International Journal Of Electronics Engineering (Issn: 0973-7383)
Volume 10 • Issue 2 pp. 175-184 June 2018-Dec 2018 www.csjournals.com

Review on Data Mining Classification Rules For Identifying Malicious Behavior in Network

[1]Janki Arora, [2]Mukesh Yadav
[1]Student, Assistant Professor[2]
Department of Computer Science & Engineering, Gurgaon Institute of Technology and Management, Gurgaon(Hr.), India
jankiarora1@rediffmail.com , mukesh.cs@ieee.org

Abstract: This paper gives background on Evolutionary Algorithms (EAs) especially focusing on Genetic Algorithms (GAs) in the data mining including Fuzzy Decision Rules. This chapter outlines the pervious related work on the EAs to KDD for discovering fuzzy decision rules and various IDS developed using Genetic Algorithm
Keywords: Evolutionary Algorithms (EAs) , Genetic Algorithms (GAs)

1 INTRODUCTION

Data classification represents an important theme and is perhaps the most commonly applied data mining technique (Fayyad et al., 1996). The classification problem becomes very hard when the numbers of possible different combinations of parameters are so high that techniques based on exhaustive search of the parameter space rapidly become computationally infeasible. Thus, it is natural to devote attention to a heuristic approach to the classification problem (Falco et al., 2005). Traditionally, classifier algorithms focused either on accuracy or interpretability. But fuzzy classifier takes into account both performance and interpretability so as to keep rule bases small and comprehensible (Roubos et al., 2001).
Fig.1 represents various Machine Learning techniques that have been applied for automated discovery of FCRs

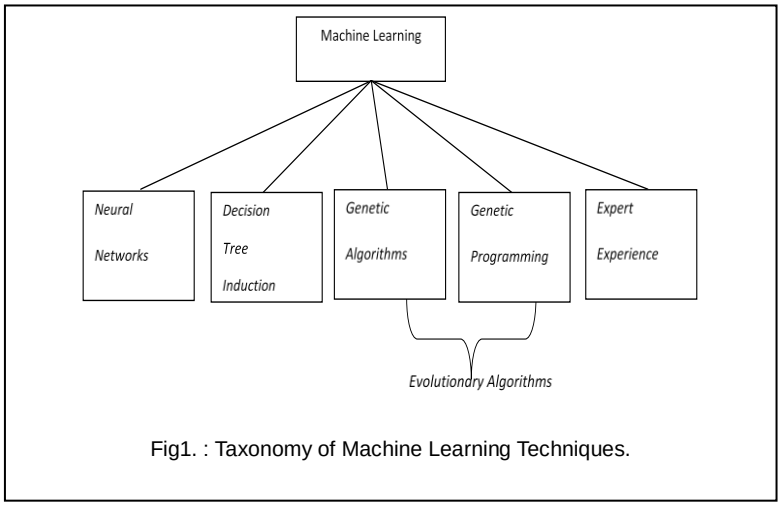


Fig1. : Taxonomy of Machine Learning Techniques.

2.1 Neural Networks

Neural Networks (Lippmann, 1987) can also be used to generate fuzzy decision rules as both neural networks and fuzzy systems are functionally equivalent despite of different structures. This functional equivalence has been explained and proved by Buckley et al. (1993) in the sense that any continuous, layered feedforward neural net can be approximated to any degree of accuracy by a fuzzy logic system and any continuous, discrete fuzzy logic system can be approximated to any degree of accuracy by a three-

layered, feedforward neural net. Hayashi and Imura (1990) suggested a two-step procedure to extract fuzzy rules. In the first step, a neural net is trained from sample data and in the subsequent step an algorithm is used to automatically extract fuzzy rules from the trained neural net. Many other methods of generating fuzzy rules through neural networks have been suggested. Kosko (1992) proposed a system called Fuzzy Cognitive Maps, which integrates neural network and fuzzy logic. A fuzzy logic system can also be constructed directly with neurons representing fuzzy logic and linguistic terms. Lin and Lee (1991) proposed a neural-network-based fuzzy logic system which consists of five layers. The first layer is the input layer. Each node in this layer represents a linguistic variable. The second and fourth layers contain term nodes which act as membership functions to represent the terms of the respective linguistic variable. The third layer is the rule node layer. Each node with its connection represents a fuzzy rule. The fifth layer is the output layer. On this layer two nodes are used for each output variable. One is for the desired output and the other is for the actual output. The system can be constructed from training examples with a hybrid learning scheme where a self-organized clustering learning method is used to locate fuzzy membership functions for the input and output terms. In this a competitive learning method is used to determine the connection of rule nodes and a supervised learning method is used to tune membership functions in order to improve the system's performance.

