Rule extraction from trained adaptive neural networks using artificial immune systems

Humar Kahramanli *, Novruz Allahverdi

Electronic and Computer Education Department, Selcuk University, 42075 Konya, Turkey

Abstract

Although artificial neural network (ANN) usually reaches high classification accuracy, the obtained results sometimes may be incomprehensible. This fact is causing a serious problem in data mining applications. The rules that are derived from ANN are needed to be formed to solve this problem and various methods have been improved to extract these rules.

Activation function is critical as the behavior and performance of an ANN model largely depends on it. So far there have been limited studies with emphasis on setting a few free parameters in the neuron activation function. ANN’s with such activation function seem to provide better fitting properties than classical architectures with fixed activation function neurons [Xu, S., & Zhang, M. (2005). Data mining – An adaptive neural network model for financial analysis. In Proceedings of the third international conference on information technology and applications].

In this study a new method that uses artificial immune systems (AIS) algorithm has been presented to extract rules from trained adaptive neural network. Two real time problems data were investigated for determining applicability of the proposed method. The data were obtained from University of California at Irvine (UCI) machine learning repository. The datasets were obtained from Breast Cancer disease and ECG data. The proposed method achieved accuracy values 94.59% and 92.31% for ECG and Breast Cancer dataset, respectively. It has been observed that these results are one of the best results comparing with results obtained from related previous studies and reported in UCI web sites.

© 2007 Elsevier Ltd. All rights reserved.

Keywords: Adaptive neural networks; Artificial immune systems; Optimization; Rule extraction; Backpropagation; Opt-aiNET

1. Introduction

An important drawback of many artificial neural networks is their lack of explanation capability (Andrews, Diederich, & Tickle, 1996). Although, ANN’s have been successfully used in many applications, the explanation of their responses is obscure, because the knowledge embedded within them is distributed over the activations and the connections of neurons (Bologna, 2004). However, they are black boxes and consequently, it is very difficult to understand how an ANN has solved a problem (Mantas, Puche, & Mantas, 2006). Thus, knowledge captured by neural networks is not transparent to users and cannot be verified by domain experts (Huang & Xing, 2002). This may cause problems in some cases. To solve this problem, researchers are interested in developing a humanly understandable representation for neural networks. This can be achieved by extracting production rules from trained neural networks (Huang & Xing, 2002).

Rule extraction techniques are grouped into three approaches named as decompositional, pedagogical and eclectic. In contrast with the decompositional approach, which analyzes the activation and weights of the hidden layers of the neural network, the pedagogical approach treats the ANN as a black box and extract rules by only looking at the input and output activations (Andrews et al., 1996; Tickle, Andrews, Golea, & Diederich, 1997). Pedagogical approach aims at extracting symbolic rules...
which map the input–output relationship as closely as possible to the way the ANN understands the relationship. The number of these rules and their form do not directly correspond to the number of weights or the architecture of ANN (Saad & Wunsch, 2007). Finally, the eclectic approach is characterized by any use of knowledge concerning the internal architecture and/or weight vectors in a trained ANN to complement a symbolic learning algorithm (Keedwell, Narayanan, & Savic, 2000a, 2000b).

This approach is named as eclectic, because it is based on both decompositional and pedagogical approaches.

The aim of this study is to develop a new adaptive activation function and a new method for rule extraction from trained neural networks using artificial immune systems. The work is organized as follows: In Section 2 previous studies related with this study are introduced. In Section 3 framework of this study and related background theory are presented. In Section 4, performance metrics are explained. In Section 5, evaluation methods are described. The results of the experiments and evaluation of these results are presented in Section 6. In the final this paper is concluded.

2. Literature review

In the literature, there are many different approaches for the rule extraction. The first rule extraction techniques from neural networks was proposed by Gallant (1988). He was working on connectionist expert systems. In this work each ANN node represents a conceptual entity. Towell and Shavlik (1993) showed how to use ANNs for rule refinement. The algorithm was called SUBSET, which is based on the analysis of the weights that make a specific neuron active. Alexander and Mozer (1995) developed a rule extraction method, based on connection weights, that supposes activation functions showing approximately Boolean behavior. Sethi and Yoo (1996) developed a rule extraction method based on the connection weights. Lu, Setiono, and Liu (1996) proposed an approach for rule extraction from ANNs based on the clustering of hidden unit activation values. Keedwell et al. 2000a, 2000b developed a system in which a genetic algorithm is used to search for rules in the ANN input space. Setiono and Leow (2000) presented the fast method is based on the relevance of hidden units, considering their information gains. Palade, Neagu, and Puscasu (2000) presented a method of rule extraction from ANNs that is based on interval propagation across the network, using a procedure of inverting an ANN. Garcez, Broda, and Gabbay (2001) presented a method to extract non-monotonic rules from ANNs formed by discrete input units. Snyders and Omlin (2001) compared the performance of symbolic rules extracted from ANNs trained with and without adaptive bias, giving empirical results for a molecular biology problem. Jiang, Zhou, and Chen (2002) proposed that combines ANNs and rule learning. The proposed algorithm utilizes a ANN ensemble as the front-end process, which generates abundant training instances for the back-end rule learning process. Setiono, Leow, and Zuarada (2002) presented an approach for extracting rules from ANNs trained in regression problems. Elalﬁ, Haque, and Elalami (2004) presented an algorithm for extracting rules from databases via trained ANN using genetic algorithm.

