Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 2465 – 2475
© TÜBİTAK
doi:10.3906/elk-1801-75

Symbolic interpretation of artificial neural networks using genetic algorithms

Dounia YEDJOUR[*], **Abdelkader BENYETTOU**, **Hayat YEDJOUR**
Signal Image Parole Laboratory, Department of Computer Science, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, El M'naouer, Oran, Algeria

Received: 08.01.2018 • **Accepted/Published Online:** 06.07.2018 • **Final Version:** 28.09.2018

Abstract: The knowledge acquired during the learning of artificial neural networks (ANNs) is coded as values in synaptic weights, which makes their interpretations difficult, hence the name of the black box. The aim of this work is to provide a comprehensible interpretation of the ANN's decisions by extracting symbolic rules. We improve the performance of our extraction algorithm by combining the ANN with a genetic algorithm. Misleading rules whose support and confidence values are less than fixed thresholds are removed and, as a result, the comprehensibility is improved. The extracted rules are evaluated and compared with other works. The results show good performance of our proposal in terms of fidelity and accuracy.

Key words: Neural networks, rule extraction, genetic algorithm

1. Introduction

Artificial neural networks (ANNs) are very powerful in classification problems, they give a good generalization of knowledge present in the training set [1], they are less sensitive to noise, and they are less vulnerable to the presence of incorrect data. Despite these advantages, neural networks also have a number of disadvantages, such as the initial network architecture, and so it is quite difficult to find the right topology of neural networks [2]. It is also difficult to determine a good choice of ANN parameters such as the initial weights and thresholds, and the number of hidden layers; a bad choice of these parameters may slow down learning time or may lead to a bad local optimum. The most penalizing problem of ANNs is the problem of the black box [3], and so the neural networks cannot explain their decisions. The knowledge remains locked in the black box and is incomprehensible to the user or even for the expert [4]. This problem is penalizing for certain applications where decision-makers generally opt for less accurate but comprehensible models. The aim of the present work is to provide a comprehensible interpretation of the ANN's decisions by extracting symbolic rules. The decisions of the ANN are represented by symbolic rules understandable by the user. The rules are first filtered to keep only the best ones, i.e. those that verify good support and good confidence. We improve the performance of our extraction algorithm by combining the ANN with a genetic algorithm (GA).

In the following sections, we describe the existing hybrid neurosymbolic systems. Next, we describe our genetic method of rule extraction from the neural network. Finally, we show some computational results and we indicate some issues for future works.

[*]Correspondence: dounia.yedjour@univ-usto.dz

2. Related work

In this section, we describe different combinations of the ANN with other machine learning algorithms. Symbolic approaches such as decision trees or GAs may be used to solve the network initial architecture problem or the network initial parameters problem. In the case of the black box problem, these combinations serve to improve the performance of the extracted rules. Among the popular existing hybrid systems, we can enumerate here the following:

- Neurofuzzy systems: there are several ways to do this hybridization: fuzzy rules can be extracted from the ANN [5] in order to solve the black box problem. Furthermore, fuzzy rules can be injected into the initial neural network architecture; here, the ANN is learned from examples and the fuzzy rules and, hence, learning time is reduced.

- Neuro-IDT: there are two possibilities to combine the ANN with decision trees: the decision tree can be extracted from the ANN [6], or, in the other case, the decision tree can be injected to the ANN.

- Neurogenetic: this hybrid system combines a GA with the ANN [7,8]; in this case, the GA can be used to have an optimal neural network architecture or to have the optimal rules that represent the hidden knowledge of the ANN.

Several methods are developed to understand neural network decisions by extracting a set of rules. Regarding extraction strategies, there are two different approaches: local and global. The local methods look for combinations of weights that activate the hidden or output neuron; then these combinations will be used to generate the rules. The local methods that are interested in each unit and each connection quickly become inefficient in the case of large neural networks. On the other hand, the global methods are insensitive to the number of the units of neural networks. In this case, the generation of the rules is only based on analysis of the responses of the neural network to the inputs. In the literature, there are several rule extraction algorithms we can enumerate here: Subset algorithm [9]: this algorithm extracted logical rules of order 0; it is based on the local method. The MofN algorithm [9] extracted a limited set of rules compared to the Subset algorithm. Taha and Ghosh presented three rule extraction techniques [10]: BIO-RE, which used the global approach, and Partial-Re and Full-Re, which used the local approach to extract partial and full rule sets from the trained feedforward. Markowska [5] proposed the global approach to extract fuzzy rules from the neural network. A survey of different rule extraction approaches can be found in [11]. In the present paper, we are interested in extracting rules from the ANN using a GA, which will be explained in the next section.

