Journal of Communications and Information Networks, Vol.2, No.4, Dec. 2017
DOI: 10.1007s41650-017-0033-7

Review paper

Feature selection techniques for intrusion detection using non-bio-inspired and bio-inspired optimization algorithms

Veeran Ranganathan Balasaraswathi*, Muthukumarasamy Sugumaran, Yasir Hamid

Department of Computer Science Engineering, Pondicherry Engineering College, Pondicherry University, Pondicherry 605014, India

* Corresponding author, Email: vrbala80@pec.edu

Abstract: As Internet access widens, IDS (Intrusion Detection System) is becoming a very important component of network security to prevent unauthorized use and misuse of data. An IDS routinely handles massive amounts of data traffic that contain redundant and irrelevant features, which impact the performance of the IDS negatively. Feature selection methods play an important role in eliminating unrelated and redundant features in IDS. Statistical analysis, neural networks, machine learning, data mining techniques, and support vector machine models are employed in some such methods. Good feature selection leads to better classification accuracy. Recently, bio-inspired optimization algorithms have been used for feature selection. This work provides a survey of feature selection techniques for IDS, including bio-inspired algorithms.

Keywords: IDS, feature selection, optimization, classification, bio-inspired algorithm

Citation: V. R. Balasaraswathi, M. Sugumaran, Y. Hamid. Feature selection techniques for intrusion detection using non-bio-inspired and bio-inspired optimization algorithms [J]. Journal of communications and information networks, 2017, 2(4): 107-119.

1 Introduction

In the late 1970's, the Internet came into existence as a result of ARPANET. The rapid advancement of the Internet has made it a potent platform for communication, business, entertainment, research, job researching, investing, and other uses. With the increase in Internet connectivity, however, there has been a corresponding increase in data breaches and security threats. Internet-based security attacks have multiplied in the last few years. Efforts to secure networks from destructive users have resulted in the development of protective software, including firewalls (designed to prevent intrusions) and IDS (to identify and operate against unauthorized data use in the case of an intrusion). IDSs are designed to test and analyze network traffic against a given set of parameters[1] to uncover potentially harmful network transactions. An IDS is a software application that examines all activities happening over the network and identifies suspicious patterns that may indicate that there has been a system or network attack by someone attempting to bypass the security mechanisms in place. An IDS can be classified in many ways, based on a collection of data, analysis of data, and actions needed to be taken. It can also be

Manuscript received Nov. 28, 2016; accepted Jun. 18, 2017

classified, based on the position of installation in the network, into two types: NIDS (Network-based IDS) and HIDS (Host-based IDS)[2,3]. An NIDS monitors and evaluates the individual packets passing through a network to detect malicious activities. An HIDS inspects such activities as login attempts, process scheduling, and system call tracing on an individual computer. Based on the detection methodology, an IDS can be classified as one of two types, namely misuse detection and anomaly detection[4]. Misuse detection is based on signatures, and is effective only in the detection of known attacks; it cannot detect unknown attacks. Anomaly detection is based on the behavior of an attacker when compared with that of a normal user. Unknown attacks can be detected by anomaly detection, but there will be high false positive rates. An IDS operates in different phases such as data collection, preprocessing, FS (Feature Selection), and classification. The FS phase is a challenging one, given that the IDS must deal with a huge amount of data. FS is the process of selecting significant features; only the selected features forming a subset of the total features are considered for classification. The major issue with an IDS is the significant computation overhead[5]. Accuracy becomes a primary concern[6], because IDS can identify a large variety of intrusions in real time. The foremost problem in designing an IDS is ranking and selecting the subset of highly discriminating features[7]. In recent years, FS methods and optimization techniques have received considerable attention for selecting the most significant features. A detailed survey of IDS can be found in our previous work[8]. To classify the dataset, FS plays an important role in selecting the most relevant and significant features[9]. Since classification is based on the class label, removal of redundant and unnecessary information is required[10]. The objective of this paper is to give an exhaustive survey of bio-inspired and non-bio-inspired techniques used for FS of IDS in the recent past. To the best of our knowledge, only a few review papers in the field are available. What differentiates this paper from existing papers is that the techniques are segregated into various groups. This survey helps researchers understand how bio-inspired and non-bio-inspired techniques have been effectively used in IDS.

The rest of the paper is organized as follows. Section 2 is a brief introduction to FS. Section 3 discusses FS using bio-inspired algorithms. Section 4 provides a brief discussion about non-bio-inspired techniques used for FS of IDS. Section 5 provides the validation methods and performance metrics of an IDS. Section 6 gives the conclusion.

2 Feature selection

In machine learning, FS is the process of selecting a subset of original features based on certain criteria. It is a significant and frequently used technique in many fields for dimension reduction. The dataset includes insignificant, redundant, and noisy features. FS is important in improving the efficiency as well as for reducing dimensions. This section will explain FS, different FS techniques and their differences, and the steps in the FS model.

2.1 Overview of FS

The FS problem involves identifying a subset of features[11]. An FS algorithm identifies and selects the subset of features that are relevant to the task to be learned. The classifier that is built with an efficient subset of relevant features gives better predictive accuracy than a classifier built from the complete set of features. Other advantages of FS include a reduction in the amount of training data needed, a process that is simpler and easier to understand, reduced computation time, and more accurate classification. Many researchers[12,13]have identified that the presence of irrelevant features may have a negative impact on the performance of learning systems. The predictive accuracy of a particular target concept might not be affected by the irrelevant features. Redundant features are those features that, although relevant to a target concept, provide information that is already provided by another feature and therefore do not contribute toward prediction.