In summary, most of the approaches described in the literature have basically two motivations. On the one hand, some authors noticed the need for simplification of neural networks to facilitate the rule extraction process, and are in favor of using specialized training schemes and architectures to perform such task. The assumption underlying these approaches is that neural networks can help the extraction of interesting rules. On the other hand, some papers have proposed algorithms mainly intended to clarify the knowledge encoded in previously trained ANNs (Hruschka & Ebecken, 2006).

This study is focused on the problem of extracting rules from previously trained ANNs. Networks with adaptive activation function seem to provide better ﬁtting properties than classical architectures with ﬁxed activation function neurons (Xu & Zhang, 2000). So in this study the ANN was trained with adaptive activation function and was offered a neuron-adaptive activation function with tree free parameters. Study on rule extraction from trained ANN is based on the work of Elalﬁ et al. (2004) and presents algorithm for extracting rules from neural networks using artiﬁcial immune systems.

3. Study environment and background theories

3.1. A neuron-adaptive activation function definition

We deﬁned a neuron-adaptive activation function as follows:

\[ \phi(x) = A1e^{-Bx} + \frac{A2}{1 + e^{-Bx}} \]  \hspace{1cm} (3.1)

where \( A1, A2, B \) are real variables which will be adjusted during training.

In this study was used a training algorithm of Xu and Zhang (2005).

The input–output relation of the \( i \)th neuron in the \( k \)th layer can be described by

\[ I_{ik} = \sum_j w_{ij,k}O_{jk-1(u)} - \theta_{ik}, \quad j = 1, 2, \ldots, k \]  \hspace{1cm} (3.2)

where \( w_{ij,k} \) the weight that connects the \( j \)th neuron in layer \( k - 1 \) and the \( i \)th neuron in layer \( k \), \( \theta_{ik} \) is the threshold of the unit, \( j \) is the number of neurons in layer \( k - 1 \), and

\[ O_{ik}(u) = \phi(I_{ik}(u)) = A1e^{-B1x} + \frac{A2}{1 + e^{-B1x}} \]  \hspace{1cm} (3.3)

To train this neural network an energy function:

\[ E = \frac{1}{2} \sum_j (d_j(u) - O_j(u))^2 \]  \hspace{1cm} (3.4)
is adopted. In (3.4) \( d(u) \) is the \( j \)th desired output value and \( m \) is total number of output layer neurons, \( l \) is the total number of constructed network layers. The aim of learning is minimize the energy function. This can be obtained by using a variation of the steepest descent gradient rule (Rumelhart & McClelland, 1986) expressed as follows:

\[
\alpha_{i,k}^{(r)} = \eta \alpha_{i,k}^{(r-1)} + \beta \frac{\partial E}{\partial \alpha_{i,k}} \tag{3.5}
\]

\[
\theta_{i,k}^{(r)} = \eta \theta_{i,k}^{(r-1)} + \beta \frac{\partial E}{\partial \theta_{i,k}} \tag{3.6}
\]

\[
A_1^{(r)} = \eta A_1^{(r-1)} + \beta \frac{\partial E}{\partial A_1^{(r)}} \tag{3.7}
\]

\[
A_2^{(r)} = \eta A_2^{(r-1)} + \beta \frac{\partial E}{\partial A_2^{(r)}} \tag{3.8}
\]

\[
B_i^{(r)} = \eta B_i^{(r-1)} + \beta \frac{\partial E}{\partial B_i^{(r)}} \tag{3.9}
\]

where \( \eta \) is the momentum and \( \beta \) is the learning rate.

