Liu et al. EURASIP Journal on Wireless Communications and Networking (2020) 2020:4
https://doi.org/10.1186/s13638-019-1567-1

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Entropy clustering-based granular classifiers for network intrusion detection

Hui Liu[1,2], Gang Hao[3*] and Bin Xing[2]

Abstract

Support vector machine (SVM) is one of the effective classifiers in the field of network intrusion detection; however, some important information related to classification might be lost in the reprocessing. In this paper, we propose a granular classifier based on entropy clustering method and support vector machine to overcome this limitation. The overall design of classifier is realized with the aid of if-then rules that consists of a premise part and conclusion part. The premise part realized by the entropy clustering method is used here to address the problem of a possible curse of dimensionality, while the conclusion part realized by support vector machines is utilized to build local models. In contrast to the conventional SVM, the proposed entropy clustering-based granular classifiers (ECGC) can be regarded as an entropy-based support function machine. Moreover, an opposition-based genetic algorithm is proposed to optimize the design parameters of the granular classifiers. Experimental results show the effectiveness of the ECGC when compared with some classical models reported in the literatures.

Keywords: Entropy clustering-based granular classifiers (ECGC), Entropy clustering method, Support vector machine (SVM), Genetic Algorithms (GAs)

1 Introduction

In the past decades, lots of techniques such as artificial intelligence and mathematical methods have been applied for many applications [1–5]. With the effectiveness in high-dimensional spaces, support vector machine (SVM) becomes one of the most important classification models when solving the problem of classification.

Many researchers have utilized the SVM for solving the classification problem in the field of network intrusion detection. Chitrakar and Huang [6] have presented the selection of candidate support vectors in incremental SVM for network intrusion detection. Shams et al. [7] have used trust aware SVM when dealing with the network intrusion detection problems. Aburomman and Reaz [8] have proposed a novel-weighted SVM multiclass classifier for the intrusion detection system. Yaseen et al. [9] have constructed multi-level hybrid SVM by means of K-means for network intrusion detection. Vijayanand et al. [10] have developed genetic-algorithm-based feature selection in the design of SVM for solving the network intrusion detection. Raman et al. [11] have proposed an efficient intrusion detection system with the aid of genetic algorithm optimized SVM. All these studies have developed SVM based on genetic algorithms or clustering methods; however, a design of SVM with both clustering methods and genetic algorithms remains open.

The entropy clustering method (ECM) [12] is a novel clustering method based on the concept of entropy that has been widely used in network intrusion detection. In comparison with the conventional clustering method such as K-means and C-Means, the ECM can obtain the number of clustering once the features of dataset are determined.

In the design of classification models, we require some crucial parameters for determining the structure. As one of the powerful optimization tools [13–17], genetic algorithms have been applied in lots of applications. In some previous studies [10], genetic algorithms have been successfully applied to optimize the support vector models. However, it should be stressed that the genetic algorithm could still get trapped in sub-optimal regions of the search space. Furthermore, the problem of finding "good" parameters in the design of the rule-based classification models remains open.

* Correspondence: Ganghaoabc@126.com
[3]School of Computer Science and Engineering, Tianjin University of Technology, Beijing, China
Full list of author information is available at the end of the article

In this study, we propose a rule-based granular classifier by means of entropy clustering method and support vector machine for network intrusion detection. The overall granular classifier is designed by means of a serial of rules that consist of a premise part and conclusion part. The premise part is realized by the entropy clustering method, while the conclusion part is realized with the aid of the support vector machine. In some senses, the proposed entropy clustering-based granular classifiers (ECGC) can be regarded as an entropy-based support function machine. Furthermore, an opposite-based genetic algorithm (OGA) is proposed to optimize the parameters of the granular classifier.

The structure of the paper is organized as follows. Section 2 presents the design of ECGC. Section 3 deals with the opposite-based genetic algorithm and the optimization of ECGC. Section 4 reports on experiments by using comparative studies. Finally, some conclusions are summarized in Section 5.

A design of the ECGCs

In the design of ECGC, the overall classification is divided into a number of rules that consist of a premise part and conclusion part. The premise part of rules determined by the entropy clustering method is to capture "rough, major structure"; while the conclusion part (local model) realized based on SVM is to capture "subtle, accurate structure." In this way, we construct ECGC. Such rule-based classifiers can be expressed with some "if-then" rules

$$\mathbf{R}^i : IF\ \mathbf{x}\ is\ in\ cluster\ C_i\ then\ y_i = f_{\mathrm{i}}(\mathbf{x}) \tag{1}$$

where ${}_{\mathrm{R}}{}^{R^i}$ represents the ith rule, C_i denotes the ith cluster, i=1, ···, n, n is equal to the number of rules, $\mathrm{f}_i(\mathbf{x})$ denotes the consequent output of the ith rule, and pattern classifiers are described by means of some discriminant functions $\mathrm{f}_i(\mathbf{x})$. An overall design of ECGC is described as shown in Fig.1.

1.1 Realization of premise part of rules using entropy clustering method

In the design of ECGC, the premise part of rules is formed by the entropy clustering method. Let $G = (V, D)$ be an undirected graph, where V denotes the vertex set and D stands for the edge set. The steps of entropy clustering method can be summarized as the following steps:

[Step 1] Calculate the entropy rate $E(P)$ by using the following expression.

$$E(P) = -\sum_i u_i \sum_j h_{ij}(P)\ \log\left(h_{ij}(P)\right). \tag{2}$$

where $E(P)$ represents ethe ntropy rate, which quantifies the uncertainty of a random process $P = \{P_t |\ t \in T\}$. Here, T denotes some index set.

