

# Learning M-of-N Concepts for Medical Diagnosis Using Neural Networks

**Yoichi Hayashi**

Department of Computer Science

Meiji University

1-1-1 Higashimita, Tama-ku, Kawasaki 214-8571, Japan

(Email: hayashiy@cs.meiji.ac.jp)

**Rudy Setiono**

School of Computing

National University of Singapore

Lower Kent Ridge Road, Singapore 119260

(Email: rudys@comp.nus.edu.sg)

**Katsumi Yoshida**

Department of Preventive Medicine

St. Marianna University School of Medicine

2-16-1 Sugao, Miyamae-ku, Kawasaki 216-8511, Japan

(Email: k2yosida@marianna-u.ac.jp)

## Abstract

Records in a medical data set may be best characterized by M-of-N concepts. For example, if a patient shows at least 2 of the 4 symptoms, then he is likely to be diagnosed as having a certain illness. In this paper, we describe how feedforward neural networks can be used to learn such concepts. We train a network where each input in the data can only have one of the two possible values, -1 or 1 and apply the hyperbolic tangent function to each connection from the input layer to the hidden layer of the network before the hidden unit activations are computed. By applying this squashing function, the activation values at the hidden units are effectively computed as the hyperbolic tangent (or the sigmoid) of the weighted inputs, where the weights have magnitudes that are near one. By restricting the inputs and the weights to binary values either -1 or 1, the extraction of the M-of-N concepts from the networks becomes trivial. We show how this approach can be used to learn concise and accurate M-of-N concepts for the diagnosis of hepatobiliary disorders.

# 1 Introduction

Neural networks have been extensively used as tools that aid medical diagnosis [2, 3, 4, 7, 8, 10]. One of the network structures that has been widely used is the feedforward neural network, where network connections are allowed only between units in one layer and those in the next layer. In many applications, a set of rules that explains the classification process of a trained network may be required. Restricting the network to have only one layer of hidden units and to have network connections from the input units to the hidden units and the hidden units to the output units greatly reduces the complexity of the rule extraction process. The decompositional approach to rule extraction [1, 17] splits the process into two steps. First, rules that explain the network outputs are generated in terms of the activation values of the hidden units. Second, rules that explain the hidden unit activation values are generated in terms of the network inputs. When these two sets of rules are merged, we obtain a set of rules that explains the network classification in terms of its inputs.

Many rule extraction algorithms generate DNF representation of the network [12, 13, 16, 18]. Under DNF representation, the classification concept is expressed as the disjunction of one or more subconcepts. However, we can expect to be able to characterize the records in a medical data set more concisely as M-of-N concepts. M-of-N concepts are expressed as rules of the following form:

if (exactly, not more than, at least) M of the N conditions are satisfied, then ...

For example, if a patient shows 2 or more of the 4 symptoms, then he is likely to be diagnosed with a certain illness. Due to their parallel processing of the inputs, neural networks are particularly suitable for use as a tool to learn this kind of concepts.

There have been algorithms developed recently for extracting M-of-N rules from neural networks. One such algorithm is MofN [18] which extracts such rules from Knowledge Based Artificial Neural Networks. Groups of network connections are checked by the algorithm for their contribution to the activation of a unit. This is done by clustering the network connections. The weights of connections in a cluster are then replaced by their average weight. Clusters with small average and a few connections are checked for possible elimination since their removal are not likely to have any effect on the network classification. A rule is formed for each hidden and output unit. This rule consists of a threshold and the weighted antecedents of the remaining connections.

In this paper, we present our results from applying the algorithm MofN3 [15] to the problem of diagnosing hepatobiliary disorders. MofN3 is an algorithm which makes use of neural networks

to learn M-of-N concepts. By assuming that both the input data and the network connections from the input units to the hidden units to be binary valued -1 or 1, the process of extracting M-of-N rules from a trained network becomes trivial. The algorithm has been shown to be very effective in learning M-of-N rules from several publicly available data sets [15]. The hepatobiliary disorder data for this study have been collected from a total of 536 patients who were admitted to a university-affiliated hospital in Japan. Nine real-valued biomedical test measurements had been obtained from each of these patients. In addition to these nine measurements, the sex of each patient was also recorded.

The outline of the paper is as follows. In Section 2, we describe our algorithm MofN3. The results from applying MofN3 to the hepatobiliary disorders data set are presented in Section 3. Comparison of MofN3 results with those from other classification methods is presented in Section 4. Section 5 concludes the paper.

## 2 The MofN3 algorithm

The outline of our algorithm is as follows.