Though neural network approach is suitable in building a fuzzy logic system with a relatively small number of numerical variables but lack of analytical guidance in determining the network configuration and trap of local optimal in the learning process limits its applicability in fuzzy environment.

2.2 Decision Tree InductionFinal Stage

Another widely used machine learning method is the induction of decision trees (Quinlan, 1986; Safavian and Landgrebe, 1991). The search method decision tree induction employs fuzzy entropy to find the most efficient decision nodes (Weber, 1992; Yuan and Shaw, 1995). Although in most of the cases this method works well but still is rendered inefficient for generating fuzzy decision tree, as it may not be able to generate best tree due to one-step ahead node splitting without backtracking. Moreover, the best tree may not be able to yield the best set of rules.

2.3 Evolutionary Algorithms

The abovementioned search techniques, neural networks and decision tree induction have problems of being trapped into local optimal. Thus, application of EAs for discovering comprehensible IF-THEN classification rules in the fuzzy environment is preferred over other searching techniques. There has been very extensive research on EAs for discovering FCRs (Cordon et al., 2004; Walter and Mohan, 1999). Simple taxonomy of Fig.2.2. represents three different evolutionary approaches used for discovering FCRs.

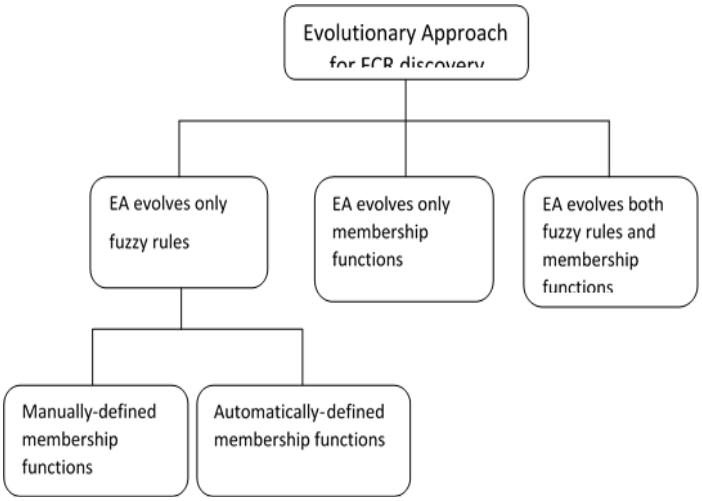


Fig.2.2: Taxonomy of using an EA for discovering FCRs EAs for generating Fuzzy Rules

In this approach EA is used to search for good combinations of attribute values that will compose fuzzy rules. However, the membership functions of the attribute values are predefined (either manually or by another algorithm), rather than being evolved by EA. Therefore, the EA is used to find combinations of attribute values that are relevant for accurately predicting the value of another attribute.

This is conceptually similar to the use of EAs for discovering crisp prediction rules. Actually, from the viewpoint of rule encoding, a fuzzy rule condition such as Age=young, where young is a linguistic value associated with a fuzzy set, is not different from a crisp rule condition involving a categorical, discrete attribute such as Age=0⋯25, where the attribute Age was previously discretized into crisp intervals such as 0⋯25. In both cases we have a rule condition of the form⟦ Attr⟧_i=⟦Val⟧_ij, where Valij is the j-th value belonging to the domain of the i-th attribute.