To derive the gradient information of \( E \) with respect to each adjustable parameter in Eqs. (3.5)–(3.9), was defined:

\[
\frac{\partial E}{\partial I_{j,k}(u)} = \zeta_{i,k} \tag{3.10}
\]

\[
\frac{\partial E}{\partial O_{i,k}(u)} = \xi_{i,k} \tag{3.11}
\]

From Eqs. (3.3), (3.4), (3.10) and (3.11) partial derivatives of \( E \) can be founded with respect to adjustable parameters as follows:

\[
\frac{\partial E}{\partial w_{i,k}} = \frac{\partial E}{\partial I_{j,k}(u)} \frac{\partial I_{j,k}(u)}{\partial w_{i,k}} = \zeta_{i,k} O_{j,k-1}(u) \tag{3.12}
\]

\[
\frac{\partial E}{\partial \theta_{i,k}} = \frac{\partial E}{\partial I_{j,k}(u)} \frac{\partial I_{j,k}(u)}{\partial \theta_{i,k}} = -\zeta_{i,k} \tag{3.13}
\]

\[
\frac{\partial E}{\partial A_{1,k}} = \frac{\partial E}{\partial O_{i,k}} \frac{\partial O_{i,k}}{\partial A_{1,k}} = \zeta_{i,k} e^{-l_{i,k}} \tag{3.14}
\]

\[
\frac{\partial E}{\partial A_{2,k}} = \frac{\partial E}{\partial O_{i,k}} \frac{\partial O_{i,k}}{\partial A_{2,k}} = \zeta_{i,k} \frac{1}{1 + e^{-b_{i,k}l_{i,k}}} \tag{3.15}
\]

\[
\frac{\partial E}{\partial B_{i,k}} = \frac{\partial E}{\partial O_{i,k}} \frac{\partial O_{i,k}}{\partial B_{i,k}} = \frac{\zeta_{i,k} A_{2,k} I_{j,k}(u) e^{-b_{i,k}l_{i,k}(u)}}{1 + e^{-b_{i,k}l_{i,k}(u)}} \tag{3.16}
\]

And for (3.10) and (3.11) following equations can be founded:

\[
\zeta_{i,k} = \frac{\partial E}{\partial I_{j,k}(u)} = \frac{\partial E}{\partial O_{i,k}} \frac{\partial O_{i,k}}{\partial I_{j,k}(u)} = \zeta_{i,k} \frac{\partial O_{i,k}(u)}{\partial I_{j,k}(u)} \tag{3.17}
\]


Algorithm works as follows:

1. Apply the input vector, \( x = (x_1, x_2, \ldots, x_N) \) to the input units.
2. Calculate the sum of weighted input signals to the hidden layer.
3. Calculate the outputs from the hidden layer.
4. Calculate the sum of weighted input signals to the output layer.
5. Calculate the outputs.
6. Calculate the error terms for the output units.
7. Calculate the error terms for the hidden units.
8. Update weights on the output and hidden layer.
9. Update real variables on the output and hidden layer.

This algorithm is not far from backpropagation algorithm. The difference is being used an updated the real variables in this algorithm. The training data are presented until the energy function \( E \) is acceptably low and the network converges. This algorithm and its application will be explain detail in Section 3.2.

### 3.2. Artificial immune system

Immune systems are naturally existing mechanisms which are responsible for detecting and coping with intruders in living organisms (Seredynski & Bouvry, 2007). The main purpose of the immune system is to recognize all cells (or molecules) within the body and categorize those cells as self or non-self (Kalini & Karaboga, 2005) and protect the organism against disease-causing cells called pathogens and to eliminate malfunctioning cells (Musilek, Lau, Reformat, & Wyard-Scot, 2006). All elements recognizable by the immune system are called antigens (Musilek et al., 2006). There are two types of antigens: self and non-self. Non-self antigens are disease-causing elements, whereas self antigens are harmless to the body (Kumar, Prakash, Shankar, & Tiwari, 2006). There are two major groups of immune cells: B-cells and T-cells which helps in recognizing an almost limitless range of antigenic patterns. It was discovered that people who had been inoculated against diseases contained certain agents that could in some way bind to other infectious agents. These agents were named antibodies (de Castro & Timmis, 2002).

AIS is a computational technique inspired by ideas coming from immunology and used to develop adaptive systems capable to solve different domain problems (Seredynski & Bouvry, 2007). The AIS (de Castro & Timmis, 2002) have become popular over the last year. Applications of AIS include pattern recognition, fault and anomaly detection, data mining and classification, scheduling, machine learning, autonomous navigation, search and optimization areas (Hou, Su, & Chang, 2008).