### Algorithm MofN3 (M-of-N rules from Neural Network)

1. Train and prune a feedforward neural network with one hidden layer.
2. Cluster the hidden unit activation values of the pruned network.
3. Generate classification rules in terms of the clustered activation values.
4. Replace the conditions of the rules generated in Step 3 by M-of-N conditions.

In order to simplify the process of rule extraction, we impose the following two assumptions on the network and its input:

Assumption 1. All attributes of the data are binary valued, -1 or 1.

Assumption 2. After the network is pruned, the network weights for the connections from the input units to the hidden units are sufficiently close to 1 in magnitude.

Assumption 1 can always be satisfied by recoding the data. To check if a pruned network satisfies Assumption 2, we replace all relevant connections by 1 if they are positive and by -1 if they are negative. Assumption 2 is satisfied if the classification accuracy of the network is not affected after the weights have been replaced by unit weights, -1 or 1.

We train a network such that the following error function is minimized

$$F(w, v) = - \sum_{p=1}^P \sum_{c=1}^C (t_{pc} \log \beta_{pc} + (1 - t_{pc}) \log(1 - \beta_{pc})) + P(w, v), \quad (1)$$

where  $P$  and  $C$  are the number of samples and the number of output units, respectively. For pattern  $\mathbf{x}_p$  and its target output  $\mathbf{t}_p$ , the network output at unit  $c$  and the activation values at hidden unit  $h$  are computed as follows

$$\beta_{pc} = \sigma \left( \sum_{h=1}^H (\alpha_{ph} v_{hc}) + \eta_c \right) \quad (2)$$

$$\alpha_{ph} = \tanh \left( \left( \sum_{n=1}^N x_{pn} \times \tanh(w_{hn}) \right) + \tau_h \right), \quad (3)$$

where  $N$  is the number of input units,  $w_{hn}$  is the weight of the connection from the  $n$ -th input unit to the  $h$ -th hidden unit,  $v_{hc}$  is the weight of the connection from the  $h$ -th hidden unit to the  $c$ -th output unit,  $\eta_c$  is the bias of output unit  $c$  and  $\tau_h$  is the bias of hidden unit  $h$ . The function  $\sigma(x)$  is the sigmoid function  $\sigma(x) = 1/(1 + e^{-x})$

$F(w, v)$  is the cross-entropy error measure augmented by the penalty term

$$P(w, v) = \epsilon \sum_{h=1}^H \left( \sum_{n=1}^N \frac{\tanh(w_{hn})^2}{1 + \tanh(w_{hn})^2} + \sum_{c=1}^C v_{hc}^2 \right) \quad (4)$$

where  $\epsilon$  is a positive penalty parameter. The weights of the connections from the input layer to the hidden layer are penalized so that smaller weights decay more rapidly than larger ones. The output layer are given a quadratic penalty to prevent these weights from getting too large as the pruning algorithm removes redundant connections based on the magnitudes of their weights. Connections with sufficiently small weights can be removed without affecting the classification accuracy of the network. After these connections are removed, the network is retrained. The process is repeated by checking if there is any connection in the retrained network that meets the criteria for removal. The criteria for removing network connections and other details of the network pruning algorithm can be found in [11].

We apply the hyperbolic tangent function

$$\tanh(x) = (e^x - e^{-x}) / (e^x + e^{-x}), \quad (5)$$

to each connection from an input unit to a hidden unit in (3) with the hope that when the training terminates, we have a trained network with weights from the input units to the hidden units that are close to 1 in magnitude.

Before rules are extracted, we cluster the hidden unit activation values in Step 2 in order to have the network classifications determined by the combinations of a small number of discretized

activation values. It can be expected that few discretized hidden unit activation values will result in a more compact set of rules. We generate classification rules where the conditions of the rules involve these discretized activation values. Once the rules are generated, we replace the conditions of the rules by M-of-N rules.

Clustering the hidden unit activation values in Step 2 of the algorithm MofN3 is equivalent to dividing the interval  $(-1, 1)$  into disjoint subintervals. Thus, the conditions of the rules generated in the next step are of the following form

$$\alpha \leq U \tag{6}$$

$$\alpha > L \tag{7}$$

or a conjunction of the above

$$L < \alpha \leq U \tag{8}$$

with  $-1 < L < U < 1$ .