The evolutionary approaches to specify the shape and number of membership functions are divided into two categories listed below:

Fixed membership functions (User-defined membership functions)

The shape and number of membership functions for each attribute being fuzzified can be simply defined by the user. An important and most cited initial contributions for FCRs discovery using this approach are advocated by Ishibuchi (1999a, 1999b, 2000). They designed EA that used user-defined membership functions for each continuous attribute thus, evolving only the combinations of relevant attributes for predicting the goal attribute which in turn reduces the search space.

The motivation for this approach is twofold. First, it can be regarded as a form of incorporating some background knowledge into the rule discovery system. The user can specify membership functions that are sensible, according to his knowledge of the application domain and the meaning of data being mined. As a result, the membership functions will be consistent with the user's previous knowledge which adds to the improvement in the comprehensibility of discovered knowledge (Pazzani, 2000).

Second, this approach avoids the computationally-expensive process of trying to optimize the shape and number of membership functions.

This approach has been extensively used by several researchers due to the fact that specification of membership functions incorporate users domain knowledge, thereby, resulting in a more comprehensible membership function for user.

Despite of quite obvious benefits this approach has several shortcomings also. It reduces the degree of autonomy of the system. Moreover, the quality of the discovered rules depends on the quality of the user-defined membership functions. In addition, different users may prefer membership functions that are quite different from each other. As a result the system may have to be run several times, once for each user. This may significantly reduce the above-mentioned benefit of reducing computation time.

Two methods for manually defining membership functions with the help of a domain expert or a set of experts discussed by Klir and Yuan (1995) are:

Direct Methods

In this experts give answers to questions directly related to the membership functions being defined. For example, a domain expert can directly assign to each element x∈X a membership grade⟦ μ⟧_A (x). The expert's answers are then used to define a membership function by using an appropriate curve-fitting method.

Indirect Methods

In this expert answer simpler questions that are only indirectly related to membership functions. For example, instead of directly assigning a membership degree μ_A (x) to each element x∈X, an expert can simply compare elements x in a pairwise fashion according to their relative degree of membership in the fuzzy set A. These methods tend to require a more elaborate post-processing of the expert's answers but they tend to be less sensitive to various biases of subjective judgement.

Automatically-defined membership functions

Instead of being manually-defined membership can be automatically defined. One possible approach for automatically-defining membership functions consists of running a clustering algorithm (Bentley, 2000; Klir and Yuan, 1995).

EAs for tuning the membership functions associated with the fuzzified attributes, appearing in fixed rules This approach is typically used when crisp rules have already been discovered by another algorithm and EA is employed only for fuzzifying the discovered crisp rules.
One example of an EA designed for this task is presented by Crockett et al. (2000). In this work a GA is used to fuzzify the decision nodes (the internal nodes) of a previously induced decision tree. In essence the GA optimizes the sizes of fuzzy regions around each decision node of the induced decision tree. For instance, consider the crisp decision tree shown in Fig.2.3.

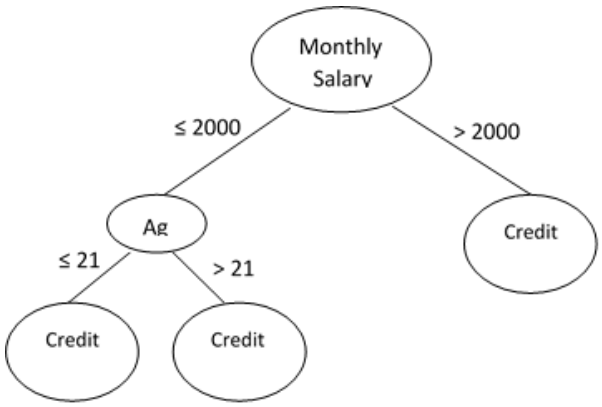
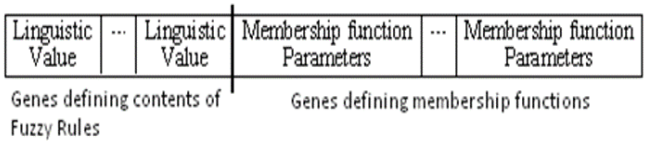


Fig.2.3: Example of induced decision tree

In this example the GA's goal is to define four membership functions, namely two opposing membership functions for the root node and two opposing membership functions for the node Age. Each membership function is determined by a gene of an individual, so in this example four genes would be necessary to define the four membership functions.

