Choosing Decision Tree-Based Boundary Patterns in the Intrusion Detection Systems with Large Data Sets

Hamidreza Ghaffari
Department of Computer Engineering, Islamic Azad University of ferdows, Iran
hghaffari@ferdowsiau.ac.ir

Abstract: *Today, due to the growing use of computer networks, the issue of security of these networks and the use of intrusion detection systems has received serious attention. A major challenge in intrusion detection systems is the enormous amount of data. The generalization capability of support vector machine has attracted the attention of intrusion detection systems in the last years. The main drawbacks of a support vector machine occur during its training phase, which is computationally expensive and dependent on the size of the input dataset. In this study, a new algorithm to speed up support vector machine training time is presented. In proposed method, First, Ant Colony Optimization (ACO) is used to find prototype samples, then a number of prototype samples is randomly selected and the approximate boundary is determined using support vector machine. Based on the approximate boundary obtained, boundary samples are determined using decision tree. Using these boundary samples, final model is obtained. To demonstrate the effectiveness of the proposed method, standard publicly available datasets have been used. The experiment results show that despite the data reduction, the proposed model produces results with similar accuracy and in a faster way than state-of-the art and the current Support Vector Machine (SVM) implementations.*

Keywords: *Intrusion detection systems, boundary patterns, support vector machine, data reduction*.

Received February 14, 2020; accepted August 31, 2021
https://doi.org/10.34028/iajit/19/3/10

1. Introduction

Intrusion Detection System (IDS) is responsible for identifying any unauthorized use of resources and data by monitoring network traffic [9, 10, 19]. In [25], the smart hybrid model was developed to explore any penetrations inside the network. The major challenge in these systems is the high volume of data that is explosively increasing. The most important approach in this regard is to reduce data volume, reduce features and extract important features related to attacks [8, 20, 21].

The generalization capability of Support Vector Machine (SVM) has attracted the attention of intrusion detection systems in the last years. The main drawbacks of a SVM occur during its training phase, which is computationally expensive and dependent on the size of the input dataset [1, 4, 11]. In this study, a new algorithm to speed up SVM training time is presented.

Figure 1 shows how scalable a standard SVM is in a large dataset. As shown in Figure 1, as the number of samples increases, the training time increases exponentially [24].

The proposed method is designed to process relatively large datasets. A similar approach to proposed method is presented in [3]; the proposed strategy is different from the one used in [3]; Initially, Ant Colony Optimization (ACO) is used to find prototype samples, then a number of prototype samples is randomly selected and the approximate boundary is determined using support vector machine. A set of support and non-support vectors are determined using this approximate separating hyper-plane. By using these boundary samples, final model is obtained.

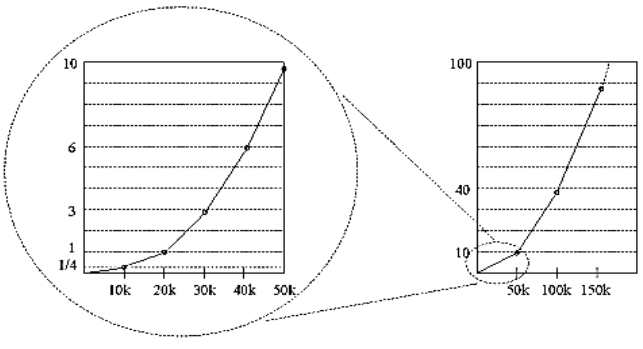

Figure 1. Inelasticity of a support vector machine. Horizontal axis number of sample data, vertical axis of training time per hour [24].

The major contributions of the proposed method are as follows:

1. In this paper, a sample reduction method is proposed to discard the redundant samples.
2. A new method is introduced to divide the training data into boundary and non-boundary patterns

based on the concept of real enemy.
3. By using the boundary samples, final model is obtained.