The acronym Opt-aiNET stands for “optimization version of an artificial immune network” (de Castro & Timmis, 2002). It is a particular type of artificial immune system developed to solve optimization problems (de Attux
et al., 2005). Opt-aiNET is capable of either unimodal or multimodal optimization and can be characterized by five main features (Timmis & Edmonds, 2004):

- The population size is dynamically adjustable.
- It demonstrates exploitation and exploration of the search space.
- It determines the locations of multiple optima.
- It has the capability of maintaining many optima solutions.
- It has defined stopping criteria.

The Opt-aiNET is a valuable tool for solving a wide range of optimization problems for two main reasons:

1. It presents a good balance between exploration and exploitation of the search-space.
2. Differently from other evolutionary proposals, it contains a mechanism devised to regulate population size and to maintain the diversity (de Attux et al., 2005).

Depending upon the extent of the infection, a large number of B-cells and T-cells may be required to handle the infection successfully and effectively. The size of sub-populations of these cells is controlled by a process termed clonal selection (Musilek et al., 2006). The clonal selection is the theory used to explain how an immune response is mounted when a non-self antigenic pattern is recognized by a B-cell (Ada & Nossal, 1987). It establish the idea that only those cells capable of recognizing an antigenic stimulus will proliferate and differentiate into effector cells, thus being selected against those that do not (de Attux et al., 2005). In brief, when a B-cell receptor (antibody) recognizes a non-self antigen with a certain affinity, it is selected to proliferate and it produces antibodies in high volumes. Proliferation in the case of immune cells is asexual, a mitotic process; the cells divide themselves (there is no crossover). During reproduction, the B-cell progenies (clones) undergo a mutation process with high rates (hypermutation) that, together with a strong selective pressure result in B-cells with antigenic receptors presenting higher affinities with selective antigen. This whole process of mutation and selection is known as affinity maturation or immune response. In addition to differentiating into antibody producing cells, the activated B-cells with high antigenic affinities are selected to become memory cells with long life spans. These memory cells are pre-eminent in future response to this same antigenic pattern, or similar one. The mainly features of the clonal selection principle are affinity proportional reproduction and mutation. In other words, the proliferation rate of each immune cell is proportional to its affinity with the selective antigen. The higher affinity, the higher number of offspring generated. The mutation suffered by each immune cell during reproduction is inversely proportional to the affinity of the cell receptor with the antigen. The higher affinity, the smaller mutation, and vice versa (de Castro & Timmis, 2002).

The immune network theory was proposed by Jerne in 1974 (de Castro & Timmis, 2002). This theory describes the immune system as being composed of cells and molecules that interact with each other in a network-like form. These self-interaction patterns suggest a dynamic immune system with eigen-behaviors even in the absence of foreign stimulation (antigens). An antigen would thus be responsible for disturbing a self-organizing and self-sustainable system (de Attux et al., 2005). The Opt-aiNET algorithm can be described as follows (de Castro & Timmis, 2002):

1. Initialization: create an initial random population of network antibodies.
2. Local search: while stopping criterion is not met, do:
   - Clonal expansion: for each network antibody, determine its fitness (an objective function to be optimized) and normalize the vector of fitnesses. Generate a clone for each antibody, i.e., a set of antibodies which are the exact copies of their antibody.
   - Affinity maturation: mutate each clone inversely proportionally to the fitness of its parent antibody that is kept unmutated. For each mutated clone, select the antibody with highest fitness, and calculate the average fitness of the selected antibodies.
   - Local convergence: if the average fitness of the population does not vary significantly from one iteration to the other, go to the next step; else, return to Step 2.
3. Network interactions: determine the affinity (similarity) between each pair of network antibodies.
4. Network suppression: eliminate all network antibodies whose affinity is less than a pre-specified threshold, and determine the number of remaining antibodies in the network; these are named memory antibodies.
5. Diversity: introduce a number of new randomly generated antibodies into the network and return to Step 2.

4. Performance metrics

Accuracy, sensitivity and specificity are the common performance metrics used in medical diagnosis tasks. The measure of the ability of the classifier to produce accurate diagnosis is determined by accuracy. The measure of the ability of the model to identify the occurrence of a target class accurately is determined by sensitivity. The measure of the ability of the model to separate the target class is determined by specificity. So that accuracy, sensitivity and specificity are calculated as follows (Loo, 2005):

\[
\text{Accuracy} = \frac{\text{Total number of correctly diagnosed cases}}{\text{Total number of cases}}
\]

\(4.1\)
5. Study environment

In this paper, we propose a novel approach for extracting rules from previously trained neural network. The idea behind suggested approach is to use artificial immune systems for optimization of function which produced from neural network.

