An Anomaly Detection Model for Network Intrusions Using One-Class SVM and Scaling Strategy

Ming Zhang(✉), Boyi Xu, and Dongxia Wang

National Key Laboratory of Science and Technology on Information System Security, Beijing Institute of System Engineering, Beijing, China
{mingle_cheung, boyi_xu}@yeah.net, WDX_76738@126.COM

Abstract. Intrusion detection acts as an effective countermeasure to solve the network security problems. Support Vector Machine (SVM) is one of the widely used intrusion detection techniques. However, the commonly used two-class SVM algorithms are facing difficulties of constructing the training dataset. That is because in many real application scenarios, normal connection records are easy to be obtained, but attack records are not so. We propose an anomaly detection model for network intrusions by using one-class SVM and scaling strategy. The one-class SVM adopts only normal network connection records as the training dataset. The scaling strategy guarantees that the variability of feature values can reflect their importance, thus improving the detection accuracy significantly. Experimental results on KDDCUP99 dataset show that compared to Probabilistic Neural Network (PNN) and C-SVM, our one-class SVM based model achieves higher detection rates and yields average better performance in terms of precision, recall and F-value.

Keywords: Intrusion detection · Anomaly detection · One-class SVM · Scaling strategy

1 Introduction

The Internet has brought with endless joy and great convenience. Especially, with the rapid growth of Web applications, everything seems so easy. However, in recent years, "attack", "intrusion" and other similar words frequently appear in people's eyes. We are suffering from increasing network threats. The well-known internet security corporation, Symantec, reminds in its annual Internet Security Threat Report (ISTR) that cybercrime remains prevalent and damaging threats from cybercriminals continue to loom over businesses and consumers [1]. Another Web security company, Cenzic, reported in 2014 that 96 % of the tested internet applications had vulnerabilities with a median of 14 per application, resulting in that hackers are increasingly focusing on and are succeeding with layer 7 (application layer) attacks [2]. These reports show that network security should not be ignored and effective security measures are much needed.

Among the important ways to solve security problems, intrusion detection is an effective and high-profile method. Intrusion detection was first introduced by Anderson

S. Guo et al. (Eds.): CollaborateCom 2015, LNICST 163, pp. 267–278, 2016.
DOI: 10.1007/978-3-319-28910-6_24

in [3]. Later, lots of researches have been carried out [4]. Generally, there are two main approaches to conduct intrusion detection: signature-based detection (misuse detection) and anomaly-based detection. The signature-based detection model has a good prior knowledge of known attacks, but seldom involves new types of attacks. Hence, in practice, it could miss a significant amount of real attacks [5]. By contrary, the anomaly detection creates a profile from normal behaviors and any violation will be reported as an intrusion. Theoretically, it is capable of detecting both known and unknown attacks. Under the current complicated network environment, the anomaly detection is much more required and has a better application foreground. In this paper, we focus on the anomaly detection.

With the network improving at an unprecedented pace, the traditional intrusion detection approaches are faced with more and more challenges. So a lot of new techniques have been introduced to conduct intrusion detection [6], among which the Support Vector Machine (SVM) is one of the widely used techniques [7, 8]. Whereas in the actual intrusion detection scenarios, the conventional two-class SVM algorithms may face some minor problems. For example, in many cases, normal network records can be obtained easily, but intrusion records are not so. So it is difficult to construct the training dataset. Actually, the intrusion detection is not a straightforward binary classification problem. The attacks can be divided into many categories. Given this, we propose to adopt the one-class SVM, which uses the normal connection records as the training dataset and can recognize normal from various attacks, to create anomaly detection model for network intrusions. Besides, the scaling strategy is introduced to improve the detection accuracy.

The rest of this paper is organized as follows. In Sect. 2, we introduce some related work about the intrusion detection. In Sect. 3, we first present the framework of our one-class SVM based intrusion detection model, and then discuss the implementation details. Experimental results and performance comparison are described in Sect. 4. Finally, Sect. 5 concludes this paper.