Randomly class-correlated features are the features that are highly correlated with the target class. Irrelevant and redundant features are worthless; hence, removing them can improve the learning process. FS is the process of identifying and removing the inappropriate, redundant, and randomly class-correlated features[14].

2.2 FS techniques

The removal of irrelevant, redundant, and noisy features speeds up the algorithm and reduces the error rate. In general, there are three common methods for FS: wrapper, filter, and hybrid.

2.2.1 Wrapper method

The wrapper method uses a learning algorithm for evaluating the subset of features selected. The learning algorithm is used as a black box for the search[15]. The objective function, an equation with certain constraints to be optimized, is used to evaluate a subset of features with the help of their predictive accuracy, which is named detection rate. This is a feedback method based on two components: search and evaluation. The search component generates parameter settings that are then evaluated using the evaluation component[16]. It uses a learning algorithm to evaluate the usefulness of features and thus produces better feature subsets. The features are selected based on the accuracy of the classifier. The wrapper method is widely used, even though it is slower than others.

Forward selection and backward elimination are the two search strategies used in this method. The forward selection strategy begins with a null set of features and includes one feature at a time. For every iteration, the feature that affects the major increase of the evaluation function with respect to current set is added[17]. If the addition of new features does not show any improvement of the evaluation function, the search process is terminated. Backward elimination begins with the full feature set and removes features one at a time. In each iteration, if removal of any feature increases the evaluation function value, that corresponding feature is removed from the set. When a removal of feature results in a decrease of the evaluation function, the search stops.

2.2.2 Filter method

As the name indicates, the filter method filters insignificant features that have fewer options in the analysis of data. It does not use any learning algorithm to evaluate the features[18]. The selected features are evaluated based on the data characteristics such as distance, consistency, correlation, and information measures in the feature space. The filter method selects subsets with a greater number of features, sometimes even all the features, so an appropriate threshold is essential to choose a subset.

2.2.3 Hybrid method

The hybrid method is a combination of the wrapper and filter approaches to achieve better performance. It implicitly or explicitly uses the FS algorithm. Examples of this method are the decision tree and the naive Bayes classifier, among others. Tab. 1 gives the advantages and disadvantages of the abovementioned FS methods.

2.3 Feature selection model

All original features are given as input to the FS method, which generates a subset of features. The selected subset is evaluated by either using a learning algorithm or by considering the implicit characteristics of the data. The steps for selecting the subset of features are given below.
Steps: 1. Generate a subset of features from the given set of data. 2. Evaluate the subset based on criteria. 3. Verify whether the ensuing goals were met. This may be an exact boundary of features that leads to a better solution or satisfactory results based on the error rate. 4. If the goals are met, validate the result and terminate the process. Fig. 1 shows the steps of FS. The results can be compared if prior output knowledge is available. Other techniques, such as conducting experiments can be applied if prior knowledge is not available.

Table 1 Comparison of FS techniques

model	advantages	disadvantages	examples
filter	fast, scalable, independent of classifier, better computational complexity	ignores interaction with classifier	chisquare, Euclidean distance, information gain, correlation-based, Markov blanket filter, etc.
wrapper	simple, interacts with classifier, models feature dependencies, good classification accuracy, reduces computational cost	computationally intensive	sequential forward selection, sequential backward selection, hill climbing, genetic algorithms, etc.
hybrid	interacts with classifier, models with feature dependencies, better computational complexity	classifier dependent selection	decision tree, weighted Naive Bayes

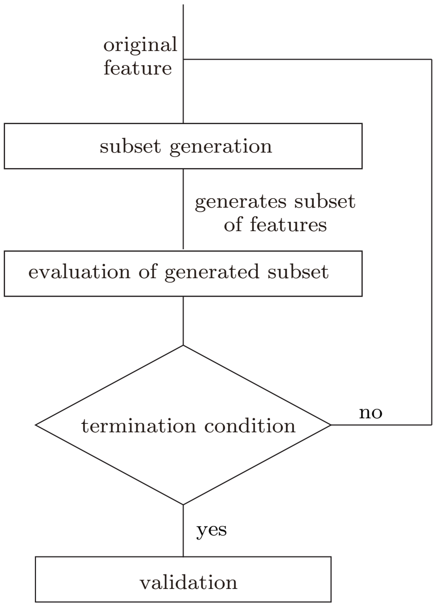

Figure 1 Steps in FS

3 FS based on bio-inspired algorithms

Given the fact that the volume of network traffic data is increasing swiftly, the need for more efficient FS methods has grown. The three above-mentioned FS techniques use complex calculations, which makes them relatively inefficient for large amounts of data. Bio-inspired algorithms are those algorithms that are inspired by nature. They solve complex problems using simple methods that exist in nature. Bio-inspired algorithms are classified into three classes: evolutionary, swarm-based, and ecology-based[19].

