Combining Conjunctive Rule Extraction with Diffusion Maps for Network Intrusion Detection

Antti Juvonen, Tuomo Sipola
Department of Mathematical Information Technology
University of Jyväskylä
Jyväskylä, Finland
{antti.k.a.juvonen, tuomo.sipola}@jyu.fi

***Abstract*—Network security and intrusion detection are important in the modern world where communication happens via information networks. Traditional signature-based intrusion detection methods cannot find previously unknown attacks. On the other hand, algorithms used for anomaly detection often have black box qualities that are difficult to understand for people who are not algorithm experts. Rule extraction methods create interpretable rule sets that act as classifiers. They have mostly been combined with already labeled data sets. This paper aims to combine unsupervised anomaly detection with rule extraction techniques to create an online anomaly detection framework. Unsupervised anomaly detection uses diffusion maps and clustering for labeling an unknown data set. Rule sets are created using conjunctive rule extraction algorithm. This research suggests that the combination of machine learning methods and rule extraction is a feasible way to implement network intrusion detection that is meaningful to network administrators.**

Keywords*—*Intrusion detection, anomaly detection, n-gram, rule extraction, diffusion map, data mining, machine learning.

I. Introduction

Web services and networks have become more and more complex in the past years. This means that services and servers face new threats and attacks. *Intrusion detection systems* (IDS) are used to detect these attacks. An IDS works generally using one of two detection principles, *signature-based* and *anomaly-based* detection [1]. Signatures are predetermined attack rules that can be used to trigger an alarm when a user's behavior matches the signature. Previous information about intrusions is required for creating these rules. This leads to a low rate of false alarms, but new and unknown threats cannot be detected. On the other hand, anomaly-based detection systems try to detect traffic that deviates from the normal behavior. New attacks can be detected but this methodology will also lead to some false alarms. Both principles can also be combined to a so-called *hybrid intrusion detection system* [2]. Figure 1 shows a simplified block diagram of the different intrusion detection approaches, demonstrating how our system relates to other approaches.

Information security reseachers have been interested in intrusion detection systems extensively, and surveys describing advances in the field have been published [3], [4]. Many machine learning methods, such as self-organizing maps [5] and support vector machines [6] have been used to cluster data and detect anomalies in these systems. Various hybrid systems combining signature and anomaly-based detection have been used [2], [7]. A two-stage adaptive hybrid system for IP

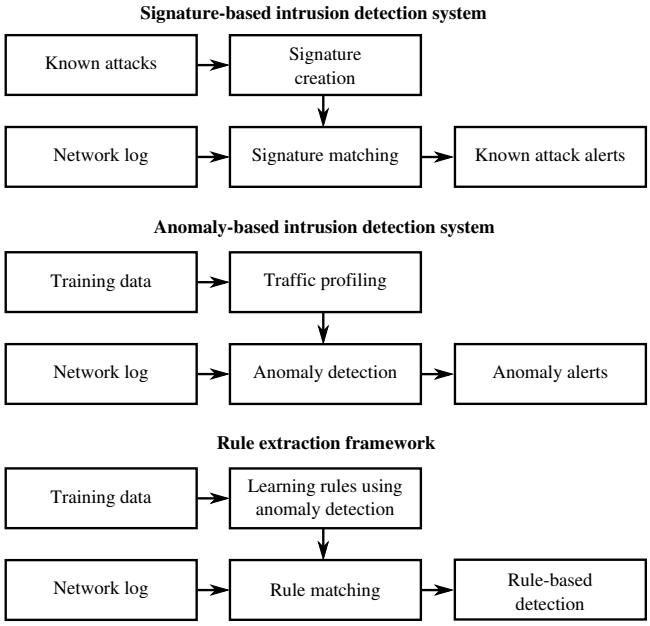

Fig. 1. Different IDS principles.

level intrusion detection has also been recently devised. A probabilistic classifier detects anomalies and a hidden Markov model narrows down attacker addresses [8]. Recently genetic algorithms have been widely used in anomaly detection and misuse detection [9], [10]. Artificial immune systems have raised the interest of intrusion detection researchers [11]. More traditional methods such as k nearest neighbors are also still researched because they can be combined with other methods, for example Dempster-Shafer theory of evidence [12]. A distributed environment has been proposed where intelligent agents analyze the network connections using data mining with association rule mining [13]. Moreover, in our previous work we have researched intrusion detection using dimensionality reduction and clustering to find anomalies from network traffic [14], [15].

