A Neural Network Based Approach to Knowledge Acqusition and Expert Systems[1]

Nicholas DeClaris and Mu-Chun Su
School of Medicine and College of Engineering
University of Maryland in Baltimore and College Park
U.S.A

Abstract **Often a major difficulty in the design of expert systems is the process of acquiring the requisite knowledge in the form of production rules. This paper presents a novel class of neural networks which are trained in such a way that they provide an appealing solution to the problem of knowledge acquisition. The value of the network parameters, after sufficient training, are then utilized to generate production rules on the basis of preselected meaningful coordinates. Futhere, the paper provides a mathematical framework for achieving reasonable generalization properties via an appropriate training algorithm (supervised decision-directed learning) with a structure that provides acceptable knowledge representations of the data. The concepts and methods presented in the paper are illustrated through one practical example from medical diagnosis.**

I. INTRODUCTION

Neural networks are attracting a lot of interest in the scientific community because of their dynamical nature: robustness, capability of generalization and fault tolerance. Neural networks have already proven useful in low level information processing (*e.g.* signal analysis). An area where neural networks find exciting applications is in medicine in cases where statistical methods can not be used such as when incomplete, or insufficient amount of data. In medical diagnose, clinicians make a series of inferences about the nature of the physiological abnormality derived from existing observations (*e.g.* historical data, physical findings, and routine laboratory tests). There are diagnostic processes which are guided by precompiled production rules. In such cases the use of rule-based expert systems is helpful.

Rule-based expert systems are rather practical development in the artificial intelligence (AI) field. They are based on the premise that expert knowledge can be encapsulated in a set of **IF...THEN...** statements. Traditionally, the design of rule-based expert systems involves a process of interaction between a domain expert and a knowledge engineer who formalizes the expert's knowledge as inference rules and encodes it in a computer. However, there are several difficulties in obtaining an adequate set of rules from human experts. Experts may not know, or may be unable to articulate, what knowledge they actually use in solving their problems. Often, the development of an expert system is time-consuming. Thus, the process of building an expert system requires much effort. Another important problems is that it is difficult to determine whether the knowledge base is correct, consistent and/or incomplete. One way to alleviate these problems is to use machine learning to automate the process of knowledge acquisition [1].

An appealing aspect of neural networks is that they can inductively acquire concepts from examples. A set of labeled examples is provided to the neural networks for training. After the process of training, the network can classify inexperienced patterns. Training procedure is accomplished by appropriately modifying its adjustable weights so that the training data is more correctly classified. Neural networks have shown promise in classification and diagnostic tasks. Nevertheless, there are still some obstacles lying in the combination of neural networks and expert systems: 1) lengthy training time; 2) no systematic way to set up a good network topology; and 3) difficulty in interpreting trained networks [2]. The difficulty in interpreting, in physiological meaningful ways, trained networks is one of the greatest problem of neural networks. A neural network can not justify its response on the bases of explicit rules or logical reasoning process. This feature is particularly important in medicine where medical experts require detailed justification for any diagnosis, whether it issues from nature or artificial intelligence.

There have been several attempts to overcome these problems. One approach is to interpret or extract rules from a trained backpropagation networks [3]. Other researchers focused on refining coarse knowledgebase by adjusting weights of neural networks [4], [5]. Sestito and Dillon [6] and Goodman, Higgins and Miller [7] considered problems of extracting rules that relate to a set of binary (or discrete) feature variables. However, there are rules which can not be extracted by such algorithms. In this paper, an architecture and algorithm of neural networks is discussed that insures the extraction of all applicable rules. Our approach is based on a new concept of utilizing "intermediate rules", by which the final decision rules are represented. Specifically, intermediate rules are represented in the form of a hyperspherical type description. A final decision rule is represented as a disjunction of intermediate rules. The training algorithm results in

Manuscript received July 8, 1993. This footnote is not numbered.
1 Supported in part under NIH Grant # 510 RR06460-01, Nicholas DeClaris, Principle Investigator

a two-layer hyperspherical composite neural network shown in Fig. 1. Each intermediate rule corresponds to a distinct node. An important feature is that the final neural network is required to learn correctly all training data.

II. REPRESENTATIONS AND PROPERTIES

Knowledge representation in expert systems must be as clear to human users as possible. The knowledge of a trained backpropagation network lies in its inter-node weights [8], in this respect, it differs from the high level representation obtained through traditional knowledge acquisition methods.