The structure of this article is organized as follows: Section 2 introduces research and work related to data reduction and intrusion detection systems. In section 3, the proposed model is presented. The effectiveness of the proposed model and its comparison with other similar methods is presented in section 4. Section 5 presents conclusions and suggestions for the future.

2. Related Works

A major challenge in intrusion detection systems is the enormous amount of data [15]. The generalization capability of SVM has attracted the attention of intrusion detection systems in the last years. The main drawbacks of a support vector machine occur during its training phase, which is computationally expensive and dependent on the size of the input dataset [6, 7, 13, 17].

The Class Boundary Maintenance Algorithm (CBP Algorithm) [16] is a multistage method for editing a training dataset. This method, is very simple and yet very effective to eliminate samples. In this method, the purpose is to preserve samples and data close to the class boundary.

In [26], to develop a high-precision model for a large training set, a compromise between speed and accuracy in the classification training step is considered. To do this, it uses a number of k-means clustering algorithm iterations. At each clustering stage, the nearest and farthest sample to the center of the cluster is selected. In the following, a multi-core SVM is taught using selected examples. The evaluation results show that this method significantly reduces the size of the training dataset as well as the training time. At the same time, it obtains a function with relatively good accuracy.

A number of ACO-based algorithms have been introduced in the literature with different classification learning approaches [18]. ADR-Miner, a recently proposed algorithm by the Anwar *et al*. [2], is an adaptation of the ACO meta-heuristic to perform data reduction via instance selection, in order to improve the predictive models of the produced classifiers.

In [3], a method is provided to accelerate SVM in the test phase, estimating SVM decision boundary using a decision tree. To reduce the training data, the decision tree is used to select the samples close to the boundary decision. After determining the samples near the boundary decision, the final classifier will be obtained.

To reduce the sample size of training data, a method called information entropy-based sample reduction is described in [14] to describe support vector data. In this method, the information entropy is computed for the distribution of each data sample. The distance between the two samples was used for the probability of uncertainty of each sample. Samples with higher entropy values are considered as samples near the data distribution boundary. All samples are removed with the entropy values below the threshold.

The adaptive Edited Natural Neighbor (ENaN) algorithm [22] eliminates the noisy patterns based on the concept of natural neighbour obtained adaptively by the search algorithm of the natural neighbour. The constraint Nearest Neighbor-Based Instance Reduction (CNNIR) algorithm was proposed by [23]. It presented a constraint nearest neighbour chain, which only consists of three samples. It is used to choose the boundary samples, which can construct a rough decision boundary. After that, a specific strategy is given to reduce the boundary set.

In [5], a novel SVM classification approach for large data sets based on Social Neighbors is introduced.

3. Proposed Model

An approach to determine the optimal subset of data is proposed. At first, it uses ACO, to find prototype samples, then a number of prototype samples is randomly selected and the approximate boundary is determined using SVM. ACO has been selected as part of the proposed method for many reasons:

1. For the data reduction problem, the ACO algorithms are easily used
2. The problem of reducing is an NP-hard optimization problem.
3. It is easily understood, so that the behavior of the algorithm is not ambiguous for many specialists.

Based on the approximate boundary obtained, boundary samples are determined using decision tree. In other words, it applies a combination of SVM and decision tree to determine boundary samples. The proposed method introduces a pre-processing step to quickly remove nonessential data. The training of classifier is done on those remained "useful" data. Figure 2 shows the proposed model and its stages.

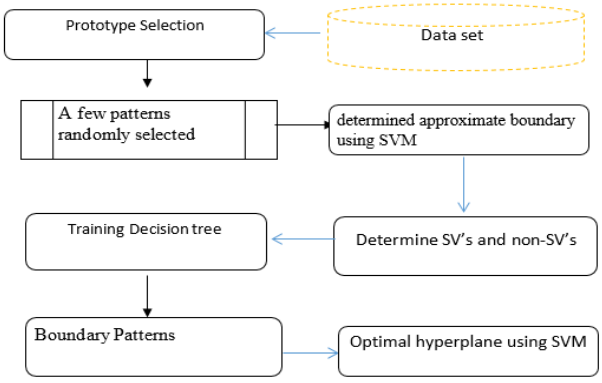

Figure 2. Schematics of the proposed model.