2.1. GA for rule extraction

2.1.1. The basic principle of GAs

GAs are exploration algorithms based on the mechanisms of natural selection and genetics. GAs are based on a coding of problems and their solutions in the form of chains, often binary chains, of fixed or variable lengths. An initial population is first generated containing potential solutions for a given problem; then the GA selects, through a fitness measure, the elements of the population that fulfill the constraints of the desired solution. Genetic operators such as crossover and mutation are then applied to this population in order to obtain a new population with better solutions than the previous generation. The process is repeated over several generations until a generation of solutions satisfies the imposed quality criteria.

2.1.2. How to use a GA for rule extraction

A GA starts with a given initial population containing a set of individuals. There are two possibilities: each individual encodes one rule (called the Michigan approach) or several rules (the Pitt approach) [12]. The goal of the GA is to search in rules space the set of rules that represent the knowledge of the ANN according to three criteria: fidelity, accuracy, and comprehensibility. The fidelity indicates the ability of the extracted rules to mimic the behavior of the network from which they were extracted [5].

Let P = {P1, ..., Pn} be a set of patterns or examples. For one pattern *Pi*, where *Pi* ⊆ P, the fidelity is equal to "*one*" if the rule set and the neural network give the same classification result of *Pi*,

$$Fid_i = \begin{cases} 1 & if\,(rule\,set\,(Pi) = neural\,network\,(Pi)) \\ 0 & otherwise \end{cases} \tag{1}$$

For N patterns, the fidelity is given by

$$Fidelity = \frac{\sum\limits_{i=1}^{N} Fid_i}{N} \tag{2}$$

Accuracy defines the number of correctly classified patterns by the rule set [3]. It is calculated independently of the ANN.

Let R = {R1, ..., Rn} be a set of rules. For one pattern *Pi*, the accuracy *acc* is equal to *one* if there is at least one rule Rj in the rule set R that satisfies *Pi*. In this case, all values of the premises that exist in Rj should exist in *Pi*.

$$\exists \subseteq Acc_i = \begin{cases} 1 & if\ Rj\ R,\, Rj\, satisfies\, Pi \\ 0 & otherwise \end{cases} \tag{3}$$

For N patterns, the accuracy is given by

$$Accuracy = \frac{\sum\limits_{i=1}^{N} Acc_i}{N} \tag{4}$$

Comprehensibility defines the degree of understanding and it is based on the number of rules. An important number of rules are difficult to understand whereas a reduced number of rules leads to good comprehensibility.

3. The proposed algorithm of rule extraction

Our algorithm is composed of three modules: the connectionist module, the genetic module (GA module) for rule extraction, and the rule pruning module (see Figure 1).

3.1. ANN component

We have used multilayer perceptron neuron network training with the backpropagation algorithm. This type of network consists of an input layer, one or more hidden layers, and an output layer. The neurons of each layer are connected to all the neurons of the adjacent layers through the synaptic weights. The main purpose of the learning phase is to find a set of optimal weights leading the neural network to give a better prediction.

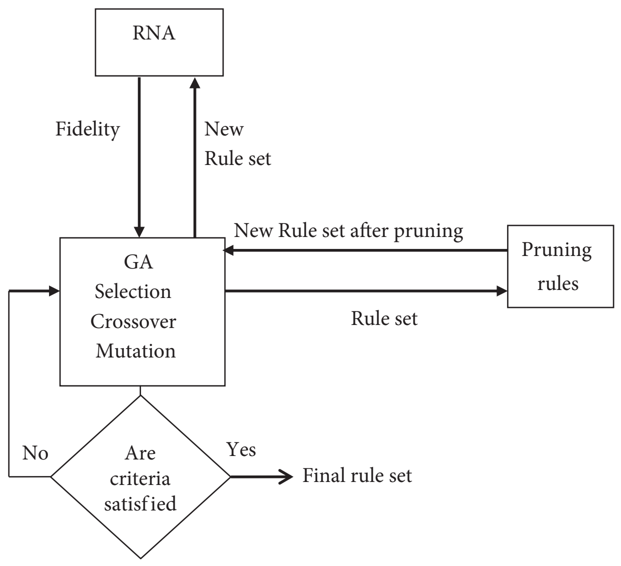

Figure 1. The proposed system.