2.4 EAs for both generating fuzzy rules and tuning membership functions Copyright Form

In this approach an EA is used for optimizing both the contents of fuzzy rules (i.e., the combination of attribute values occurring in the rules) and the membership functions of the linguistic values of the fuzzified attributes (Pena-Reyes and Sipper, 1999; Xiong and Litz, 1999; Mota et al., 1999; Chen and Ho, 2001). The user-defined membership function approach has advantage over current approach.
In this case the genotype (concerning individual encoding and genetic operators) of an individual has atleast two kinds of genes i.e. an individual consist of two strings as illustrated in Fig.2.4.



2.4 Genotype for optimizing both rules and membership functions

The first part of the genotype defines the attribute values occurring in the antecedents of the rules contained in an individual. The consequent is the same for all rules in all individuals of the population i.e., it is fixed for the entire GA run.
The second part of the genotype defines for each attribute being fuzzified two parameters that specify the exact shape of the membership functions.
The type of EA to be used depends on the application domain and various EAs that are most commonly employed are: Yuan and Zhuang (1996) proposed a genetic approach for discovering FCRs. Several techniques such as encoding, fitness function, and viability check and rule extraction are used to improve the

efficiency and the effectiveness of the algorithm. Despite of such an effective genetic approach for rule discovery, this method didn't provide any solution for fuzzifying numeric variables and can work well only on the data set with pre-assumed membership values for each fuzzy modifier.

Roubous et al. (2001) employed a covariance based model initialization method to obtain an initial fuzzy classifier from data. For computing the membership degrees, this method uses a covariance matrix and a fuzzy partition matrix along with GA based tuning.

Besides FCRs, GA can also be used to discover interesting fuzzy prediction rules (Romao et al., 2002). For this, subjective approach is preferred as application domain is science and technology data and user is expert in this domain rather than Noda et al. (1999) GA which followed objective approach. This approach is user independent and has more generality and autonomy.

Mansoori et al. (2008) proposed a novel approach called SGERD (SGERD: Steady-State Genetic Algorithm for Extracting Fuzzy Classification Rules from Data), for extracting fuzzy classification rules from labeled or numerical data. It uses a non-random mechanism to select best individuals. SGERD Algorithm is simple, intuitive and fast and generates few rules that are short, accurate and interpretable. Also, its population size and number of generations are small, so saves memory space.

Recently sequential and parallel GA approaches have been presented for discovery of rules in the form of Censored Production Rules (CPRs -> PRs + Exceptions) (Saroj and Bharadwaj, 2009). Both of these genetic algorithms have limitation that they can work only with categorical attributes and not with continuous attributes.

Genetic Programming

Besides using GA, Genetic Programming (GP) which has emerged as an extension of GA proposed by Cramer (1985) and Salustowicz and Schmidhuber (1997) is also applicable for discovering FCRs when a direct search is impossible. The main difference between GA and GP lies in the representation of the structure they manipulate and the meaning of the representation. Furthermore, GA usually operates on a population of fixed-length binary strings whereas; GP typically operates on population of parse trees or syntactic trees to represent the problem (Michalewicz, 1996; Cordon, 2004). For instance, consider a rule If X1 is M and X2 is L Then Y,corresponding syntactic tree depicting GP individuals is represented in Fig2.5.

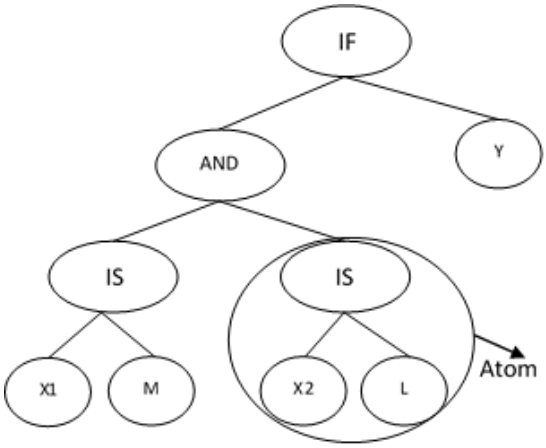


Fig.2.5: Syntactic Tree

Where, an atom is syntactically a predicate of the form operator (attr, operator, value) i.e., has three arguments: attribute name, relational operator (is) and attribute values of the data set being mined.