[Step 2] Calculate the balancing term $B(P)$ by using the following formula.

$$B(P) = -\sum_i \frac{|\ S_i\ |}{|\ V\ |}\ \log\left(\frac{|\ S_i\ |}{|\ V\ |}\right) - N_P, i = \{1, ..., N_P\} \tag{3}$$

where N_P represents the number of connected components in the graph, $S_P = \{S_1, S_2, ..., S_{N_P}\}$ which means the graph partitioning for P.

[Step 3] Set $E \leftarrow \varnothing$ and $U \leftarrow D$.

[Step 4] Set $a1 \leftarrow \mathrm{argmax}_{a \in U}\ F(P \cup \{a\}) - F(P)$, where $F(P) = E(P) + kB(P)$, k denotes the number of clusters.

[Step 5] If $P \cup \{a1\} \in I$ then set $P \leftarrow P \cup \{a1\}$; else set $\mathrm{U} \leftarrow \mathrm{U} \cup \{a1\}$.

[Step 6] Repeat steps 4–5 until $\mathrm{U} = \varnothing$.

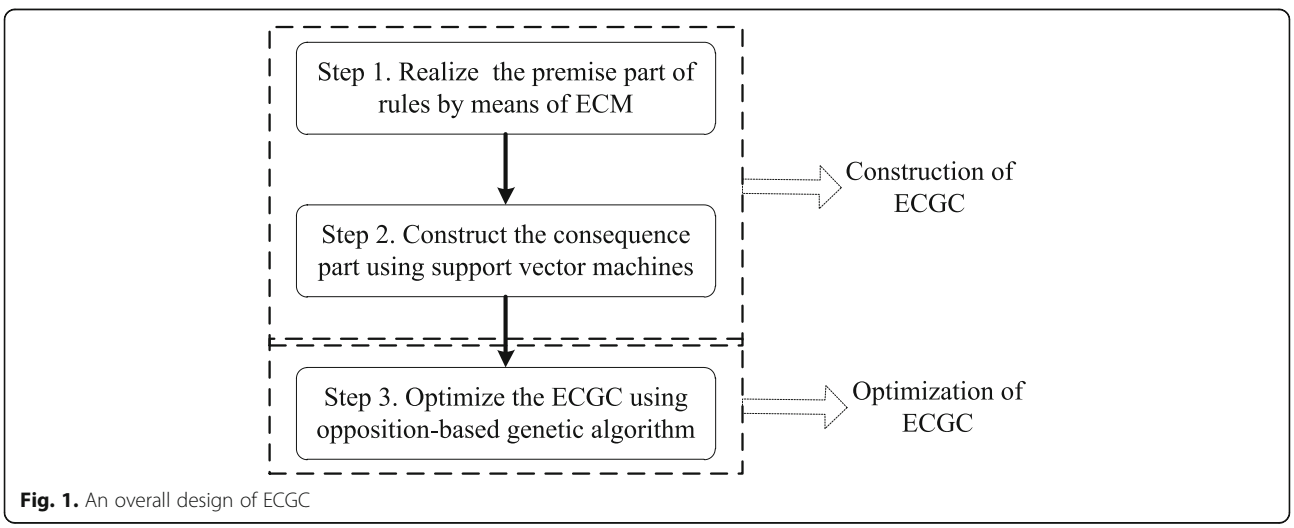

Fig. 1. An overall design of ECGC

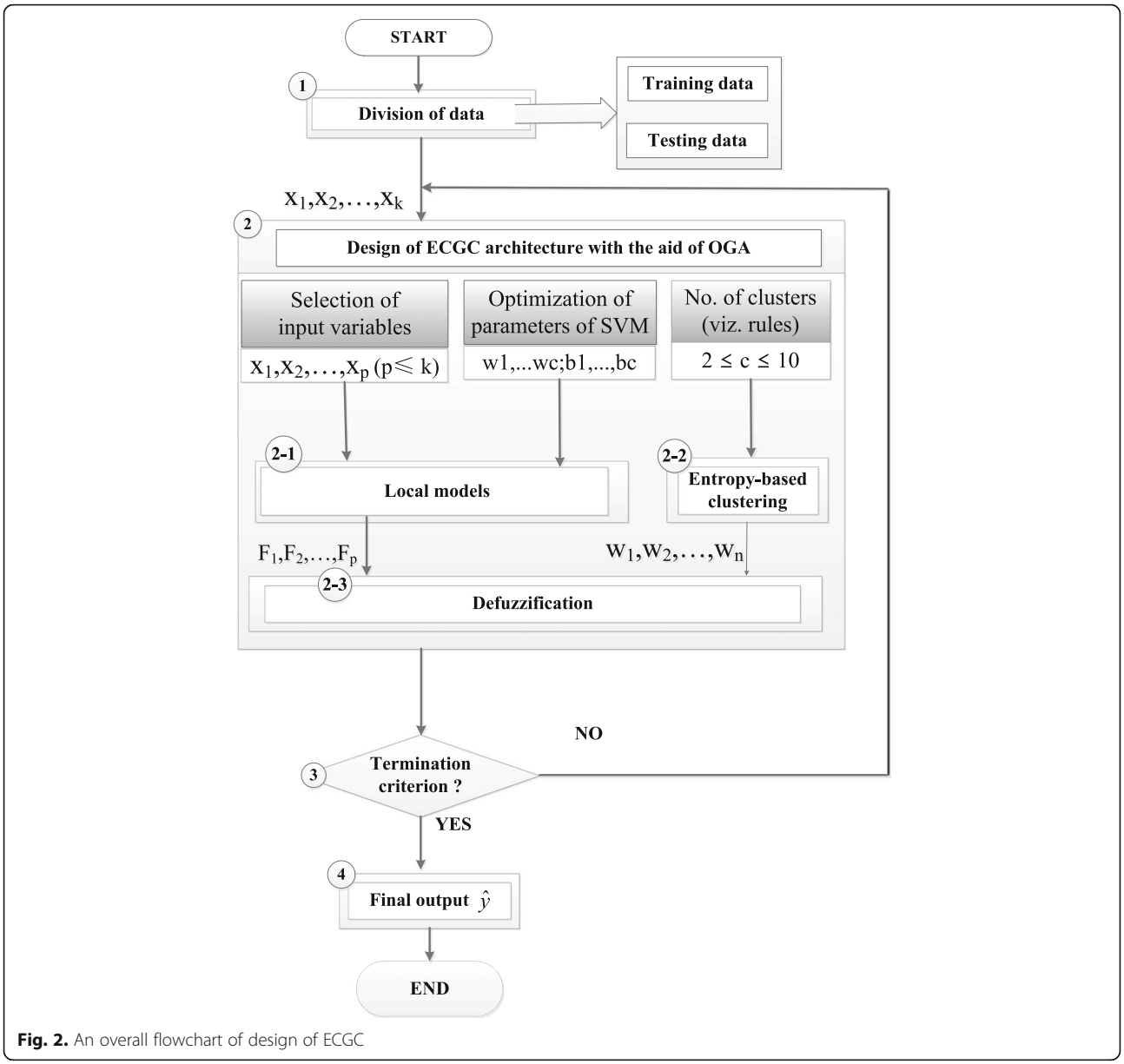

Fig. 2. An overall flowchart of design of ECGC

1.2 Construction of conclusion part of rules using support vector machines

The conclusion part of the rules is realized by means of support vector machines [12]. Assume that the training dataset T is formed by the following expression:

$$T = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m), ..., (x_n, y_n)\} \quad (4)$$

Where x_m represents a training sample, $y_m \in \{-1, 1\}$ is the class label of x_m, and n stands for the total number of training samples, $m = 1, 2, ..., n$.

Suppose that w and b are parameters of hyperplane functions that can be expressed as follows:

$$f(x) = wx + b \quad (5)$$

Then, optimal value of w and b can be calculated by the following model:

$$\begin{aligned} &\min_p \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_m y_j p_m p_j K(x_m, x_j) - \sum_{k=1}^{n} p_k \\ &s.t. \quad \sum_{l=1}^{n} y_l p_l = 0, \\ &\quad 0 \le p_l \le C. \end{aligned} \quad (6)$$

where $p = (p_1, ..., p_n)^T$ denotes the Lagrange multiplier vector, C represents a penalty parameter, K stands for a kernel function, and x_m, x_j, y_m, y_j are the mth input sample, the jth input sample, label of the mth input sample,

and label of the jth input label, respectively. The steps of the support vector machine can be summarized as the following steps:

[Step 1] Divide the entire dataset into the training dataset and testing dataset.

[Step 2] Estimating the parameters w and b based on the expression (6).

[Step 3] Calculate the decision function of the support vector machine according to the expression (5).

[Step 4] Calculate the labels of testing data based on the decision function.

