I. J. Mathematical Sciences and Computing, **2020, 4, 1-11**
Published Online August 2020 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmsc.2020.04.01

A Flow-Based Technique to Detect Network Intrusions Using Support Vector Regression (SVR) over Some Distinguished Graph Features

Yaser Ghaderipour
Department of Computer Science, University of Tabriz, East Azerbaijan Province, Tabriz, Iran

Hamed Dinari
Department of Computer Engineering (CE), Iran University of Science and Technology (IUST), Tehran Province, Tehran, Iran
E-mail: dinari.hamed@yahoo.com

Received: 23 April 2020; Accepted: 13 May 2020; Published: 08 August 2020

Abstract: Today unauthorized access to sensitive information and cybercrimes is rising because of increasing access to the Internet. Improvement in software and hardware technologies have made it possible to detect some attacks and anomalies effectively. In recent years, many researchers have considered flow-based approaches through machine learning algorithms and techniques to reveal anomalies. But, they have some serious defects. By way of illustration, they require a tremendous amount of data across a network to train and model network's behaviors. This problem has been caused these methods to suffer from desirable performance in the learning phase. In this paper, a technique to disclose intrusions by Support Vector Regression (SVR) is suggested and assessed over a standard dataset. The main intension of this technique is pruning the remarkable portion of the dataset through mathematics concepts. Firstly, the input dataset is modeled as a Directed Graph (DG), then some well-known features are extracted in which these ones represent the nature of the dataset. Afterward, they are utilized to feed our model in the learning phase. The results indicate the satisfactory performance of the proposed technique in the learning phase and accuracy over the other ones.

Index Terms: Cyber Attack, Intrusion Detection System (IDS), Network Security, Machine Learning, Support Vector Regression (SVR)

1. Introduction

These days, due to the ever-increasing use of digital devices connected to the Internet most organizations and corporations have been encountered the danger of illegitimate access to susceptible information. These issues with regard to security and privacy have been caused hundreds of billions of dollars cost [*1*]. The most common cyber-attacks are usually done by performing malware or malicious software. They are defined as "a software program that runs on a system and allows an attacker to carry out some unwanted actions. To tackle these chief problems, some researches have proposed a concept named the Intrusion Detection System (IDS).

The idea of intrusion detection began in the mid-1980s by Denning and became the foundation for the Intrusion Detection Expert System (IDES) [*2*]. When this idea was introduced, it was just utilized in critical military and commercial systems because of their high-processing load. Today, with the significant advancement in computer and electronic technologies they are applied in a wide spectrum of computer systems and network environments. This notion according to NIST's definition is specified as, "a software program that monitors and analyzes events over a network or on a single system to disclose possible intrusion" [*3*]. More specifically, three main targets of an intrusion detection system are as follows:

- Monitoring and Evaluating
- Discovery
- Reaction

Figure 1 illustrates the general architecture of an intrusion detection system. Initially, data is collected from diverse data sources such as the network's traffic, system logs, and etc. Afterward, they are passed toward the processing engine. Ultimately, they are analyzed to detect anomalies and generates proper responses [4].

Some intrusion detection systems are designed and developed in such a way that inspect the payload of each network's packet [5,6]. Due to the increase in data transmission rate in computer networks, the process of network's packet inspection is a tedious and time-consuming operation and in most cases impossible. To deal with these problems some researchers have allotted their time to investigate the flow-based IDS. Because flow-based IDS just analyze communication patterns instead of packets' payload inspection [7]. Internet Protocol Flow Information Export (IPFIX)'s group recognizes a flow as "a set of IP packets that pass across a special point over a network during a certain time interval. All packets belonging to a particular flow share a set of properties which are recognized as flow keys that including IP Protocol, source and destination IP addresses, source and destination port number" [8,9].

In recent years, scientists have employed graph-based approaches to managing the obstacles concerning the network's intrusion detection. In these approaches, each vertex of the graph represents an entity in the network like client and server and each edge exposes communication between two entities [10]. Furthermore, many researchers have used machine learning algorithms to discern anomalies in high-speed networks [11,12]. However, they have some major drawbacks. For instance, they need an immense amount of the network's traffic data to train and stimulate the network's behaviors. This problem has resulted in an unpleasing performance in the learning phase.

Intrusion detection methods are classified into three broad categories: Signature-based IDS, Anomaly-based IDS, and Stateful Protocol Analysis (SPA). In signature-based IDS, the patterns of captured events are compared to the pattern of well-known attacks or vulnerabilities. This type of IDS is also specified as knowledge-based detection or misuse detection. An anomaly-based IDS builds a model for normal behavior and compares it to the observed events in order to recognize significant attacks. This IDS type is also called behavior-based detection. SPA is similar to anomaly-based methods but SPA depends on vendor-developed generic profiles [13].

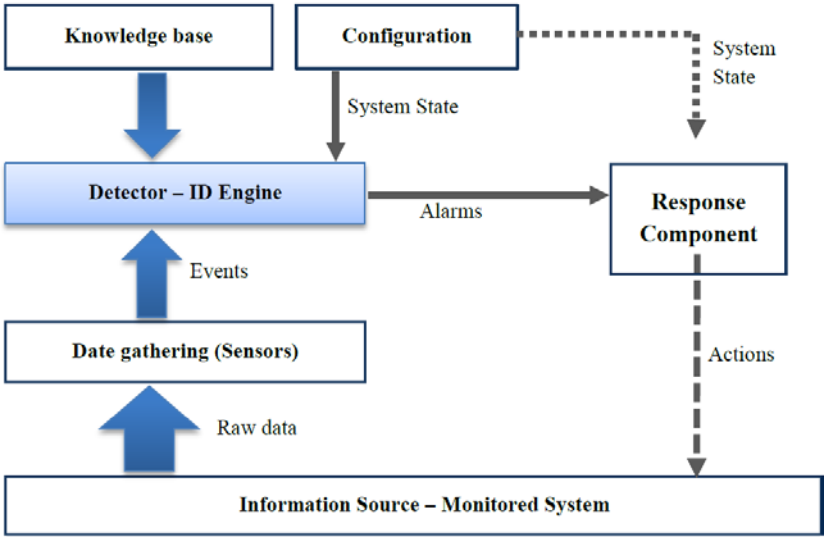


Fig 1. An architectural view of an Intrusion Detection System (IDS) [*4*]