3.1 Evolutionary algorithms

Evolutionary algorithms are based on Darwinian principles of nature's ability to evolve living beings well adapted to their environment[20]. Evolutionary algorithms can be GA (Genetic Algorithms), GP (Genetic Programming), ES (Evolutionary Strategies), or evolutionary programming. They use procedures motivated by biological evolution, such as reproduction, mutation, recombination, and selection. A set of solutions plays the role of individuals in a population, and the fitness function evaluates the quality of the solutions. The evolution of the population happens after the repeated application of the mutation and crossover operators. Evolutionary algorithms often generate approximate solutions to all types of problems, and therefore, they are successfully applied to diverse fields such as engineering, biology, marketing, economics, art, operations research, physics, chemistry, social sciences, and genetics. Fig. 2 represents the steps to generate the optimized subset using an evolutionary algorithm.

3.1.1 Genetic programming

The GP proposed by Koza[21] in 1992 is a systematic, domain-independent method for computers to solve problems automatically. GP is an extension of GA, with the difference being that in GP, the individuals in the population are computer programs. The programs are transformed into the populations for each generation, and solutions to problems are generated

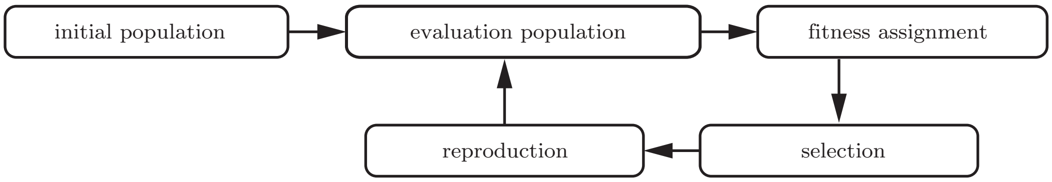

Figure 2 Flow diagram of evolutionary algorithm

in the form of parameterized topologies by applying crossover and mutation.

3.1.2 Genetic algorithm

The GA proposed by Holland in 1973[22] is an evolutionary algorithm with a global search. The GA is the most successful class of evolutionary algorithms inspired by the evolutionary facts of natural genetics. These algorithms generate primary individuals from a population; each individual represents a solution for the problem. GA solves problems in an easier manner since it is free of mathematical derivation. It also produces optimal solutions, making it highly suitable for FS. The genetic operators in GA are selection, crossover, and mutation.

3.1.3 Evolution strategies

ES are global optimization algorithms inspired by the theories of adaptation and evolution. They were developed by Bienert et al. (1964) at the Technical University in Berlin in 1964[23]. Specifically, this technique is inspired by the macro-level or species-level processes of evolution (phenotype, hereditary, variation). The main feature of ES is the employment of self-adaptive mechanisms for controlling the application of mutation.

3.2 Swarm-based algorithms

In an SBA (Swarm-Based Algorithm), the behavior of the system is composed of many individual swarms interacting with the environment and also locally with each other[24]. Self-organization and decentralized control are used by swarms to achieve their goals. Swarm-based systems are developed to resolve complex problems. Swarms are uncomplicated creatures with narrowly focused intellectual abilities and limited mode of communication. Swarms show intelligent behavior and provide significant solutions for complex problems such as finding the shortest path, the “knapsack problem”, and predator evasion.

3.2.1 Ant colony optimization

ACO (Ant Colony Optimization), introduced by Dorigo in 1990[25] is a nature-inspired metaheuristic algorithm that obtains good solutions for rigid combinatorial optimization problems with a reasonable computation time. The ants find the shortest routes due to their ability to deposit pheromones as they move; other ants follow, moving in the direction where the chemical content is richest. Since the pheromone decays over time, the less popular paths end up with less of the pheromone. The traversal rate by ants is more in the shortest path[26].

3.2.2 Particle swarm optimization

Kennedy and Eberhart (1995) proposed another bio-inspired algorithm called PSO (Particle Swarm Optimization). In this model, a fitness function is used that measures the quality of the current solution. It uses the metaphor of the grouping behavior of birds to solve optimization problems. In PSO, more entities (particles) are stochastically created in the search space. The particles flutter through the problem space by succeeding the current optimum particles. All particles have fitness values that are evaluated by the fitness function to be optimized and have velocities that direct the flying of the particles. A swarm consists of particles flying around in a search space. Every particle swarm has a certain kind of topology describing the interconnections among the particles. The key features of PSO are speed and inherent ability to find the best solution globally.

The speed of each particle is calculated using the findings of both that particle and rest of the swarm. The global best solution is conveyed among all the particles of the swarm[27].

3.2.3 Bee colony algorithm

The BC (Bee Colony) algorithm is based on the behavior of the bees in nature and is classified as foraging behavior and mating behavior. Proposed by Karaboga and Basturk (2007), BC simulates the intelligent foraging behavior of a honey bee. A BC contains three groups: worker bees, onlookers, and scouts. Onlooker bees wait in the dance region to make a decision about selecting a food source. Worker bees visit the food source. Scout bees perform a random search to determine new food sources. The position of a food source corresponds to a potential solution to the optimization problem, and the nectar quantity of a food source matches the class of the associated solution. A swarm of virtual bees is produced and begins to move arbitrarily in two-dimensional search space. Bees act together when they locate some target nectar and the solution is attained from the intensity of bee interactions[28].

