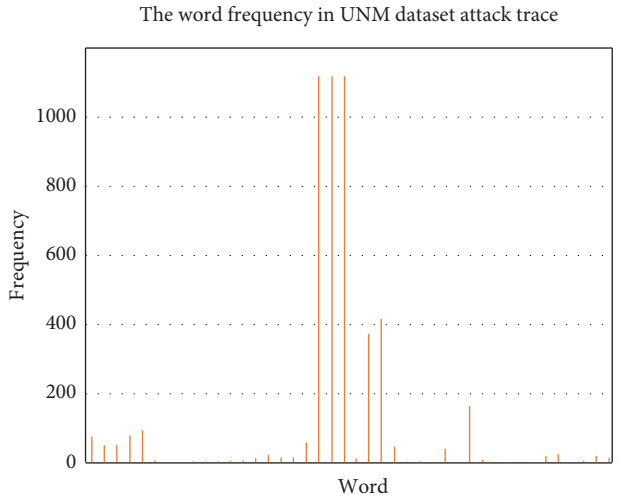

Figure 13: The word frequency in UNM attacking traces.

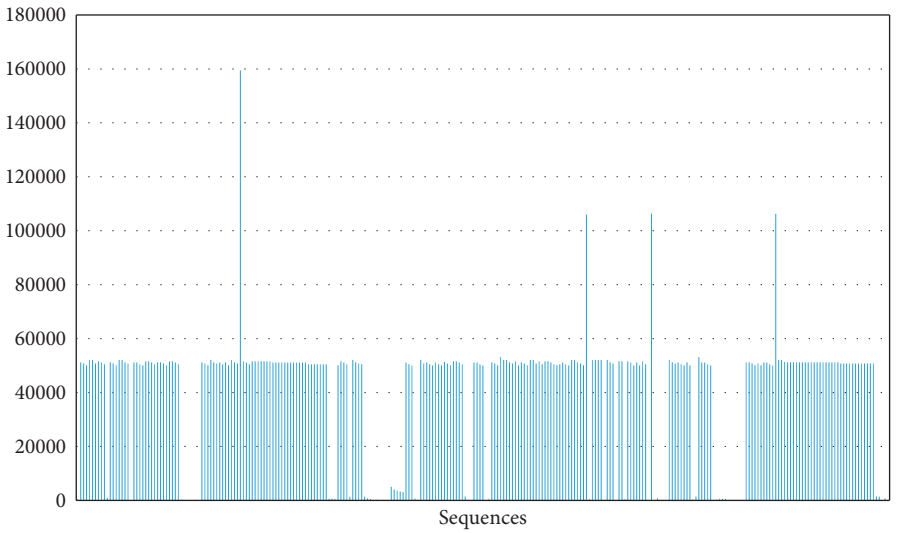

Figure 14: The variable-length sequences frequency in UNM normal traces.

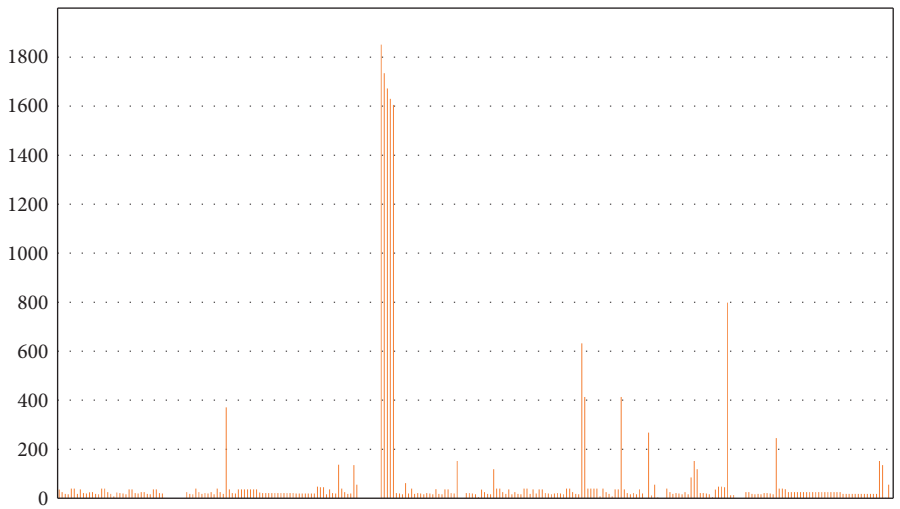

Figure 15: The variable-length sequences frequency in UNM attacking traces.