The intrusion detection system provides useful information about the attacks. Some purposes of them include:

- Monitoring and analyzing the users' activities on computers and networks
- Checking the network specifications and its vulnerabilities
- Recognizing patterns with regard to attacks
- Specifying patterns of abnormal behaviors

To get more knowledge about the structure of intrusion detection systems in detail, you can see [*4*]. Since to train the model of the machine learning-based method, the immense volume of the network's traffic slowed down this stage and previous methods could not be used in the operational environment, the main purpose of this research is to find a solution to decrease the input data in the learning phase so that the quality of the input data be preserved and the detection process be done with high accuracy. The remainder of the paper is organized as follows: at first, some basic terminologies are introduced. An overview of related works is described concisely in section two. Section three represents our proposed technique point by point. Section four indicates the experimental results comprehensively. Finally, conclusion and future works would be illustrated in section five.

Basic Concepts:

In order to assist the reader to grasp this paper in-depth, we tend to allude several notions and terminologies:

Computer Network: A computer network often referred to as a network, is a group of computers and devices connected together that can communicate through either wired or wireless media. The computer network facilitates communication between users and allows them to share resources.

Intrusion: A series of illegal actions that endanger the integrity, confidentiality, and accessibility to a resource.

Adversary: Any internal or external intruder agent that its aim is to damage the system or perform unauthorized operations.
Intrusion Detection System (IDS): A system that its primary goal is to identify any unauthorized access that may damage the system and usually performed by both internal and external users. Intrusion detection and prevention systems play a notable role to guarantee the security of network and computer systems. Plus, they are used as a key component alongside the firewalls.

2. Previous Studies

In recent years there has been growing interest in intrusion detection. Many researchers have conducted immense papers in this area. In the following, we would denote some of them briefly. Li and et al. [14] proposed an anomaly-based IDS method to divulge Denial Of Services (DOS) attacks in high-speed networks using merely flow information. Authors in [15] presented an algorithm to disclose DOS and Distributed Denial of Service (DDOS) attacks. They defined an adaptive threshold to extract the flow's features from the network in various conditions, then through ones predict the anomalies. Hellemons and et al. [16] developed a flow-based intrusion detection system to expose SSH-dictionary attacks in a real-time manner. Their algorithm specified suspect traffic via two metrics: packet-per-flow and a minimum number of flow records. Winter and et al. [11] Sheikhan and et al. in [12] used machine learning algorithms to uncover anomalies in high-speed networks. To decrease CPU usage in the learning phase, they chose a small subset of a flow-based dataset released by Sperotto and et al. [17].

In 1996 Staniford-Chen and et al. [18] introduced a graph-based IDS (GrIDS) to disclose some attacks like worm-propagation but their work had not satisfactory security [19]. Ellis and et al. [20] presented a secure worm detection approach, although it was not appropriate to implement on the hardware. Illiofotou and et al. [21] suggested an intrusion detection approach that could be implemented in hardware. Zhou and et al. [22] recommended a time-series graph mining method to determine DDOS attacks. Sun and et al. [23] proposed a Compact Matrix Decomposition (CMD) to decompose the adjacency matrix and used reconstruction sum-square-error to find anomalies over the network's traffic which modeled as a graph. Authors in [24] presented a graph clustering algorithm to discern anomalies, albeit it was memory-consuming and had poor performance.

Kelton and et al. [25] proposed a nature-based approach to reveal abnormal or malicious behaviors. They used meta-heuristic optimization to improve their algorithm. Ma, Jiefei, and his co-authors proposed a distributed algorithm to uncover distributed signature-based intrusion. They used some common features between multiple paths to reach their ends. [26]. In [27] authors offered a method to uncover anomalies based on genetic algorithm. In [28] authors combined IPFIX-based flow monitoring with Network-Based Intrusion Detection (NIDS). Their method exploits the HTTP related flow Information Elements (IEs) to divulge intrusion in high-speed networks. Jelidi, Mohamed And et al. developed an algorithm to recognize intrusion in cloud environments. They utilized two detection methods: Signature-based and Anomaly-based methods [29].

3. Proposed Technique

In this section, our technique is stated in two fundamental steps. Firstly, some features are extracted from the network's traffic data. After that, Support Vector Regression (SVR) algorithm is trained through the features' values that elicited in the previous step to construct a model. Next, this model is applied to unveil anomalies. Figure 2 shows the general architecture of the suggested technique. In the following, it would be expressed in great detail.

To model our system, we refer to our mathematical knowledge. We are aware that if the network in different time intervals visualizes as a Directed Graph (DG) so that the vertices represent the network's components (computer systems and smartphones, etc.), the edges show the directed links between them, and weight of each edge exhibits the number of packets that exchanged among two nodes, hence it is straightforward to analyze the network's behavior. First of all, because the real dataset we chose to evaluate our technique contains approximately seven billion records, we attempted to summarize it by exploring some most noteworthy features, otherwise, we faced an exhausting action to train our model. After this preprocessing, we solely had 105 records to work on. As indicates in Figure 4, the flow-based network's traffic is modeled as a Traffic Dispersion Graph (TDG) in several specific time intervals (here, each time interval is supposed as 10 minutes.), then some graph's features are withdrawn. Applying graph to model network's traffic and deriving its principal features lead to a sharp reduction in the number of instances for the SVR training model, consequently, performance in the learning phase improves. These features were determined based on characteristics of DDOS, Scan, and SSH-dictionary attacks, in such a way the highest value for the first two options and the lowest amount for others ones can be signs of an anomaly. Each of the features is suitable to reveal a certain type of anomalies. These features have been listed as follows:

1) The number of edges. 2) Total weight of edges. 3) The average weight of edges. 4) Percentage of edges with minimum weight. 5) Percentage of edges that their weight less than the average minus standard deviation 6) Number of nodes with a degree more than average plus standard deviation. 7) Maximum degree of the graph. 8) Number of nodes with non-zero degree. 9) The multiplication of maximum degree by the proportion of maximum degree per total input

weight.

