The Second International Conference "Problems of Cybernetics and Informatics"
September 10-12, 2008, Baku, Azerbaijan. Section #2 "Intellectual Systems"
www.pci2008.science.az/2/20.pdf

EXTRACTING RULES FROM NEURAL NETWORKS USING ARTIFICIAL IMMUNE SYSTEMS

Novruz Allahverdi, Humar Kahramanli

Electronic and Computer Education Department, Selcuk University, Konya, Turkey
{*noval*, *hkahramanli*}*@selcuk.edu.tr*

Neural Networks adjust their internal parameters performing vector mappings from the input to the output space. Although they may achieve high accuracy of classification, the knowledge acquired by such systems is represented in a large number of numerical parameters and network architectures, in a way that is incomprehensible for humans [1]. 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 [2].

In the literature, there are many different approaches for the rule extraction. The one of the first rule extraction techniques from neural networks was proposed by Gallant [3]. He was working on connectionist expert systems. In this work each ANN node represents a conceptual entity. Towell and Shavlik showed how to use ANNs for rule refinement [4]. The algorithm was called SUBSET, which is based on the analysis of the weights that make a specific neuron active. Alexander and Mozer developed a rule extraction method, based on connection weights, that supposes activation functions showing approximately Boolean behavior [5]. Sethi et al. developed a rule extraction method based on the connection weights [6]. Lu et al. proposed an approach for rule extraction from ANNs based on the clustering of hidden unit activation values [7]. Setiono and Leow presented the fast method is based on the relevance of hidden units, considering their information gains [8]. Palade et al. presented a method of rule extraction from ANNs that are based on interval propagation across the network, using a procedure of inverting an ANN [9]. Elalfi et al. presented an algorithm for extracting rules from databases via trained ANN using genetic algorithm [10].

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 [11]. This study is focused on the problem of extracting rules from previously trained ANNs. Study on rule extraction from trained ANN is based on the work of Elalfi et al. [10]. 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.

Every data in dataset are coded as binary string and presented as input to ANN. The following method has been used for coding [10]. 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 substring $\{b_{n1}, b_{n2}, \ldots, b_{nm_n}\}$. If attribute A_n belongs to substring a_i $(i=1,2,\ldots,m_n)$, b_{nj} is givenby:

$$b_{nj} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}, \quad j = 1,2\ldots,m_n \tag{1}$$

Thus, the input vector of ANN can be givenby:

$$X = \bigcup_{n=1}^{N} \bigcup_{i=1}^{m_n} b_{ni} \tag{2}$$

The length of input vector *X* is determined as follows:

$$m = \sum_{n=1}^{N} m_n \tag{3}$$

The dataset 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 sum of weighted input signals for *j*th neuron of hidden layer is calculated as:

$$G_j = \sum_{i=1}^{m} x_i * w_{ij} \ , \ \ j = 1,2....k \tag{4}$$

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. Output of *j*th neuron of hidden layer is calculated as follows:

$$CG_j = \frac{1}{1+e^{-G_j+\theta_j}}, \quad j = 1,2....k \tag{5}$$

where, θ_j is threshold for *j*th neuron of hidden layer. The sum of weighted input signals for output neuron is calculated as:

$$O = \sum_{j=1}^{k} CG_j * v_j \tag{6}$$

where, v_j is the weight between *j*th neuron of hidden layer and output neuron. Output neuron is calculated as follows:

$$C = \frac{1}{1+e^{-O+\xi}} \tag{7}$$

where ξ is threshold for output neuron.
Thus, we obtain a nonlinear function *C* that depends on *X*.

$$C(X) = \left(1+e^{-\sum_{j=1}^{k} \left(1+e^{-\sum_{i=1}^{m} x_i * w_{ij} + \theta_j} \right)^{-1} * v_j + \xi} \right)^{-1} \tag{8}$$

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.