Mendes et al. (2001) proposed a Co-Evolutionary System for discovering FCRs. The System used two EAs. A GP algorithm which evolves a population of fuzzy decision rule sets and a simple EA for evolving a population of membership functions. Such evolutionary systems may achieve on rule accuracy front but

these are complex and have to search a large and complex search space simultaneously. An advantage of using co-evolutionary approach is that fitness is evaluated across several rule sets rather than single fuzzy set. But this increases processing time which is then reduced by evaluating membership function across best GP individuals satisfying syntactical restrictions. The basic idea of co-evolution was proposed by Delgado et al. (1999) who used three populations, two of which were evolved by GA. In contrast to Delgado's work, Mendes et al. (2001) used GP rather than GA for discovering FCRs. GP works by evolving a population of randomly generated initial population using a fitness measure. The main intent behind using GP is its ability to automatically inherent data dependencies in its varied evolving structures rather than fixed size solutions. Also, the applicability of GP is preferred over other search techniques due to its global search mechanism and inherited stochastic search properties which make it less likely to get stuck in local optimum.

Comparable to the Mendes et al. (2001) work, Akbarzadeh et al. (2008) proposed a Relational Fuzzy Classification Rule Derivation (RFCRD) system. This method used two populations. One for encoding FCRs, evolved by constrained-syntax GP and another keeps membership functions definition, evolved by mutation-based EA. These two populations co-evolve to better classify the underlying data set. Unlike other approaches that use fuzzification of data set containing continuous attributes for discovering FCRs, this system fuzzifies the relational operators.

The application of standard GP to the classification task is relatively straightforward, as long as all the attributes are numeric. In this case we can include in the function set several kinds of mathematical function appropriate to the application domain and include in the terminal set the predicting attributes and possibly a random-constant generator. Once we apply functions in the internal nodes of a GP individual to the values of the attributes in the leaf nodes of that individual, the system computes a numerical value that is output at the root node of the tree. Assuming a two-class problem, if this output is greater than a given threshold the system predicts a given class; otherwise the system predicts other class.

Despite of quite obvious benefits of using evolutionary approach called GP there are several domains where it is not applicable as it suffers from a closure problem which arises when multiple data types are used. The closure property means that the output of a node in a GP tree can be used as the input to any parent node in the tree. Though this property is satisfied when standard GP is applied to numerical data as in principle, the number returned by mathematical function can be used as the input to another mathematical function. But the problem arises when GP is to be used for discovering high-level, comprehensible prediction (IF-THEN) rules, rather than just producing a numerical signal in the root node. For instance, consider a parse tree given in Fig.2.6

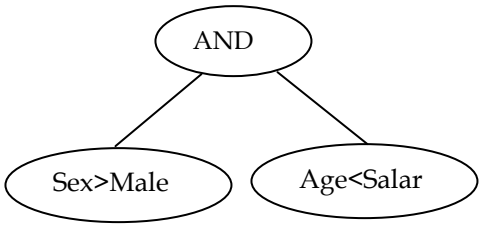


Fig.1.6: Closure Problem

The tree shown in this Fig. illustrates two kinds of violation of data type restrictions. The rule condition [Sex>Male] is an invalid propositional logic condition, since the operator ">" does not make sense for a categorical attribute such as sex.

Also, the rule condition [Age<Salary] can be considered as an invalid first-order-logic condition (from a semantical viewpoint), since the attributes age and salary have different incompatible domains.

Several solutions have been proposed to cope with the closure requirement of GP in the context of prediction-rule discovery. These proposed solutions are divided into three groups:

Booleanizing all terminals and using only Boolean functions.

Using a constrained-syntax or strongly-typed GP.

Using a grammar-based GP.

To resolve closure problem data type standardization must be employed. That is, if data set being mined contains attributes with different data type then we must booleanize all attributes by converting them into a single standard data type. Thereby, including in the function set only functions having arguments and producing outputs of that data type.