As previously mentioned, in this technique the SVR algorithm has been employed to construct the detection model. In the following section, it would be narrated briefly.

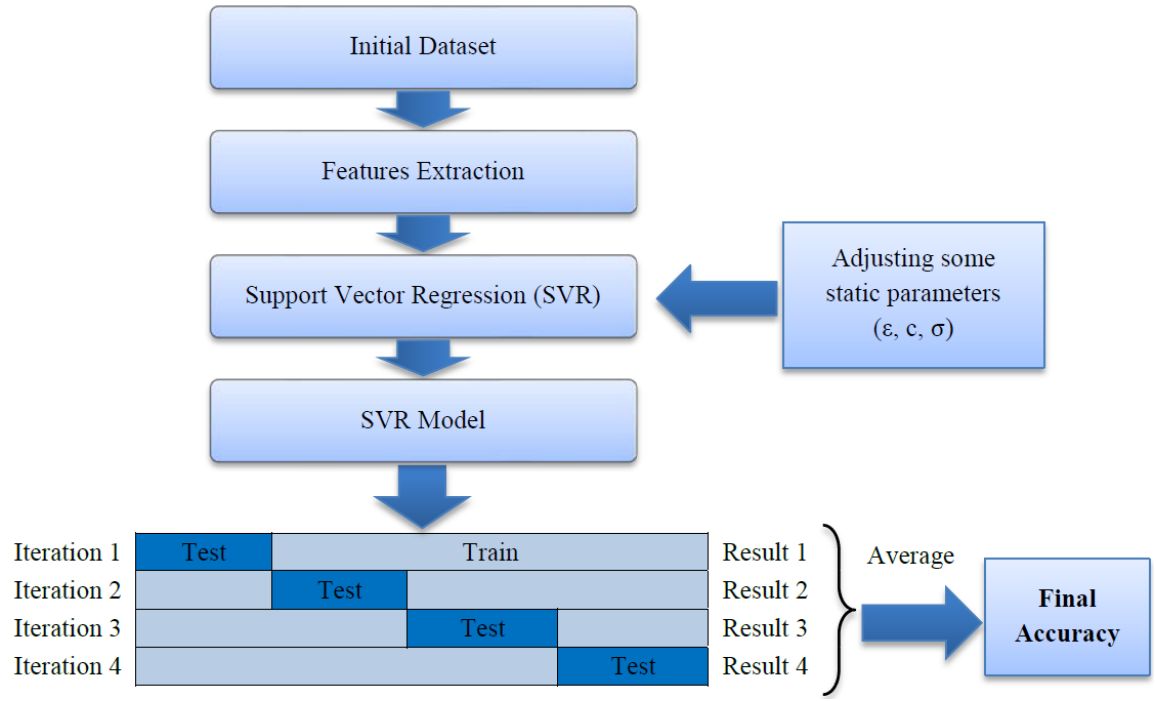


Fig 2. A general structure of our intrusion detection technique

Support Vector Regression (SVR)

This is one of the most powerful machine learning algorithms. It is proper for such problems in which there are complicated relationships among their features and a few numbers of instances input. It was developed at AT&T Bell Laboratories by Cortes and Vapnik [*30*]. Support Vector Machine (SVM) also can be used in both nonlinear regression (SVR) and classification (SVC) problems. It is based on the structural risk minimization principle of the statistical learning theory [*31*]. Below the standard SVR is introduced to solve the approximation problems:

$$f(x) = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) k(x_i, x) + b \tag{1}$$

Where α_i^* and α_i are lagrange multipliers. The kernel function $k(x_i, x)$ has been defined as a linear dot product of the nonlinear mapping, i.e.

$$k(x_i, x) = \varphi(x_i)\varphi(x) \tag{2}$$

The coefficients α_i^* and α_i in (1) have been obtained by minimizing the following regularized risk functional:

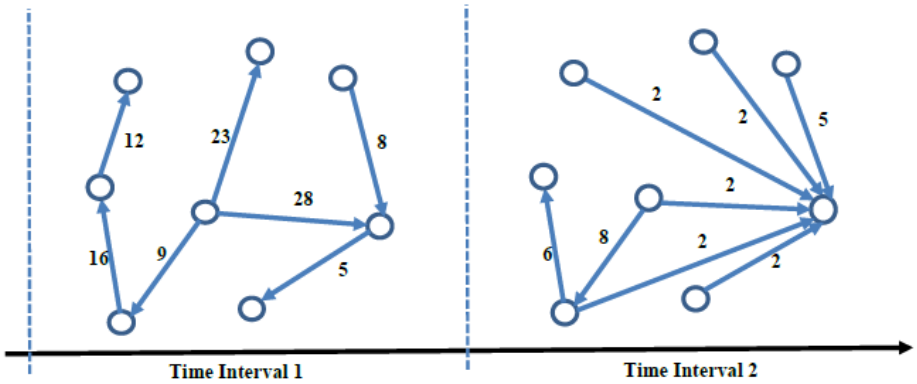
$$R_{reg}[f] = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{l} L_\varepsilon(y) \tag{3}$$

The term $\|\omega\|^2$ has been characterized as the model complexity, *C* as a constant which determines the trade-off and the ε-insensitive loss function $L_\varepsilon(y)$ is computed as follows:

$$L_\varepsilon(y) = \begin{cases} 0, & for\ |f(x) - y| < \varepsilon \\ |f(x) - y| - \varepsilon & otherwise \end{cases} \tag{4}$$

After the training, an evaluation is performed through the k-fold cross-validation algorithm. This algorithm is repeated k times, in which each of the k subsamples used exactly once as the testing data. Then the k results are averaged to obtain the final accuracy.