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 [10]:

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

Two different datasets are used in this study for application part. The used datasets were medical datasets consisting of Statlog Heart Dataset and Postoperative Patient Dataset. The dataset chosen for this first experiment was the Statlog Heart Dataset from UCI Machine Learning Repository [12]. This dataset contains 270 samples that were taken from patients with heart problem. The dataset has 13 attributes. The attributes have different range values in the

database and these ranges of the data can be seen in Table 1. The sub-intervals used to coding of attribute values are summarized in the third column of the Table 1. Each bit of a string was either 0 or 1 depending on which sub-interval the original value was located to. For example, a resting blood pressure value at 130 would be coded as {0, 1, 0}. Sub-intervals for the blood pressure are: [94, 115], (115, 140), (140, 200]. So the binary string for blood pressure has tree bits. Because of $130 \in (115,\ \ 140)$, second bit of this string equals to 1 and all others equal to 0. A resting ecg result of 2 would be coded as {0, 0, 1}.

Table 1. Attribute names, range values and coding of the attributes for Statlog Heart Disease database

Attribute	Range	Subintervals	No. Of inputs
age	[29, 77]	[29, 50], (50, 60), [60, 77]	3
sex	{0, 1}	{0}, {1}	2
chest pain type	{1, 2, 3, 4}	{1}, {2}, {3}, {4}	4
resting blood pressure	[94, 200]	[94, 115], (115, 140), [140, 200]	3
serum cholesterol in mg/dl	[126, 564]	[126, 220], (220, 300), [300, 564]	3
fasting blood sugar > 120 mg/dl	{0, 1}	{0}, {1}	2
resting electrocardiographic results	{0, 1, 2}	{0}, {1}, {2}	3
maximum heart rate achieved	[71, 202]	[71, 120], (120, 150), [150, 202]	3
exercise induced angina	{0, 1}	{0}, {1}	2
oldpeak	[0, 6]	[0, 0.6], (0.6, 1.6), [1.6, 6]	3
the slope of the peak exercise ST segment	{1, 2, 3}	{1}, {2}, {3}	3
number of major vessels (0-3) colored by flouroscopy	{0, 1, 2, 3}	{0}, {1}, {2}, {3}	4
thal	{3, 6, 7}	{3}, {6}, {7}	3

With the coding scheme shown in Table 1 we had a total of 38 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 five. The allowable error equals to 0.0001. The Opt-aiNET algorithm has been applied to solve the equation *C(X)* (see 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 10000 generations for each classification. All parameters were chosen empirically for the best convergence rate between the actual and desired output. This generates 43 rules for class 1 and 44 rules for class 2. Both of the maximum and minimum of output antibodies have been determined and will be translated into rules.

In summary, for rule extraction firstly, ANN was designed which classifies the dataset. Then Opt-aiNET algorithm was executed for extraction rules from this ANN. Finally the extracted rules were decoded. The classification accuracy of the extracted rules for suggested approach is 95.19%. Accuracies, are obtained for the same problem related to the other methods in the literature [13] are between 58.5% and 76.7%.

The dataset that is chosen for the second experiment was the Postoperative Patient Data from the same database [12]. The dataset has 8 attributes. Classes were coded as 0 and 1. The value 0 means patient prepared to go home. The value 1 means patient sent to Intensive Care Unit or to general hospital floor. The attributes have different values in the database and these ranges of the data can be seen in Table 2. With the coding scheme shown in Table 2 we had a total of 26 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 four. The allowable error equals to 0.0001.

The Opt-aiNET was then run with a population of 20 for 40000 generations for each classification. This generates 10 rules for class 0 and 12 rules for class 1. Both of the maximum and minimum of output antibodies have been determined and will be translated into rules. The

classification accuracy of the extracted rules for suggested approach is 83.33%. The example of the extracted set of rules for the Postoperative Patient Database has been presented in Table 3.