3.2.4 Fish swarm algorithm

Li et al. (2002)[29] proposed an innovative swarm intelligent algorithm inspired by the natural schooling behavior of fish called FSA (Fish Swarm Algorithm). FSA possesses a powerful ability to keep away from local minimums to attain global optimization. FSA replicates three distinctive behaviors: searching, swarming, and following. Searching is a random search for food, with an inclination toward food concentration. Swarming tries to satisfy food intake needs, engage swarm members, and attract new swarm members. In the following, neighboring individuals follow the fish that locates the food. The parameters involved in FSA are the visual distance (visual), maximum step length (step), and a crowd factor. FSA effectiveness is primarily influenced by the visual and step parameters.

3.2.5 Firefly algorithm

FA (Firefly Algorithm) is an unusual swarm-based heuristic algorithm inspired by the blinking behavior of fireflies[30]. A population-based iterative algorithm with numerous agents (perceived as fireflies), FA concurrently solves the optimization problem. Fireflies communicate with each other through bioluminescent glowing that helps them discover cost-function space more efficiently. This technique is based on the theory that solution of an optimization problem can be perceived as agents (fireflies) that glow proportionally to their excellence in a measured problem setting. Consequently, the brighter fireflies attract more partners, which makes the search easier and more efficient.

3.3 Ecology-based algorithms

Difficult engineering and computer science problems can be solved by the mechanisms of natural ecosystems. EBA (Ecology-Based Algorithms) are composed of populations of individuals wherein each population grows according to an optimization strategy. Hence, individuals of each population are modified according to the mechanisms of intensification and diversification, with initial parameters specific to each optimization strategy. The ecology-based system can be formed in two ways, namely homogeneous and heterogeneous. In the homogeneous model, all populations are evolved according to the same optimization strategy, configured with the same parameters. In the heterogeneous model, optimization strategies or parameters can be changed. The ecological inspiration stems from the use of some ecological concepts, such as habitats, ecological relationships, and ecological successions. Once detached in the search space, populations of individuals recognized in the same region form an ecological habitat. A hypersurface may have numerous habitats. Populations can move around through the environment. However, each population may belong to only one habitat at a given moment of time[31].

Table 2 Evolutionary algorithms used for FS of IDS

S.No	reference	technique	dataset	feature length	classifier	measures
1	[32]	GA	KDD CUP'99	32	decision tree	–
2	[33]	GA	KDD CUP'99	27	multiobjective genetic fuzzy rule classifier	AC: 99.24
3	[34]	PCA + GA	KDD CUP'99	22	SVM	DR: 99.60
4	[35]	PCA + GA	KDD CUP'99	12	multilayer perception	AC: 99.00
5	[36]	neural network+ GA	KDD CUP'99	16	decision tree	DR: 98.38
6	[37]	PCA + GA	KDD CUP'99	10	SVM	DR: 99.51
7	[38]	hybrid kernel principal component analysis+ GA	KDD CUP'99	12	multilayer SVM classifier	DR: 94.22
8	[39]	hybrid method of GA and SVM	KDD CUP'99	10	SVM	true positive value: 0.97, false positive value: 0.01

3.4 Review of works applying bio-inspired techniques

This section provides a review of bio-inspired techniques used for FS of IDS. For each work, the FS technique used, the dataset considered, the resulting reduced feature length, the base classifier employed, and the performance metrics taken have been recorded.

3.4.1 Works applying evolutionary algorithms

A comparison of evolutionary algorithms based FS for IDS is given in Tab. 2.

3.4.2 Works applying swarm based algorithms

ACO, PSO, the bee algorithm, and the cuttlefish algorithm were used for FS of IDS. ACO is used more widely than the other techniques. To improve the performance, SBAs are combined with other algorithm types to create hybrid forms. In terms of detection rate, the ACO and cuttlefish algorithms produced a better result. The error rate is lower in CFA and the bee algorithm. The works using SBA are shown in Tab. 3.

4 Review of works applying non-bio-inspired techniques

A comparison of nonbio-inspired FS techniques used for IDS is given in Tab. 4.

As Tab. 4 suggests, different combinations can provide different advantages. The information theory concept combined with correlation-based FS algorithm eliminates irrelevant features. Correlation analysis calculates the strength of feature to class and feature to feature. PCA is used to find the significant features from the high-dimensional vectors of input data. PCA with support vector machine improves accuracy and also generalization ability of the classifiers. LDA (Linear Discriminant Analysis) is also used to select significant features. Mutual information concept combined with binary gravitational search algorithm will select relevant features; this hybrid results in selecting the optimal subset of features and also increases the classification performance. The rough-set-based FS approach, when used, also provides problem simplification and faster and more accurate detection rates. A combination of PCNN (Pulse-Coupled Neural Network) algorithm and Gaussian support vector machine gives better generalization ability and also successfully detects intrusion. Using the entropy-based FS model with layered classifier improves the preci-