It uses ACO Algorithm to select prototype samples. This method is discussed in more details in section 3.

3.1. Prototypes Selection

It uses ACO Algorithm to select prototype samples. n. This method is presented in Algorithm (1).

- **Initializing the Parameters**

To execute the algorithm, at first the number of ants, the number of process executions, the base pheromone value and the heuristic function are initialized. In the ant algorithm, the heuristic function η (*i*, *j*) represents the inverse of the value of the distance between two samples *i* and *j*.

$$\eta(i,j) = \frac{1}{d(i,j)} \quad (1)$$

- **Selecting the Samples and Calculate its Probability Function**

For each of the ants, a number of samples from the training set will be randomly assigned, which would not be the final selection of those samples by the ant, and the final condition for the selection of the ant random sample i is based on the result of the probability function of that sample and finally the sample will be selected by the Roulette Wheel.

Note: The Roulette Wheel mechanism is as follows. If the pheromone level of the sample selected is greater, then the probability function obtained for that sample will be greater and in the end, the chance of a final selection will be higher.

Probability Function Formula

$$p^{m}_{i(t)} = \frac{[\tau_{i(t)}]^{\alpha}.[\eta_{i(t)}]^{\beta}}{\sum_{i\epsilon j^{m}}[\tau_{i(t)}]^{\alpha}.[\eta_{i(t)}]^{\beta}} \quad (2)$$

$p^{m}_{i(t)}$	The probability function of sample *i* by ant *m* at time *t*
$\tau_{i(t)}$	Amount of pheromone of sample *i* at time *t*
$\eta_{i(t)}$	Amount of pheromone of heuristic function *i* at time *t*
α	Determine the amount of impact the heuristic function on the sample path
B	Determine the amount of pheromone effect present in the sample pathway

- **Evaluation of Ant Performance in Sample Selection**

At each run of the program, the final samples were extracted by the ants (based on the probability function value). Then in order to evaluate the performance of each ant in the final selection of samples, SVM is used and an ant is determined based on the maximum accuracy.

- **Pouring pheromones locally**

Once the samples have been selected by an ant, local pheromone operations are performed on the selected samples, this will increase the amount of pheromone available on the samples selected by the ant compared to the other samples. As a result, the chances of those samples being re-elected by other ants will increase. The following will explain the parameters used in the local update formulas.

$$\Delta\tau^{m}_{i}(t) = \varphi * S^{m}(t) + \frac{(1-\varphi)*(n-L_K)}{n} \quad (3)$$

φ	Local pheromone evaporation coefficient (Local evaporation coefficient is equal to .3)
S^m	The accuracy of classification of the ant m
N	Total number of training set samples
L_K	The number of samples selected by the ant *m*

- **Global Update Operation (Global Pheromone)**

At the end of the ant operation at each stage, a global update (pheromone update) is performed to perform the next step. In this case the base pheromone value of all samples will be updated.

In fact, global update operations are performed to avoid ants' rapid convergence to an inappropriate path. We will continue to explain the parameters used in the global update formula.

$$\tau_i(t+1) = (1-p)*\tau_i(t) + \sum_{m=1}^{N}\Delta\tau^{m}_{i}(t) + \Delta\tau^{g}_{i}(t) \quad (4)$$

p	Evaporation rate of global pheromone effect
$\tau_i(t)$	The pheromone value of sample *i* at time *t.*
$\sum\Delta\tau^{m}_{i}$	The sum of the pheromone values poured by ants onto sample *i*
$\Delta\tau^{g}_{i}$	The amount of ant pheromone selected
N	Total number of ants

- **Identify the best Ant and Choose the Best Examples**

By extracting the most effective ant samples from the training set at each stage, and separately calculating their performance precision in selecting the sample, the criterion for selecting the best ant will be based on the highest accuracy obtained for that ant, after determining the best ant among the other ants. Finally, the number of samples extracted by that ant will also be considered as the best samples.