The activation  $\alpha$  has been computed as (3)

$$\alpha = \tanh \left( \left( \sum_n x_n \times \tanh(w_n) \right) + \tau \right)$$

The pattern index and the hidden unit index have been dropped in the above equation for simplicity. Let us define

- $\bar{L} = \lceil \tanh^{-1}(L) - \tau \rceil$  and  $\bar{U} = \lfloor \tanh^{-1}(U) - \tau \rfloor$ .
- $\bar{N}$ : the number of input connections connected to the hidden unit.
- $\bar{x}$ : not  $x$ , hence by Assumption 1,  $\bar{x} = -x$ .
- $\mathcal{W}_+$ : the set of connections with positive weight.
- $\mathcal{W}_-$ : the set of connections with negative weight.
- $\mathcal{I}_+$ : the set of inputs with positive connections.
- $\mathcal{I}_-$ : the set of inputs with negative connections.

We have the following

$$L < \alpha \leq U$$

$$\Updownarrow$$

$$\tanh^{-1}(L) < \sum_n x_n \times \tanh(w_n) + \tau \leq \tanh^{-1}(U)$$

$$\begin{aligned}
& \Updownarrow \\
\tanh^{-1}(L) - \tau & < \sum_{w_n \in \mathcal{W}_+} x_n \times \tanh(w_n) + \sum_{w_n \in \mathcal{W}_-} x_n \times \tanh(w_n) \leq \tanh^{-1}(U) - \tau \\
& \Updownarrow \\
\tanh^{-1}(L) - \tau & < \sum_{w_n \in \mathcal{W}_+} x_n - \sum_{w_n \in \mathcal{W}_-} x_n \leq \tanh^{-1}(U) - \tau \tag{9} \\
& \Updownarrow \\
\bar{L} & \leq \sum_{w_n \in \mathcal{W}_+} x_n + \sum_{w_n \in \mathcal{W}_-} \bar{x}_n \leq \bar{U} \tag{10}
\end{aligned}$$

Inequality (9) follows from Assumption 2, while inequality (10) follows from the definition of  $\bar{x}$ ,  $\bar{L}$  and  $\bar{U}$ , and Assumption 1.

Let  $\mathcal{C}$  be the disjunction

$$\left( \bigvee_{I_n \in \mathcal{I}_+} (I_n = 1) \right) \vee \left( \bigvee_{I_n \in \mathcal{I}_-} (I_n = -1) \right)$$

The extracted rule is: if  $M$  of the  $\bar{N}$  conditions of  $\mathcal{C}$  are satisfied, then  $L < \alpha \leq U$ . Let  $\bar{M}$  be the number of conditions of  $\mathcal{C}$  that are **not** satisfied. It follows from (10), that we must have

$$\bar{L} \leq M - \bar{M} \leq \bar{U} \tag{11}$$

Since  $M + \bar{M} = \bar{N}$ , if we let  $M = \lfloor \frac{1}{2}(\bar{N} + \bar{U}) \rfloor$  and  $\bar{M} = \bar{N} - M$ , we have a pair of values  $(M, \bar{M})$  that satisfies condition (11). All other pairs of  $(M, \bar{M})$  that satisfy this condition can be found by simply decreasing  $M$  by one and increasing  $\bar{M}$  by one as long as  $M - \bar{M} \geq \bar{L}$ ,  $M \geq 0$ .

### 3 Diagnosis of hepatobiliary disorders

Each record in the hepatobiliary disorder database consists of the patient's sex and the results of nine biochemical tests for hepatobiliary disorders [5]. The measurements from these tests are Glutamic Oxaloacetic Transaminase (GOT), Glutamic Pyruvic Transaminase (GPT), Lactate Dehydrogenase (LDH), Gamma Glutamyl Transpeptidase (GGT), Blood Urea Nitrogen (BUN), Mean Corpuscular Volume of red blood cells (MCV), Mean Corpuscular Hemoglobin (MCH), Total Bilirubin (TBil) and Creatinine (CRTNN). The unit of the measurements, the minimum and maximum values as well as the two cut-off levels used to discretize each measurement are shown in Table 1. Measurements from a total of 536 patients had been collected. The patients were clinically and pathologically diagnosed by physicians at a university-affiliated hospital in

Table 1

The nine measurements of the hepatobiliary data set

Measurement	Unit	min. value	max. value	cut-off level	
				1	2
GOT	Karmen unit	8	4356	40	100
GPT	Karmen unit	3	1124	40	100
LDH	iu/l	179	6327	500	700
GGT	mu/ml	4	3075	60	100
BUN	mg/dl	3.3	91.0	15	20
MCV	fl	66.7	160.5	90	100
MCH	pg	20.3	52.5	30	36
TBil	mg/dl	0.1	37.0	2.0	5.0
CRTNN	mg/dl	0.4	4.3	0.9	1.3

Japan and each was diagnosed as suffering from one of the four disorders: Alcoholic Liver Damage (ALD), Primary Hepatoma (PH), Liver Cirrhosis (LC) and Cholelithiasis (C).