The problem with deploying anomaly detection systems in the commercial sector is that some algorithms, such as neural networks, work like a black box [16]. The systems are automated and it is difficult to know exactly how the decisions are made. To overcome this problem, *rule extraction* methods have been proposed [17]. These rules can be directly applied to the original data for efficient web traffic filtering. In addition,

This is the authors' postprint version of the article. The original print version appeared as: A. Juvonen and T. Sipola, "Combining conjunctive rule extraction with diffusion maps for network intrusion detection," in *In The Eighteenth IEEE Symposium on Computers and Communications (ISCC 2013)*. IEEE 2013. Available online at http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6754981&isnumber=6754912

this symbolic knowledge can be read and inspected by humans. This can lead to a better understanding of the data and will aid user acceptance especially in real-life company networks.

One way of extracting these rules is taking a decompositional approach [18]. This can be achieved, e.g., by decomposing a neural network architecture. However, methods of this type are algorithm dependent and the rules themselves may not be sufficiently comprehensible [16]. Another way to extract rules is by using pedagogical approach [17]. This approach takes only the input data and output results into account. Therefore, it is not specific to any particular classification method. Any suitable algorithm can be used to find anomalies or cluster data. Also, the produced rules are directly related to original data and can therefore be easily understood. Because of these reasons, we take the pedagogical approach in our system. Various methods have been used to create different kinds of rule sets and trees. Recent research seems to focus on methods based on heuristic algorithms or creating intelligent wrapper methods [19]. A less researched option is to use conjunctive rules [17]. These rule extraction methods should not be confused with association rule mining [20].

We propose a framework for detecting network anomalies and extracting rules from a data set. Figure 1 shows how it differs from other common approaches. This framework is a supplementary module for signature-based intrusion detection systems, such as next generation firewalls. In this approach, network logs or other similar data is collected and preprocessed to extract features and form numerical matrices to be analyzed further. The dimensionality of this data is reduced for more efficient clustering. After clustering the data to normal and anomalous traffic, the obtained clustering is used to create labels for the data. Subsequently, this information is used to create a rule set for the high-dimensional features. This rule set can then be used to classify traffic and detect intrusions in real time. The proposed framework enables rule creation in an unsupervised manner for previously unknown data. Our contribution is combining unsupervised data analysis with rule extraction techniques to create an online anomaly detection system.

II. Methodology

The proposed framework uses training data to create a rule set which can then be used to classify testing data or actual network traffic data. Thus, our approach is divided into two phases: *rule set learning* and *traffic classification*. The first phase takes the approach of learning the clustering of the data using dimensionality reduction and creating conjunctive rules to describe these clusters in the initial feature space. These rules will then be used to classify new incoming traffic in the second phase. This process is described in Figure 2, which shows the needed input data sets, produced rule set and classification results.

The rule set learning phase aims to find rules that describe the training data. This is done by clustering and labeling the training data set. The resulting rule set classifies data according to the obtained clustering. Architecture of the rule set learning process is shown in Figure 3, which shows the labeling and rule extraction phases in more detail. The methods in individual modules are not fixed, meaning that the specific methods

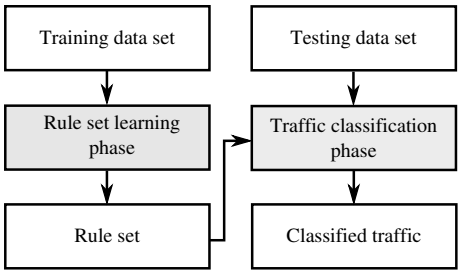

Fig. 2. Block diagram of the overall process.

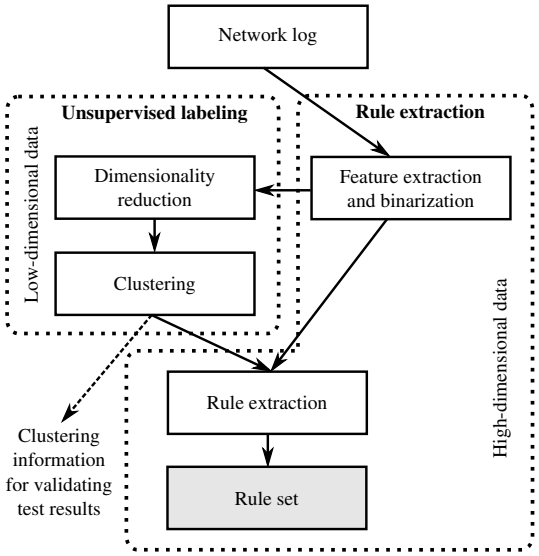

Fig. 3. Block diagram of the rule set learning process.