2 Related Work

The research on intrusion detection began from Anderson's famous literature [3]. In [3], the author proposed a model established from statistics of users' normal behaviors, so as to find the "masquerader" that deviates from the established normal model, which laid the foundation of intrusion detection and revealed the basic idea of anomaly detection. Later researches on anomaly detection also employ various statistical methods including multivariate statistics [9], Bayesian analysis [10], principal component analysis [11], and frequency and simple significance tests [12]. The signature-based detection (also called misuse detection) was first introduced by Denning in [13]. The author proposed an intrusion detection model that can be regarded as a rule-based pattern matching system. Both the misuse detection and statistics based anomaly detection have some limitations, such as low intelligence and poor ability to adapt to various application scenarios. And when encountering with larger datasets, the detection results would become worse [14].

To solve the limitations of above models, a number of machine learning techniques have been used [15, 16], of which the most widely used techniques may be Artificial Neural Networks (ANNs) [17] and Support Vector Machines (SVMs) [18]. A common practice is to use ANN and SVM to construct the hybrid model to detect intrusions [19, 20]. In this paper, our work relates to SVM and ANN is used as a comparison.

Multi-class SVM is also an alternative in intrusion detection. In [21], the author applied multi-class SVM classifiers, using one-against-one method, for anomaly as well as misuse detection to identify attacks precisely by type. But like the two-class SVM, the multi-class SVM is also faced with the difficulties to construct the training dataset.

Some other studies concern combining cluster algorithms with SVM techniques. In [22, 23], a hierarchical clustering method was applied to preprocess the originally enormous dataset to provide a reduced dataset for the SVM training. Thus the intrusion detection system could greatly shorten the training time. In this paper, we are more concerned about how to improve the detection accuracy, and seldom care about the learning speed. But the clustering method to reduce the dataset can also be used in our model.

Based on the related work, we propose an anomaly detection model for network intrusions by using one-class SVM and scaling strategy. One-class SVM can overcome the difficulties that the common two-class SVM and multi-class SVM encounter. Scaling strategy can greatly improve the detection accuracy.

3 One-Class SVM Based Anomaly Detection

In this section, we expound our one-class SVM based intrusion detection model. We first present the framework of the model, and then discuss how each constituent module works.

3.1 Framework of One-Class SVM Model

Our one-class SVM based intrusion detection model consists of the following three modules, as illustrated in Fig. 1

Module I: Feature extracting module. Feature extracting is the necessary step to make the detection module work correctly. Our intrusion detection model integrates a feature extracting module mainly to extract useful features from the raw data and then generates manageable formatted data for the detection module.

Module II: Scaling module. As an enhancing module, the scaling module normalized the data before inputting them to the detection module. In many circumstances, scaling the feature values to a small range can help to get better detection results and avoid numerical difficulties during the calculation.

Module III: One-class SVM module. Working as the detection module, one-class SVM involves two processes. The training process accepts the normalized training data and then generates a decision model. The testing process takes both the decision model and the normalized testing data as inputs, and then produces the detection results.