3.2. GA component

Neural networks are treated as a black box. The aim of the GA is to search, from a set of initial rules, those that express knowledge of the neural network. These rules should verify good fidelity, good accuracy, and good comprehensibility.

3.2.1. Fitness function

In order to converge to a single solution, the fitness function is defined as the weighted average of the following parameters: fidelity, accuracy, and comprehensibility.

$$fitness = \alpha \times \ fidelity + \ \beta \times accuracy + \delta \times comprehensibility \tag{5}$$

The fidelity and the accuracy measures take their values in the interval [0,1], whereas the comprehensibility (number of rules) values are always greater or equal to one. Therefore, the comprehensibility should be normalized to the interval [0,1] before processing using the following formula:

$$Compr_{Norm} = {}^{(MaxRule- \ comprehensibility)}/_{(MaxRule - 1)}\,, \tag{6}$$

where MaxRule is the maximum number of rules, *comprehensibility* is the number of rules obtained by the algorithm, and Compr$_{Norm}$ is the normalized value of *comprehensibility.* For example, if *MaxRule* = 5 and *comprehensibility* = 5 then *Compr*$_{Norm}$ is set to 0 and if *MaxRule* = 5 and *comprehensibility* = 1, Compr$_{Norm}$ is set to 1. The objective is to maximize Compr$_{Norm}$ in order to have a minimum of rules. Therefore, the fitness function of Eq. (5) is modified as follows:

$$fitness = \alpha \times \ fidelity + \ \beta \times accuracy + \delta \times Compr_{Norm} \tag{7}$$

$$\{\alpha, \beta, \delta\} \in [0,1] and \alpha + \beta + \delta = 1$$

The parameters α, β and δ should be determined according to the preferences of the decision-makers. An important issue in the rule extraction is finding a compromise between the qualities of extracted rules.

3.2.2. The form of chromosomes

The chromosome represents a possible solution to the proposed problem. It consists of a vector of rules. One rule is coded as a string of integer values between 0 and 1. "-" means that the attribute is not involved in the rule," 0" means that the attribute is written "Not (Ai)" in the rule generated, and "1" means that the attribute is written as "Ai" in the rule generated. For example, Rule 4 in Figure 2 is written as "if A1 and A4 and not (A5) then ClassX"

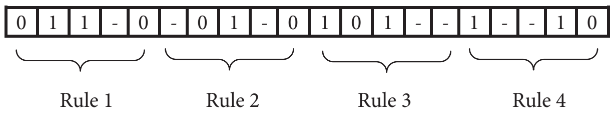

Figure 2. Chromosome form of a given class.

3.2.3. Genetic operators

A wheel selection operator was used to choose parents for the next generation. Crossover combined two individuals (or parents) to form a new individual (or child) of the next generation by using one-point crossover technique. The mutation operator takes one parent from the population and randomly changes one gene.

3.3. Pruning rules component

Let A = {A1, ..., An} be a set of premises or attributes and C the target class. Let *ri* be a rule of the form X $\Rightarrow$ C, where X $\subseteq$ A.

The support of *ri* indicates the number of patterns satisfying both the A and the C divided by the total number of patterns. The confidence of *ri* indicates the number of patterns satisfying the consequent (C) among those verifying the premise (X) [13].

For example, let us assume Table 1, which represents a set of examples. The support of the rule (*If A1 and A6* $\rightarrow$ *C2)* = 2/9 = 0.22 whereas the confidence = 2/3 = 0.66.

Table 1. Sample dataset.

A1	A2	A3	A4	A5	A6	A7	Class
1	0	1	0	0	1	1	C1
0	1	0	0	0	0	0	C1
1	0	1	0	1	0	1	C1
0	1	1	1	0	1	0	C1
0	0	1	0	0	0	1	C2
1	0	1	1	0	0	1	C2
1	0	1	1	0	0	0	C2
1	1	1	0	0	1	1	C2
1	0	0	0	0	1	0	C2

The pruning rules component consists of eliminating the invalid rules whose support and confidence values are less than fixed thresholds. Once filtered, the rules are used by the genetic module for optimization.

The choice of thresholds should be made with caution. A very small support threshold value may lead us to uninteresting rules whereas a very large value may skip interesting rules.