can be changed if better alternatives are found. The rule set learning phase consists of the following steps:

- Feature extraction from training data
- Unsupervised labeling
  - Dimensionality reduction
  - Clustering
- Rule extraction

In the traffic classification phase new incoming traffic is preprocessed and classified using the generated rule set. Because of the conjunctive nature of the rules simple matching is sufficient. This phase validates how well the rules apply to data that was not part of the training data set. The steps are as follows:

- Feature extraction from testing data
- Classification by rule matching

The following subsections describe the methods used in previously mentioned phases in detail.

A. Feature extraction

Network log files consist of text lines that need to be converted to numeric feature vectors. An n-gram is a consecutive sequence of n characters that represents extracted semantic

information [21]. Our study uses 2-gram features generated from the network logs. This approach produces a rather sparse feature matrix [14]. The rule extraction algorithm works with symbolic conjunctive rules. This means that only nominal and binarized data can be used. Converting data to this kind of format ensures that the feature matrix may be used with the overall learning pipeline.

The feature matrix consists of binary values representing whether an n-gram is present in a specific log line or not. Let us consider the following example. Having two strings containing the words `anomaly` and `analysis`, we can construct the feature matrix in the following way:

an	no	om	ma	al	ly	na	ys	si	is
1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1

In this study, 2-grams are used. However, it is possible to use longer n-grams as well. This will result in more dimensions in the matrix, because there will be more unique n-grams, slowing down the process. For the purposes of this research, 2-grams contained enough information for separating normal and anomalous traffic. Also, using $n = 1$ will give the character distribution. Single characters may not contain enough semantic information, and therefore higher values of n are often used.

B. Dimensionality reduction and clustering

Clustering high-dimensional data is facilitated by dimensionality reduction. We employ diffusion map training to identify the attacks in the training data set. The features describing the dataset are numerous and sometimes hard to interpret together. Therefore, a dimensionality reduction approach using diffusion map is taken. Diffusion map training produces a low-dimensional model of the data, which reveals the internal structure of the dataset and facilitates anomaly detection. In addition, it can cope with non-linear dependencies in the data. Diffusion map retains the diffusion distance in the initial feature space as the Euclidean distance in the low-dimensional space [22], [23], [24].

One log line is represented by feature vector $x_i \in \mathbb{R}^n$. The whole data set is $X = \{x_1,\ x_2,\ x_3,\ \dots\ x_N\}$, from which the affinity matrix

$$W(x_i, x_j) = \exp\left(\frac{-||x_i - x_j||^2}{\epsilon}\right)$$

can be calculated. As seen, the Gaussian kernel is used for the distance matrix and the bandwith parameter ϵ is selected from the optimal region in the weight matrix sum [25]. D, which collects W's row sums on its diagonal, and the transition matrix $P = D^{-1}W$ form the symmetric matrix

$$\tilde{P} = D^{\frac{1}{2}} P D^{-\frac{1}{2}} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}.$$

The singular value decomposition (SVD) of $\tilde{P}$ yields the eigenvectors v_k and eigenvalues λ_k. Now, the low-dimensional coordinates corresponding each original log line are found: $x_i \to [\lambda_1 v_1(x_i),\ \lambda_2 v_2(x_i) \dots\ \lambda_d v_d(x_i)]$. Most of the information is retained in the first eigenvectors and less meaningful ones are left out. Some information is lost because not all eigenvectors are used, but lower dimensionality makes clustering easier.

The k-means method is used to group the data points into clusters. This method is simple and well-known clustering algorithm and it has been used in various data mining tasks. The algorithm description and examples of use can be found in literature [26], [27], [28]. The k-means method relies heavily on the parameter k which determines the number of clusters. Silhouette expresses the quality of clustering for each data point. The optimal number of clusters for the k-means is determined using average silhouette value [29]. An alternative clustering method could be used.