The data set was divided randomly into a training set and a test set. The training set consists of 373 records and the test set consists of the remaining 163 records. The number of output units is 4 and the number of hidden units is 5. Records from patients diagnosed as having ALD, PH, LC and C are given the target output of  $(1, 0, 0, 0)$ ,  $(0, 1, 0, 0)$ ,  $(0, 0, 1, 0)$  and  $(0, 0, 0, 1)$ , respectively. Since each continuous measurement is divided into three subintervals, three input units are needed for each measurement. The thermometer coding is used. For example, a GOT value of 8 is below the first cut-off level of 40. It falls in the first subinterval and is coded in binary as  $(-1, -1, 1)$ . GOT values that fall in the second and the third subintervals are represented as  $(-1, 1, 1)$  and  $(1, 1, 1)$ , respectively. Hence, when binary encoding of the data is used, the total number of input units needed for the network is  $9 \times 3 + 1 + 1 = 29$ . One input unit is for the sex of the patients and an additional input unit is used for the bias (or threshold) at the hidden units. The input value for the last input unit is set to 1 for all input samples.

We have used the same two criteria for measuring correctness of the network classification as the criteria used in previous studies [5, 6]. Under criterion 1, if the highest output value from the network is found in the output unit that corresponds to the position of 1 in the binary encoded target output of the record, then the diagnosis is correct. Criterion 2 is more relaxed

than criterion 1 as we also consider the second highest network output. Under this condition, a sample with target ALD for example, is considered to be correctly classified if the highest or second highest network output value is found in output unit 1.

Thirty neural networks were trained using different random initial weights. Connections were removed as long as the networks could still correctly classify at least 84% of the training samples using criterion 2. Two of the pruned networks are selected for rule extraction.

*Example 1.*

One of the smallest pruned networks only has 3 hidden units and 16 connections left. Only 4 connections are from the input units to the hidden units, the rest connect units in the hidden layer to the output units.

We save the activation values of the training samples that have been correctly classified by the network. Instead of applying two separate algorithms for discretization and rule generation, we have used C4.5 [9] to directly generate classification rules from the hidden unit activation values. For data with continuous attributes, C4.5 computes a measure called the information gain to determine the best split levels for these attribute values as it builds a classification tree. The input to C4.5 is a data file consisting the activation values of 315 patterns that have been correctly classified by the network. The dimensionality of these patterns is 3, the number of hidden units left in the network after pruning. To each of these patterns we assign one of the possible 12 combinations of highest and second highest network outputs. For example, if the the highest network output is in the first output unit and the second highest is the the second output unit, then the pattern will be assigned a target value of 1. A predicted value of 1 given by the rules would correspond to ALD as the first diagnosis choice and PH as a possible alternative diagnosis.

The rules generated by C4.5 from the hidden activations of the samples that have been correctly classified by the pruned network are as follows

**Rule 1:** if  $\alpha_1 \leq 0$  and  $\alpha_2 > -0.76$  and  $\alpha_3 > -0.76$ , then predict (first choice = PH, second choice = ALD)

**Rule 2:** else if  $\alpha_2 > -0.76$  and  $\alpha_3 \leq -0.76$ , then predict (C, ALD)

**Rule 3:** else if  $\alpha_1 > 0$  and  $\alpha_2 > -0.76$ , then predict (PH, C)

**Rule 4:** else if  $\alpha_2 \leq -0.76$  and  $\alpha_3 > -0.76$ , then predict (LC, PH)

**Rule 5:** else if  $\alpha_2 \leq -0.76$  and  $\alpha_3 \leq -0.76$ , then predict (C, LC)

**Default rule:** predict (PH,ALD)

We now examine the connections from the input units to the 3 hidden units and explain how the rule conditions above can be replaced by their equivalent M-of-N rules involving the inputs.