Table 3 SBA used for FS of IDS

S.No	reference	technique	dataset	feature length	classifier	measures
1	[40]	PSO +support vector machine	KDD CUP'99	41	support vector machine	DR: 99.84
2	[41]	particle swarm optimization	KDD CUP'99	12	support vector machine	AC: 94.49
3	[42]	bees algorithm	KDD CUP'99	8	support vector machine	DR: 95.75
4	[43]	swarm based rough set	KDD CUP'99	6	SSO with weighted local search	AC: 93.60
5	[44]	bee algorithm using membrane computing	KDD CUP'99	41	support vector machine	DR: 89.11AC: 95.6
6	[45]	ant colony optimization + feature weighting support vector machine	KDD CUP'99	–	support vector machine	DR: 98.38 False Alarm Rate: 0.004
7	[46]	cuttle fish algorithm	NSL KDD	–	C5.0+one class SVM	–AC: 98.20 False Alarm Rate: 1.70
8	[47]	cuttle fish algorithm	KDD CUP'99	10	decision tree	DR: 92.05
9	[48]	ant colony optimization	KDD CUP'99	14	support vector machine	TPR: 98.00
10	[49]	ant colony optimization	NSL KDD and KDD CUP'99	8	ACO	AC: 98.90 FPR: 2.59
11	[50]	bat algorithm	KDD CUP'99	–	support vector machine	DR: 92.94 Detection time: 0.43 s
12	[51]	artificial bee colony optimization	KDD CUP'99	–	feed-forward neural network classifier	–

sion value. Many non-bio-inspired techniques such as mutual information, rough-set theory, clustering approach, and effect-based, PCA, LDA, PCNN, SVM, and correlation-based techniques have been applied to NSL and KDD CUP'99 datasets. Among all the techniques, correlation-based technique improves the accuracy rate the most, whereas the entropy-based model increases the detection rate and precision value. PCA reduces the computational time.

5 Validation and performance measures

This section describes validation methods and performance metrics for an IDS.

5.1 Validation methods

The efficiency of the selected set of features has been evaluated with the help of a test set that was not known to the FS method. In a few cases, the data were distributed into training and test sets. The training set was used to carry out the FS process and the test set was used to calculate the aptness of the selection. All the datasets were not originally partitioned; in many cases, to partition the data set, several validation techniques were used. To check whether the classifier produced the expected output, the test set was validated against the training set. Validation methods included k-fold cross-validation, leave-one-outcross-validation, bootstrap validation, and holdout validation[69]. In k-fold cross-validation methods, the dataset was split into k subsets, where $k-1$ subsets were used for training and the remaining partition was spared for testing. This process was repeated until all the sets were tested. Leave-one-out was alternated with k-fold with k as the number of samples. A single instance was left out every time. In bootstrap, the number of samples was drawn equally with replacements from the original data; thus, some samples might appear multiple times, whereas others might not have appeared at all. Holdout validation randomly splits the available data, with two-thirds for training and one-third for testing used as a common partition.

Table 4 Non-bio-inspired FS techniques used for IDS

S.No	Ref.	technique	dataset	feature length	classifier	features	results
1	[52]	principal component analysis(hybrid method)	KDD CUP'99	–	support vector machine	improves generalization ability of classifiers	training time: 33.3 s, testing time: 14.4 s
2	[53]	information theory concept+ correlation based FS algorithm	KDD CUP'99	41	fuzzy belief KNN	provides superior performance	computation time: 0.25 s
3	[54]	linear discriminant analysis	KDD CUP'99	–	–	detection rate is good	FPR: 3.7
4	[55]	gradually feature removal method	KDD CUP'99	19	support vector machine	improves accuracy	AC: 98.62
5	[56]	pulse-coupled neural networks	KDD CUP'99	–	Gaussian kernel of support vector machine	reduces feature subset, increase accuracy	FPR: less than 1
6	[57]	Naive Bayes classifier+Kruskal Wallis statistical test	KDD CUP'99	13	C4.5	classifies significant and relevant feature	AC: 99.95
7	[58]	intelligent agent-based attribute selection algorithm	KDD CUP'99	10	support vector machine	1. minimizes computation time 2. reduces false positive rate	AC (DOS: 99.11 Probe: 96.16 U2R: 96.00 R2L: 96.00)
8	[59]	filter based	KDD CUP'99, NSL-KDD dataset, Kyoto 2006	25, 28, 15 SVM	improves performance	AC: 99.94	
9	[60]	intraclass and interclass correlation coefficient-based FS algorithm	KDD CUP'99	–	–	1. reduces false alarm rate 2. reduces execution time	–
10	[61]	correlation algorithm+ redundancy algorithm	KDD CUP'99	–	Naive Bayes classifier	reduces time	AC: 94
11	[62]	attribute ratio	NSL-KDD	22	decision tree	improves accuracy	AC: 99.79
12	[63]	rough set based	DD CUP'99	24	–	1. simplifies the problem 2. maximises accuracy	–
13	[64]	mutual information concept+binary gravitational search algorithm	NSL-KDD	5	support vector machine	1. selects optimal subset of features 2. increases the classification performance	AC: 88.36
14	[65]	entropy based FS	KDD CUP'99	DoS: 17, probe: 21, R2L: 14	layered classifier	improves recall value	DR: 99.16 FPR: 0.74, Recall Value: 99.03
15	[66]	enhanced correlation based model	KDD CUP'99	12	–	1. speeds up detection process 2. high performance rate	AC (Probe: 99.98 DOS: 99.3 R2L: 98.10 U2R: 72.20)
16	[67]	conditional random field based	KDD CUP'99	41	layered approach based algorithm	reduces false alarm rate	AC (Probe: 98.83 DOS: 97.62 R2L: 32.43 U2R: 86.91)
17	[68]	random forest	KDD CUP'99	25	random forest	performance is improved	precision (DoS: 98.94 Probe: 55.33 R2L-66.11 U2R: 17.14)