Note: If two or more ants have the same maximum accuracy, the second condition for selecting the best ants will be to have the least sample extracted by the ant.

Algorithm 1: Prototype Selection Using Ant Colony

Input: datasets (D)
Output: Prototypes selection
1- Initialize the parameters
Start of ant operations
2- Select samples randomly
3- Calculate the probability function of samples
4- Ant evaluation in sample selection
5- Pouring pheromones locally
6- Global Update Operation (Global Pheromone)
7- Finishing the ants job
8- Pheromone update
9- N=N-1

10- If N==1 go to 2
11- Identify the best ant and choose the best examples
12- Selected Prototypes

3.2. Extracting Boundary Samples

From obtained prototype samples in previous stage (T), small amounts and a representative of the data from data sets are selected to make the training of the classifier SVM possible. To make training (huge amount of data) possible, a subset S from the prototype samples is selected so that: | S | <| T | i.e., the cardinality of the selected subset (S) is less than the cardinality of the set (T). First, instead of obtaining the exact boundary, an approximate boundary is obtained. To obtain the approximate boundary, it is trained with a few patterns randomly selected (a number of data are randomly selected). The hyper-plane obtained before is used to determine support vector and non-support vector samples (Tsv).

To determine which samples (T) are most likely to be boundary set, a classifier (decision tree or neural network) is trained based on Tsv data set. Using this classifier, Boundary samples of T is determined. Samples of T which are similar to SV's are selected as boundary samples. By using the boundary patterns, the final model is made using the SVM.

4. Experimental Setup

Two benchmark datasets NSL-KDD and KYOTO [12] were used to evaluate the proposed intrusion detection system. NSL-KDD is a updated version of KDD cup99 data set which suggested to solve some problems of previous version. The NSL-KDD dataset is a UCI dataset available from https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. The details of this dataset are presented in Table 1.

Table 1. NSL-KDD dataset.

Feature no.	Feature name	Feature no.	Feature name	Feature no.	Feature name
1	Duration	15	Su attempted	29	Same srv rate
2	Protocol type	16	Num root	30	Diff srv rate
3	Service	17	Num file creations	31	Srv diff host rate
4	Flag	18	Num shells	32	Dst host count
5	Source bytes	19	Num access files	33	Dst host srv count
6	Destination bytes	20	Num outbound cmds	34	Dst host same srv rate
7	Land	21	Is Host login	35	Dst host diff srv rate
8	Wrong fragment	22	Is guest login	36	Dst host same sreport rate
9	Urgent	23	Count	37	Dst host diff host rate
10	Hot	24	Srv count	38	Dst host serror rate
11	Number failed Login	25	Serror rate	39	Dst host srvserror rate
12	Logged in	26	Srvserror rate	40	Dst host rerror rate
13	Num compromised	27	Rerror rate	41	Dst host srvrerror rate
14	Root shell	28	Srvrerror rate	42	Class lable

As it can be seen in Table 1, each record in the NSL-KDD dataset has 41 attributes and each record falls into either normal (non-attack) or unusual (attack) traffic modes. Therefore, each record in this dataset belongs to the normal class or belongs to one of the types of attacks. The total number of attacks in this dataset is 22.

Table 2. Kyoto dataset.

Feature no.	Feature Name	Feature no.	Feature name
1	Duration	10	Dst host srv count
2	Service	11	Dst host same sre port rate
3	Source bytes	12	Dst host serror rate
4	Destination bytes	13	Dst host srvserror rate
5	Count	14	Flag
6	Same Srv count	15	Source port number
7	Serror rate	16	destination port number
8	Srvserror rate	17	Lablel
9	Dst host count	18	-

The Kyoto dataset is from the KYOTO University dataset available at http://www.takakura.com/Kyoto_data/. As shown in Table 2, each record in the Kyoto dataset has 17 attributes, and each record belongs to either normal (non-attack) or unusual traffic.
In these datasets, each record has different attributes, some of which are discrete or persistent, number or string. Non-number attributes are converted to number (for example attributes such as: protocol, service). Some properties also have nominal or interval values, so pre-processing is required before entering the classification.