Table 2. Attribute names, range values and coding of the attributes for Postoperative Patient Dataset

Attribute	Values	Subintervals	No. Of inputs
L-CORE	High, mid, low	{High}, {mid}, {low}	3
L-SURF	High, mid, low	{High}, {mid}, {low}	3
L-O2	Excellent, good, fair, poor	{Excellent}, {good}, {fair}, {poor}	4
L-BP	High, mid, low	{High}, {mid}, {low}	3
SURF-STBL	stable, mod-stable, unstable	{stable}, {mod-stable}, {unstable}	3
CORE-STBL	stable, mod-stable, unstable	{stable}, {mod-stable}, {unstable}	3
BP-STBL	stable, mod-stable, unstable	{stable}, {mod-stable}, {unstable}	3
COMFORT	5, 7, 10, 15	{5}, {7}, {10}, {15}	4

Table 3. The example of the extracted set of rules for Postoperative Patient Dataset

1- If L-core≠low & L-surf≠high & L-02≠poor & L-bp≠low & Core-stbl=unstable & Bp-stbl=stable Then Class 0

1- If L-core ≠mid & L-surf≠high & (L-02=good or L-02=poor) & L-bp=mid & Surf-stbl≠unstable & Core-stbl=stable & Bp-stbl=mod-stable & Comfort≠10 Then Class 0

2- If L-core ≠low & L-02≠fair & L-bp≠mid & Bp-stbl≠stable & (Comfort=10 or Comfort=15) Then Class 1

3- If L-core =mid & L-02≠fair & Surf-stbl=unstable & Core-stbl=stable & Bp-stbl=stable & Comfort=15 Then Class 1

Literature

1. W. Duch, R. Adamczak, K. Grabczwski. A new methodology of extraction, optimization and application of crisp and fuzzy logical rules. IEEE Transactions on Neural Networks, Vol 11, no 2, (2000).
2. S.H. Huang, H. Xing. Extract intelligeble and concise fuzzy rules from neural networks. Fuzzy Sets and Systems 132, (2002), 233-243.
3. S.I. Gallant. Connection expert systems. Commun. ACM 31 (2), (1988), 152-169.
4. G.G. Towell, J. Shavlik. Extracting refined rules from knowledge based neural Networks. Mach. Learn, 13, (1993), 71-101.
5. J.A. Alexander & M.C. Mozer. Template-based algorithm for connectionist rule extraction. in:G. Tesauro, D. Touetzky, T. Leen (Eds.). Advances in Neural Information Processing Systems. Vol.7 MIT Pres, Cambridge, MA, (1995).
6. I. Sethi & J. Yoo. Multi-valued logic mapping of neurons in feed-forward Networks. Eng. Intel. Syst, 4 (4), (1996), 243-153.
7. H. Lu, R. Setiono, H. Liu, Effective data mining using neural Networks. IEEE Trans Knowledge Data Eng 8 (6), (1996), 957-961.
8. R. Setiono, K. Leow. FERNN: an algorithm for fast extraction of rules from neural Networks. Appl. Intel, 12 (1-2), (2000), 15-25.
9. V. Palade, D. Neagu, G. Puscasu. Rule extraction from neural Networks by interval propogation. Proceedings of the Fourth IEEE International Conference on Knowledge-Based Intelligent Engineering Systems, Brighton-UK, (2000), 217-220.
10. A.E. Elalfi, R. Haque, M.E. Elalami. Extracting rules from trained neural network using GA for managing E-business. Applied Soft Computing 4, (2004), 65-77.
11. E.R. Hruschka, N.F.F. Ebecken. Extracting rules from multilayer perceptrons in classification problems: A clustering-based approach. Neurocomputing 70, (2006), 384-397.
12. Hettich, S. and Bay, S. D. (1999). The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science, Last accessed: 27 March 2008.
13. www.fizyka.umk.pl/kmk/projects/datasets.html, Last accessed: May 2008.