[Step 5] Obtain the classification results based on training and testing data.

1.3 Optimization of ECGC using opposition-based genetic algorithms

Like the other classification models, the performance of ECGC is dramatically affected by the parameters. Here we present an opposition-based genetic algorithm as the vehicle for the optimization of parameters in the design of ECGC. The mechanism of opposition-based learning (OBL) [18, 19] has been shown to be an effective concept to enhance various optimization approaches. Let us recall the basic concept.

Opposition-based point [19]: let $P = (x_1, x_2, ..., x_D)$ be a point in a D-dimensional space, where $x_1, x_2, ..., x_D \in R$ and $x_i \in [a_i, b_i]$, $\forall i \in \{1, 2, ..., D\}$. The opposite point $P\cup = (x\cup_1, x\cup_2, ..., x\cup_D)$ is completely defined by its components

$$\breve{x}_i = a_i + b_i - x_i \tag{7}$$

Opposition-based optimization (OBL) [19]: let $P = (x_1, x_2, ..., x_D)$ be a point in a D-dimensional space (i.e., a candidate solution). Assume $f(\bullet)$ is a fitness function. According to the definition of the opposite point, we say that $P\cup = (x\cup_1, x\cup_2, ..., x\cup_D)$ is the opposite of site $P = (x_1, x_2, ..., x_D)$. Now, if $f(P\cup) \geq f(P)$, then the point P can be replaced with $P\cup$. Hence, the point and its opposite point are evaluated simultaneously in order to continue with one of the highest fitness.

With the opposition concept, we develop the opposition-based genetic operator. The overall opposition-based genetic algorithm can be summarized as follows:

[Step 1] Randomly generate the population of genetic algorithm, where the performance of ECGC is the objective function, the parameters in the design of ECGS are considered as chromosome.

[Step 2] Update the population based on *opposition-based population operator*.

[Step 2.1] Find the interval boundaries $[a_i, b_i]$ in the population set P1, where $a_j = \min(x_j^k), b_j = \max(x_j^k)$, $j = 1, 2, ..., h$; $k = 1, 2, ..., d$. Here, h denotes the size of population, and d represents the dimension of an individual.

[Step 2.2] For each individual, generate a new individual $X^{\text{new}} = (x_1^{\text{new}}, ..., x_j^{\text{new}}, ..., x_n^{\text{new}})$ based on the expression $x_j^{\text{new}} = a_j + b_j - x_j$.

[Step 2.3] Obtain the opposition population set P2 by calculating the fitness value of each X^{new}.

[Step 2.4] Obtain the final population P^{new} by selecting the best h individuals based on the P1 ∪ P2.

[Step 3] Generate the new individual based on crossover.