- Hidden unit 1 is connected to input  $I2$  and  $I6$ , the weights of the connections are -1 and 1, respectively. We would like to know the input combinations that produce  $\alpha_1 \leq 0$  for Rule 1 and  $\alpha_1 > 0$  for Rule 3. We first compute  $\tanh^{-1}(0) = 0$  and  $\bar{U} = 0$ . We have  $M = \lfloor \frac{1}{2}(\bar{N} + \bar{U}) \rfloor = \lfloor \frac{1}{2}(2 + 0) \rfloor = 1$ . We can conclude that  $\alpha_1 \leq 0$  if and only if at most one of the two conditions ( $I2 = -1, I6 = 1$ ) is satisfied. Also,  $\alpha_1 > 0$  if and only if both conditions ( $I2 = -1, I6 = 1$ ) are satisfied.
- Hidden unit 2 is connected to only input  $I12$  with connection weight equal to 1. Hence,  $\alpha_2 > -0.76$  if and only if  $I12 = 1$ .
- Hidden unit 3 is connected to input unit  $I3$  with connection weight equal to 1. We have  $\alpha_3 \leq -0.76$  if and only if  $I3 = -1$ .

The relationships between these inputs and the original measurements are as follows:

- $I2 = 1$  if and only if  $GOT \geq 100$
- $I3 = 1$  if and only if  $GOT \geq 40$
- $I6 = 1$  if and only if  $GPT \geq 40$
- $I12 = 1$  if and only if  $GGT \geq 60$

Hence, the classification rules in terms of the original input attributes of the data are as follows:

**Rule set 1:**

**Rule 1:** if ((not more than 1 of ( $GOT < 100, GPT \geq 40$ )) and ( $GGT \geq 60$ ) and ( $GOT \geq 40$ )) then predict (PH, ALD)

**Rule 2:** else if (( $GGT \geq 60$ ) and ( $GOT < 40$ )) then predict (C, ALD)

**Rule 3:** else if (( $GOT < 100$ ) and ( $GPT \geq 40$ ) and ( $GGT \geq 60$ )) then predict (PH, C)

**Rule 4:** else if (( $GGT < 60$ ) and ( $GOT \geq 40$ )) then predict (LC, PH)

**Rule 5:** else if (( $GGT < 60$ ) and ( $GOT > 40$ )) then predict (C, LC)

**Default rule:** predict (PH,ALD) predict (PH, ALD)

Since 2 of the 3 conditions of Rule 1 involve the input GOT, the rule may be easier to describe by splitting it into 2 rules:

**Rule 1a:** if  $((\text{GOT} \geq 100) \text{ and } (\text{GGT} \geq 60))$  then predict (PH, ALD)

**Rule 1b:** else if  $((\text{GOT} \geq 40) \text{ and } (\text{GOT} < 100) \text{ and } (\text{GGT} \geq 60) \text{ and } (\text{GPT} < 40))$  then predict (PH, ALD)

The accuracy of the rules is the same as that of the pruned networks, 84.45% and 85.28% on the training data set and on the test data set, respectively.