Table 5: ADFA-LD and UNM dataset: the information gain results for fixed-length sequences.

Dataset	Sequences type	Average information gain
ADFA-LD	Selected	0.185
	Unselected	0.014
UNM	Selected	0.025
	Unselected	0.015 3

Table 6: ADFA and UNM datasets the results for Mann–Whitney U.

Dataset	Sequences type	Statistic	P-Value
ADFA-LD	TF-IDF fixed-length sub-sequences	14692.5	3.31×10^{-10}
	Variable-length sub-sequences	144565.5	1.15×10^{-109}
UNM	TF-IDF fixed-length sub-sequences	875.5	4.18×10^{-2}
	Variable-length sub-sequences	2915.5	4.26×10^{-79}

Table 7: Comparison results.

Algorithm	AUC	False positive rate
One class SVM [8]	70%	20%
Data feature retrieval and decision engine [10]	60%	23%
Semantic-based method [25]	82%	4.2%
EWR-SVM	99%	2.4%

abnormal class sizes to release the affection of imbalanced data distribution between the two classes. And the slack factors in EWR-SVM make the outliers have less influence on the optimal hyperplane and ensure a large margin between the two classes.

5. Conclusions

In the paper, both fixed-length and variable-length subsequence of system calls are taken into consideration for the host-based intrusion detection. The semantic weight is incorporated into the selection process of fixed-length subsequences. On the other hand, the variable-length subsequence is automatically selected from the suffix tree of each call sequence. In order to deal with the imbalance data distribution and samples outliers of the call sequences, a cost-sensitive relaxed support vector machine EWR-SVM is proposed, in which the restricted penalty-free slack is split independently between the two classes in proportion to the number of samples in each class with different weights. Both

the theoretical analysis and experimental comparison results demonstrated our methods are more suitable for the detection of attacking traces in system call sequences.

Data Availability

Host intrusion detection datasets in this article are public datasets. Web pages are as follows: (1) http://www.cs.unm.edu/ĩmmsec/systemcalls.htm. (2) https://research.unsw.edu.au/projects/adfa-ids-datasets.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (020YFB1805405, and 2019QY0800), and the Natural Science Foundation of China Nos. 61 872 255, U19A2068, and U1736212.

References

[1] S. Forrest and S. A. Hofmeyr, "Sense of self for unix processes," in *Proceedings of the Security and Privacy*, pp. 120–128, IEEE, Oakland, CA, USA, May 1996.

[2] S. A. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion detection using sequences of system calls," *Journal of Computer Security*, vol. 6, no. 3, pp. 151–180, 1998.

[3] C. Warrender and S. Forrest, "Detecting intrusions using system calls: alternative data models," in *Proceedings of the Security and Privacy*, pp. 1–20, IEEE, Oakland, CA, USA, May 1999.

[4] B. Jewell and J. Beaver, "Host-based data exfiltration detection via system call sequences," in *Proceedings of the Conf. Inform. Warf. Secur.*, pp. 134–137, Columbia University, New York, NY, USA, January 2011.

[5] P. Helman and J. Bhangoo, "A statistically based system for prioritizing information exploration under uncertainty," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 27, no. 4, pp. 449–466, 1997.

[6] W. Lee and S. J. Stolfo, "Learning patterns from unix process execution traces for intrusion detection," in *Proceedings of the AAAI Work AI Approaches to Fraud Detect Risk Manag*, pp. 50–56, Columbia University, New York, NY, May 1997.

[7] M. Xie and J. Hu, "Evaluating host-based anomaly detection systems: a preliminary analysis of adfa-ld," in *Proceedings of the Int. Congr. Image Signal Process.*, pp. 1711–1716, IEEE, Hangzhou, China, December 2013.

[8] M. Xie, J. Hu, and J. Slay, "Evaluating host-based anomaly detection systems: application of the one-class svm algorithm to adfa-ld," in *Proceedings of the Int. Conf. Fuzzy Syst. Knowl. Discovery*, pp. 978–982, IEEE, Xiamen, China, August 2014.