Table 5 Performance measures

S.No	metric	definition	formula
1	DR	the ratio between a total number of instances detected by the system and the total number of instances present in the dataset	-
2	TN	percentage of negative samples classified correctly as negative	-
3	FN	percentage of positive samples incorrectly classified as negative	-
4	TP	percentage of positive samples classified correctly as positive	-
5	FP	percentage of negative samples incorrectly classified as positive	-
6	sensitivity or TPR	the proportion of positive samples correctly classified over a total number of positive samples	TP/(TP+ FN).
7	specificity or TNR	proportion of negative samples incorrectly classified as negative over the total number of negative samples	TN/(TN+FP)
8	AC	accuracy measures how an IDS works correctly. It is the proportion of the total number of the correct predictions to the actual dataset size	(TN+TP)/ (TN+TP+FN+FP)
9	misclassification rate	percentage of input samples that are misclassified	(FN+FP)/(TP+FP+FN+TN)
10	FPR	the proportion of normal instances incorrectly classified as attack over the total number of normal instances	FP/FP+TN
11	P	the proportion of attack instances that were correctly predicted relative to the predicted size of the attack class	TP/(TP+FP)
12	confusion matrix	compares the actual class labels with predicted ones	-
13	ROC	ROC is a graph that illustrates the relation between TPR and FPR	-.

5.2 Performance metrics

A good performance metric should have a quantitative basis, allow accurate and detailed comparisons, and lead to a correct conclusion. The metric set that has been used to evaluate IDS are AC (Accuracy), DR (Detection Rate), TN (True Negative), FN (False Negative), TPR (True Positive Rate), FPR (False Positive Rate), sensitivity, specificity, error rate, recall, P (Precision), and ROC (Receiver Operating Characteristics)[70]. Tab. 5 gives the performance measures used for IDS evaluation.

6 Conclusion

This paper provides an overview of different FS methodologies and approaches for an IDS. Each technique was applied to the same data and measured using the same metrics. Some techniques produced better results for certain metrics. No single technique ranked as being the best in every metric or produced overall a better result. As this study suggests, there is good reason for the trend in recent years toward hybrid methods. The best results seem to be derived from bio-inspired algorithms combined with data mining techniques, soft computing techniques, neural networks, rough set theory, and fuzzy logic. The hybridization of bio-inspired algorithms with other techniques improves performance and provides effective problem solving.

References

[1] Heasman, John, S. Movle. Intrusion detection system [M]. U.S. Patent Application No. 10/511, 775, 2003.

[2] D. Anderson, T. Frivold, A. Valdes. Next-generation intrusion detection expert system(nides): a summary [C]//SRI International Computer Science Laboratory Menio Park, CA, 1995.

[3] R. Martin. Snort: lightweight intrusion detection for networks [J]. LISA, 1999, 99 (1): 229-238.

[4] H. Debar, M. Dacier, A. Wespi. Towards taxonomy of intrusion detection systems [J]. Computer networks 1999, 31(9), 805-822.

[5] H. Debar, M. Dacier, A. Wespi. A revised taxonomy for intrusion-detection system [J]. Annals of telecommuni-

cations, 2000, 55: 361-378.

[6] P. Garca-Teodoroa, J. Daz-Verdejoa, G. Maci-Fernndez, et al. Anomaly-based network intrusion detection: techniques, systems and challenges[J]. Computers and security, 2009, 28(1): 18-28.

[7] T. F. Lunt. A survey of intrusion detection techniques [J]. Computers and security, 1993, 12(4): 405-418.

[8] Y. Hamid, M. Sugumaran, V. R. Balasaraswathi. IDS using machine learning-current state of art and future directions [J]. British journal of applied science and technology, 2016, 15(3).

[9] S. Chebrolu, A. Abraham, J. P. Thomas. Feature deduction and ensemble design of intrusion detection systems [J]. Computers and security, 2005, 24(4): 295-307.

[10] D. Manoranjan, H. Liu. Feature selection for classification [J]. Intelligent data analysis, 1997, 1(3): 131-156.

[11] G. H. John, R. Kohavi, K. Pfleger. Irrelevant features and the subset selection problem [C]//Proceedings of ML-94, 1994: 121-129.

[12] J. Frank. Artificial intelligence and intrusion detection: Current and future directions [C]//Proceedings of the 17th National Computer Security Conference, 1994, 10: 1-12.

[13] W. Lee, S. J. Stolfo. A framework for constructing features and models for intrusion detection systems [C]//ACM Transactions on Information and System Security, 2000, 3(4): 227-261.

[14] K. Kira, L. A. Rendell. The feature selection problem: traditional methods and a new algorithm [M]. AAAI-92.2, MIT Press, 1992: 129-134.

[15] B. Kumar, T. Swarnkar. Filter versus wrapper feature subset selection in large dimensionality micro array: a review [J]. International journal of computer science and information technologies, 2011, 2(3): 1048-1053.

[16] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos. A review of feature selection methods on synthetic data [J]. Knowledge and information system, 2013, 34(3): 483-519.

[17] S Maldonado, R Weber, F Famili, et al. Feature selection for high-dimensional class-imbalanced data sets using support vector machines [J]. Information sciences, 2014, 286: 228-246.

[18] S. Binitha, S. S. Sathya. A survey of bio inspired optimization algorithms [J]. International journal of soft computing and engineering, 2012, 2(2): 137-151.