Parameters tuning is an important section in the SVR algorithm. It has a direct impact on the accuracy of the model. To do so, the evolutionary algorithm is applied. The major objective of this algorithm is to find the optimal values for three static parameters of the SVR algorithm, namely ε, c, and σ. Table 1 pictures all characteristics of this evolutionary algorithm. Value encoding is the best choice for floating-point numbers. For more information regarding evolutionary algorithms refer to [32].



Graph Features	Time Interval 1	Time Interval 2
Total weight of edges	101	29
The average weight of edges	14.4	3.6
Number of edges	7	8
Percentage of edges with minimum weight	14.2	62.5
Percentage of edges with a weight less than an average minus standard deviation	14.2	0
Number of nodes with a degree more than average plus standard deviation	1	1
Maximum degree of graph	2	6
Number of nodes with non-zero degree	6	3
The multiplication of maximum degree and proportion of maximum degree per total input weight	0.1	2.4

Fig 3. Graph Visualization and Feature Extraction

Table 1. Designing an evolutionary algorithm for SVR static parameters tuning

1	Encoding	Value Encoding (ε,c,σ)
2	Generating Initial Population	Generating Random values in a uniform distribution
3	Fitness function	SVR Accuracy
4	Selection	*1)* Roulette Wheel Selection
5	Crossover	Uniform Crossover
6	Mutation	Adding or subtracting a small number to the selected value
7	Replacement	Generalization + Elitism
8	Stop Condition	A certain number of iteration or to reach a wanted accuracy

4 Experimental Results

This section describes the dataset which we used to evaluate our technique as well as the results of experiments. The dataset used in this research is a labeled flow-based dataset released by Sperotto et al. [*17*]. It was prepared by monitoring a honeypot in the network of the University of Twente. The records of flow are labeled as malicious or benign flows. The entire dataset classified into some sections as follows:

Malicious traffic: this section involves all flow records in which the honeypot is the victim of attacks or it is used to perform attacks against others.

Side-effect traffic: this part of traffic was streamed when attacks were carried out, however, it cannot be considered as malicious traffic.

Unknown traffic: this class includes a small subset of dataset (5968 flows) in which the malicious or benign of flows cannot be specified.

The two first sections of this dataset are used to evaluate the proposed technique in the current paper in which the side-effect class is considered as benign traffic.

Due to the lack of a complete flow-based dataset that covers all types of attacks, in this study, it was not possible to test all attacks. Because the previous techniques employed a large amount of network's traffics in the learning phase and could not be practical in operational environments, our proposed technique alleviated the input data of the SVR algorithm to make the learning phase more efficient and suitable for the operational environments. To achieve this goal, Firstly, we visualized the dataset as a graph to withdraw the important features and use them in the learning phase as inputs. On the other hand, because other methods required a boring process to train their model as well as our technique was far superior in this step, so we neglected to compare our experimental results with others.

We used MATLAB to implement our work, then performed on a system with the following configuration: RAM: 8GB, CPU: Intel(R) Core(TM) i7-4710HQ@ 2.8GHz (8CPUs), Operating System (OS): Windows 8.1-64bit. Using Mathworks Matlab 2014a software. The dataset is divided into 104 subsets according to 10-minute time intervals. The malicious flows are ones in which they could have a sign of anomaly or intrusion. There are no malicious flows in the 8-time intervals, as illustrated in Figure 3. These features are presented in Figures 4 through 13. As mentioned before, each of these features is exploited to detect a specific type of attack. For instance, the 'max degree' in Figure 10 is used to uncap the DOS attacks. However, there is a situation in which the high value of this feature can be considered as the normal behavior of the network. Malicious flows can be detected by analyzing these features manually. Support Vector Regression (SVR) is used in the next step to disclose the anomalies. Table 2 shows the value of input parameters which has been estimated by the Evolutionary Algorithm (EA). The four iterations of the k-fold cross-validation algorithm (k=4) are presented in Figures 14 to 17. The final accuracy is calculated by averaging the outputs of the validation algorithm. It is equal to 98.2975%. The values of input parameters for SVR estimated from the evolutionary algorithm are as follow:

1) $\boldsymbol{\varepsilon}$***:*** 5.35463428729795E-4 2) ***c:*** 1.0570000697821605E+3 3) $\boldsymbol{\sigma}$***:*** 2.925292600260995E+2

Discussion

The current technique is proper for high-speed networks because it is a flow-based technique, and it does not need to inspect the packets' payloads. In this technique, the Support Vector Regression (SVR) was used to detect anomalies, plus the problem of waste of CPU's time in the training phase by graph visualization was resolved. If the parameters "d" and "n" are the number of features and the number of instances respectively, the time complexity of SVR is equal to ***O(max(n,d), min(n,d)2)***. Moreover, the time complexity of the SVR, just like other machine learning algorithms highly depends on the number of instances. In comparison to previous works in which the features have been specified on the individual flows, in current technique features are recognized over a graph, furthermore the number of the instances is extremely decreased. For example, the dataset used in current research includes 104 ten-minute time intervals and approximately 6000 flows in each time interval by average. Based on previous approaches, [*11*,*12*] that features are defined on flows, the total number of instances is equal to 4,368,000. Additionally, they had to use a small subset of the dataset. Whereas in our technique one graph is visualized for every time intervals and the number of instances would be 104. As mentioned earlier, the dataset includes DDOS, Scan, and SSH-dictionary attacks. this technique would be able to detect attacks by an accuracy equal to 98.2975%.