[Step 4] Generate the new individual based on mutation.

[Step 5] Generate the new individual based on *opposition-based genetic operator*.

[Step 5.1] Find the interval boundaries $[a_i, b_i]$ in the population set P1, where $a_j = \min(x_j^k), b_j = \max(x_j^k)$, $j = 1, 2, ..., h$; $k = 1, 2, ..., d$. Here, h denotes the size of the population, and d represents the dimension of an individual.

[Step 5.2] For each individual, generate a new individual $X^{\text{new}} = (x_1^{\text{new}}, ..., x_j^{\text{new}}, ..., x_n^{\text{new}})$ based on the expression $x_j^{\text{new}} = a_j + b_j - x_j$.

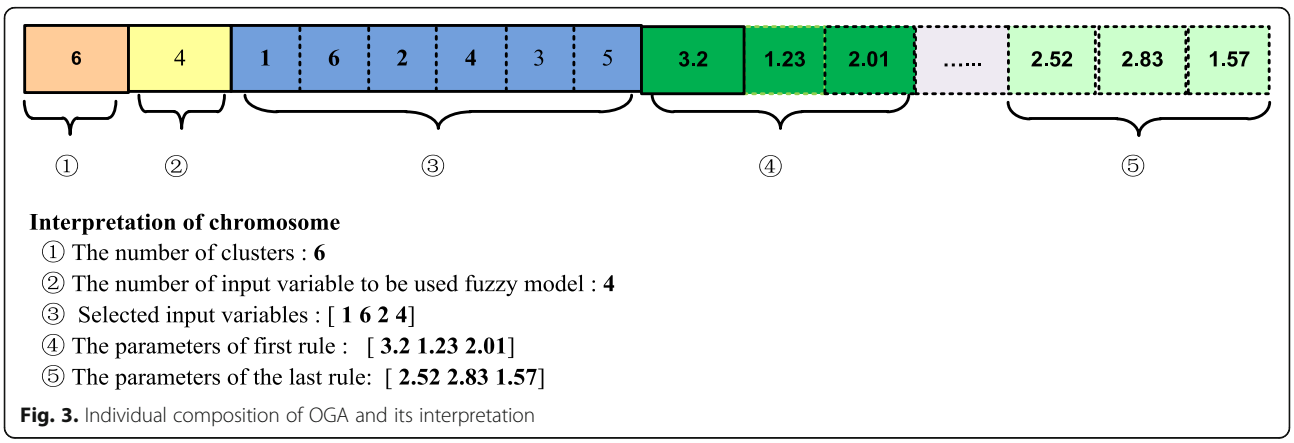

Fig. 3. Individual composition of OGA and its interpretation

Table 1 List of the parameter of the OGA

OGA parameters	
Number of generations	100
Number of individuals	20
Crossover rate	0.4
Mutation rate	0.1
[a1]Number of input variables	2~10
Number of clusters (rules)	2~10
[b1]Value of *w*	[0.5**w*0,1.5**w*0]
[c1]Value of *b*	[0.5**b*0,1.5**b*0]

*[a1]If the number of input variables in dataset smaller than 10, then we use the number of all input variables
*[b1], [c1] *w*0 and *b*0 are the results obtained by the original SVM

[Step 6] Select the new individual in the current population.

[Step 7] Repeat steps 3–6 until the terminal condition is satisfied.

2 A design procedure of the ECGCs

The overall design methodology of entropy-based clustering granular classification is described in this section. The design of ECGC can be summarized in the following steps see (Fig. 2).

2.1 Step 1: Division of dataset

The original data is divided into training and testing datasets. Training data is used to construct the model of ECGC, while the testing data is utilized to evaluate the performance of ECGC. Suppose that the original input–output dataset is denoted as (xi,yi) = (x1i, x2i, …, xni, yi), i = 1, 2, …, N, where N is the number of data points. Let T be the number of correct classification patterns. The classification rate (CR) can be represented as follows

$$CR = \frac{T}{N} \times 100\% \tag{8}$$

Furthermore, let TR be the classification rate for the training data, and TE be the classification rate for the testing data. It is evident that TR records the objective function (viz. performance index, PI) and TE stands for the testing performance index (TPI).

2.2 Step 2: Design of ECGC architecture with the aid of OGA

The overall design of ECGC can be regarded as the construction of rules that comprises the premise part and conclusion part. Here, the premise part is realized based on the entropy-clustering method, while the conclusion part is realized by means of SVMs. OGA is used here to optimize the parameters not only in the entropy-based clustering method but also in the design of SVMs. Specifically, In the ECGC, an individual is denoted as a vector comprising the number of clusters, the number of selected input variables, the input variable to be selected, and the parameters for each rule as shown in Fig. 3. The overall length of the individual corresponds to the number of clusters (viz. rules) to be used.

2.3 Step 3: Check the termination criterion

As to the termination criterion, we have used two different conditions. The first condition is that the number of loops is not more than a predetermined number, while