*Example 2.*

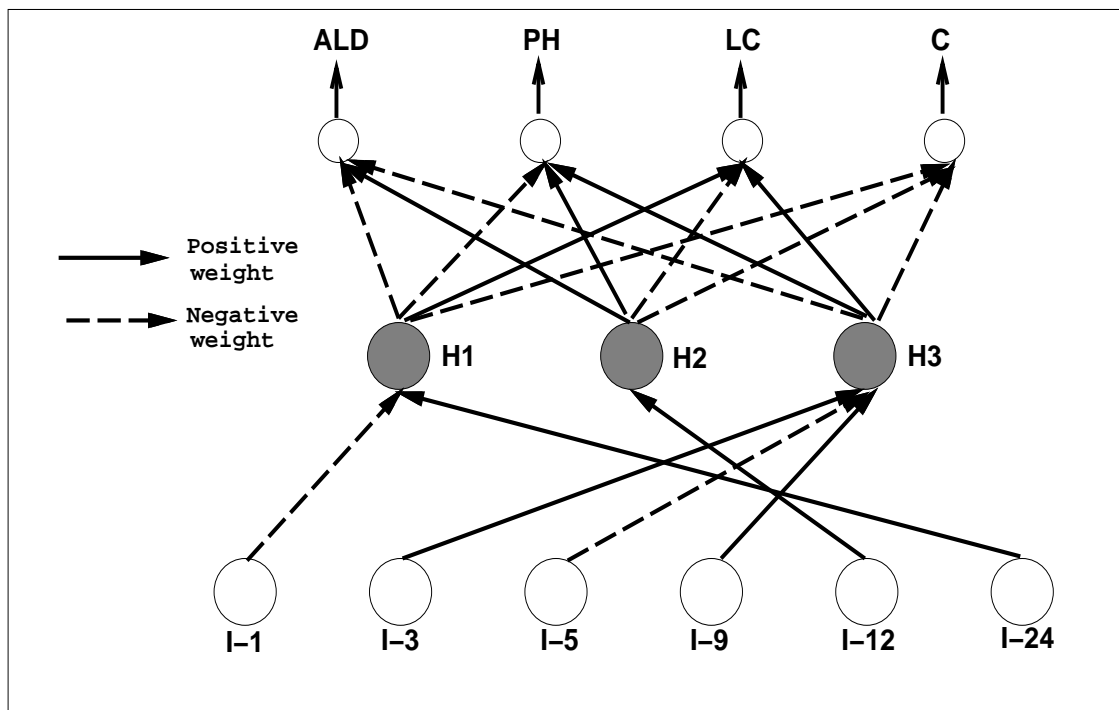


Figure 1: A pruned neural network with unit weights for the connections between units in the input layer and hidden layer. Its classification accuracy rates on the training and test data sets are 86.06% and 87.12%, respectively.

Six network inputs are still present in the pruned network that we use to extract a second set of rule from the hepatobiliary data set. These inputs are  $I1, I3, I5, I9, I12$  and  $I24$  (Figure 1). Three hidden units remain in the pruned network:

- Hidden unit 1 is connected to inputs  $I1$  and  $I24$ , with connection weights of -1 and 1, respectively.

- Hidden unit 2 is connected to input  $I_{12}$  with connection weight equals to 1.
- Hidden unit 3 is connected to inputs  $I_3, I_5$  and  $I_9$ , with connection weights of 1, -1 and 1, respectively.

The relationships between these inputs and the original measurements are as follows:

- $I_1 = 1$  if and only the patient is male.
- $I_3 = 1$  if and only if  $GOT \geq 40$
- $I_5 = 1$  if and only if  $GPT \geq 100$
- $I_9 = 1$  if and only if  $LDH \geq 500$
- $I_{12} = 1$  if and only if  $GGT \geq 60$
- $I_{24} = 1$  if and only if  $TBil \geq 2.0$

The rules generated by C4.5 are as follows

**Rule 1:** if  $\alpha_1 \leq 0$  and  $\alpha_2 > -0.76$  and  $\alpha_3 \leq -0.76$ , then predict (ALD,C)

**Rule 2:** else if  $\alpha_1 \leq 0$  and  $\alpha_2 > -0.76$  and  $\alpha_3 > -0.76$ , then predict (PH,ALD)

**Rule 3:** else if  $\alpha_2 \leq -0.76$  and  $\alpha_3 > -0.76$ , then predict (LC,PH)

**Rule 4:** else if  $\alpha_1 > 0$  and  $\alpha_2 > -0.76$  and  $\alpha_3 > -0.76$ , then predict (PH,LC)

**Rule 5:** else if  $\alpha_1 > -0.96$  and  $\alpha_2 \leq -0.76$  and  $\alpha_3 > -0.76$ , then predict (LC,C)

**Rule 6:** else if  $\alpha_1 \leq -0.96$  and  $\alpha_2 \leq -0.76$  and  $\alpha_3 \leq -0.76$ , then predict (C,LC)

**Default rule:** predict (PH,ALD)

Consider the first condition of Rule 1:  $\alpha_1 \leq 0$ . In order to replace the conditions of the above rule by M-of-N conditions, we must first compute  $\tanh^{-1}(0) = 0$ . Hence,  $\overline{U} = \lfloor 0 \rfloor = 0$ . We have  $M = \lfloor \frac{1}{2}(\overline{N} + \overline{U}) \rfloor = \lfloor \frac{1}{2}(2 + 0) \rfloor = 1$ . We can conclude that  $\alpha_1 \leq 0$  if and only if at most one of the two conditions ( $I_1 = -1, I_{24} = 1$ ) is satisfied.

The second condition of Rule 1 is  $\alpha_2 > -0.76$ . Since the second hidden unit is connected only input  $I_{12}$  with connection weight of +1, it is straightforward to conclude that the condition is satisfied if and only if  $I_{12} = 1$  or ( $GGT \geq 60$ ).