[19] D. Karaboga, B. Akay. A comparative study of artificial bee colony algorithm [J]. Applied mathematics and computation, 2009, 214(1): 108-132.

[20] T. Back. Evolutionary algorithms in theory and practice [M]. Oxford: Oxford University Press, 1996.

[21] J. R. Koza. Genetic programming: on the programming of computers by means of natural selection (Vol. 1) [M]. Cambridge: MIT press, 1992.

[22] J. H. Holland. Genetic algorithms and the optimal allocation of trials [J]. SIAM journal on computing, 1973, 2(2): 88-105.

[23] H. G. Beyer, H. P. Sshwefel. Evolution strategies [J]. Natural computing, 2002: 3-52.

[24] E. Bonabeau, M. Dorigo, G. Theraulaz. Swarm intelligence [M]. Oxford: Oxford University Press, 1999.

[25] M. Dorigo, C. Blum. Ant colony optimization theory: a survey [J]. Theoretical computer science, 2005, 344(2): 243-278.

[26] M. Dorigo, V. Maniezzo, A. Colorn. Ant system: optimization by a colony of cooperating agents [C]//IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 1996, 26(1): 29-41.

[27] R. Poli, J. Kennedy, T. Blackwell. Particle swarm optimization [J]. Swarm intelligence, 2007, 1(1): 33-57.

[28] D. Karaboga, B. Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm [J]. Journal of global optimization, 2007, 39(3): 459-471.

[29] X. L. Li, Z. J. Shao, J. X. Qian. An optimizing method based on autonomous animats: fish-swarm algorithm [J]. System engineering theory and practice, 2002, 22(11): 32-38.

[30] X. S. Yang. Fire fly algorithm for multimodal optimization [D]. Cambridge: International Symposium on Stochastic Algorithms, 2009: 169-178.

[31] S. Goings, H. Goldsby, B. H. C. Cheng, et al. An ecology-based evolutionary algorithm to evolve solutions to complex problems [J]. Artificial life, 2012, 13: 171-177.

[32] G. Stein, B. Chen, A. S. Wu, et al. Decision tree classifier for network intrusion detection with GA-based feature selection [C]//Proceedings of the 43rd Annual Southeast Regional Conference. 2005, 2: 136-141.

[33] C. H. Tsang. Network-based anomaly intrusion detection using ant colony clustering model and genetic-fuzzy rule mining approach [D]. Hong Kong: City University of Hong Kong, 2006.

[34] I. Ahmad, A. B. Abdulla, A. S. Alghamdi, et al. Intrusion detection using feature subset selection based on MLP [J]. Scientific research and essays, 2011, 24(7-8): 1671-1682.

[35] I. Ahmad, A. B. Abdulah, A. S. Alghamdi, et al. Feature subset selection for network intrusion detection mechanism [C]//Proceedings of CSIT, 2011, 5.

[36] S. S. S. Sindhu, S. Geetha, A. Kannan, et al. Decision tree based light weight intrusion detection using a wrapper approach [J]. Expert systems with applications, 2012, 39(1): 129-141.

[37] I. Ahmad, M. Hussain, A. Alghamdi, et al. Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components [J]. Neural computing and applications, 2014, 6(34): 6804-6810.

[38] F. J. Kuang, W. H. Xu, S. Zhang. A novel hybrid KPCA and SVM with GA model for intrusion detection [J]. Ap-

plied soft computing, 2014, 18, 178-184.

[39] B. M. Aslahi-Shahri, R. Rahmani, M. Chizari, et al. A hybrid method consisting of GA and SVM for intrusion detection system [J]. Neural computing and applications, 2015, 1-8.

[40] J. Wang, X. Hong, R. R. Ren, et al. A real-time intrusion detection system based on PSO-SVM [C]//Proceedings of the International Workshop on Information Security and Application, 2009: 319-321.

[41] T. Jiang, H. Gu. Anomaly detection combining one-class SVMs and particle swarm optimization algorithms [J]. Nonlinear dynamics, 2010, 61(1-2): 303-310.

[42] A. Osama, Z. A. Othman. Bees algorithm for feature selection in network anomaly detection [J]. Journal of applied sciences research, 2012, 8(3): 1748-1756.

[43] C. Chung, Y. Ying, N. Wahid. A hybrid network intrusion detection system using simplified swarm optimization (SSO) [J]. Applied soft computing, 2012, 12(9): 3014-3022.

[44] K. I. Rufai, R. C. Muniyandi, Z. A. Othman. Improving bee algorithm based feature selection in intrusion detection system using membrane computing [J]. Journal of networks, 2014, 9(3): 523-529.

[45] X. Z. Wang. ACO and SVM selection feature weighting of network [J]. International Journal of Security and Its applications, 2015, 9(4): 129-270.

[46] M. S. Rani, S. B. Xavier. A hybrid intrusion detection system based on C5.0 decision tree and one-Class SVM [J]. International journal of current engineering and technology, 2015, 5(3): 2001-2007.

[47] A. S. Eesa, Z. Orman, A. M. A. Brifcani. A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems [J]. Expert systems with applications, 2015, 42(5): 2670-2679.

[48] T. Mehmod, H. B. M. Rais. Ant colony optimization and feature selection for intrusion detection [J]. Advances in machine learning and signal processing, 2016, 305-312.