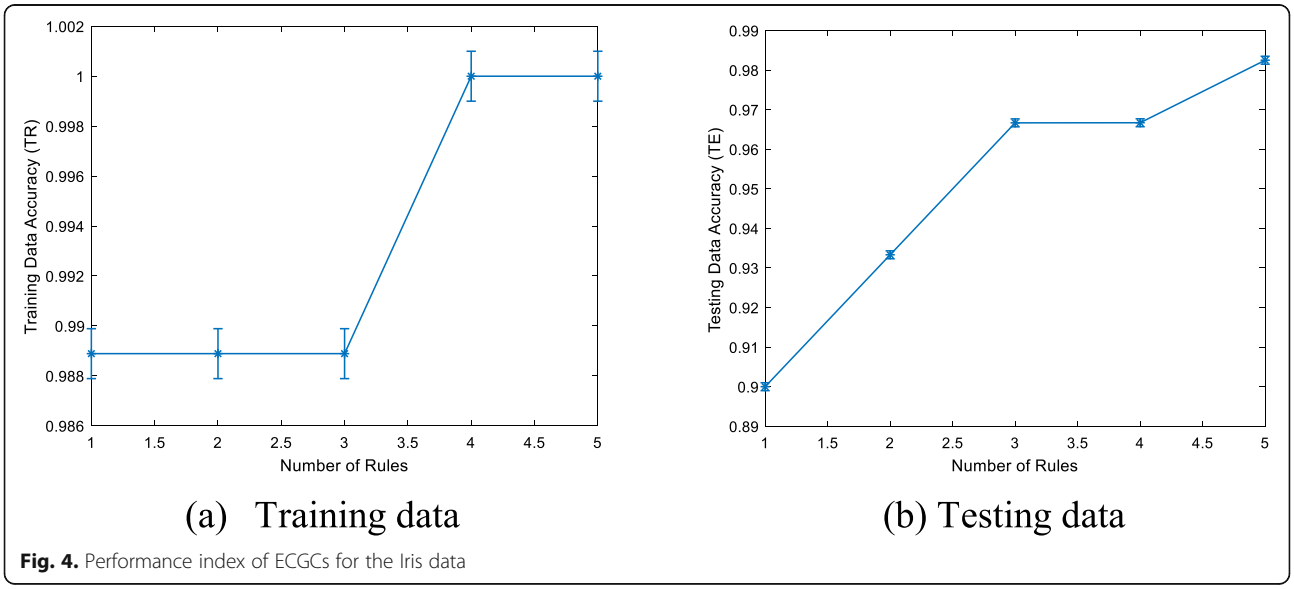

Fig. 4. Performance index of ECGCs for the Iris data

the second condition is that the performance of the current local model is worse than a predetermined value [20]. It has stressed that the final optimal ECGC has been experimentally determined based on a sound compromise between the high accuracy and the low complexity of models.

2.4 Step 4: Final output

Report the optimal ECGC and final output.

3 Results and discussion

This section reports the experimental results of the proposed ECGC models. To evaluate the performance of the ECGC, we first experimented some benchmark machine learning data [21–26], and then applied the ECGC in the network intrusion detection KDDCUP 99 data. The symbols used in these experiments are listed as follows: TR denotes the performance index of training data, while TE represents the performance index of testing data. Furthermore, the parameters and boundaries of OGA are summarized as shown in Table 1. (The selection of these specific values of parameters is referred to reference [10, 27]).

3.1 Machine learning data

Some machine learning data are used to evaluate the performance of the proposed ECGC. In these experiments, datasets are partitioned into two parts: 80% of data is considered as training data, while the rest 20% of data is regarded as testing data.

3.1.1 Iris data

The iris flower dataset is a multivariate dataset introduced by Sir Ronald Fisher as an example of discriminant analysis. This is a classical dataset consisting of