4. Results and discussion

We have used a multilayer perceptron trained by the backpropagation algorithm. It does not exclude the possibility of working with other ANN types or with other learning algorithms. Initially, the neural network is trained with a set containing 70% of the examples of the dataset. The remaining (30%) are used for testing its accuracy.

4.1. Datasets [14]

We chose to work on 3 datasets: breast cancer, Austra (Australian Credit Approval), and Pima (Pima Indians Diabetes). A brief description is given in Table 2.

Table 2. Datasets used in our computational experiments.

Datasets	Number of patterns	Attributes	Classes	NN	
				NNH	Acc (%)
Breast cancer	699	9	2	3	97.11
Austra	690	14	2	3	90.08
Pima	768	8	2	6	78.00

(NNH = number of hidden nodes, Acc = accuracy).

4.2. MLP module

Our rule extraction algorithm requires binary data. Numerical or real data should be converted into binary data using the following formula [10]:

$$Y_i = \begin{cases} 1 & if\, x_i \geq u_i \\ 0 & else \end{cases} , \tag{8}$$

where xi is the value of the attribute Xi, ui is the average value of Xi, and yi is the corresponding binary value. Tables 3 and 4 show the confusion matrix of the best results on the training and test sets on the breast cancer dataset. Several measures are calculated to check the validity of the network, namely accuracy, misclassification rate, sensitivity, and specificity:

Table 3. Confusion matrix for the best network on the training set.

	Benign	Malignant
Benign	318 (true positive)	1 (false negative)
Malignant	13 (false positive)	159 (true negative)

$$Accuracy = 100\% \times \left(\frac{true\, positive\, cases + true\, negative\, cases}{total} \right) \tag{9}$$

$$Misclassification = 100\% \times \left(\frac{false\, positive\, cases + false\, negative\, cases}{total} \right) \tag{10}$$

Table 4. Confusion matrix for the best network on the test set.

	Benign	Malignant
Benign	123 (true positive)	2 (false negative)
Malignant	4 (false positive)	79 (true negative)

$$Sensitivity = 100\% \times (\frac{true\,positive\,cases}{true\,positive\,cases + false\,negative\,cases}) \tag{11}$$

$$Specificity = 100\% \times (\frac{true\,negative\,cases}{true\,negative\,cases + false\,positive\,cases}) \tag{12}$$

Table 5 shows the performance of the neural network on test and training sets of the breast cancer dataset. Table 2 shows the accuracy of neural networks for all datasets used.

Table 5. Performance of the ANN.

	Accuracy	Misclassification	Sensitivity	Specificity
Train	97.14%	0.028%	99.67%	92.44%
Test	97.11%	0.028%	98.40%	95.18%

4.3. Genetic module for rule extraction

The time taken by our system to converge to the solution depends largely on parameters such as the initial population, number of iterations, mutation probability value, and crossover probability value. After several tests, the parameters values were set as follows: crossover probability = 0.8, mutation probability = 0.2, population size = 100, the number of individuals = 20 rules, minimum value of support = 0.1, and minimum value of confidence = 0.5. MaxRule = 5 (we suppose that, for each class, the maximum number of rules is set to 5). We first applied our extraction method to the breast cancer dataset; the aim of this rule extraction application is to search for the automatic breast cancer diagnostic through the generation of a set of rules. Figures 3 and 4 show the obtained results on benign and malignant classes in terms of accuracy, fidelity, and compr_norm (the normalized comprehensibility), by varying the parameters $\{\alpha,\beta,\delta\}$ in the interval [0, 1]. The results are sorted according to the ascending order of accuracy values. In the middle of Figure 3, note that the three curve lines are closer together, which means a good compromise between the three measures in this region. Outside this region, the accuracy and the comprehensibility tend to be contradictory since as accuracy increases comprehensibility decreases. We can make the same observation about the malignant class (see Figure 4). For the value $\beta = 1$, the algorithm extracted 4 rules for the benign class with an accuracy value of 1. The fidelity for this value is low (close to 0.74). By focusing on the fidelity and the accuracy, the algorithm extracted 3 rules whose fidelity is equal to 1 and with an accuracy value of 0.9779. Moreover, by focusing on comprehensibility, the algorithm extracted one rule with fidelity = 100% and with accuracy close to 0.95. Table 6 shows the trade-off between the accuracy and the comprehensibility for the benign class by modifying the parameters values $\{\alpha,\ \beta,\ \delta\}$.