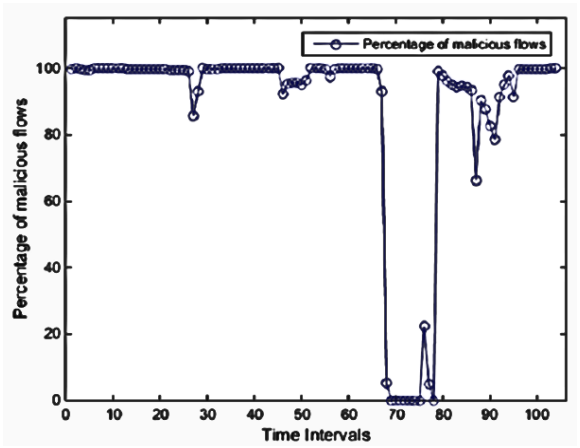


Fig 4. Percentage of malicious flows in each time interval

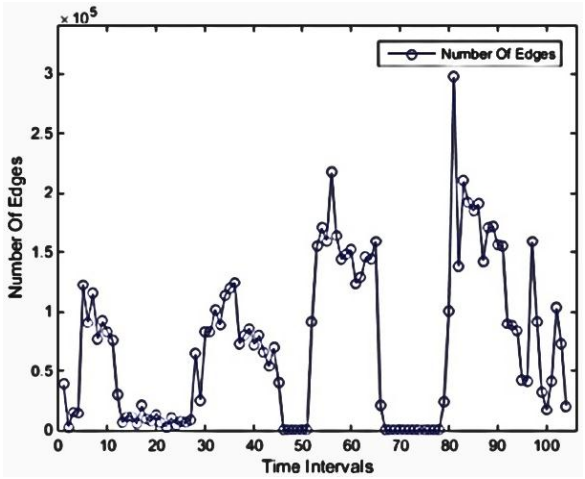


Fig 5. Number of edges

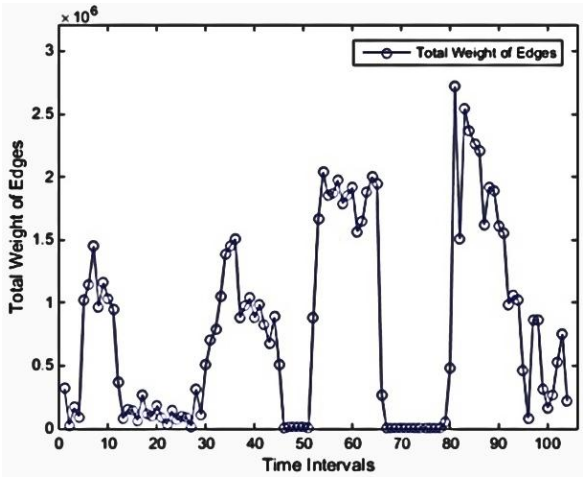


Fig 6. Total weight of edges

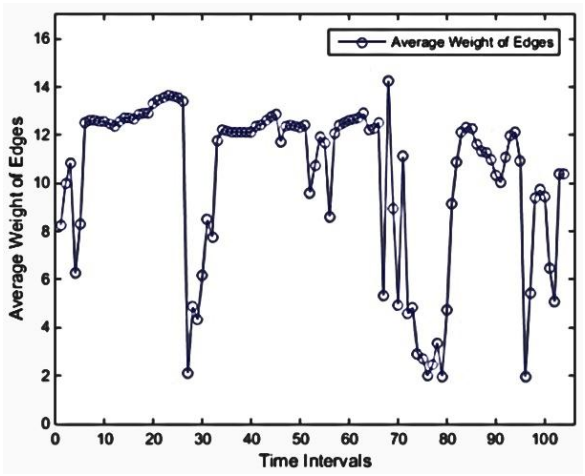


Fig 7. The average weight of edges

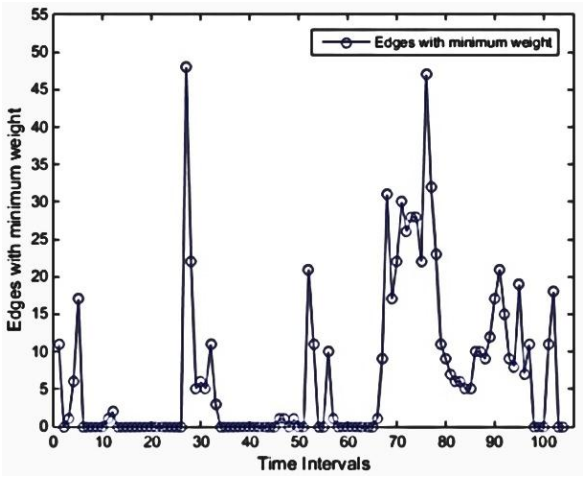


Fig 8. Edges with minimum weight

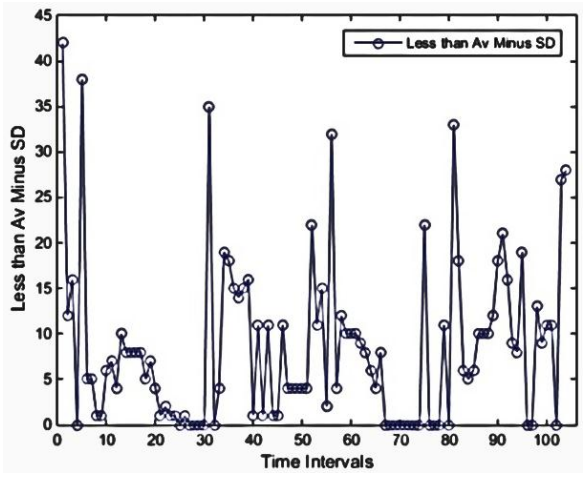