The performance of the proposed method is compared with three new methods:

1. Fitcsvm implemented new version of SVM and state of the art for SVM training (Mat lab 2016.b),
2. Data selection based on decision tree for SVM class in the set large Data (DSDTSVM) [3] and Ant Colony Optimization [1], the FIFDR method [21], shell extraction (shel) [13].

For the experiments, the training set is divided into ten equal parts using the validation method 10-fold. All the simulations are done in MATLAB R2016 on a system with a Core i2 processor and 4GB of memory. Newer versions of SVM in Matlab (Matlab.2016b) have better performance in terms of performance, training time and test time than other versions. Since the experiments in previous sources used more Libsvm, the comparison of performance, training time and test time between the two new versions of the support vector machine with increasing number of samples is shown in Figure 3. As it can be seen in each of the three sections, Fitcsvm has better scalability. Therefore, the proposed model uses this version as the base model. The experiment was performed on the NSL-KDD dataset. The number of

training data samples starts from 1000 data samples and increases to 100,000 data samples.

a) Y-axis: performance. b) Y-axis: training time in hours. c) Y-axis: test time in hours.

Figure 3. Comparing LIBSVM and Fitcsvm. X-axis: # of data points.

To evaluate models, it is necessary to use relevant criteria in this area. In this study, the accuracy criterion was used to evaluate the performance of the proposed method. Tables 3 to 8 show the simulation results of the proposed system for each dataset.

Table 3. Comparison of the average accuracy of the proposed method and other algorithms.

Dataset\Method	svm	ACO	DSDT SVM	shel	Fifdr	Proposed method
NSL-KDD	98.7	99.5	99.3	85.4	95.3	98.9
Kyoto	97	98	96.8	72.8	86.2	97.3

Table 4. Average Training Time of the proposed method and other algorithms.

Dataset\Method	svm	ACO	DSDT SVM	shel	Fifdr	Proposed method
NSL-KDD	105	185	68.6	1.8	0.7	11.6
Kyoto	138	283	28.2	6.7	3.7	24.2

Table 5. Average Test Time of the proposed method and other algorithms.

Dataset\Method	Svm	ACO	DSDT SVM	shel	Fifdr	Proposed method
NSL-KDD	16.0	10.4	5.8	0.3	0.8	3.9
Kyoto	13.7	6.3	9.0	1.7	1.11	5.1

Table 6. Average number of support vectors of the proposed method and other algorithms.

Dataset\Method	svm	ACO	DSDT SVM	Shel	Fifdr	Proposed Method
NSL-KDD	6389	3503	2247.5	701	1625	1559
Kyoto	6441	2019	3105	1204	1377	1574

Table 7. Average condensation (Cond) percentages of the proposed method and other compared algorithms.

Dataset\Method	ACO	DSDT SVM	Shel	Fifdr	Proposed Method
NSL-KDD	57.6	69.8	80.6	91	85.70
Kyoto	48.5	79.2	82.7	76.6	91.12

Table 8. Comparison of the average preprocessing time of the proposed method and other algorithms.

Dataset\Method	ACO	DSDT SVM	Shel	fifdr	Proposed Method
NSL-KDD	6.7E+11	0.853	0.6	2.9	0.552
Kyoto	1.0E+12	0.736	0.88	7.58	0.673

5. Analysis of Results

In this study, an algorithm that reduces the size of the training dataset is proposed. It is clear that the proposed method eliminates a large amount of data which may have a direct impact on performance, because the reduced subset may miss important features. The experimental results show that in some cases, the accuracy obtained with the proposed method is better than the classification of the data with the whole dataset. This result shows that the proposed algorithm eliminates the boundary instances that cause noise in the decision process. However, the proposed method improves training time, test time (Tables 4 and 5) and reduces the number of support vectors (Table 6) compared to other methods. The proposed algorithm uses a function that helps to select the best examples in the data set.