The activation values at hidden unit 3 are determined by inputs  $I_3, I_5$  and  $I_9$ . For the third condition of Rule 1, we compute  $\tanh^{-1}(-0.76) = -0.996$ ,  $\overline{U} = \lfloor -0.996 \rfloor = -1$ ,  $M =$

$\lfloor \frac{1}{2}(\overline{N} + \overline{U}) \rfloor = \lfloor \frac{1}{2}(3 - 1) \rfloor = 1$  and  $\overline{M} = 3 - 1 = 2$ . Hence,  $\alpha_3 \leq -0.76$  if and only if at most one of the conditions ( $I3 = 1, I5 = -1, I9 = 1$ ) is satisfied. Hence,  $\alpha_3 \leq -0.76$  if and only if at most one of the conditions ( $GOT \geq 40, GPT < 100, LDH \geq 500$ ) is satisfied.

Let us denote  $\mathcal{C}_1 = (Female, Tbil \geq 2.0), \mathcal{C}_2 = (GGT \geq 60), \mathcal{C}_3 = (GOT \geq 40, GPT < 100, LDH \geq 500)$  and let  $S_k$  be the number of conditions in  $\mathcal{C}_k$  that is satisfied by a sample. The complete set of M-of-N rules from the neural network is then

**Rule set 2:**

**Rule 1:** if  $\mathcal{S}_1 \leq 1$  and  $\mathcal{S}_2 = 1$  and  $\mathcal{S}_3 \leq 1$ , then predict (ALD,C)

**Rule 2:** else if  $\mathcal{S}_1 \leq 1$  and  $\mathcal{S}_2 = 1$  and  $\mathcal{S}_3 > 1$ , then predict (PH,ALD)

**Rule 3:** else if  $\mathcal{S}_2 = 0$  and  $\mathcal{S}_3 > 1$ , then predict (LC,PH)

**Rule 4:** else if  $\mathcal{S}_1 = 2$  and  $\mathcal{S}_2 = 1$  and  $\mathcal{S}_3 > 1$ , then predict (PH,LC)

**Rule 5:** else if  $\mathcal{S}_1 > 0$  and  $\mathcal{S}_2 = 0$  and  $\mathcal{S}_3 \leq 1$ , then predict (LC,C)

**Rule 6:** else if  $\mathcal{S}_1 = 0$  and  $\mathcal{S}_2 = 0$  and  $\mathcal{S}_3 \leq 1$ , then predict (C,LC)

**Default Rule:** (PH,ALD)

The above set of rules correctly classifies 87.12% of the samples in the test data set, exactly the same as the predictive accuracy of the pruned network from which the rules are extracted. We compare the accuracy and the complexity of this rule set to those from other methods in the next section.

## 4 Comparison with other methods

The predictive accuracy rates obtained on the same hepatobiliary disorder data set by several other classification algorithms are shown in Table 2. In the columns C/N in the table, we show the number of correct classifications (C) and the total number of patterns in a class (N). The methods that have been used in the past include linear discriminant analysis (LDA) and fuzzy neural networks (Fuzzy NN) [5]. We have also applied our neural network rule extraction methods NeuroRule and NeuroLinear [14, 19]. Fuzzy NN extracts fuzzy if-then rules from a network that has been trained using fuzzified input of the data. NeuroRule generates symbolic classification rules from the discretized data, while NeuroLinear generates classification rules that involve one or more linear discriminant functions in the each rule condition.

Table 2.

Comparison of the accuracy rates obtained by various methods for the diagnosis of hepatobiliary disorders.

	Linear DA		Fuzzy NN		NeuroRule		NeuroLinear		MofN3	
	C/N	(%)	C/N	(%)	C/N	(%)	C/N	(%)	C/N	(%)
ALD	19/33	57.6	23/33	69.7	29/33	87.9	32/33	97.0	31/33	93.9
PH	33/51	64.7	42/51	82.4	47/51	92.2	50/51	98.0	48/41	94.1
LC	23/35	65.7	25/35	71.4	28/35	80.0	26/35	74.3	28/35	80.0
C	28/44	63.6	36/44	81.8	40/44	90.9	39/44	88.6	35/44	79.6
Total	103/163	63.2	126/163	77.3	144/163	88.3	147/163	90.2	142/163	87.12

The performance of MofN3 rules is slightly lower than that of NeuroRule and NeuroLinear, but higher than that of LDA and FNN. NeuroLinear achieves the best performance. This could be attributed to the fact that NeuroLinear is applied on the original data set and does not require discretization of the input attributes. MofN3 accuracy is not as high as those of NeuroRule and NeuroLinear, however, the number of rules extracted by this method is smaller.