Fig 9. Edges with a weight less than an average minus standard deviation

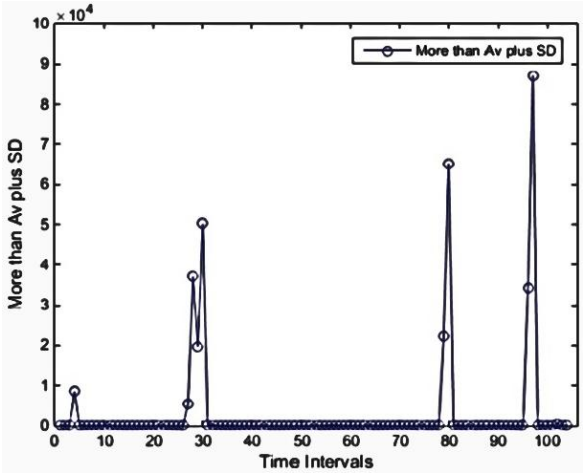


Fig 10. Nodes with a degree more than an average plus standard deviation

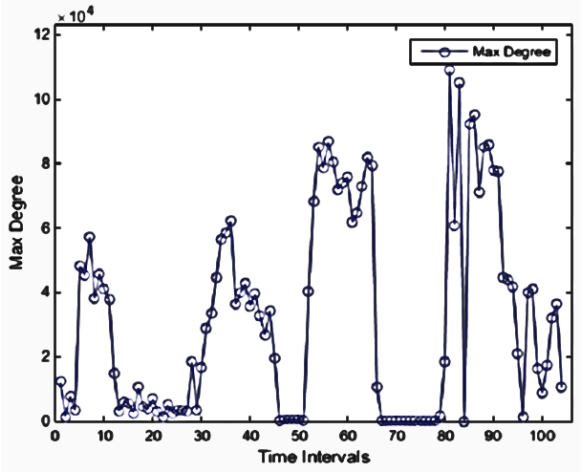


Fig 11. Max degree of the graph

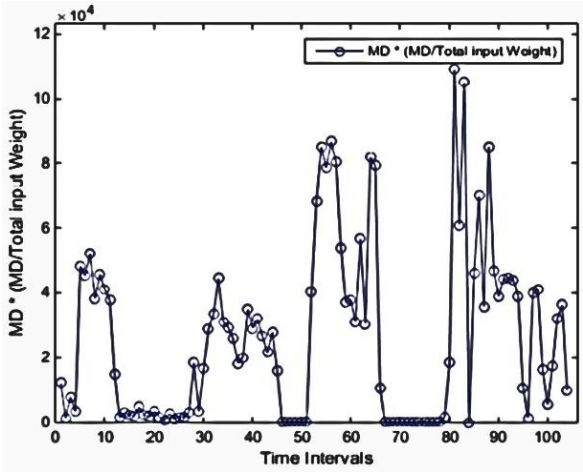


Fig 12. Multiplication of max degree and proportion of max degree per total input weight