[49] M. H. Aghdam, P. Kabiri. Feature selection for intrusion detection system using ant colony optimisation [J]. International journal of network security, 2016, 18(3): 420-432.

[50] C. Y. Cheng, L. Y. Bao, C. H. Bao. Network intrusion detection with bat algorithm for synchronization of feature selection and support vector machines [C]//International Symposium on Neural Networks. Springer International Publishing Switzerland, 2016: 401-408.

[51] W. A. H. M. Ghanem, A. Jantan. Novel multi-objective artificial bee colony optimization for wrapper based feature selection in intrusion detection [J]. International journal of advance soft computing applications, 2016, 8(1).

[52] X. Xu, X. N. Wang. An adaptive network intrusion detection method based on pca and support vector machines [C]//ADMA, 2005: 696-703.

[53] T. S. Chou, K. K. Yen, J. Luo. Network intrusion detection design using feature selection of soft computing paradigms [J]. International journal of computational intelligence, 2008, 4(3): 196-208.

[54] Z. Y. Tan, A. Jamdagni, X. He. et al. Network intrusion detection based on LDA for payload feature selection [C]//2010 IEEE Globecom Workshops, 2010: 1545-1549.

[55] Y. H. Li, J. B. Xia, S. L. Zhang, et al. An efficient intrusion detection system based on support vector machines and gradually feature removal method [J]. Expert systems with applications, 2012, 39(1): 424-430.

[56] A. Shrivastava, M. Baghel, H. Gupta. A novel hybrid feature selection and intrusion detection based on pcnn and support vector machine [J]. International journal of computer technology and applications, 2013, 4 (6), 922.

[57] P. Louvieris, N. Clewley, X. Liu. Effects-based feature identification for network intrusion detection [J]. Neurocomputing, 2013, 121, 265-273.

[58] S. Balakrishnan, K. Venkatalakshmi, A. Kannan. A intrusion detection system using feature selection and classification technique [J]. International journal of computer science and application (IJCSA), 2016, 3(4).

[59] M. Ambusaidi, X. J. He, P. Nanda, et al. Building an intrusion detection system using a filter-based feature selection algorithm [C]//IEEE transactions on computers, 2014, 65(10): 2986-2998.

[60] A. R. Vasudevan, S. Selvakumar. Intraclass and interclass correlation coefficient-based feature selection in NIDS dataset [J]. Security and communication networks, 2015, 8(18): 3441-3458.

[61] C. Y. Yin, L. Y. Ma, L. Feng, et al. A feature selection algorithm towards efficient intrusion [J]. International journal of multimedia and ubiquitous engineering, 2015, 10(11): 253-264.

[62] H. S. Chae, B. O. Jo, S. H. Choi, et al. Feature selection for intrusion detection using NSL-KDD [J]. Recent advances in computer science, 2015, 960-978.

[63] V. Rampure, A. Tiwari. A rough set based feature selection on KDD CUP 99 data set [J]. International journal of database theory and application, 2015, 8(1): 149-156.

[64] H. Bostani, M. Sheikhan. Hybrid of binary gravitational search algorithm and mutual information for feature selection in intrusion detection systems [J]. Soft computing, 2015, 1-18.

[65] S. Ramakrishnan, S. Devaraju. Attack's feature selection-based network intrusion detection using fuzzy control language [J]. International journal of fuzzy systems, 2016, 1-13.

[66] A. I. Madbouly, T. M. Barakat. Enhanced relevant feature selection model for intrusion detection systems [J]. International journal intelligent engineering informatics, 2016, 4(1): 21-45.

[67] S. Ganapathy, P. Vijayakumar, et al. An intelligent CRF based feature selection for effective intrusion detection [J]. The international arab journal of information technology, 2016, 16(2).

[68] S. Rastegari, P. Hingston, C. P. Lam. Evolving statistical rulesets for network intrusion detection [J]. Applied soft computing, 2015, 33: 348-359.

[69] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita. Network anomaly detection: methods, systems and tools [J]. IEEE communications surveys and tutorials, 2014, 16(1): 303-336.

[70] C. H. Tsang, S. Kwong. Multi-agent intrusion detection system in industrial network using ant colony clustering approach and unsupervised feature extraction [C]//IEEE International Conference on Industrial Technology, 2005: 51-56.

About the authors

Veeran Ranganathan Balasaraswathi [corresponding author] received her B.E. degree in computer science and engineering from Madras University in 2002 and M.E. degree in computer science and engineering in 2008 from Anna University, Chennai. She is now a Ph.D. candidate of Pondicherry Engineering College, India. Her research interests include information security, data mining, data structures and algorithms. (Email: vrbala80@pec.edu)

Muthukumarasamy Sugumaran received his M.Sc. degree in mathematics from University of Madras in 1986 and M.Tech. degree in computer science and data processing from Indian Institute of Technology, Karagpur in 1991 and obtained his Ph.D. degree from Anna University, Chennai in 2008. He is currently working as a professor and head of computer science at Pondicherry Engineering College, India. His areas of interests are theoretical computer science, analysis of algorithms, parallel and distributed computing and spatial temporal data. (Email: sugu@pec.edu)

Yasir Hamid received his Master's degree in computer applications from University of Kashmir in 2014. He is now a Ph.D. candidate of Pondicherry Engineering College, India. His research interests include machine learning, network security, non linear dimension reduction and data visualization. (Email: bhatyasirhamid@pec.edu)