[9] M. Xie, J. Hu, and E. Chang, "Evaluating host-based anomaly detection systems: application of the frequency-based algorithms to adfa-ld," *Network and System Security*, vol. 8792, pp. 542–549, 2014.

[10] W. Haider, J. Hu, and M. Xie, "Towards reliable data feature retrieval and decision engine in host-based anomaly detection systems," in *Proceedings of the 10th IEEE Conf. Ind. Electron. Appl.*, pp. 513–517, IEEE, Auckland, New Zealand, June 2015.

[11] X. Kan, Y. Fan, Z. Fang et al., "A novel iot network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network," *Information Sciences*, vol. 568, pp. 147–162, 2021.

[12] H. Bian, T. Bai, M. A. Salahuddin, N. Limam, A. A. Daya, and R. Boutaba, "Uncovering lateral movement using authentication logs," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 1049–1063, 2021.

[13] Y. Shin and K. Kim, "Comparison of anomaly detection accuracy of host-based intrusion detection systems based on different machine learning algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 2, 2020.

[14] G. Creech and J. Hu, "A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns," *IEEE Transactions on Computers*, vol. 63, no. 4, pp. 807–819, 2014.

[15] Y. Liao and V. R. Vemuri, "Use of k-nearest neighbor classifier for intrusion detection," *Computers & Security*, vol. 21, no. 5, pp. 439–448, 2002.

[16] Z. Zhang and H. Shen, "Application of online-training svms for real-time intrusion detection with different considerations," *Computer Communications*, vol. 28, no. 12, pp. 1428–1442, 2005.

[17] P.-F. Marteau, "Sequence covering for efficient host-based intrusion detection," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 994–1006, 2019.

[18] M. A. Ambusaidi, X. He, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986–2998, 2016.

[19] E. A. Shams, A. Rizaner, and A. H. Ulusoy, "A novel context-aware feature extraction method for convolutional neural network-based intrusion detection systems," *Neural Computing & Applications*, vol. 33, no. 20, pp. 13647–13665, 2021.

[20] B. Subba, "A tfidfvectorizer and singular value decomposition based host intrusion detection system framework for detecting anomalous system processes," *Computers & Security*, vol. 100, p. 102084, 2020.

[21] A. Laszka, W. Abbas, and S. Shankar, "Optimal thresholds for intrusion detection systems," in *Proceedings of the Symp. Bootcamp Sci. Secur.*, pp. 72–81, ACM, New York, NY, USA, April 2016.

[22] J. R. Ullmann, "A binary n-gram technique for automatic correction of substitution, deletion, insertion and reversal errors in words," *The Computer Journal*, vol. 20, no. 2, pp. 141–147, 1977.

[23] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.

[24] A. V. Aho and M. J. Corasick, "Efficient string matching," *Communications of the ACM*, vol. 18, no. 6, pp. 333–340, 1975.

[25] M. Crochemore and W. Rytter, "On-line construction of suffix trees," *Jewels Of Stringology:Text Algorithms*, vol. 14, 2015.

[26] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[27] O. Şeref, W. A. Chaovalitwongse, and J. P. Brooks, "Relaxing support vectors for classification," *Annals of Operations Research*, vol. 216, no. 1, pp. 229–255, 2014.

[28] O. Seref, T. Razzaghi, and P. Xanthopoulos, "Weighted relaxed support vector machines," *Annals of Operations Research*, vol. 249, no. 1-2, pp. 1–37, 2017.

[29] G. Creech and J. Hu, "Generation of a new ids test dataset: time to retire the kdd collection," in *Proceedings of the Wireless Communications and Networking Conference (WCNC), 2013*, pp. 4487–4492, IEEE, Shanghai, China, April 2013.

[30] Computer Science Department, "University of New Mexico intrusion detection dataset oct 2020," 2020, http://www.cs.unm.edu/immsec/systemcalls.htm.

[31] J. Whitney, "Testing for differences with the nonparametric mann-whitney u test," *The Journal of Wound, Ostomy and Continence Nursing: Official Publication of The Wound, Ostomy and Continence Nurses Society*, vol. 24, no. 1, p. 12, 1997.

[32] M. Anandapriya and B. Lakshmanan, "Anomaly based host intrusion detection system using semantic based system call patterns," in *Proceedings of the IEEE International Conference on Intelligent Systems & Control*, pp. 1–5, Coimbatore, India, January 2015.