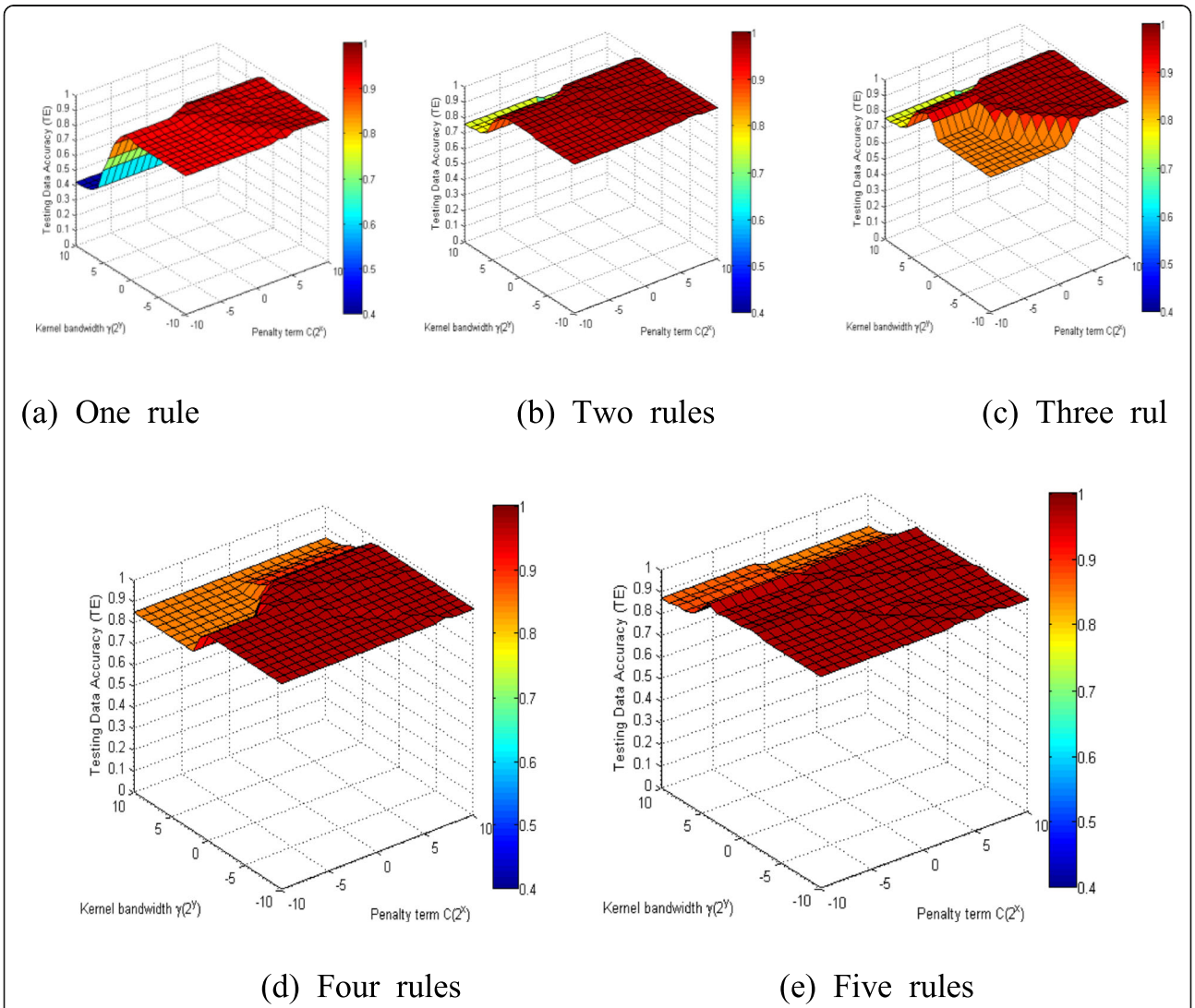

Fig. 5. Performance index (TE) range from one rule to five rules for the Iris data

Table 2 Comparison of classification rate with previous classifiers (Iris data)

Classifier	TR	TE
MLP [21]	–	66.4
KN N[22]	–	94.6
TS-KN N[22]	–	96.7
PF C[23]	–	93.3
SV M[24]		95.6
Our classifier (rule = 5)	98.42 ± 0.62	98.25 ± 0.21

150 input-output pairs, four input variables, and three classes.

Figure 4 depicts the values of the performance index (TR and TE) vis-à-vis the ECGCs with the increasing rules. As shown in Fig. 4, the value of the performance index for training data TR is increased with the increasing prediction abilities of GCs. It is clear that the optimal classifier could have emerged with the layer of assigned rules (clusters). The testing performance TE increases in the case of two rules, while it becomes the same as the rules are equal to five. This tendency illustrates that the substantial increase of the rules improves the prediction abilities.

Figure 5 Displays the values of performance index TE range from one rule to five rules for the Iris data when selecting different parameters (penalty term and Kernel bandwidth of conclusion part). In most cases, the value of performance index TE raises with the increasing number of rules. This tendency demonstrates that the number of rules is beneficial to the enhanced TE.

Table 2 summarizes the experimental results. It is shown that the proposed ECGCs arrive at 98.25 ± 0.21 with five rules.

3.1.2 Some selected machine learning data

Five selected machine learning data are further used to evaluate the performance of the proposed ECGC model.

Here, the selected data with different number of input and variables are summarized as illustrated in Table 3.

Table 4 further shows the comparative results of the proposed ECGC and some well-known machine learning models. As shown in Table 4, the proposed ECGC outperforms the better accuracy of classification as well as the prediction when compared to the models reported in the literatures.

Table 3 Description of five selected machine learning data

Datasets	Variables	Patterns	Classes
Glass	7	9	185
E-coli	8	7	336
Ionosphere	6	10	350
Diabetes	2	8	768
Banana	2	2	5300

3.2 KDDCUP99 data

In the field of network intrusion detection [27–29], some datasets can be obtained to evaluate the performance of models. To evaluate the performance of ECGC, here we experiment the ECGC on the benchmark KDDCUP99 data.

The KDDCUP99 data has 5,000,000 labeled records (viz. patterns) and 41 features (viz. input variables) provided by the Massachusetts Institute of Technology. This dataset consists of 24 different types of attacks that are divided into four groups: DDOS, Probe, U2R, and R2L. According to some studies [30–32], the filtered 10% KDDCUP99 data described as shown in Table 5 is used when dealing with lots of network intrusion detection issues. In the experiments, the dataset is partitioned into two parts: 50% of data is utilized as training data and the remaining 50% data is considered as testing data. Moreover, in order to compare with other models, we also used the existing performance index [27–32]:

True positive (TP). A TP represents one correct detection of an attack of network intrusion detection;

False positive (FP). A FP denotes an indication of an attack on traffic that should have been classified as "normal";

True negative (TN). A TN stands for one correct classification of "normal Traffic" of network intrusion detection;

False negative (FN). A FN is written as a real attack that was misidentified as "Normal" traffic;

Table 4 Description of five selected machine learning data

Data sets	Glass data	E-coli data	Lonosphere data	Diabetes data	Banana data
LDA [25]	77.62	88.12	Null	76.60	76.60
SVM+LD A[25]	79.17	88.29	Null	76.58	53.03
SV M[24]	74.81 ± 0.64	87.28 ± 0.36	95.71 ± 0.02	76.76 ± 0.08	89.40 ± 0.02
KN N[22]	72.00	*Null	Null	Null	Null
TS-KN N[22]	80.40	Null	Null	Null	Null
The proposed ECGC	1.1. 84.26 ± 1.60	1.2. 90.77 ± 0.08	1.3. 96.60 ± 0.24	1.4. 78.12 ± 0.23	1.5. 89.56 ± 0.12

*Null represents the results of the model are unknown

Table 5 Distribution for the 10% KDD Cup 99 dataset

Class	Traffic type	Training data	Percent of total (%)
1	Normal	87,832	60.28
2	DDOS	54,572	37.51
3	Probe	2130	1.48
4	U2R	999	0.68
5	R2L	52	0.04
	Total	145,585	

Accuracy. Accuracy is the common metric used for assessing the overall effectiveness of a classifier.