The obtained clustering is believed to describe behavior of the data. If the high-dimensional features can differentiate normal and intrusive behavior, this should be apparent from the resulting low-dimensional clusters. The actual nature of the clusters should be confirmed with domain area experts.

If performance becomes an issue with larger data sets, the learning process could be expanded with out-of-sample extension. However, representative selection of training data is usually a more challenging problem.

C. Rule extraction

A rule is a way to determine the class of a data point based on certain conditions. Ideally a rule would be easily interpretable by a network administrator. All the possible rules span such a huge space that it is not feasible to go through all of them. This means that a sub-optimal but efficient method needs to be used. Such systems have been applied with neural networks [17], [16] and support vector machines [30], [31], [32].

Conjunctive rule is a logical expression containing truth values about the inclusion of binary features. These rules tell whether a symbol should be included, excluded or if it does not matter. Let us assume that we have binary features a, b, c, d, e. Thus, the feature matrix contains five columns corresponding to each binary feature. For the sake of example we have a rule set containing three rules:

$$\begin{aligned} r_1 &= \neg a && \text{for class } c_1, \\ r_2 &= a \wedge b \wedge c \wedge \neg d \wedge e && \text{for class } c_1, \\ r_3 &= a \wedge b \wedge \neg c && \text{for class } c_2. \end{aligned}$$

The rule set for class c_1 would be expressed in logical form as $R_1 = r_1 \vee r_2$. In practice, there are usually multiple rules for each class. Note that in rule r_1, values of features b, c, d and e do not matter. Similarly, for r_3 values of d and e can be anything.

For implementation purposes, the rules are expressed as vectors. The length of these vectors is equivalent to the number of features. The logical truth values are converted to 1 and -1. The values that do not matter are expressed as 0. In the previous example, the rules would be vectors of length 5. Rule r_1 is expressed as a vector $(-1 \quad 0 \quad 0 \quad 0 \quad 0)$. It is easy to match feature vectors to this kind of rule vectors. Note that in

this research a rule symbol corresponds to an n-gram feature as described in II-A.

The conjunctive rule extraction algorithm [17] finds rule-based classifier that approximates the clustering obtained in the unsupervised labeling step. Conjunctive rule extraction is presented in Algorithm II.1. Note that a rule r consists of symbols $r = s_1 \wedge s_2 \wedge s_3 \wedge \ldots \wedge s_n$.

Algorithm II.1: Conjunctive rule extraction.

Input: data points E, classes C
Output: rules R_c that cover E with classification C

```
repeat
  e := get new training observation from E
  c := get the classification of e from C
  if e not covered by the rules R_c then
    r := use e as basis for new rule r
    for all symbols s_i in r do
      r' = r with symbol s_i dropped
      if all instances covered by r are of the same class
      as e then
        r := r'
      end if
    end for
    add rule r to the rule set R_c
  end if
until all training data analyzed
```

The obtained rules separate the training data into the clusters. These rules can now be matched to new incoming data points. Their performance depends on how well the training data covers the behavior of the data. If the point matches one of the rules, the exact type of the abnormal or normal state can be interpreted. If a data point does not fall under any of the rules, then it can be considered abnormal.

Created rules are valid for the classification task while the essential profile of the data remains the same. This is often not the case for extended periods of time, especially for network traffic or similar data. Therefore, rules can be recreated periodically, e.g., daily.

III. Results

This section contains the classification results using real-world network log data. The goal is to perform preliminary validation on real data to test the feasibility of rule extraction in a practical IDS application. The previously described framework was implemented and applied to this data. Data acquisition and analysis are presented below. These results illustrate that the rule set learning phase works on a data that comes from a real-world source.

A. Data acquisition and processing

We use the same network log database that has been used in our previous related research [15]. The data comes from a real-life web server used by a company. Different kinds of intrusion attempts and other abnormal log lines are included in the data. We examine two log files that correspond to different resource URIs. The servers are using Apache server software, which logs network traffic using Combined Log Format. A single log line contains information about the HTTP query:

```
127.0.0.1 - -
[01/January/2012:00:00:01 +0300]
"GET /resource.php?parameter1=value1
&parameter2=value2
HTTP/1.1"
200 2680
"http://www.address.com/webpage.html"
"Mozilla/5.0
(SymbianOS/9.2;...)"
```

The HTTP GET request part of the log line might contain information about SQL injections and other kinds of attacks. This request part is preprocessed using the methods described in Section II-A. Consequently, we get a binary matrix representing whether an n-gram is present in a specific log line or not. The resulting data points are then clustered into normal and anomalous clusters as described in Section II-B. Because the data set is unlabeled, the unsupervised labeling is performed for the whole data set. This is information is used for test result validation as shown in Figure 3.