A particular approach for Booleanizing is to booleanize all terminals in the terminal set and using only Boolean functions [logical AND, OR, NOT etc.] in the function set. The Booleanization of terminal set is usually done as a pre-processing step for the GP. As a result of Booleanization, in general each terminal in the terminal set corresponds to a Boolean rule condition or to a Boolean attribute.

Another approach is to use constrained-syntax GP. The key idea is that, for each function available in the function set, the user specifies the type of its arguments and the type of its result (Bhattacharya et al., 1998). Crossover and Mutation are then modified to create only valid trees, by respecting the user-defined restrictions on tree syntax. An example is shown in Table 2.1, where, for each row, the second and third columns specify the data type of the input and of the output of the function specified in the first column. Once this kind of specification is available to the GP, the system can generate individuals as the one shown in Fig.2.7. This figure assumes that Atr1 is a categorical (nominal) attribute, whereas Atr3, Atr5, Atr6 are real-valued attributes.

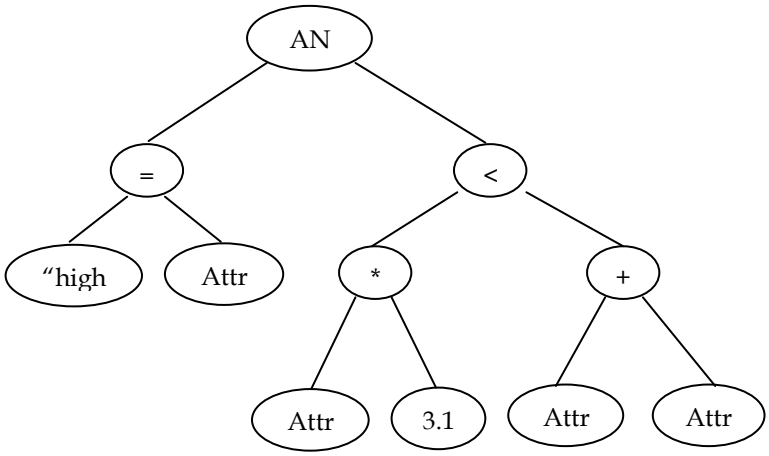


Fig.1.7: Example of GP individual: (representing a rule antecedent) meeting the data type restrictions specified in Table 1.1

Functions	Data type of Input Arguments	Data type of Output Arguments
+, -, *, /	(real, real)	real
≤, >	(real, real)	boolean
=	(nominal, nominal)	boolean
AND, OR	(boolean, boolean)	Boolean

Table1.1: Example of data type definitions for input and output functions.

Another evolutionary scheme proposed by Marghny and El-Semman (2005) called GEP seems more promising for discovering FCRs and is applicable when data type is a grammar based on GP or is an atomic representation than. GEP is a genotype (chromosomes) or phenotype (expression trees) system that evolves computer programs encoded in linear chromosomes of fixed length.

Another approach to construct fuzzy classification system using Genetic-based machine learning approach is to apply expert experience. In the method proposed by Li et al. (2007), experts experience was translated into fuzzy sets by similarity measures and was integrated into fuzzy genetic based learning mechanism. This classification algorithm was able to achieve better accuracy and interpretability. The main advantage behind applying expert experience to fuzzy classification system is that knowledge of man and machine is similar, so it is convenient to design interpretable fuzzy classification system. Moreover, applying expert experience will reduce the computing complexity. Pittsburgh (Smith, 1980) and Michigan (Holland, 1986) approaches are the two genetic-based machine learning approaches for rule discovery. In the former each rule set is handled as an individual thus it represents a complete solution whereas in latter each rule is handled as an individual thus it represents a partial solution. The choice between two approaches depends on rule to be discovered. But in order to translate expert experience into fuzzy sets, the former is preferred over latter as it can directly optimize fuzzy rule-based system. For this, modifiers are used which act on atomic words and modify membership of fuzzy sets.
A hybrid co-evolution algorithm, which combines both the machine learning approaches, can also be used to construct fuzzy classification systems. In this Pittsburgh-style algorithm is used to evolve rule bases and Michigan-style algorithm is used to evolve rules of fuzzy modeling (Limin et al., 2008).