Table 3. Comparison of the rules extracted by NeuroRule, NeuroLinear and MofN3

Method	No. of rules	No. of conditions	No. rules/condition
NeuroRule	14	39	2.79
NeuroLinear	11	25	2.08
MofN3	7	17	2.43

Table 3 compares the rule sets extracted by NeuroRule, NeuroLinear and MofN3. NeuroRule and MofN3 have been applied to the same discretized data set. From the figures in the table, we can see that both the number of rules and the number of conditions per rule of MofN3 are approximately half the corresponding numbers from NeuroRule. The accuracy rates of rules differ by only 1.2%.

## 5 CONCLUSION

The algorithm MofN3 for learning M-of-N concepts can be applied to any medical diagnosis problem. It requires the standard feedforward neural networks as its backbone. Two assumptions

are required by the algorithm. First, the data must be discrete and have only the binary values -1 or 1. This assumption can be easily met by a simple recoding of the data. Second, the weights of the connections from the input to the hidden units are also binary valued, -1 or 1. The algorithm attempts to find a network that meets this assumption by applying the hyperbolic tangent function to the network connection weights.

This paper presents an application of MofN3 for the diagnosis of hepatobiliary disorders. For the data that have been collected, we show that the algorithm can extract a concise set of rules with few conditions per rule. The predictive accuracy of the extracted MofN3 rules is comparable to another method that extracts a larger set of DNF rules.

## References

- [1] R. Andrews, J. Diederich and A.B. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks," *Knowledge Based Systems* 8(6) (1995) 373–389.
- [2] S. Cho and J.A. Reggia, Multiple disorder diagnosis with adaptive competitive neural networks, *Artificial Intelligence in Medicine* 5 (1993) 469–487.
- [3] C. Ulbricht, G. Dorffner, and L. Andreas, Neural networks for recognizing patterns in cardiocograms, *Artificial Intelligence In Medicine* (12)3 (1998) 271–284.
- [4] D.B. Fogel, E.C. Wasson III, E.M. Boughton, and V.W. Porto, Evolving artificial neural networks for screening features from mammograms, *Artificial Intelligence In Medicine* (14)3 (1998) 317–326.
- [5] Y. Hayashi, Neural expert system using fuzzy teaching input and its application to medical diagnosis, *Information Sciences: Applications* 1 (1994), 47–58.
- [6] S. Mitra, Fuzzy MLP based expert system for medical diagnosis, *Fuzzy Sets and Systems* (65) (1994), 285–296.
- [7] C.D. Nugent, J.A.C. Webb, M. McIntyre, N.D. Black, G.T.H. Wright, Computerised electrocardiology employing bi-group neural networks, *Artificial Intelligence In Medicine* (13)3 (1998) 167–180.
- [8] E. Pesonen, M. Eskelinen, and M. Juhola, Treatment of missing data values in a neural network based decision support system for acute abdominal pain, *Artificial Intelligence In Medicine* (13)3 (1998) 139–146.

- [9] J.R. Quinlan. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.
- [10] R. Setiono, Extracting rules from pruned neural networks for breast cancer diagnosis, *Artificial Intelligence in Medicine*, (8)1 (1996) 37–51.
- [11] R. Setiono, A penalty function approach for pruning feedforward neural networks, *Neural Computation* (9)1 (1997) 185–204.
- [12] R.Setiono, Extracting rules from neural networks by pruning and hidden-unit splitting, *Neural Computation*, (9)1 (1997) 205–225.
- [13] R.Setiono and H. Liu, Symbolic representation of neural networks, *IEEE Computer*, (29)3 (1996) 71–77.
- [14] R.Setiono and H. Liu, NeuroLinear: from neural networks to oblique decision rules, *Neurocomputing*, (17)1 (1997) 1–24
- [15] R. Setiono, Extracting M-of-N rules from trained neural networks, *IEEE Transactions on Neural Networks* (11)2 (2000) 512–519.
- [16] I. Taha and J. Gosh, Symbolic interpretation of artificial neural networks, *IEEE Transactions on Knowledge and Data Engineering*, (11)3 (1999) 448–462.
- [17] A.B. Tickle, R. Andrews, M. Golea, and J. Diederich, The truth will come to light: Directions and challenges in extracting the knowledge embedded within artificial neural networks, *IEEE Transactions on Neural Networks* (9)6 (1999) 1057–1068.
- [18] G.G. Towell and J.W. Shavlik, Extracting refined rules from knowledge-based neural networks, *Machine Learning*, 13(1) (1993) 71–101.
- [19] Y. Hayashi, R. Setiono and K. Yoshida, A comparison between two neural network rule extraction techniques for the diagnosis of hepatobiliary disorders, *Artificial Intelligence in Medicine*, 2000, forthcoming.