Proposed new rule extraction algorithm is composed of three parts:

1 – Data coding.
2 – Classification of coding data.
3 – Rule extraction.

5.1. Data coding

Every data in dataset are coded as binary string and presented as input to ANN. The following method has been used for coding (Elalfi et al., 2004). Let the data have \( N \) attributes. Every attribute \( A_n \) \( \{n = 1, 2, \ldots, N\} \) has been divided in to \( m_n \) sub-strings as \( \{a_1, a_2, \ldots, a_{m_n}\} \) and coded as binary sub-string \( \{b_1, b_2, \ldots, b_{m_n}\} \). If attribute \( A_n \) belongs to sub-string \( a_i \) \( \{i = 1, 2, \ldots, m_n\} \), \( b_j \) is given by

\[
b_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad j = 1, 2, \ldots, m_n
\]

Thus, the input vector of ANN can be given by

\[
X = \bigcup_{n=1}^{N} \bigcup_{i=1}^{m_n} b_i
\]

The length of input vector \( X \) is determined as follows:

\[
m = \sum_{n=1}^{N} m_n
\]

5.2. Data classification

The datasets that will be applied in this study consist of two classes, so output layer of ANN can consist of one neuron. Output will 1, when the presented vector belongs to class 1 and it will be 0, when the presented vector belongs to class 0. The ANN model that is seen in Fig. 1 has been used for classification.

The sum of weighted input signals for \( j \)th neuron of hidden layer is calculated as

\[
G_j = \sum_{i=1}^{m} x_i w_{ij} - \theta_j, \quad j = 1, 2, \ldots, k
\]

where \( k \) is number of neurons in hidden layer, \( w_{ij} \) is the weight between \( i \)th neuron of input layer and \( j \)th neuron of hidden layer and \( \theta_j \) is threshold for \( j \)th neuron of hidden layer. Output of \( j \)th neuron of hidden layer is calculated as follows:

\[
CG_j = A_1 e^{-G_j} + \frac{A_2}{1 + e^{-B_j \theta_j}}, \quad j = 1, 2, \ldots, k
\]

where \( A_{1,2}, B_j \) are adjustable variables for activation function of \( j \)th neuron in hidden layer.

The sum of weighted input signals for output neuron is calculated as

\[
O = \sum_{j=1}^{k} CG_j v_j - \xi
\]

where \( v_j \) is the weight between \( j \)th neuron of hidden layer and output neuron and \( \xi \) is threshold for output neuron. Output neuron is calculated as follows:

\[
C = A_1 e^{-O_2} + \frac{A_2}{1 + e^{-B_c O}}
\]

where \( A_{1,2}, B_c \) are adjustable variables for activation function of output neuron.

Thus, we obtain a nonlinear function \( C \) that depends on \( X \).

\[
C(X) = A_1 e \left[ \mathbf{1} + 2 \left( A_1 e \left[ \sum_{j=1}^{k} \left( \frac{A_1 e}{1 + e^{-B_j \theta_j}} \right)^2 \right] + A_2 \left( \frac{A_1 e}{1 + e^{-B_c O}} \right)^{-1} \right) \right]^{-1}
\]

\[
A_1 e \left[ \sum_{j=1}^{k} \left( \frac{A_1 e}{1 + e^{-B_j \theta_j}} \right)^2 \right] + A_2 \left( \frac{A_1 e}{1 + e^{-B_c O}} \right)^{-1}
\]

\[
A_1 e \left[ \sum_{j=1}^{k} \left( \frac{A_1 e}{1 + e^{-B_j \theta_j}} \right)^2 \right] + A_2 \left( \frac{A_1 e}{1 + e^{-B_c O}} \right)^{-1}
\]
The vectors that make output value $C$ “1” and “0”, are needed to be found to extract rules from ANN. This is also an optimization problem. For optimization Opt-aiNET algorithm has been used.

5.3. Extraction of rules

As result of optimization, the binary vectors which have $m$ length are produced. Each vector means a rule. To see the rules these vectors must be decoded. For extracting a rule belongs to class 0 or class 1 the best antibody must be decoded as follows (Elalfi et al., 2004):

\[
\{
\text{The best antibody is divided into } N \text{ segments. Each segment represents one attribute, } A_n (n = 1, 2, \ldots, N), \text{ and has a corresponding bits length } m_n \text{ which represents their values.}
\text{The attribute values are existed if the corresponding bits in the best antibody equal one and vice versa.}
\text{The operators “OR” and “AND” are used to correlate the existing values of the same attribute and the different attributes, respectively.}
\}
\]

6. Evaluation

Two different datasets are used in this study for application part. The used datasets were medical datasets consisting of Breast Cancer data and echocardiogram data. The dataset chosen for this first experiment was the Ljubljana Breast Cancer dataset from UCI machine learning repository (http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998). The dataset has nine attributes, some of which are linear and some are nominal. There are: (1) age, (2) menopause, (3) tumor-size, (4) inv-nodes, (5) node-caps, (6) deg-malig, (7) breast, (8) breast-quad and (9) irradiat. Classes was coded as 0 and 1. Value 0 means the events that did not recurrence. Value 1 means the events that recurrence. The attributes have different range values in the database and these ranges of the data can be seen in Table 1.