B. Data analysis

The first data set for initial testing contains 4292 log lines. After preprocessing we find that there are 490 unique 2-grams in the data, resulting in 4292×490 sized feature matrix. Each datapoint now has a label (normal or anomalous) based on the clustering results. This information can be used to extract the rules. We select randomly 2000 data points for rule creation. The whole data set contains 2292 log lines that are not present during rule set learning phase. These remaining lines are our testing data set.

First, we discover that the used algorithm creates 6 rules, 2 for the normal traffic cluster and 4 for the anomalous one. After testing the rules with the whole dataset, all the data points except one match the correct rules. One anomalous data point is not covered by any rule. All of the normal traffic data points match one of the rules. In this case the system works with almost 100% accuracy, which means that the training data represents the testing data well enough.

The second data set contains 10935 log lines. In this data, 414 unique n-grams are found, resulting in a matrix of size 10935×414. After dimensionality reduction, the number of clusters k is determined using the average silhouette value, as described in Section II-B. Figure 4 shows that the data seems to form 4 clusters that are found using k-means algorithm. For rule set learning phase, 8000 data points are used. Other 2935 are used for traffic classification testing. Figure 5 shows all of the data points after dimensionality reduction and clustering used for unsupervised labeling step. As we can see from this visualization, cluster c_4 contains clearly more points than the others.

Rule extraction from the training set produces 15 rules describing 3 of the classes. One class is not featured in the training data and therefore no rules were generated for this class. The testing data set does not contain any samples belonging to class c_1. Out of the 493 data points of class c_3,

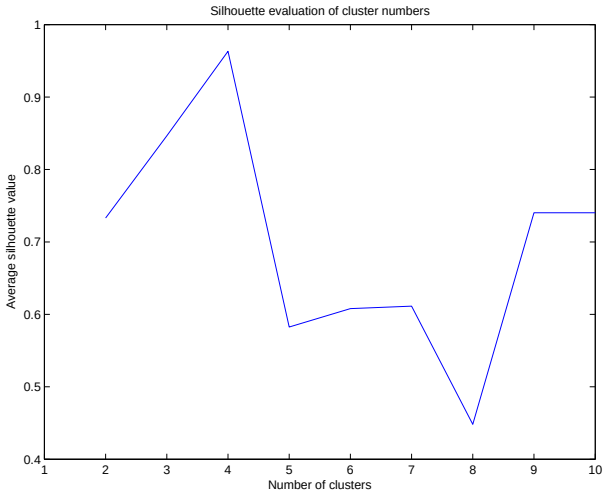

Fig. 4. Optimal cluster number for k-means.

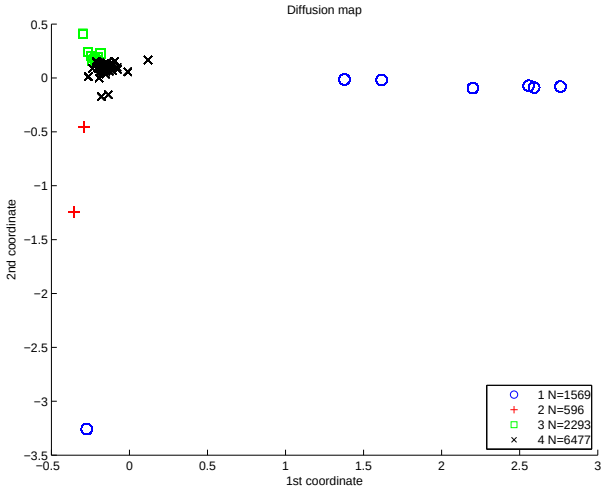

Fig. 5. Two-dimensional visualization of diffusion map of the whole data set.

the extracted rules successfully identify 349 (71%). Test data set contains 2742 data points of class c_4, out of which 1990 are found using the rules (73%). The reason these percentage figures are so low is that the training data differs from the testing data too much. However, the conjunctive rule extraction algorithm always covers the whole training data with 100% identification rate.