Figure 5 and 6 show the results obtained in terms of average support and average confidence on benign and malignant classes (the results are sorted according to the ascending order of accuracy values). Several solutions

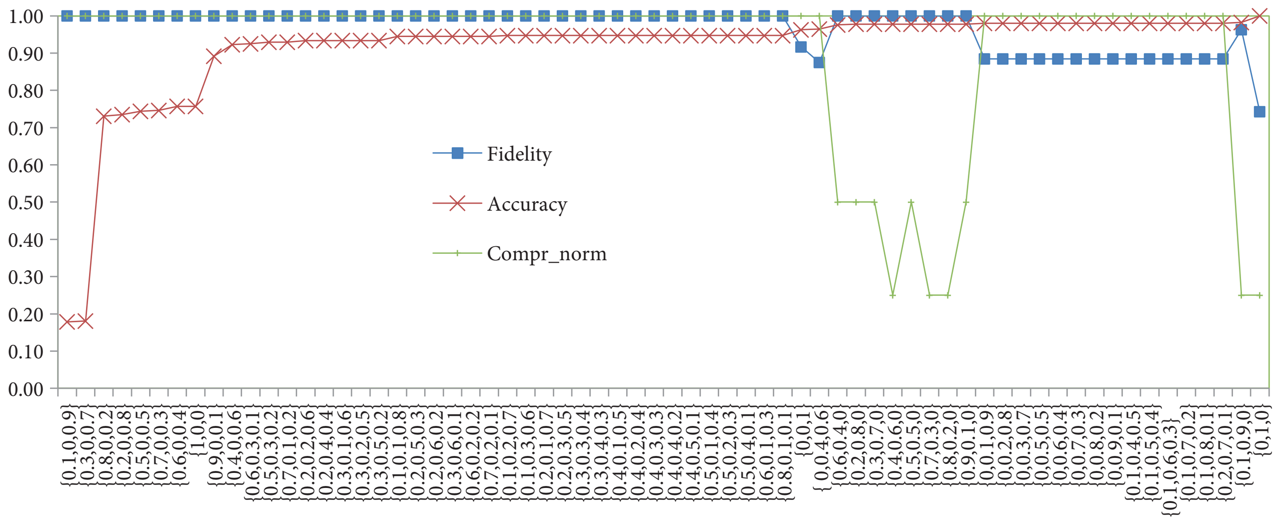

Figure 3. Obtained results for the benign class by varying the $\{\alpha,\ \beta,\ \delta\}$ parameters.

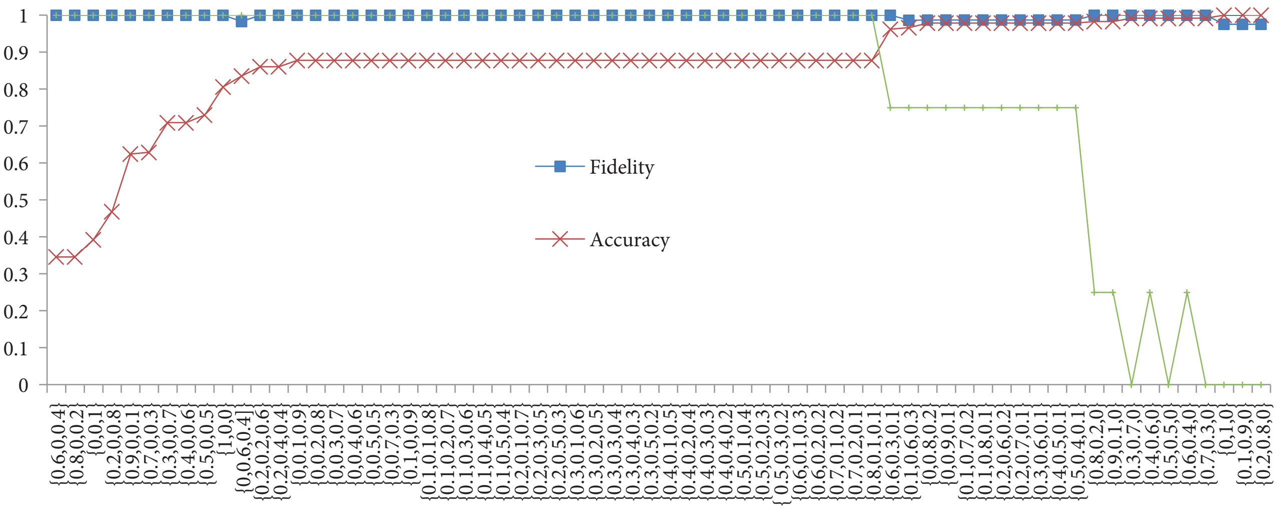

Figure 4. Obtained results for the malignant class by varying the $\{\alpha,\ \beta,\ \delta\}$ parameters.