The construction of a rule-based expert system involves the process of acquiring the production rules. The production rules are often represented as "IF *condition* THEN *act*". Backpropagation networks do not always arrive to this type of representation. A suitable class are hyperspherical composite networks because production rules can be easily extracted from them. The symbolic representation of a hyperspherical neural node is shown in Fig. 2 and it is described by the following equations,

$$net(X) = d^2 - \sum_{i=1}^{n} (x_i - c_i)^2 \tag{1}$$

and

$$Out(X) = f(net(X)) \tag{2}$$

where

$$f(x) = \begin{cases} 1 & if\ x \geq 0 \\ 0 & if\ x < 0 \end{cases}, \tag{3}$$

c_i, and $d \in \mathbf{R}$ are adjustable weights, $\mathbf{R}$ is the set of real number, n is the dimensionality of input variables, $x_i \in \mathbf{R}$, X is an n-component column vector of x_i and $Out(X) : R^n \to \{0, 1\}$ is an output function of a hyperspherical neural node.

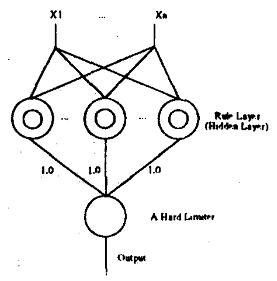


Fig. 1 A two-layer hyperspherical composite neural network

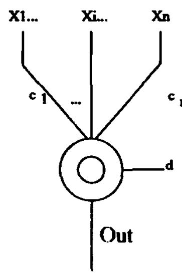


Fig. 2 A hyperspherical neural node

According to (1) & (2), we know the network outputs one only under the following condition:

$$\sum_{i=1}^{n} (x_i - c_i)^2 \leq d^2. \tag{4}$$

Then the classification knowledge can be described in the form of a production rule. The *if-then* rule corresponds to the following statement

$$IF\left(\sum_{i=1}^{n} (x_i - c_i)^2 \leq d^2\right) \\ THEN\ Out = 1. \tag{5}$$

The domain defined by (5) is an n-dimensional hypersphere. DeClaris and Su have done considerable works with this class of quadratic junction neural-type networks [9]. The decision region formed by a neural node with quadratic junctions is in general a hyperellipsoid whose axes may be oblique with respect to the axes of the input space in n-dimensional space. Use of a nondiagonal covariance matrix allows each general hyperellipsoid to tilt in the direction of the maximum data spread. However, the inclusion of correlation coefficients for each general hyperellipsoid increases the numbers of parameters. The class of neural nodes studied in this paper uses constrained quadratic junctions—a trade-off between the flexibility of hyperellipsoids and the number of parameters.

Owing to characteristics of data, such as *dispersion characteristic*, it may be not adequate to use a simple classification rule represented as a hypersphere to solve all pattern recognition problems. Dispersion characteristic of data leads to an existence of many distinct clusters in input space. A reasonable idea of clustering data would be in the form of a hyperspherical or hyperellipsoidal "cluster" of patterns. In this way a complex concept is represented as *intermediate concepts* explicitely extracted by the set of hyperspheres. In the two-layer neural network configuration as shown in Fig. 1, input variables are assigned input nodes, intermediate concepts are assigned hidden nodes (rule nodes), and the induced concept (final classification decision rule) is assigned an output node. Each hidden node is connected, with weight value

1.0, to the output node. The output node is just a hard limiter. The network outputs one whenever there exists one hidden node whose output is one. Fig. 3 illustrates decision regions formed by hyperspherical composite neural networks. Therefore, if there are k hidden nodes, a decision rule can be represented as

$$
\begin{aligned}
&IF\left(\sum_{i=1}^{n}(x_i - c_{1i})^2 \le d_1^2\right)\\
&THEN\ output = 1,\\
&\dots\\
&ELSE\ IF\left(\sum_{i=1}^{n}(x_i - c_{ki})^2 \le d_k^2\right)\\
&THEN\ output = 1.
\end{aligned}
\tag{6}
$$

III. TRAINING ALGORITHM FOR HYPERSPHERICAL COMPOSITE NEURAL NETWORKS

Fig. 4 shows the algorithm called supervised decision-directed learning algorithm for training hyperspherical composite neural networks. Here, we apply the idea of inductive machine learning into the training procedure of the class of hyperspherical composite neural networks. In setting up the network topology, this algorithm generates a two-layer feed-forward network in a sequential manner by adding hidden nodes as need. After the training procedure, all training patterns are correctly recognized. At the same time the decision rule can be easily extracted. The algorithm is based on a concept of division of the domain of inputs into appropriate subsets.