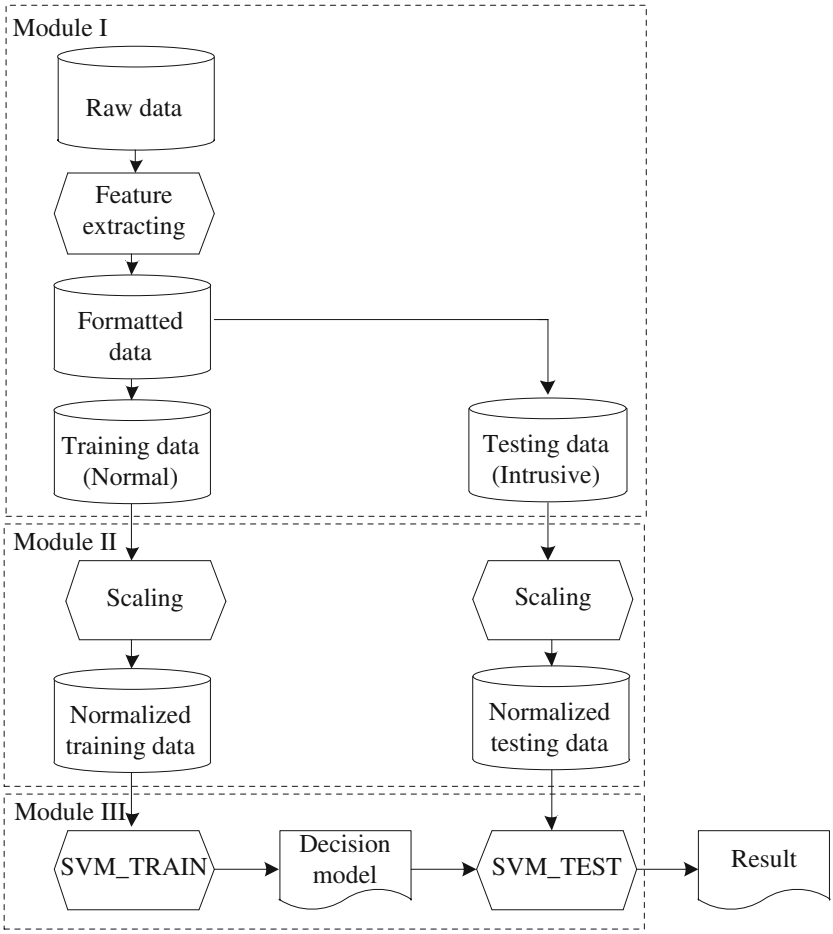

Fig. 1. Framework of one-class SVM based model

3.2 Feature Extracting Module

Almost no intrusion detection model can distinguish between intrusive connections and normal connections directly from original packets. They must be inputted with formatted data. Feature extracting is to obtain useful information from raw data and then format it, so that it can be interpreted by the detection module. There is no permanent standard to extract features. It may be better to extract features based on the actual network environment to find whether some attacks are hidden in connections. Extracting proper features helps the detection module to make more accurate predictions. In terms of network intrusions, some frequently-used features need paying attention to, such as the length (number of the seconds) of the connection, the type of the protocol, e.g. tcp, udp, etc., the number of data bytes transferred, the number of "root" accesses and so forth. In our one-class SVM based detection model, the feature extracting module takes the raw data as inputs, and then extracts expected features to

form the formatted data. Moreover, the feature extracting module is charged with dividing the formatted data into two divisions, the training data and the testing data. This process is fairly simple. The normal records comprise the training data and the rest (intrusive) records comprise the testing data. This relates to the detection mechanism of one-class SVM (detailed later).

3.3 Scaling Module

Scaling a value means to add or subtract a constant and then multiply or divide by a constant, to make the value lie in an expected range. So it is also called "normalizing". Scaling before applying one-class SVM is very important. The main advantage of scaling is to avoid features in greater numeric ranges dominating those in smaller numeric ranges. In intrusion detection models, we extract features from many aspects. These features may have great difference in numerical values. For example, the feature, length of the connection, may have a range of 0 to 10, while another feature, number of data bytes transferred, can have a range of 0 to 65535. Then the contribution of the first feature to the detection result will be swamped by the second. So it is crucial to scale the feature values so that their variability reflects their importance. Another advantage is to avoid numerical difficulties during the calculation. Because kernel functions used in one-class SVM usually need complex calculations of feature vectors, and large feature values might cause numerical problems. It is recommended to scale the feature values to a small range. In our one-class SVM based detection model, the scaling module scales both training and testing data to the same range [0, 1]. We use the following min-max normalization method.

$$x^* = \frac{x - min}{max - min}, \tag{1}$$

where x is the initial feature value, x^* is the new scaled value, *min* denotes the minimum value of the same features and *max* denotes the maximum value.

3.4 One-Class SVM Module

Here, we adopt the one-class SVM proposed by Scholkopf [24]. First, consider the training dataset:

$$D = \{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathbb{R}^n, y_i = +1\}_{i=1}^{l}, \tag{2}$$

where $\boldsymbol{x}_i$ is the feature vector with dimension n, $y_i = +1$ means all the training patterns are normal observations, and l is the number of training patterns.