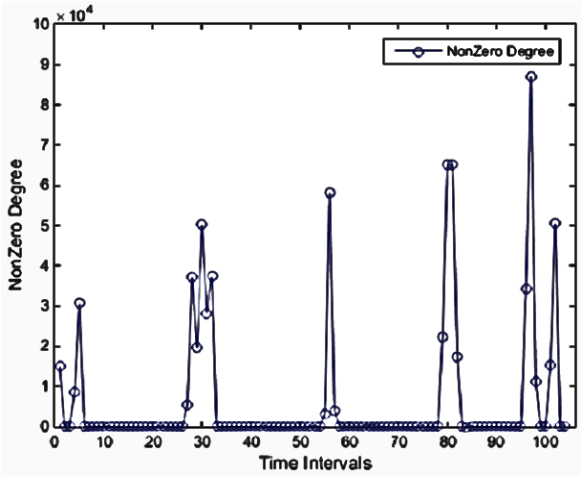


Fig 13. Number of edges with nonzero degree

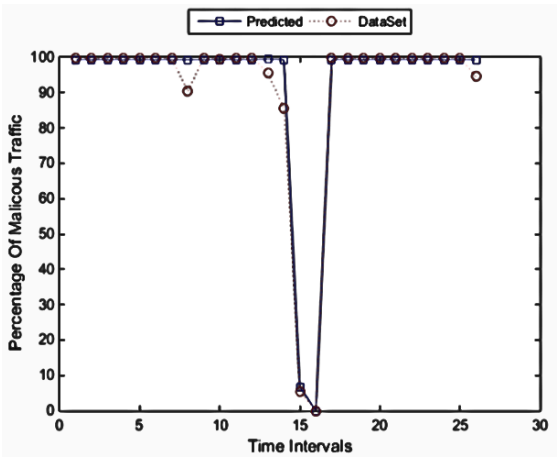


Fig 14. Iteration 1, Accuracy: 98.31%

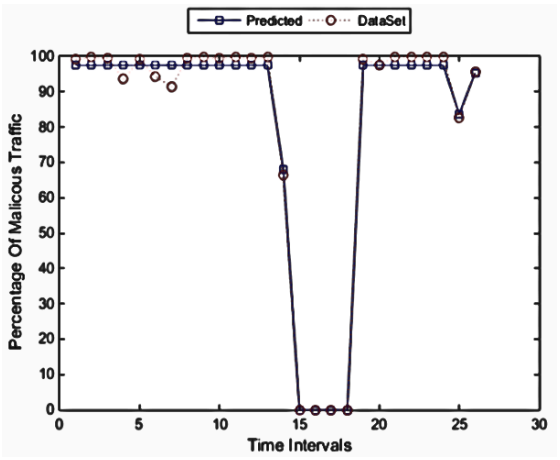


Fig 15. Iteration 2, Accuracy: 98.11%

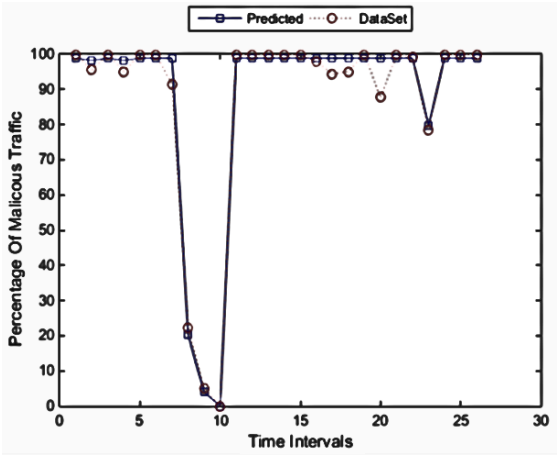


Fig 16. Iteration 3, Accuracy: 97.96%

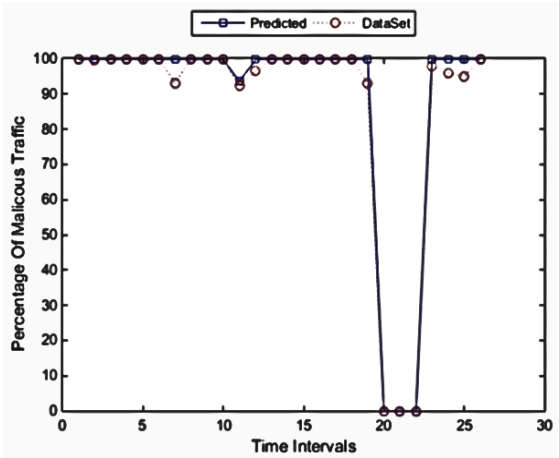


Fig 17. Iteration 4, accuracy: 98.81%

5 Summary, Conclusion, and Future Works

To sum up, in this paper, firstly, we addressed some essential terminologies in order to grasp the intrusion detection area profoundly. After that, a concise literature review concerning some intrusion detection approaches, techniques, and mechanisms as well as some prominent weak points of them was discussed. Then, our proposed technique was stated elegantly. It was evaluated by a flow-based real-world dataset. The results of our experiments depicted that our technique was superior to others in terms of accuracy and performance, particularly in the learning phase.

Several attacks such as DDOS, Scan, and SSH-dictionary were exposed with an accuracy of 98.295575%. Furthermore, because it used some leading features, as a result, it had a notable performance in the training phase of the SVR algorithm. By preparing more datasets in the future, we can offer better solutions to unveil intrusions by combining machine learning techniques as well as powerful statistical methods like the Markov chain model. Furthermore, if we can generate comprehensive flow-based datasets that cover all the attacks that have been identified, these datasets can be used by other researchers to evaluate their methods and techniques with a higher standard. Due to the outstanding performance of our suggested technique in the learning phase, it can also be a competent solution in high-speed networks. Finally, since it is worthwhile to detect botnet activities from traffic dispersion graphs by extracting new features, we are going to conduct research in the future.

References

[1] Lewis, James and Baker, Stewart, The economic impact of cybercrime and cyber espionage. McAfee, 2013.
[2] Mukherjee, Biswanath and Heberlein, L Todd and Levitt, Karl N, "Network intrusion detection," IEEE network, vol. 8, no. 3, pp. 26-41, 1994.
[3] Scarfone, Karen and Mell, Peter, "Guide to intrusion detection and prevention systems (idps)," 2012.
[4] Lazarevic, Aleksandar and Kumar, Vipin and Srivastava, Jaideep, "Intrusion detection: A survey," in Managing Cyber Threats. Springer, 2005, pp. 19-78.
[5] Paxson, Vern, "Bro: a system for detecting network intruders in real-time," Computer networks, vol. 31, no. 23-24, pp. 2435-2463, 1999.
[6] Roesch, Martin and others, "Snort: Lightweight intrusion detection for networks.," in Lisa, 1999, vol. 99, pp. 229-238.
[7] Sperotto, Anna and Schaffrath, Gregor and Sadre, Ramin and Morariu, Cristian and Pras, Aiko and Stiller, Burkhard, "An overview of IP flow-based intrusion detection," IEEE communications surveys & tutorials, vol. 12, no. 3, pp. 343-356, 2010.
[8] Quittek, J and Zseby, T and Claise, B and Zander, S, "Requirements for IP flow information export (IPFIX)," 2004.
[9] Claise, Benoit, "Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information," 2008.
[10] Akoglu, Leman and Tong, Hanghang and Koutra, Danai, "Graph based anomaly detection and description: a survey," Data mining and knowledge discovery, vol. 29, no. 3, pp. 626-688, 2015.
[11] Winter, Philipp and Hermann, Eckehard and Zeilinger, Markus, "Inductive intrusion detection in flow-based network data using one-class support vector machines," in 2011 4th IFIP international conference on new technologies, mobility and security. IEEE, 2011, pp. 1-5.
[12] Sheikhan, Mansour and Jadidi, Zahra, "Flow-based anomaly detection in high-speed links using modified GSA-optimized neural network," Neural Computing and Applications, vol. 24, no. 3-4, pp. 599-611, 2014.
[13] Liao, Hung-Jen and Lin, Chun-Hung Richard and Lin, Ying-Chih and Tung, Kuang-Yuan, "Intrusion detection system: A comprehensive review," Journal of Network and Computer Applications, vol. 36, no. 1, pp. 16-24, 2013.
[14] Li, Zhichun and Gao, Yan and Chen, Yan, "HiFIND: A high-speed flow-level intrusion detection approach with DoS resiliency," Computer Networks, vol. 54, no. 8, pp. 1282-1299, 2010.