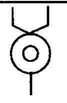
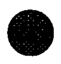
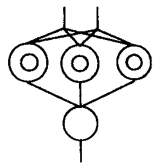
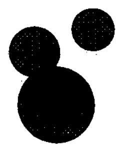
STRUCTURE	TYPE OF DECISION REGION	DECISION REGION
	CONVEX (HYPERSPHERE)	
	ARBITRARY (UNION OF THE HYPERSPHERES)	

Fig. 3 Decision regions formed by hyperspherical composite neural networks

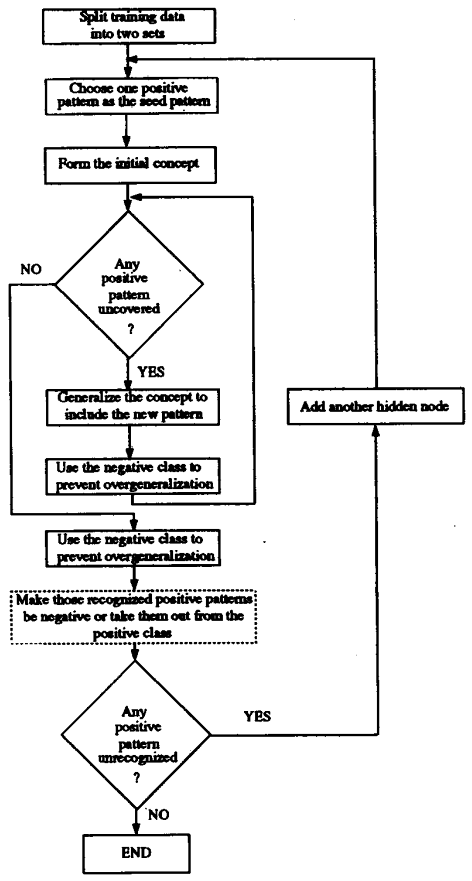

Split training data into two sets
Choose one positive pattern as the seed pattern
Form the initial concept
Any positive pattern uncovered ?
NO
YES
Generalize the concept to include the new pattern
Use the negative class to prevent overgeneralization
Use the negative class to prevent overgeneralization
Make those recognized positive patterns be negative or take them out from the positive class
Any positive pattern unrecognized ?
YES
NO
END
Add another hidden node

Fig. 4 Flowchart of the training algorithm for hyperspherical composite neural networks

Often inputs (features) are measured on different units. For instances, in medical diagnostic process, data are derived from nature existing observations such as historical data, physical findings, and routine laboratory tests. Therefore, patterns need to be normalized so that no single input overwhelms data merely because of scale. The method of normalization used in this paper is that data have been normalized to zero mean and unit variance in each direction. Thus, a hypersphere in normalized input space is indeed a hyperellipsoid in original input space as shown followed

$$
\begin{aligned}
&\sum_{i=1}^{n}(x_i' - c_i')^2 \le d^2 \quad x_i' = \frac{x_i - m_i}{\sigma_i}\\
&\Leftrightarrow \sum_{i=1}^{n}\frac{(x_i - c_i)^2}{\sigma_i^2} \le d^2 \quad c_i = m_i + \sigma_i c_i'
\end{aligned}
\tag{7}
$$

where m_i and σ_i are mean value and squared variance of x_i, and x_i' is the normalized input.

In this training algorithm, we want to find a set of hyperspheres which cover all patterns belonging to the same categories. To begin with, training patterns are divided into two sets: 1) a *positive class* from which we want to extract (induce) the **concept** and 2) a *negative class* which provides counterexamples with respect to the concept. Then a *seed* pattern X' is used as the basis of an initial concept (the seed pattern is any one chosen from the positive class). Then weights are initialized in the following manner:

$$\begin{cases} c_i'(0) = x_i' & 1 \le i \le n \\ d(0) = \delta & \delta \textit{ is a small positive number.} \end{cases} \tag{8}$$

Next we use all counterexamples to prevent overgeneralization (the induced concept should not be so general as to include any of counterexamples) formed by the initialization of weights. The following step is to fetch the next positive pattern and to generalize the initial concept to include the new positive pattern. This process involves growing the original hypersphere to make it larger to include the new positive pattern. The size and position of the present hypersphere is adapted from a better knowledge of the local distribution of positive patterns falling inside the present hypersphere. The way of computing a new center and radius is shown as follows:

$$c_i'(t+1) = \left(\sqrt{\sum_{i=1}^{n} (x_i' - c_i'(t))^2} + d(t)\right) \cdot \frac{(c_i'(t) - x_i')}{2\sqrt{\sum_{i=1}^{n} (x_i' - c_i'(t))^2}} + x_i' \quad 1 \le i \le n. \tag{9}$$

and

$$d(t+1) = \frac{\left(\sqrt{\sum_{i=1}^{n} (x_i' - c_i')^2} + d(t)\right)}{2} + \epsilon \tag{10}$$

where ϵ, a small positive number, is the control parameter which decides the degree of generalization contributed by a positive pattern. After the process of generalization, again we use counterexamples to validate the size of the expanded hypersphere. Here let us assume X' is the nearest counterexample to the center of the original hypersphere (*i.e.* at time t). Then