The algorithm basically separates all the training data points from the origin. Suppose the hyperplane has the form:

$$w \cdot \phi(\boldsymbol{x}) - \rho = 0, \tag{3}$$

then the distance from the hyperplane to the origin is $\frac{\rho}{||w||}$. Maximizing the distance results to solving the following quadratic programming problem:

$$\min_{w,\boldsymbol{\xi},\rho} \frac{1}{2}\|w\|^2 + \frac{1}{vl}\sum_{i=1}^{l}\xi_i - \rho, \quad \text{s.t. } w \cdot \phi(\boldsymbol{x}_i) \geq \rho - \xi_i, \xi_i \geq 0. \tag{4}$$

Here, $\phi(\boldsymbol{x}_i)$ is the feature mapping function that maps $\boldsymbol{x}_i$ from its input space to a feature space, ξ_i is the slack variable for outlier $\boldsymbol{x}_i$ that allows it to lie on the other side of the decision boundary (hyperplane), and $v \in (0, 1]$ is the regularization parameter that is an upper bound on the fraction of outliers and a lower bound on the fraction of support vectors.

For convenience, we can introduce the kernel function, which is defined as follows:

$$k(\boldsymbol{x}_i, \boldsymbol{x}) = \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}). \tag{5}$$

For the one-class SVM used in our detection model, we use the Gaussian kernel function:

$$k(\boldsymbol{x}_i, \boldsymbol{x}) = e^{-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}\|^2}. \tag{6}$$

After introducing the kernel function, we can get the following dual form of the primal quadratic programming problem:

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \quad \text{s.t. } 0 \leq \alpha_i \leq \frac{1}{vl}, \sum_{i=1}^{l}\alpha_i = 1. \tag{7}$$

The answer to the dual problem (Eq. 7) is also the answer to the primal quadratic programming problem (Eq. 4). Furthermore, solving the dual problem is much easier and more feasible. We use the SMO (Sequential Minimal Optimization) algorithm [25] to solve the dual problem. Once solving the problem, we can get the following decision function:

$$f(\boldsymbol{x}) = \text{sgn}\left(\sum_{i=1}^{l}\alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) - \rho\right). \tag{8}$$

That is, if $w \cdot \phi(\boldsymbol{x}) - \rho \geq 0$, $\boldsymbol{x}$ is regarded as a normal event, otherwise, it is declared as intrusive.

4 Experiments and Discussions

To evaluate the performance of our one-class SVM based intrusion detection model, we conducted a series of experiments on KDDCUP99 [26] dataset.

4.1 Data Preparation

In 1998, DARPA Intrusion Detection Evaluation Program was prepared and managed by MIT Lincoln Labs. A standard dataset [27] was provided. The KDDCUP99 dataset used in our experiments is a version of this dataset.

The raw training data contains about five million TCP connection records from seven weeks of network traffic. Similarly, the two weeks of testing data yields around three million records. Each connection record has 41 derived features that help in distinguishing normal connections from attacks, and is labeled as either normal, or as an attack, with exactly one specific attack type. Attacks fall into four main categories: DOS, Probe, R2L and U2R.

In experiments, we used stratified random sampling to reduce the size of dataset. For one-class SVM used in our intrusion detection model, the training data must contain only normal patterns and does not contain any attacks. So we selected a random sample of normal records in the raw training data. The sampling proportion is about 3 %. To test the model's ability to detect different kinds of attacks, we randomly selected different types of records in the raw testing data. The sampling proportion is about 1 %. Some types of attacks such as R2L and U2R were totally selected due to their low proportion in KDDCUP99 dataset. Finally, 32426 normal connection records in the raw training data and 31415 connection records in the raw testing data were randomly selected. Table 1 shows the details about different categories of records. "Other" indicates the new types of attacks not present in the four main categories.