Table 6. Performance of the extracted rules from the benign class.

$\{\alpha\ ,\ \beta,\ \delta\}$	Fidelity	Accuracy	Compr_{Norm}	#Rules	Mean sup	Mean conf
$\{0.4, 0.2, 0.4\}$	1.0000	0.9470	1.00	1	0.62	0.99
$\{0.5, 0.5, 0\}$	1.0000	0.9779	0.50	3	0.57	0.98
$\{0.1, 0.9, 0\}$	0.9630	0.9823	0.25	4	0.60	0.98
$\{0, 1, 0\}$	0.7429	1.0000	0.25	4	0.5	0.91

have been obtained with the same values of fidelity, accuracy, and comprehensibility (see Figures 3 and 4). In this case, the support and confidence values can help us to make a decision by choosing the best values. Finally, Tables 7–9 show the obtained results in terms of mean fidelity, mean accuracy, mean comprehensibility (#rules), mean support (sup), and mean confidence (conf) on the breast cancer, Austra, and Pima datasets.

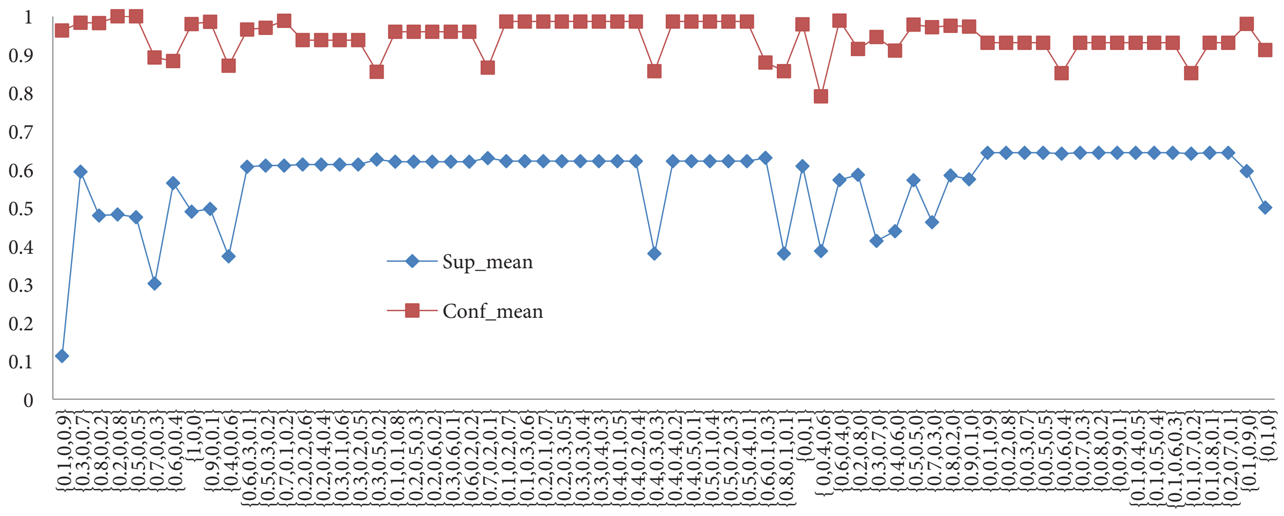

Figure 5. Average support and confidence for the benign class.