The expression of Accuracy can be formulated as follows:

$$Accuracy = (TP + TN)/(FP + FN + TP + TN) \quad (9)$$

The experimental results of ECGC are compared with the results of several well-known models reported in the literatures as shown in Table 6. It is evident that the proposed ECGC outperforms the cited approaches in case of consistently good detection across all four types of attack classes.

4 Conclusions

In the reprocessing, conventional support vector machines have some inevitable limitations. One fact is that some important information related to classification might be lost. In this study, we have proposed ECGC to overcome this limitation. In the design of ECGC, SVMs are explored here as local models that are considered as the consequence part of rules, while the premise part of rules is realized with the aid of entropy-based clustering method. Genetic algorithm is utilized to optimize the parameters when constructing the ECGC. It is evident that the proposed ECGC can be regarded as the extended SVMs to some extent. Experimental results on several well-known datasets demonstrate the effectiveness of the ECGC, especially for the network intrusion detection dataset. More importantly, with the proposed ECGC,

Table 6 Comparison results for the 10% KDD Cup 99 dataset

Classifier	Testing samples	TP/FP	DDoS	Probe	U2R	R2L	Mean TP/FP ratio
FNT [27]	11,982	TP	98.75	98.39	99.7	99.09	203.84
		FP	0.62	1.39	0.22	0.75	
ID3 [28]	311,029	TP	99.52	97.85	49.21	92.75	756.66
		FP	0.04	0.55	0.14	10.03	
Naïve Bayes [29]	77,287	TP	79.2	94.8	12.2	0.1	209.01
		FP	1.7	13.3	0.9	0.3	
SVM [30]	15,437	TP	96.32	63.81	34.48	85.64	40.00
		FP	1.63	0.94	0.05	4.4	
J48 [31]	15,437	TP	96.8	75.2	12.2	0.1	148.75
		FP	1	0.2	0.1	0.5	
K-means [32]	311,029	TP	97.3	87.6	29.8	6.4	103.86
		FP	0.4	2.6	0.4	0.1	
SOM [31]	15,437	TP	96.4	74.3	13.3	0.1	125.35
		FP	0.8	0.3	0.1	0.4	
GAU [32]	311,029	TP	82.4	90.2	22.8	9.6	60.28
		FP	0.9	11.3	0.5	0.1	
OneR [32]	49,596	TP	94.2	12.9	10.7	10.7	63.80
		FP	6.8	0.1	2	0.1	
Bayes DT combo [29]	311,029	TP	87.9	76.23	12.33	30.6	44.68
		FP	0.67	1.7	8.9	23.8	
NBTree [31]	15,437	TP	97.4	73.3	1.2	0.1	40.00
		FP	1.2	1.1	0.1	0.5	
Naïve Baye s[31]	15,437	TP	79.2	94.8	12.2	0.1	16.90
		FP	1.7	13.3	0.9	0.3	
The proposed ECGC	15,437	TP	99.98	97.85	100	97.57	3055
		FP	0.01	0.05	0	0.89	

one can efficiently construct the optimal model (viz. optimization of the parameters in the design of model), which is the key issue to improve the performance when constructing models.

For future studies, new optimization algorithm can be included. By taking into account new optimization algorithm, one can obtain optimized ECGC. Furthermore, several objectives can be considered to construct ECGC, one can also develop multiobjective optimized ECGC.

5 Methods/experimental

This study aims at the design of classification for network intrusion detection. A granular classifier based on entropy-clustering method and supported vector machine is constructed to overcome the shortcoming that most of the conventional classifiers such as SVM may lose some important information in the reprocessing. The proposed granular classifier that is designed by means of a serial of rules can also be regarded as an entropy-based support function machine. Experiments illustrate that the performance of the granular classifier obtains "good" results in comparison with some well-known classifiers.

It has to be stressed that, granular classifiers can further improve the performance with the aid of opposite-based genetic algorithm. Experimental results show that the performance of granular classifier can be generally improved. Also, the number of rules is quite effective in the final performance of the granular classifiers. Generally, with the growth of rules, the performance of granular classifiers is gradually increasing while its complex is rising.

Abbreviations

CR: Classification rate; ECGC: Entropy clustering-based granular classifiers; ECM: Entropy clustering method; FN: False negative; FP: False positive; GA: Genetic algorithms; OBL: Opposition-based learning; OGA: Opposition-based genetic algorithm; PI: Performance index; SVM: Support vector machine; TE: Classification rate for testing data; TN: True negative; TP: True positive; TPI: Testing performance index; TR: Classification rate for training data

Authors' contributions

All authors contribute to the concept, the design and developments of the theory analysis and algorithm, and the simulation results in this manuscript. All authors read and approved the final manuscript.

Funding

This work was not supported by any fund.

Availability of data and materials

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Competing interest

The authors declare that they have no competing interests.

Author details

[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China. [2]Beijing Institute of Information Application Technology, Beijing, China. [3]School of Computer Science and Engineering, Tianjin University of Technology, Beijing, China.