Table 1. Number and distribution of training and testing data

Category	Training dataset		Testing dataset	
Normal	32426	100 %	6060	19.29 %
DOS	/	/	22429	71.40 %
Probe	/	/	315	1.00 %
R2L	/	/	622	1.98 %
U2R	/	/	39	0.12 %
Other	/	/	1950	6.21 %
Total	32426	100 %	31415	100 %

4.2 Evaluation Criteria

In order to evaluate the performance of IDS, some accepted measurements are proposed. We use *TP*, *FN*, *TN* and *FP* to represent the number of true positives, false negatives, true negatives and false positives, respectively. Usually, we use the detection rate to evaluate the IDS' ability to detect real attacks. For some category of attacks, the detection rate is the fraction of detected attacks accounting for the total ones. In addition to the detection rate, another three criteria are also widely used for performance evaluation, especially for performance comparison. They are precision, recall

and F-value. Precision is the fraction of true positives in total determined positives (i.e. the sum of true positives and false positives). Recall has the same formula as the detection rate. F-value considers both the precision and the recall to compute the evaluation value. The precision, recall and F-value are defined as follows.

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

$$F - value = \frac{2 * Recall * Precision}{Recall + Precision} \tag{11}$$

4.3 Results and Discussions

In this section, we compare our one-class SVM based model with other two well-knowns, probabilistic neural network (PNN) [28] and C-SVM (proposed by Cortes and Vapnik in [29]), given that they both adopt the radial basis function (Gaussian function or Gaussian kernel) as the one-class SVM does and are often used to detect intrusions due to their good classification performance. PNN used in our experiments is taken from the MATLAB R2013b toolbox and C-SVM from the software LIBSVM [30]. Because the training data used by PNN and C-SVM must contain both normal and abnormal records, we conducted a stratified random sampling for the raw training data in KDDCUP99 with the proportion around 1 %. The final training data contains 49567 records, including 9728 Normals (19.63 %), 39167 DOSs (79.02 %), 412 Probes (0.83 %), 208 R2Ls (0.42 %), and 52 U2Rs (0.10 %). The three models use the same testing data as described in Sect. 4.1.

In experiments, the parameter γ (gamma) in radial basis function was set to 0.5, the cost parameter c in C-SVM was set to 1 and the parameter v (nu) in one-class SVM was set to 0.05. First, we compare and discuss the detection rates of these three models for different categories of attacks. The results are shown in Table 2 and Fig. 2, and are produced in this way—first, any attack that can be detected by one-class SVM is declared abnormal without any distinction, then we compute the detection rate for different category of attacks according to the labels in the testing data. We can see that for DOS attacks, the three models get perfect results (all above 99 %). For Probe attacks, one-class SVM can reach the top detection rate 100 %, while the detection rates of PNN and C-SVM are relatively lower, respectively 98.73 % and 86.98 %. We should note that for R2L, U2R and "Other" categories of attacks, the results of all the three models are not very satisfactory. We believe one of the main reasons is that the number of attacks in these three categories is relatively small (see Table 1), so the test results have some limitations. Another reason may be that the attacks are too covert to be detected by the models. But even so, the detection results of one-class SVM are

Table 2. Detection rates of different models

	PNN	C-SVM	One-class SVM
DOS	0.9979	0.9958	0.9950
Probe	0.9873	0.8698	1.0000
R2L	0.0804	0.0322	0.2685
U2R	0.1284	0.4872	0.6923
Other	0.0687	0.1421	0.2067

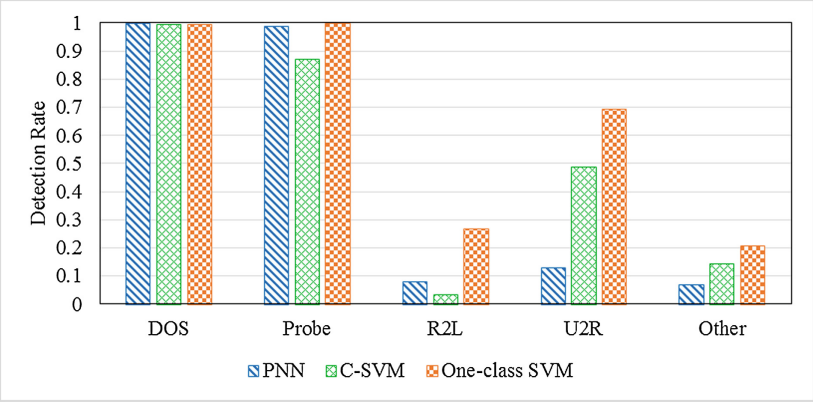

Fig. 2. Detection rate comparison of different models

considerably better than two others'. Furthermore, for PNN and C-SVM, the "Other" category of attacks are new attacks not present in their training data, so it is especially difficult for them to detect such attacks. But for one-class SVM, the new attacks receive the same treatment as with other categories of attacks, without any difference.