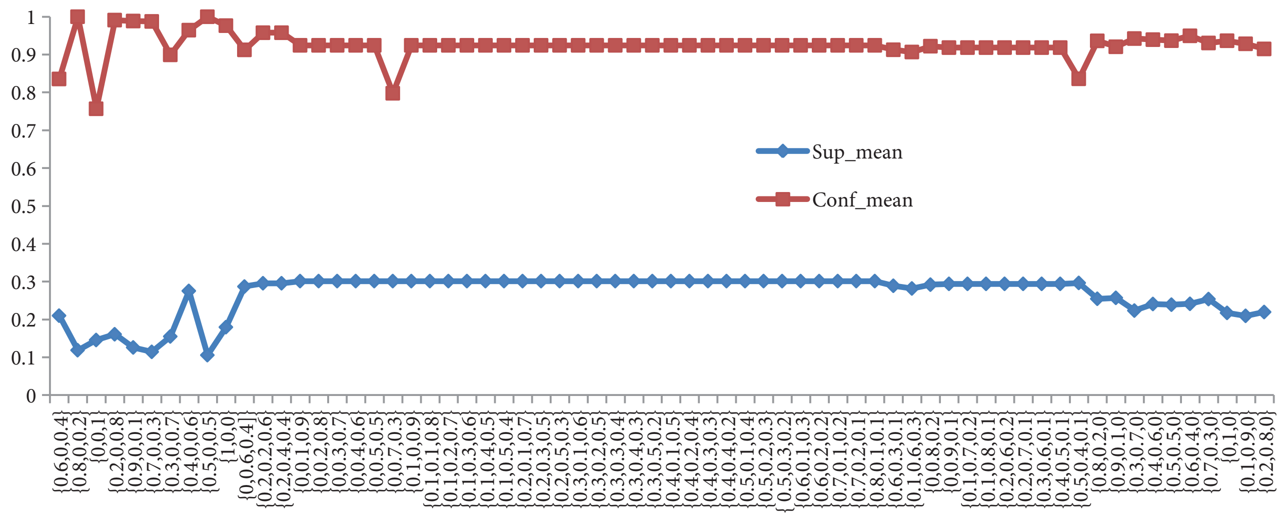

Figure 6. Average support and confidence for the malignant class.

Table 7. Obtained results on the breast cancer dataset.

Rules	Class	Sup	Conf	Fidelity (%)	Accuracy (%)	#rules
if A2 $<$ 3.1 and A6 $<$ 3.5	Benin	0.6217	0.9862	99.51	98.50	05
if A4 $<$ 2.8 and A5 $<$ 3.2	Benin	0.6130	0.9379			
if A3 $\geq$ 3.2	Malignant	0.3014	0.9244			
If A6 $\geq$ 3.5	Malignant	0.2870	0.9124			
if X8 $\geq$ 2.9	Malignant	0.2768	0.9009			

5. Comparison with other works

To confirm the performance of our method we have compared our obtained results with seven algorithms: Full_RE [10], NeuroRule [15], BIO-RE [10], HYPINV [16], NeuroLinear [17], Kim et al. [18], and GEX [5].

Table 8. Obtained results on the Austra dataset.

Rules	Class	Sup	Conf	Fidelity (%)	Accuracy (%)	#rules
if $A_8 \geq 0.52$	class1	0.41	0.79	94.96	89.42	03
if $A_8 < 0.52$ and $A_{10} < 2.40$	class0	0.43	0.93			
if $A_5 < 7.37$ and $A_7 < 2.90$	class0	0.26	0.80			

Class = 1 indicates that the credit card application has been approved by the bank.

Table 9. Obtained results on the Pima dataset.

Rules	Class	Sup	Conf	Fidelity	Accuracy	#rules
if $A_8 < 33.2$	Class0	0.46	0.74	86.2	77.87	6
if $A_6 < 32$	Class0	0.38	0.77			
if $A_2 < 120.9$ and $A_6 \geq 32$	Class0	0.18	0.75			
if $A_2 \geq 120.9$ and $A_6 \geq 32$	Class1	0.18	0.65			
if $A_1 \geq 3.8$ and $A_2 \geq 120.9$	Class1	0.14	0.60			
if $A_3 \geq 69.1$ and $A_6 \geq 32$ and $A_8 \geq 33.2$	Class1	0.10	0.57			

(class1 is interpreted as "tested positive for diabetes")

Comparison results in Table 10 indicate that the proposed algorithm produces a small number of rules. It extracts rules for all classes without the necessity of the default rule. Our proposal achieves high performance for the breast cancer and Austra datasets (see Table 11) and surpasses the others methods. The application of our algorithm to the Pima database showed the the lack of convenience of our algorithm on complex datasets. The binarization used by Eq. (8) severely degrades the performance of the Pima neural network and subsequently the performance of the rules extracted. Therefore, a more natural representation of the problem offers more effective solutions. Finally, the proposed algorithm provides interesting rules by taking into account the support and the confidence measures while the others do not.