To solve the problem, first a neural network was constructed. Each attribute value was coded as a binary string for use as input to the network. The sub-intervals used to coding of attribute values are summarized in Table 2. Each bit of a string was either 0 or 1 depending on which sub-interval the original value was located. For example, an age value at 45 would be coded as \( 0,0,1,0,0,0 \). Sub-intervals for the age are: \([10,30), [30,40), [40,50), [50,60), [60, 70), [70, 80)\). So the binary string for epss has six bits. Because of \( 45 \in [40,50) \), third bit of this string is equal to 1 and all others equal to 0. For the binary attribute, a node-caps of 0 would be coded as \{0\}, 1 as \{1\}.

With the coding scheme shown in Table 2, we had a total of 41 binary inputs. As the patients were classified into two classes a single unit of the output layer was sufficient. The target output was 1 if the patient belonged to Class 1, and 0, otherwise. The number of neurons in the hidden layer was taken as six.

The Opt-aiNET algorithm has been applied to solve the equation $C(X)$ (see 5.8) and in order to get the vectors which maximizes or minimizes that function. Multiplying factor is 0.5, mutation rate is 10. The Opt-aiNET was then run with a population of 20 for 10,000 generation for each classification. All parameter was chosen empirically for the best convergence rate between the actual and desired output. This generates 32 rules for Class 0 and 27 rules for Class 1. Both the maximum and minimum of output antibodies has been determined and will be translated into rules.

In summary, for rule extraction first ANN was designed which classifies the dataset. Then Opt-aiNET algorithm was executed for extraction rules from this ANN. Finally extracted rules was decoded.

The classification accuracy of extracted rules for suggested approach is shown in Table 3 with the accuracies which are obtained for the same problem related to other methods in http://www.fizyka.umk.pl/kmk/projects/datasets.html for Breast Cancer database. Table shows that, presented algorithm is the best performed rule extractor.

When suggested rule extraction algorithm was applied to standard ANN, 88.46% accuracy is being reached by using 78 rules as sum; while 92.31% accuracy is being reached by using 59 rules when it was applied to adaptive
This also means increment of accuracy and decrement of decision time when adaptive ANN is used. As can be seen in Table 4, classification accuracy, sensitivity and specificity of the proposed system for Breast Cancer database according to (4.1), (4.2) and (4.3) are 92.31%, 97% and 81.18%, respectively.

It is not possible to show all generated rules due to lack of space. The example of the extracted set of rules for Breast Cancer database have been presented in Tables 5 and 6.

The dataset that is chosen for the second experiment was the Echocardiogram data from the same database (http://www.ics.uci.edu/~mlearn/MLRepository.html). The data set has seven attributes, of which six are continuous and one is Boolean. There are: (1) age at heart attack, (2) pericardial effusion, (3) fractional shortening, (4) e-point septal separation (epss), (5) left ventricular end-diastolic dimension (lvdd), (6) wall motion score and (7) wall motion index. Classes were coded as 0 and 1. The value 0 means patient was either dead after one year or had been followed for less than one year. Value 1 means patient was alive at one year. The attributes have different range values in the database and these ranges of the data can be seen in Table 7.

With the coding scheme shown in Table 8 we had a total of 25 binary inputs. The number of hidden units was taken as four. The AIS was then run with a population of 20 for 5000 generation for each classification. All parameter was
chosen empirically for the best convergence rate between the actual and desired output. This generates five rules for each class. Both the maximum and minimum of output antibodies has been determined and will be translated into rules.

For rule extraction, the same methodology that is mentioned above for Breast Cancer database is followed. The accuracy that is obtained by suggested approach is shown in Table 9 with the accuracies which are obtained for the same problem related to other approach in Keedwell et al. 2000a, 2000b for ECG dataset. The table shows that, presented algorithm is the best performed rule extractor.