[15] David, Jisa and Thomas, Ciza, "Intrusion Detection Using Flow-Based Analysis of Network Traffic," in International Conference on Computer Science and Information Technology. Springer, 2011, pp. 391-399.
[16] Hellemons, Laurens and Hendriks, Luuk and Hofstede, Rick and Sperotto, Anna and Sadre, Ramin and Pras, Aiko, "SSHCure: a flow-based SSH intrusion detection system," in IFIP International Conference on Autonomous Infrastructure, Management and Security. Springer, 2012, pp. 86-97.
[17] Sperotto, Anna and Sadre, Ramin and Van Vliet, Frank and Pras, Aiko, "A labeled data set for flow-based intrusion detection," in International Workshop on IP Operations and Management. Springer, 2009, pp. 39-50.
[18] Staniford-Chen, Stuart and Cheung, Steven and Crawford, Richard and Dilger, Mark and Frank, Jeremy and Hoagland, James and Levitt, Karl and Wee, Christopher and Yip, Raymond and Zerkle, Dan, "GrIDS-a graph based intrusion detection system for large networks," in Proceedings of the 19th national information systems security conference. Baltimore, 1996, vol. 1, pp. 361-370.
[19] Axelsson, Stefan, "Intrusion detection systems: A survey and taxonomy," 2000.
[20] Ellis, D and Aiken, John G and McLeod, Adam M and Keppler, David R and Amman, Paul G, "Graph-based worm detection on operational enterprise networks," McLean, VA, USA: MITRE Corporation, 2006.
[21] Iliofotou, Marios and Pappu, Prashanth and Faloutsos, Michalis and Mitzenmacher, Michael and Singh, Sumeet and Varghese, George, "Network traffic analysis using traffic dispersion graphs (TDGs): techniques and hardware implementation," 2007.
[22] Zhou, Yingjie and Hu, Guangmin and He, Weisong, "Using graph to detect network traffic anomaly," in 2009 International Conference on Communications, Circuits and Systems. IEEE, 2009, pp. 341-345.
[23] Sun, Jimeng and Xie, Yinglian and Zhang, Hui and Faloutsos, Christos, "Less is more: Sparse graph mining with compact matrix decomposition," Statistical Analysis and Data Mining: The ASA Data Science Journal, vol. 1, no. 1, pp. 6-22, 2008.
[24] Mingqiang, Zhou and Hui, Huang and Qian, Wang, "A graph-based clustering algorithm for anomaly intrusion detection," in 2012 7th International Conference on Computer Science \& Education (ICCSE). IEEE, 2012, pp. 1311-1314.
[25] Kelton, AP and Luis, AM and Rodrigo, YM and Clayton, R and Joao, P and Xavier, Alexandre and others, "A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks," Information Sciences, vol. 294, pp. 95-108, 2015.
[26] Ma, Jiefei and Le, Franck and Russo, Alessandra and Lobo, Jorge, "Detecting distributed signature-based intrusion: The case of multi-path routing attacks," in IEEE Conference on Computer Communications (INFOCOM), 2015, pp. 558-566.
[27] Bronte, Robert and Shahriar, Hossain and Haddad, Hisham M, "A signature-based intrusion detection system for web applications based on genetic algorithm," in Proceedings of the 9th International Conference on Security of Information and Network, 2016, pp. 32-39.
[28] Erlacher, Felix, and Falko Dressler, "FIXIDS: A high-speed signature-based flow intrusion detection system," in IEEE/IFIP Network Operations and Management Symposium, 2018, pp. 1-8.
[29] Jelidi, Mohamed and Ghourabi, Abdallah and Gasmi, Karim, "A Hybrid Intrusion Detection System for Cloud Computing Environments," in International Conference on Computer and Information Sciences (ICCIS), 2019, pp. 1-6.
[30] Cortes, Corinna and Vapnik, Vladimir, "Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273-297, 1995.
[31] Basak, Debasish and Pal, Srimanta and Patranabis, Dipak Chandra, "Support vector regression," Neural Information Processing-Letters and Reviews, vol. 11, no. 10, pp. 203-224, 2007.
[32] Eiben, Agoston E and Smith, James E and others, Introduction to evolutionary computing. Springer, 2003, vol. 53.
[33] C. a. V. V. Cortes, "Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273-297, 1995.

Authors' Profiles

Yaser Ghaderipour is from Kurdistan Province, Iran. He is an M.Sc. graduate in Computer Science from the University of Tabriz, Tabriz, Iran. His main research topics are Data Science and Intrusion Detection. He is currently working as a software developer and tries to produce secure and reliable codes in the banking and payment industry. He is responsible for developing and maintaining Payment Switch projects for Behpardakht Mellat (Best Payment Service Provider in the Middle East). He likes to play traditional instruments, read psychology books, and go swimming in his free time.

Mr. Hamed Dinari is from Abdanan, Ilam Province, located in the west of IRAN. He is an M.Sc. graduate in Computer Engineering (Software) from the Department of Computer Engineering (CE), Iran University of Science and Technology (IUST), Tehran, IRAN. His research interests lie primarily in the area of Database Systems, Data Mining, Graph Mining, Indexing, and Distributed Systems. He has published several papers about Graph Mining and Distributed Systems. He currently serves as a Software Engineer to develop and maintain banking projects using Java-based technologies. In his leisure time, he likes to listen to music and study psychology and linguistics books.

How to cite this paper: Yaser Ghaderipour, Hamed Dinari. " A Flow-Based Technique to Detect Network Intrusions Using Support Vector Regression (SVR) over Some Distinguished Graph Features ", International Journal of Mathematical Sciences and Computing (IJMSC), Vol.6, No.4, pp.1-11, 2020. DOI: 10.5815/ijMSC.2020.04.01