Next, we use three other criteria, precision, recall and F-value to conduct performance comparison. The results are shown in Table 3 and Fig. 3. As illustrated by Fig. 3, one-class SVM produces a slightly lower precision than PNN and C-SVM. But the precisions of all the three models are very high (above 99 %). Apparently, the recall and F-value of one-class SVM are higher than others'.

Table 3. Precision, recall and f-value of different models

	PNN	C-SVM	One-class SVM
Precision	0.9988	0.9957	0.9903
Recall	0.8916	0.9041	0.9161
F-value	0.9422	0.9477	0.9518

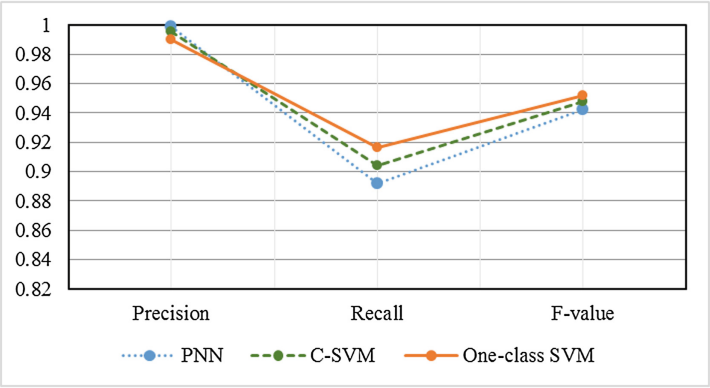

Fig. 3. Performance comparison of different models

5 Conclusion

We propose a novel anomaly detection model for network intrusions by using one-class SVM and scaling strategy. One-class SVM is a one-versus-rest classifier, which is very suitable for anomaly detection. Although the commonly used two-class SVM algorithms have been applied in intrusion detection, they are facing the difficulties of constructing the training dataset. That is because in many true application scenarios, it is easy to obtain normal connection records, but difficult to obtain attack records, or the number of attack records is very limited. Whereas to a great extent, the distribution of training records affects the detection results of the two-class SVM. Hence, we propose to use one-class SVM, which adopts only the normal network connection records as the training data, to conduct the anomaly detection. The scaling strategy scales the feature values to a small range so that their variability reflects their importance, thus greatly improving the detection accuracy and avoiding numerical difficulties during the calculation. The experimental results on KDDCUP99 dataset show that compared to PNN and C-SVM, our one-class SVM achieves higher detection rates for different categories of attacks and has an average better performance in terms of precision, recall and F-value. The deficiency lies in that both our one-class SVM based and other two models show relatively low detection rates for low-frequent attacks, such as R2L and U2R. Affecting the accuracy of results, the insufficient number of data is partially to blame. But the detection model could also be enhanced. We leave this as future work.

Acknowledgement. The work of this paper is supported by the National Natural Science Foundation of China Project under grant No. 61271252.