Although suggested rule extraction algorithm reaches the same accuracy value when it is applied to standard

Table 6
The example of the extracted set of rules for Class 1 for the Breast Cancer database

<table>
<thead>
<tr>
<th>No</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>If (age ∈ [10, 30) OR age ∈ [40, 70]) &amp; menopause ≠ lt40 &amp; (tumor-size ∈ (4, 9] OR tumor-size ∈ [14, 24] OR tumor-size ∈ (29, 44]) &amp; (inv-nodes ∈ [6, 9) OR inv-nodes ∈ [12, 24]) &amp; node-caps = 1 &amp; deg-malig = 3 Then Class 1</td>
</tr>
<tr>
<td>2</td>
<td>If age ∉ [50, 60] &amp; (tumor-size ∈ (4, 9] OR tumor-size ∈ [14, 29] OR tumor-size ∈ (34, 44]) &amp; inv-nodes ≥ 3 &amp; deg-malig ≠ 1 &amp; breast = left &amp; (breast-quadrant = left-low OR breast-quadrant = right low) &amp; irradiation = no Then Class 1</td>
</tr>
<tr>
<td>3</td>
<td>If age ∈ [40, 60] &amp; menopause = gl40 &amp; (tumor-size ∈ (0, 4] OR tumor-size ∈ [14, 24] OR tumor-size ∈ (39, 49]) &amp; (inv-nodes ∈ [3, 24] &amp; node-caps = 1 &amp; breast-quadrant ≠ left-low &amp; breast-quadrant = central Then Class 1</td>
</tr>
<tr>
<td>4</td>
<td>If (age ∈ [30, 40] &amp; age ∈ [50, 80]) &amp; menopause = gl40 &amp; (tumor-size ∈ (0, 14] OR tumor-size ∈ [19, 34] OR tumor-size ∈ (39, 49]) &amp; inv-nodes ∈ [3, 24] &amp; node-caps = 1 &amp; deg-malig = 3 &amp; (breast-quadrant = left-up OR breast-quadrant = left-low) Then Class 1</td>
</tr>
<tr>
<td>5</td>
<td>If (age ∈ [10, 40] OR age ∈ [60, 70]) &amp; menopause ≠ lt40 &amp; (tumor-size ∈ [0, 9] OR tumor-size ∈ [19, 24] OR tumor-size ∈ (39, 44] OR tumor-size ∈ (49, 59]) &amp; inv-nodes ∉ [6, 12] &amp; deg-malig = 3 &amp; (breast-quadrant = left-low OR breast-quadrant = left-low) Then Class 1</td>
</tr>
</tbody>
</table>

Table 7
Range values and attribute names for ECG database

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age at heart attack</td>
<td>0–86</td>
</tr>
<tr>
<td>Pericardial effusion</td>
<td>0–1 (Boolean)</td>
</tr>
<tr>
<td>Fractional shortening</td>
<td>0–0.61</td>
</tr>
<tr>
<td>Eps</td>
<td>0–40</td>
</tr>
<tr>
<td>Lvdd</td>
<td>0–6.73</td>
</tr>
<tr>
<td>Wall motion score (WMS)</td>
<td>0–39</td>
</tr>
<tr>
<td>Wall motion index (WMI)</td>
<td>0–3</td>
</tr>
<tr>
<td>Alive at 1 (Class)</td>
<td>0–1 (Boolean)</td>
</tr>
</tbody>
</table>

Table 8
Coding of the attributes of ECG dataset for neural network input

<table>
<thead>
<tr>
<th>Attribute</th>
<th>No. of inputs</th>
<th>Sub-intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age at heart attack</td>
<td>4</td>
<td>[0, 50], (50, 60], (60, 70], (70, 86)</td>
</tr>
<tr>
<td>Pericardial effusion</td>
<td>1</td>
<td>[0], [1]</td>
</tr>
<tr>
<td>Fractional shortening</td>
<td>4</td>
<td>[0, 0.15], (0.15, 0.3], (0.3, 0.45], (0.45, 0.61]</td>
</tr>
<tr>
<td>Eps</td>
<td>4</td>
<td>[0, 9], (9, 13], (13, 17], (17, 40]</td>
</tr>
<tr>
<td>Lvdd</td>
<td>4</td>
<td>[0, 4], [4, 5], [5, 6], [6, 6.73]</td>
</tr>
<tr>
<td>Wall motion score (WMS)</td>
<td>4</td>
<td>[0, 10], (10, 13], (13, 17], (17, 39]</td>
</tr>
<tr>
<td>Wall motion index (WMI)</td>
<td>4</td>
<td>[0, 1], (1, 1.5], (1.5, 2], (2, 3]</td>
</tr>
</tbody>
</table>