The number of support vectors of the proposed method is much lower than the other methods (Table 6), and due to the relationship between data complexity and the number of support vectors, the complexity of this method is lower than of other comparable methods. And thus it's more generalizable.

According to the structure of the SVMs, the training time of the SVM is exponentially dependent on the number of training samples. Therefore, by applying the output of this method to training will reduce the time of training. By examining the results of Table 7, it is concluded that the compression rate of the proposed method is higher than the other methods and despite its high compression rate, it has relatively high efficiency.

By examining Tables 3 to 8, it is concluded that by accepting a pre-processing time (Table 8), the proposed method is scalable for the data set of intrusion detection systems and at the same time operates with high classification accuracy.

6. Conclusions

In this paper, a new method was introduced, that reduces the size of the training dataset. The main challenge in classifying large-scale data sets is that it is difficult to train a model in a time-efficient and high-accuracy manner. The proposed method is designed to process relatively large datasets. This study was conducted to present a useful method for reducing the number of data, in which the training set is divided into the boundary, non-boundary. Boundary

patterns contain amounts of information to exactly describe the decision function. On the contrary, non-boundary patterns far away from the class boundaries have small effect on the classification performance, but reserving the suitable non-boundary patterns can help to improve the performance. A framework is proposed, which retains a suitable number of samples while discarding the redundant samples to achieve a good accuracy as well as a fast training and testing speed. Experimental results show that the proposed algorithm can improve the reduction rate, while maintaining or improving the accuracy.

References

[1] Aburomman A. and Reaz M., "A Novel SVM-KNN-PSO Ensemble Method for Intrusion Detection System," *Applied Soft Computing*, vol. 38, pp. 360-372, 2016.

[2] Anwar I., Salama K., and Abdelbar A., "Instance Selection with Ant Colony Optimization," *Procedia Computer Science*, vol. 53, pp. 248-256, 2015.

[3] Cervantes J., Lamont F., López-Chau A., Mazahua L., and Ruíz J., "Data Selection Based on Decision Tree for SVM Classification on Large Data Sets," *Applied Soft Computing*, vol. 37, pp. 787-798, 2015.

[4] Chitrakar R. and Huang C., "Selection of Candidate Support Vectors in Incremental SVM for Network Intrusion Detection," *Computers and Security*, vol. 45, pp. 231-241, 2014.

[5] Ghaffari H., "Speeding up the Testing and Training Time for the Support Vector Machines with Minimal Effect on The Performance," *The Journal of Supercomputing*, vol. 77, no. 2, pp. 11390-11409, 2021.

[6] Ghaffari H. and Yazdi H., "Multiclass Classifier Based on Boundary Complexity," *Neural Computing and Applications*, vol. 24, no. 5, pp. 985-93, 2014.

[7] Guo L. and Boukir S., "Fast Data Selection for SVM Training Using Ensemble Margin," *Pattern Recognition Letters*, vol. 51, pp. 112-119, 2015.

[8] Ji S., Jeong B., Choi S., and Jeong D., "A Multi-Level Intrusion Detection Method for Abnormal Network Behaviors," *Journal of Network and Computer Applications*, vol. 62, pp. 9-17, 2016.

[9] Joldzic O., Djuric Z., and Vuletic P., "A Transparent and Scalable Anomaly-Based Dos Detection Method," *Computer Networks*, vol. 104, pp. 27-42, 2016.

[10] Kevric J., Jukic S., and Subasi A., "An Effective Combining Classifier Approach Using Tree Algorithms for Network Intrusion Detection," *Neural Computing and Applications*, vol. 28, no. 1, pp. 1051-1058, 2017.