2.5 GA BASED IDS

GAs and GP have been used of or network intrusion detection in different ways. Some approaches directly use GAs to derive the classification rules, while some others use different AI methods for acquisition of rules, where GAs are used to select appropriate features or to determine the optimal parameters of some functions. The early effort of using GAs for intrusion detection can be dated back to 1995, when Crosbie et applied the multiple agent technology and GP to detect network anomalies. Each agent monitors one parameter of the network audit data and GP is used to find the set of agents that collectively determine anomalous network behaviors. This method has the advantage of using many small autonomous agents, but the communication among them is still a problem. Also the training process can be time-consuming if the agents are not appropriately initialized.
Bridges et al. develop a method that integrates fuzzy data mining techniques and genetic algorithms to detect both network misuses and anomalies. In most of the existing GA based IDSs, the quantitative features of network audit data are either ignored or simply treated, though such features are often involved in intrusion detection. This is because of the large cardinalities of quantitative features. The authors propose a way to include quantitative features by introducing fuzzy numerical functions. Their preliminary experiments show that the inclusion of quantitative features and the fuzzy functions significantly improved the accuracy of the generated rules. In this approach, a GA is used to find the optimal parameters of the fuzzy functions as well as to select the most relevant network features.
Lu et al.present an approach that uses GP to directly derive a set of classification rules from historical network data. The approach employs the support confidence framework as the fitness function and is able to generally detect or precisely classify network intrusions. However, the use of GP makes implementation more difficult and more data or time is required to train the system.
Li propose a GA-based method to detect anomalous network behaviors. Both quantitative and categorical features of network data are included when deriving classification rules using GA. The inclusion of quantitative features may lead to increased detection rates. However, no experimental results are available yet.
Xiao et al. present an approach that uses information theory and GA to detect abnormal network behaviors. Based on the mutual information between network features and the types of network intrusions, a small number of network features are closely identified with network attacks. Then a linear structure rule is derived using the selected features and a GA. The use of mutual information reduces the complexity of GA, and the single resulting linear rule makes intrusion detection efficient in real-time environment.

4 RESEARCH DIRECTIONS

Several directions for the future research which aim at discovering truly interesting FCRs are:

1)Reduction of number of Misclassifications

Though researchers have been able to discover compact and accurate classifiers, further experiments are needed to reduce the number of misclassifications as accuracy of the model depends on the misclassification rate (Roubous et al., 2001).

2)Improvement of Selection mechanism

Mansoori et al. (2008) has stated that there is needed to improve the selection mechanism as the non-random selection mechanism cannot utilize rules co-operation while producing offspring. Moreover, Mendes et al. (2001) have suggested that further experimentation must be carried out for designing "Intelligent" operator which selects tree nodes to be pruned on the basis of predictive power unlike "Blind" operator used for tree pruning which selects nodes randomly. Also, performance needs to be improved as current algorithm may not cope with data sets having large number of attributes

3)Fuzzification of other relational operators

As evolutionary system called RFCRD proposed by Akbarzadeh (2008) fuzzified only "greater than" and "less than" operator so fuzzification of other relational operators can be an interesting search. Another important research extension lies in the replacement of the GP algorithm with the Gene Expression Programming (GEP) algorithm. This substitution must be carried out because of difference in evolutionary strategies of both techniques. GEP allows evolution of multi-genic chromosomes i.e. each gene codes for a different sub-expression or a specific classification rule set. For instance, if a problem includes n classes i.e. parse tree must yield n classification rule sets than in case of GP, system must run n times. On the other hand, by using multi-genic GEP system, any given run would simultaneously evolve n classification rule sets.

4)Comparative study of results

Another significant future search would be comparative study of accuracy of FCRs evolved by various machine learning techniques with the results explored by other traditional rule induction and decision-tree-induction methods. Like in case of genetic system proposed by Romao et al. (2002) future direction lies in comparative study of degree of interestingness of rules discovered by GA and by another Data Mining algorithm.