Received: 24 May 2019 Accepted: 23 September 2019
Published online: 03 January 2020

References

1. P. Huijse, P.A. Estevez, P. Protopapas, J.C. Principe, P. Zegers, Computational intelligence challenges and applications on large scale astronomical time series databases. IEEE Computational Intelligence Mag. **9**(3), 27–39 (2014)
2. W. Huang, J. Wang, The shortest path problem on a time-dependent network with mixed uncertainty of randomness and fuzziness. IEEE Transac Intelligent Trans Syst. **17**(11), 3194–3204 (2016)
3. L. Wang, H. Zhen, X. Fang, S. Wan, W. Ding, Y. Guo, A unified two-parallel-branch deep neural network for joint gland contour and segmentation learning. Future Gen Comp Syst. **100**(316-324) (2019)
4. Q. Xu, L. Wang, X.H. Hei, P. Shen, W. Shi, L. Shan, Gl/Geom/1 queue based on communication model for mesh networks. Int J Comm Syst. **27**(11), 3013–3029 (2014)
5. W. Huang, L. Ding, The Shortest Path Problem on a Fuzzy Time-Dependent Network. IEEE Transac Comm. **60**(11), 3376–3385 (2012)
6. R. Chitrakar, C. Huang, Selection of candidate support vectors in incremental SVM for network intrusion detection. Comp Sec. **45**, 231–241 (2014)
7. E.A. Shams, A. Rizaner, A.H. Ulusoy, Trust aware support vector machine intrusion detection and prevention system in vehicular ad hoc networks. Comp Sec. **78**, 245–254 (2018)
8. A.A. Aburromman, M.B.I. Reaz, A novel weighted support vector machines multiclass classifier based on differential evolution for intrusion detection systems. Info Sci. **414**, 225–246 (2017)
9. W. L. A. Yaseen, Z. A. Othman, M. Z. A. nazri, Expert Systems with Applications. Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection systems. 67, 296-303 (2017).
10. R. Vijayanand, D. Devaraj, B. Kannapiran, Comp Sec.. Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection 77, 304-314 (2018).
11. M.R.G. Raman, N. Somu, K. Kirthivasan, R. Liscano, V.S.S. Sriram, knowledge-based systems. An efficient intrusion detection system based on hypergraph genetic algorithm for parameter optimization and feature selection in support vector machine. 134, 1-12 (2017).
12. M.Y. Liu, O. Tuzel, S. Ramalingam, R. Chellappa, Entropy-rate clustering: cluster analysis via maximizing a submodular function subject to a matroid constraint. IEEE Transac Pattern Anal Machine Intell. **36**(1), 99–111 (2014)
13. R. Zhang, P. Xie, C. Wang, G. Liu, S. Wan, classifying transportation mode and speed from trajectory data via deep multi-scale learning. Computer Networks. **162**, 1–13 (2019)
14. W. Huang, S.K. Oh, W. Pedrycz, Hybrid fuzzy wavelet neural networks architecture based on polynomial neural networks and fuzzy set/relation inference-based wavelet neurons. IEEE Transac Neural Networks Learn Syst. **29**(8), 3452–3462 (2018)
15. W. Li, X. Liu, J. Liu, P. Chen, S. Wan, X. Cui, On improving the accuracy with auto-encoder on conjunctivitis. App Soft Computing. **81**, 1–11 (2019)
16. A.R. Solis, G. Panoutsos, Interval type-2 radial basis function neural networks: a modeling framework. IEEE Transac Fuzzy Syst. **23**, 457–473 (2015)
17. W. Huang, L. Ding, Project-Scheduling problem with random time-dependent activity duration times. IEEE Transac Eng Management. **58**(2), 377–387 (2011)
18. W. Huang, S.K. Oh, Z. Guo, W. Pedrycz, A space search optimization algorithm with accelerated convergence strategies. App Soft Computing. **13**, 4659–4675 (2013)
19. S. Rahnamayan, H.R. Tizhoosh, M.A. Salama, Opposition-Based differential evolution. IEEE Transac Evol Comp. **12**(1), 64–79 (2008)
20. W. Huang, S.K. Oh, W. Pedrycz, IEEE Transac Fuzzy Syst Fuzzy Wavelet Neural Networks: Analysis and Design. 25(5), 3452-3462, 1329-1341 (2017).
21. M.E. Tipping, Adv. Neural Inf. Process. Syst. The relevance vector machine. 12, 652-658 (2000).
22. M.A. Tahir, A. Bouridane, F. Kurugollu, Simultaneous feature selection and feature weighting using hybrid tabu search/K-nearest neighbor classifier. Pattern Recog Letters. **28**(4), 438–446 (2007)
23. J.P. Mei, L. Chen, Fuzzy clustering with weighted methods for relational data. Pattern Recog. **43**(5), 1964–1974 (2010)
24. V. Vapnik, Spring-Verlag. The nature of statistical learning theory. (1995).
25. T. Xiong and V. Cherkassy, Proceedings of the International Joint Conference on Neural Networks. A combined SVM and LDA approach for classification. 1455-1459 (2005).

26. Wei Huang, Sung-Kwun Oh, Witold Pedrycz, Neural Networks. Design of hybrid radial basis function neural networks (HRBFNNs) realized with the aid of hybridization of fuzzy clustering method (FCM) and polynomial neural networks (PNNs). 60, 166-181 (2014).
27. V. Jaiganesh, S. Mangayarkarasi, P. Sumathi, An efficient algorithm for network intrusion detection system. Int J Comp Applications. **90**(12), 12–16 (2014)
28. W. Wei, X.L. Yang, P.Y. Shen, B. Zhou, Holes detection in anisotropic sensornets: Topological methods. Int J Distr Sensor Networks. **8**(10), 1–10 (2012)
29. X. Wang, Z. Zhang, J. Li, Y. Wang, H. Cao, Z. Li, L. Shan, An optimized encoding algorithm for systematic polar codes. EURASIP J Wireless Comm Network. **193**, 1–12 (2010)
30. S.J. Jang, C.H. Han, K.E. Lee, S.J. Yoo, Reinforcement learning-based dynamic band and channel selection in cognitive radio ad-hoc networks. EURASIP J Wireless Comm Network. **131**, 1–25 (2019)
31. H. A. Nguyen, D. Choi, Berlin Heidelberg: Springer. Application of data mining to network intrusion detection: Classifier selection model. 2014.
32. M. Sabhnani, G. Serpen, Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context. In MLMTA, 2003.

Publisher's Note