[11] Kumar M. and Gopal M., "A Hybrid SVM based Decision Tree," *Pattern Recognition*, vol. 43, no. 12, pp. 3977-87, 2010.

[12] Kyoto University Benchmark Dataset (2009), http://www.takakura.com/Kyoto_data/. 703, Last Visited, 2021.

[13] Liu C., Wang W., Wang M., Lv F., and Konan M., "An Efficient Instance Selection Algorithm to Reconstruct Training Set for Support Vector Machine," *Knowledge-Based Systems*, vol. 116, pp. 58-73, 2017.

[14] Li D., Wang Z., Cao C., and Liu Y., "Information Entropy Based Sample Reduction for Support Vector Data Description," *Applied Soft Computing*, vol. 71, pp. 1153-60, 2018.

[15] Lin W., Ke S., and Tsai C., "CANN: An Intrusion Detection System Based on Combining Cluster Centers and Nearest Neighbors," *Knowledge-based systems*, vol. 78, pp. 13-21, 2015.

[16] Nikolaidis K., Goulermas J., and Wu Q., "A Class Boundary Preserving Algorithm for Data Condensation," *Pattern Recognition*, vol. 44, no. 3, pp. 704-715, 2011.

[17] Ougiaroglou S., Diamantaras K., and Evangelidis G., "Exploring the Effect of Data Reduction on Neural Network and Support Vector Machine Classification," *Neurocomputing*, vol. 280, pp. 101-110, 2018.

[18] Sharbaf F., Mosafer S., and Moattar M., "A Hybrid Gene Selection Approach for Microarray Data Classification Using Cellular Learning Automata and Ant Colony Optimization," *Genomics*, vol. 107, no. 6, pp. 231- 238, 2016.

[19] Sharma A., Manzoor I., Kumar N., "A Feature Reduced Intrusion Detection System Using ANN Classifier," *Expert Systems with Applications*, vol. 88, pp. 249-57, 2017.

[20] Shen X., Mu L., Li Z., Wu H., Gou J., and Chen X., "Large-Scale Support Vector Machine Classification with Redundant Data Reduction," *Neurocomputing*, vol. 172, pp. 189-97, 2016.

[21] Singh R., Kumar H., and Singla R., "An Intrusion Detection System Using Network Traffic Profiling and Online Sequential Extreme Learning Machine," *Expert Systems with Applications*, vol. 42, no. 22, pp. 8609-24, 2015.

[22] Yang L., Zhu Q., Huang J., and Cheng D., "Adaptive Edited Natural Neighbor Algorithm," *Neurocomputing*, vol. 230, pp. 427-33, 2017.

[23] Yang L., Zhu Q., Huang J., Wu Q., and Cheng D., Hong X., "Constraint Nearest Neighbor for Instance Reduction," *Soft Computing*, vol. 23, no. 11, pp. 13235-45, 2019.

[24] Yu H., Yang J., and Han J., "Classifying Large Data Sets Using Svms with Hierarchical Clusters," *in Proceedings of the 9th ACM SIGKDD International Conference on*

Knowledge Discovery and Data Mining, Washington, pp. 306-315, 2003.

[25] Tabash M., Abd Allah M., Tawfik B., "Intrusion Detection Model Using Naive Bayes and Deep Learning Technique," *The International Arab Journal of Information Technology*, vol. 17, no. 2, pp. 215-24, 2020.

[26] Tang T., Chen S., Zhao M., Huang W., and Luo J., "Very Large-Scale Data Classification Based on K-Means Clustering and Multi-Kernel SVM," *Soft Computing*, vol. 23, no. 1, pp. 3793-3801, 2019.

Hamidreza Ghaffari completed his B.Sc. degree in computer science at Sharif University of Technology and his M.Sc. degree at the University of South Tehran and his Ph.D. at Ferdowsi University. He is currently a faculty member and assistant professor at the Faculty of Computer Engineering, Ferdows Azad University. His interest research areas are machine learning, pattern recognition and image processing.