Table 10. Results obtained by different algorithms in terms of number of rules and accuracy, using the breast cancer dataset.

	Full-RE	BIO-RE	NeuroRule	Our proposal
Accuracy (%)	96.19	96.63	97.21 with default rule	98.50
#Rules	05	10 + default rule	4 (3 + default rule)	5

Table 11. Results obtained by different algorithms in terms of number of rules and accuracy, using the Austra and Pima datasets.

Datasets	Measures	NeuroLinear	Kim et al.	HYPINV	Gex	Our proposal
Austra	Accuracy	83.64	-	82.52	64.3	89.42
	#Rules	6.6	-	2	67.52	3
Pima	Accuracy	-	73.4	78.59	0.889	77.87
	#Rules	-	23.6	1	27.8	6

6. Conclusion

A new method of rule extraction from ANNs has been proposed. The performance of our extraction algorithm is improved by combining the ANN with a genetic algorithm. The algorithm starts by removing the misleading rules by applying the support and confidence measures. The results show that the proposal achieves high performance when comparing it with others works. However, the results are strongly dependent on α, β and δ parameters. This may affect the feasibility of the method. To overcome this problem, an interesting extension is the use of another optimization strategy that offers multiobjective optimization in the Pareto sense. Another interesting extension is the application of our method to extract rules from neural networks trained with original input features without any binarization.

References

[1] Pandya MD, Patel JR. A survey: artificial neural network for character recognition. In: Proceedings of Fifth International Conference on Soft Computing for Problem Solving; 2016; Singapore: Springer. pp. 403-410.

[2] Hayashi Y, Setiono R, Azcarraga A. Neural network training and rule extraction with augmented discretized input. Neurocomputing 2016; 207: 610-622.

[3] Augasta M, Kathirvalavakumar T. Reverse engineering the neural networks for rule extraction in classification problems. Neural Process Lett 2012; 35: 131-150.

[4] Craven MW. Extracting Comprehensible Models from Trained Neural Networks. PhD, University of Wisconsin, Madison, WI, USA, 1996.

[5] Markowska-Kaczmar U. Evolutionary approaches to rule extraction from neural networks. Stud Comput Intell 2008; 82: 177-209.

[6] Craven M, Shavlik J. Extracting tree-structured representations of trained networks. Adv Neur Info Proc Sys 1996; 8: 24-30.

[7] Etemadi H, Ahmadpour A, Moshashaei SM. Earnings per share forecast using extracted rules from trained neural network by genetic algorithm. Comput Econ 2015; 46: 55-63.

[8] Yedjour D, Benyettou A, Yedjour H. Combining Quine Mc-Cluskey and genetic algorithms for extracting rules from trained neural networks. Asian J Appl Sci 2011; 4: 72-80.

[9] Towell G, Shavlik JW. The extraction of refined rules from knowledge-based neural networks. Mach Learn 1993; 131: 71-101.

[10] Taha I, Ghosh J. Symbolic interpretation of artificial neural networks. IEEE T Knowl Data Eng 1999; 11: 448-463.

[11] Andrews R, Diederich J, Aickle AB. Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowl Based Syst 1995; 8: 373-389.

[12] Santos R, Nievola J, Freitas A. Extracting comprehensible rules from neural networks via genetic algorithm. In: Proceedings of the IEEE Symposium on Combination of Evolutionary Algorithm and Neural Network; San Antonio, TX, USA. 2000, pp. 130-139.

[13] Dhanalaxmi B, Naidu GA, Anuradha K. Adaptive PSO based association rule mining technique for software defect classification using ANN. Procedia Comput Sci 2015; 46: 432-442.

[14] Blake CC, Merz C. UCI Repository of Machine Learning Databases. University of California, Irvine, Dept. of Information and Computer Sciences 1998.

[15] Setiono R, Liu H. Understanding neural networks via rule extraction. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI); 1995. pp. 480-485.

[16] Saad EW, Wunsch DC. Neural network explanation using inversion. Neural Networks 2007; 20: 78-93.

[17] Setiono R, Liu H. NeuroLinear: From neural networks to oblique decision rules. Neurocomputing 1997; 17: 1-24.

[18] Kim D, Lee J. Handling continuous-valued attributes in decision tree with neural network modeling. In: Proceedings of the 11th European Conference on Machine Learning; 31 May–2 June 2000; Barcelona, Catalonia, Spain. Berlin, Germany: Springer, pp. 211-219.