5)Reducing time complexity of Evolutionary Algorithms

Evolutionary algorithms have been widely applied in data mining. However, use of genetic algorithms is somewhat restricted due to their typically large running time.

6)Use of filtering techniques

One of the very important research directions is to improve the efficiency of GAs by employing filtering techniques to reduce search space and augmenting GAs with cache like memory to reduce the number of fitness evaluations.

include footnotes in the abstract and avoid using a footnote in the first column of the article. This will cause it to appear of the affiliation box, making the layout look confusing

References

[1] D. B. Ekta Gandotra and S. Sofat, "Malware analysis and classification: a survey," Journal of Information Security, vol. 5, pp. 56–64, 2014. P. Wang and Y.-S. Wang, "Malware behavioural detection and vaccine development by using a support vector model classifier," Journal of Computer and System Sciences, vol. 81, no. 6, pp. 1012–1026, 2015. View at Publisher ·

[2] G. Ollmann, "The evolution of commercial malware development kits and colour-by-numbers custom malware," Computer Fraud and Security, vol. 2008, no. 9, pp. 4–7, 2008. View at Publisher · M. Ghiasi, A. Sami, and Z. Salehi, "Dynamic VSA: a framework for malware detection based on register contents," Engineering Applications of Artificial Intelligence, vol. 44, pp. 111–122, 2015. View at Publisher ·

[3] D. Bruschi, L. Martignoni, and M. Monga, "Detecting self-mutating malware using control-flow graph matching," in Detection of Intrusions and Malware & Vulnerability Assessment, R. Büschkes and P. Laskov, Eds., vol. 4064, pp. 129–143, Springer, Berlin, Germany, 2006

[4] M. R. Chouchane and A. Lakhotia, "Using engine signature to detect metamorphic malware," in Proceedings of the 4th ACM Workshop on Recurring Malcode, Alexandria, Va, USA, 2006.

[5] N. Kuzurin, A. Shokurov, N. Varnovsky, and V. Zakharov, "On the concept of software obfuscation in computer security," in Information Security, J. Garay, A. Lenstra, M. Mambo, and R. Peralta, Eds., vol. 4779, pp. 281–298, Springer, Berlin, Germany, 2007.

[6] M. Christodorescu and S. Jha, "Testing malware detectors," SIGSOFT Software Engineering Notes, vol. 29, no. 4, pp. 34–44, 2004. View at Publisher ·

[7] L. K. Mehedy Masud and B. Thuraisingham, Data Mining Tools for Malware Detection, vol. 1, CRC Press, 2012.

[8] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," ACM Computing Surveys, vol. 44, pp. 1–42, 2008.

[9] S. P. Monire Norouzi and A. Mahjur, "A new approach for formal behavioral modeling of protection services in antivirus systems," International Journal in Foundations of Computer Science & Technology, vol. 4, pp. 57–67, 2014.

A. Safarkhanlou, A. Souri, M. Norouzi, and S. E. H. Sardroud, "Formalizing and verification of an antivirus protection service using model checking," Procedia Computer Science, vol. 57, pp. 1324–1331, 2015

[10] Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, "Opcode sequences as representation of executables for data-mining-based unknown malware detection," Information Sciences, vol. 231, pp. 64–82, 2013.

[11] N. Abdelhamid, A. Ayesh, and F. Thabtah, "Phishing detection based associative classification data mining," Expert Systems with Applications, vol. 41, no. 13, pp. 5948–5959, 2014

[12] G. Jacob, H. Debar, and E. Filiol, "Behavioral detection of malware: from a survey towards an established taxonomy," Journal in Computer Virology, vol. 4, no. 3, pp. 251–266, 2008. View at Publisher ·

[13] S. B. Kotsiantis, "Supervised machine learning: a review of classification techniques," in Proceedings of the Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies, pp. 3–24, IOS Press, 2007.

[14] U. Bayer, A. Moser, C. Kruegel, and E. Kirda, "Dynamic analysis of malicious code," Journal in Computer Virology, vol. 2, no. 1, pp. 67–77, 2006. View at Publisher ·

[15] Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," Journal of Machine Learning Research, vol. 7, pp. 2721–2744, 2006.