IV. Conclusion

Using modern data mining technology in network security context can become problematic when facing end-user needs. Even if the technology produces tangible results, the user rarely has understanding of the methodology. Therefore, this so-called black box system is not a desirable end goal. Simple conjunctive rules are easier to understand, and rule extraction from the complex data mining techniques might facilitate user acceptance. In this research, we have combined rule extraction methodology with diffusion map training framework in order to produce a rule-based network security system.

The main benefit of this framework is that the final output is a set of rules. No black box implementation is needed as the end result is a simple and easy to understand rule matching system. The training data may contain intrusions and anomalies, provided that the clustering step can differentiate them. In addition, rule matching is a fast operation compared to more complex algorithms.

The experimental data sets in this study are suitable for rule generation. The number of created rules is not too high for practical purposes and the accuracy with the first data set is high enough. Data points that do not match any rule could still be flagged as an anomaly in a practical intrusion detection system. The most important thing is to recognize normal traffic accurately. However, if new data points introduced after rule generation are very different from the training data set, the accuracy of classification using the rules might suffer considerably. Periodical rule updating will solve this issue. The second test data set demonstrates how important it is to have a training set that corresponds to the real situation as accurately as possible. If some types of data points are not featured in the rule generation phase, corresponding rules are not generated and these points will not be classified correctly. With proper training data the generated rules give much better accuracy. The created clustering may not represent reality but it is convenient while actual data labels are unknown. Another concern is overfitting of the rules, but the rules can be generalized to mitigate this problem.

The proposed framework is useful in situations where high-dimensional data sets need to be used as a basis for anomaly detection and quick classification. Such data sets are common nowadays in research environments as well as in industry, because collecting data is wide-spread. Our example case has been network security, which bears real benefits to anyone using modern communication networks. The provided tools are useful for network administrators who are trying to understand anomalous behavior in their networks.

Future topics include dynamic rule update as systems evolve, rule set optimization and using the rule set to filter real-time data sets. The modular structure of the framework enables these additions to be implemented conveniently. The applicability of the system to a wider network security context should also be tested, meaning cooperation with other security systems and components such as next-generation firewalls and other signature-based systems.

Acknowledgment

This research was supported by the Foundation of Nokia Corporation. Thanks are extended to Kilosoft Group Oy.

References

[1] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," *NIST Special Publication*, vol. 800, no. 2007, p. 94, 2007.

[2] M. A. Aydın, A. H. Zaim, and K. G. Ceylan, "A hybrid intrusion detection system design for computer network security," *Computers & Electrical Engineering*, vol. 35, no. 3, pp. 517–526, 2009.

[3] A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion detection: A survey," *Managing Cyber Threats*, pp. 19–78, 2005.

[4] F. Sabahi and A. Movaghar, "Intrusion detection: A survey," in *Systems and Networks Communications, 2008. ICSNC'08. 3rd International Conference on*. IEEE, 2008, pp. 23–26.

[5] M. Ramadas, S. Ostermann, and B. Tjaden, "Detecting anomalous network traffic with self-organizing maps," in *Recent Advances in Intrusion Detection*, G. Vigna, E. Jonsson, and C. Kruegel, Eds. Springer, 2003, pp. 36–54.

[6] Q. Tran, H. Duan, and X. Li, "One-class support vector machine for anomaly network traffic detection," *China Education and Research Network (CERNET), Tsinghua University, Main Building*, vol. 310, 2004.

[7] H. Om and A. Kundu, "A hybrid system for reducing the false alarm rate of anomaly intrusion detection system," in *Recent Advances in Information Technology (RAIT), 2012 1st International Conference on*, march 2012, pp. 131–136.

[8] R. Rangadurai Karthick, V. Hattiwale, and B. Ravindran, "Adaptive network intrusion detection system using a hybrid approach," in *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, jan. 2012, pp. 1–7.

[9] L. Li, G. Zhang, J. Nie, Y. Niu, and A. Yao, "The application of genetic algorithm to intrusion detection in mp2p network," in *Advances in Swarm Intelligence*, ser. Lecture Notes in Computer Science, Y. Tan, Y. Shi, and Z. Ji, Eds. Springer Berlin Heidelberg, 2012, vol. 7331, pp. 390–397.

[10] M. Goyal and A. Aggarwal, "Composing signatures for misuse intrusion detection system using genetic algorithm in an offline environment," in *Advances in Computing and Information Technology*, ser. Advances in Intelligent Systems and Computing, N. Meghanathan, D. Nagamalai, and N. Chaki, Eds. Springer Berlin Heidelberg, 2012, vol. 176, pp. 151–157.