Table 9
Classification accuracies obtained by our algorithm and other classifiers for ECG dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy, % test</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule extraction from adaptive ANN using AIS</td>
<td>94.59</td>
<td>Suggested approach in this study</td>
</tr>
<tr>
<td>Rule extraction from standard ANN using AIS</td>
<td>94.59</td>
<td>Keedwell et al. (2000a)</td>
</tr>
<tr>
<td>GANN-C</td>
<td>91.89</td>
<td>Keedwell et al. (2000a)</td>
</tr>
</tbody>
</table>

Table 10
Obtained ECG dataset classification accuracy, sensitivity and specificity

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>94.59%</td>
<td>96%</td>
<td>91.7%</td>
</tr>
</tbody>
</table>

For rule extraction, the same methodology that is mentioned above for Breast Cancer database is followed.

The accuracy that is obtained by suggested approach is shown in Table 9 with the accuracies which are obtained for the same problem related to other approach in Keedwell et al. 2000a, 2000b for ECG dataset. The table shows that, presented algorithm is the best performed rule extractor.

Although suggested rule extraction algorithm reaches the same accuracy value when it is applied to standard

Table 11
The example of the extracted set of rules for Class 0 for ECG dataset

<table>
<thead>
<tr>
<th>No</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>If age at heart attack ≤ 60 &amp; Pericardial effusion = 0 &amp; Fractional shortening ≤ 0.45 &amp; Lvdd ≠ (5, 6] &amp; WMS &gt; 10 &amp; WMI ≤ 2 Then Class 0</td>
</tr>
<tr>
<td>2</td>
<td>If pericardial effusion = 0 &amp; (Fractional shortening ∈ (0, 0.15] OR Fractional shortening ∈ (0.3, 0.45]) &amp; Eps notin (9, 13] &amp; Lvdd ≤ 6 &amp; WMI ≤ 1.5 Then Class 0</td>
</tr>
<tr>
<td>3</td>
<td>If age at heart attack &gt;50 &amp; Pericardial effusion = 0 &amp; Fractional shortening &gt;0.15 &amp; WMS ∈ (10, 13] &amp; WMI ≠ (1, 5, 2] Then Class 0</td>
</tr>
<tr>
<td>4</td>
<td>If age at heart attack ∈ [50, 70] &amp; Pericardial effusion = 1 &amp; Fractional shortening &gt;0.15 &amp; (Eps ∈ (9, 13] OR Eps ∈ (17, 40]) &amp; Lvdd &gt; 4 &amp; (WMS ∈ (0, 10] OR WMS ∈ (13, 17]) &amp; WMI ≤ 1.5 Then Class 0</td>
</tr>
<tr>
<td>5</td>
<td>If age at heart attack ∈ [50, 70] &amp; Pericardial effusion = 0 &amp; Fractional shortening ≠ (0.3, 0.45] &amp; Eps ∈ (13, 17] &amp; Lvdd &gt; 4 &amp; WMS &gt; 10 &amp; WMI ≤ 2 Then Class 0</td>
</tr>
</tbody>
</table>
and adaptive ANN, number of rules decreases when adaptive ANN is used. For every class 13 rules occur when standard ANN is used. This number decreases to 12 when adaptive ANN is used and this also means the decision time to become short somewhat.

As it can be seen in Table 10, classification accuracy, sensitivity and specificity of the proposed system for ECG dataset according to (4.1), (4.2) and (4.3) are 94.59%, 96% and 91.7%, respectively. The example of the extracted set of rules for the ECG dataset have been presented in Tables 11 and 12.

7. Conclusion

Mining classification rules is an important task of data mining. In this paper, a new algorithm for extracting comprehensible rules has been presented. This algorithm takes all input attributes into consideration and extracts rules from the trained neural network with adaptive activation function efficiently. This study demonstrates a new neuron-adaptive activation function and the use of AIS algorithms in extracting rules from trained neural networks. The approach for extracting rules from ANN consists of three phases:

1 – Data coding.
2 – Classification of coding data.
3 – Rule extraction.

The presented approach was applied to two real world classification problems. The data were obtained from University of California at Irvine (UCI) machine learning repository. Suggested rule extraction algorithm has been applied to standard and adaptive ANN. Eventually, increment of accuracy and decrement of number of extracted rules has been seen when adaptive ANN is used. This also means decrement of decision time. So, the results of comparison experiments show that proposed approach can generate more accurate rules.

Although the method is successful, it is seen that a great number of rules are formed. In future works, being realized the classification with fewer rules by decreasing this number is aimed.

Acknowledgement

This study is supported by the Scientific Research Projects Unit of Selcuk University.

References