References

1. Symantec Enterprise.: Internet Security Threat Report 2014. http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v19_21291018.en-us.pdf. accessed 15 April 2015

2. Cenzic.: Application Vulnerability Trends Report 2014. http://www.cenzic.com/downloads/Cenzic_Vulnerability_Report_2014.pdf. accessed 15 April 2015
3. Anderson, J.P.: Computer security threat monitoring and surveillance. vol. 17. Technical report, James P. Anderson Company, Fort Washington, Pennsylvania (1980)
4. Axelsson, S.: Intrusion detection systems: A survey and taxonomy. vol. 99. Technical report, 2000
5. Kruegel, C., Tóth, T.: Using decision trees to improve signature-based intrusion detection. In: Vigna, G., Kruegel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 173–191. Springer, Heidelberg (2003)
6. Patcha, A., Park, J.-M.: An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput. Netw. **51**(12), 3448–3470 (2007)
7. Li, Y., Li, W., Wu, G.: An intrusion detection approach using SVM and multiple kernel method. Int. J Adv. Comput. Technol. IJACT **4**(1), 463–469 (2012)
8. Li, Y., et al.: An efficient intrusion detection system based on support vector machines and gradually feature removal method. Expert Syst. Appl. **39**(1), 424–430 (2012)
9. Taylor, C., Alves-Foss, J.: Low cost network intrusion detection (2000)
10. Barbara, D., Wu, N., Jajodia, S.: Detecting novel network intrusions using Bayes estimators. In: SDM (2001)
11. Shyu, M.-L., et al.: A novel anomaly detection scheme based on principal component classifier. Miami Univ Coral Gables FL Dept of Electrical and Computer Engineering (2003)
12. Qin, M., Hwang, K.: Frequent episode rules for intrusive anomaly detection with internet datamining. In: USENIX Security Symposium (2004)
13. Denning, D.E.: An intrusion-detection model. IEEE Trans. Softw. Eng. **2**, 222–232 (1987)
14. Wang, G., et al.: A new approach to intrusion detection using artificial neural networks and fuzzy clustering. Expert Syst. Appl. **37**(9), 6225–6232 (2010)
15. Sinclair, C., Pierce, L., Matzner, S.: An application of machine learning to network intrusion detection. In: 15th Annual Computer Security Applications Conference (ACSAC 1999) Proceedings. IEEE (1999)
16. Tsai, C.-F., et al.: Intrusion detection by machine learning: a review. Expert Syst. Appl. **36** (10), 11994–12000 (2009)
17. Ryan, J., Lin, M.-J., Miikkulainen, R.: Intrusion detection with neural networks. In: Advances in neural information processing systems 943–949 (1998)
18. Kim, D.S., Park, J.S.: Network-based intrusion detection with support vector machines. In: Kahng, H.-K. (ed.) ICOIN 2003. LNCS, vol. 2662, pp. 747–756. Springer, Heidelberg (2003)
19. Sung, A.H., Mukkamala, S.: Identifying important features for intrusion detection using support vector machines and neural networks. In: 2003 Symposium on Applications and the Internet, Proceedings, pp. 209–216. IEEE (2003)
20. Mukkamala, S., Janoski, G., Sung, A.: Intrusion detection using neural networks and support vector machines. In: Proceedings of the 2002 International Joint Conference on Neural Networks, IJCNN 2002. vol. 2. IEEE (2002)
21. Ambwani, T.: Multi class support vector machine implementation to intrusion detection. In: Proceedings of the International Joint Conference on Neural Networks, vol. 3. IEEE (2003)
22. Khan, L., Awad, M., Thuraisingham, B.: A new intrusion detection system using support vector machines and hierarchical clustering. Int. J. Very Large Data Bases **16**(4), 507–521 (2007)
23. Horng, S.-J., et al.: A novel intrusion detection system based on hierarchical clustering and support vector machines. Expert Syst. Appl. **38**(1), 306–313 (2011)
24. Schölkopf, B., et al.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)

25. Platt, J.: Sequential minimal optimization: a fast algorithm for training support vector machines (1998)
26. UCI KDD Archive.: KDDCUP99 dataset. http://kdd.ics.uci.edu/databases/kddcup99/. accessed 15 April 2015
27. MIT Lincoln Laboratory.: DARPA Intrusion Detection Data Sets. http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/index.html. accessed 15 April 2015
28. Specht, D.F.: Probabilistic neural networks. Neural Netw. **3**(1), 109–118 (1990)
29. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
30. Chang, C.-C., Lin, C.-J.: LIBSVM : a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**, 27:1–27:27 (2011). http://www.csie.ntu.edu.tw/∼cjlin/libsvm