[11] A. Parashar, P. Saurabh, and B. Verma, "A novel approach for intrusion detection system using artificial immune system," in *Proceedings of All India Seminar on Biomedical Engineering 2012 (AISOBE 2012)*, ser. Lecture Notes in Bioengineering, V. Kumar and M. Bhatele, Eds. Springer India, 2013, pp. 221–229.

[12] D. Dave and S. Vashishtha, "Efficient intrusion detection with knn classification and ds theory," in *Proceedings of All India Seminar on Biomedical Engineering 2012 (AISOBE 2012)*, ser. Lecture Notes in Bioengineering, V. Kumar and M. Bhatele, Eds. Springer India, 2013, pp. 173–188.

[13] I. Brahmi, S. Yahia, H. Aouadi, and P. Poncelet, "Towards a multiagent-based distributed intrusion detection system using data mining approaches," in *Agents and Data Mining Interaction*, ser. Lecture Notes in Computer Science, L. Cao, A. Bazzan, A. Symeonidis, V. Gorodetsky, G. Weiss, and P. Yu, Eds. Springer Berlin Heidelberg, 2012, vol. 7103, pp. 173–194.

[14] T. Sipola, A. Juvonen, and J. Lehtonen, "Anomaly detection from network logs using diffusion maps," in *Engineering Applications of Neural Networks*, ser. IFIP Advances in Information and Communication Technology, L. Iliadis and C. Jayne, Eds. Springer Boston, 2011, vol. 363, pp. 172–181.

[15] ——, "Dimensionality reduction framework for detecting anomalies from network logs," *Engineering Intelligent Systems*, vol. 20, pp. 87–97, 2012.

[16] N. Ryman-Tubb and A. d'Avila Garcez, "SOAR – Sparse oracle-based adaptive rule extraction: Knowledge extraction from large-scale datasets to detect credit card fraud," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–9.

[17] M. W. Craven and J. W. Shavlik, "Using sampling and queries to extract rules from trained neural networks," in *In Proceedings of the Eleventh International Conference on Machine Learning*. Morgan Kaufmann, 1994, pp. 37–45.

[18] A. d'Avila Garcez, K. Broda, and D. Gabbay, "Symbolic knowledge extraction from trained neural networks: A sound approach," *Artificial Intelligence*, vol. 125, no. 1, pp. 155–207, 2001.

[19] D. Martens, B. Baesens, and T. Van Gestel, "Decompositional rule extraction from support vector machines by active learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, no. 2, pp. 178–191, 2009.

[20] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining–a general survey and comparison," *ACM SIGKDD Explorations Newsletter*, vol. 2, no. 1, pp. 58–64, 2000.

[21] M. Damashek, "Gauging similarity with n-grams: Language-independent categorization of text," *Science*, vol. 267, no. 5199, p. 843, 1995.

[22] R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Nadler, F. Warner, and S. Zucker, "Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 21, pp. 7426–7431, 2005.

[23] R. R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, 2006.

[24] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, "Diffusion maps, spectral clustering and reaction coordinates of dynamical systems," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 113–127, 2006.

[25] R. Coifman, Y. Shkolnisky, F. Sigworth, and A. Singer, "Graph laplacian tomography from unknown random projections," *Image Processing, IEEE Transactions on*, vol. 17, no. 10, pp. 1891–1899, oct. 2008.

[26] P. Tan, M. Steinbach, and V. Kumar, "Cluster analysis: Basic concepts and algorithms," *Introduction to data mining*, pp. 487–568, 2006.

[27] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[28] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice Hall., 1988.

[29] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, no. 0, pp. 53 – 65, 1987.

[30] H. Núñez, C. Angulo, and A. Català, "Rule extraction from support vector machines," in *In European Symposium on Artificial Neural Networks Proceedings*, 2002, pp. 107–112.

[31] N. Barakat and J. Diederich, "Learning-based rule-extraction from support vector machines," in *The 14th International Conference on Computer Theory and applications ICCTA'2004*, 2004.

[32] N. Barakat and A. P. Bradley, "Rule extraction from support vector machines: A review," *Neurocomputing*, vol. 74, no. 1–3, pp. 178–190, 2010.