$$c_i'(t+2) = \left(\sqrt{\sum_{i=1}^{n} (x_i' - c_i'(t))^2} + d(t)\right) \cdot \frac{(c_i'(t) - x_i')}{2\sqrt{\sum_{i=1}^{n} (x_i' - c_i'(t))^2}} + x_i' \quad 1 \le i \le n. \tag{11}$$

and

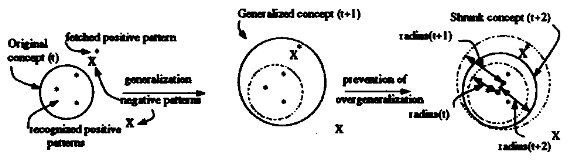


Fig. 5 Process of learning in hyperspherical composite neural networks

$$d(t+2) = \frac{\left(\sqrt{\sum_{i=1}^{n} (x_i' - c_i'(t))^2} + d(t)\right)}{2} - \epsilon \tag{12}$$

Fig. 5 illustrates this process of learning. This process is repeated for all the remaining positive patterns.

Here, we want to point out that during the process of preventing overgeneralization, a most important consideration is that whenever we shrink the expanded hypersphere (at time t+1) in order to exclude counterexamples, the shrunk hypersphere should include the original hypersphere (at time t). This criteria can guarantee that at least one positive pattern (*e.g.* seed pattern) is correctly recognized after the training procedure. In other words, it prevents the new learning from washing away the memories of prior learning. Thus if there exists any unrecognized positive pattern, another hidden node is self-generated and the process of learning is repeated again and again until all positive patterns are recognized correctly. In the worst case, the number of the hidden nodes is equal to the number of positive patterns, however, if the data clusters well, the number of hidden nodes will be as small as possible.

A multi-output system can always be separated into a group of single-output systems. Therefore, we may break down a large training task into pieces of small tasks. Finally, each hidden node is associated with a value, V, representing the total number of positive examples explained (covered) by the respective intermediate concept. The V-value may be interpreted as a measure of its representativeness of the intermediate concept as a concept description.

The RCE networks also adopt the representation of hyperspheres [10]. During its training procedure, each new pattern that is not correctly recognized results in the creation of a new hypersphere whose center is defined by the new pattern. Misclassifications reduce the size of the created hyperspheres, but they are not moved, nor deleted. Thus the number of hyperspheres created by the RCE networks does not reflect the number of underlying clusters of data. Nevertheless, in our training algorithm, we do attempt to minimize the number of hyperspheres in normalized input space. On the other hand, a popular choice for choosing radial basis units for the radial basis function (RBF) networks is the K-means algorithm [11], [12]. It require the number of units be predetermined, nevertheless, this predetermined number may not the right number of clusters. Our algorithm provides another choice for training RBF networks.

IV. EXAMPLE: A MEDICAL DIAGNOSIS EXPERT NETWORK

Diabetes mellitus is one of the major non-communicable chronic diseases and as such it constitutes a major medical challenge. Diabetes is a syndrome involving both metabolic and vascular abnormalities. It results in a range of complications which affect circulatory systems, eyes, and nerves. There are two primary categories of diabetes mellitus: type I—insulin dependent diabetes mellitus (IDDM); and type II—non-insulin dependent diabetes mellitus (NIDDM) [13]. The pathogenesis of IDDM is well understood, but, little progress has been made in discovering the mechanisms which trigger NIDDM. In hopes of discovering a method of early detection of future NIDDM onset, and ultimately, a means to delay or avert the onset of this catastrophic metabolic disorder, Hansen and Bodkin [14] are studying its pathogenesis in Rhesus monkeys. Currently, nine *phases* of progression have been identified in the development of the type II diabetes mellitus. These phases are identified by fluctuations in various measurements taken on the monkeys. Unfortunately, these relationships are rather complex and difficult to be described by human experts. Lin [15] used a multilayer backpropagation network to classify the data in order to decide on which phase the monkey is. The result was satisfactory. However, the classification knowledge can not be articulated as comprehensible diagnosis rules so its use is rather limited. We shall use hyperspherical composite neural networks to elicit the knowledge required to identify the phases on the basis of physiological variable.

In collaboration with Hamsen we used data from 42 monkeys which had been followed through the course of various phases of diabetes mellitus. The set of physiological variables used for phase identification is as follows:

1. age (C_{ge})—monkey's age (years),
2. wgt (C_{gt})—monkey's weight (kg),
3. fpg (C_{pg})—fasting plasma glucose (mg/dl),
4. fpi (C_{pi})—fasting plasma insulin (μU/ml),
5. gpr (C_{pr})—glucose disappearance rate.

The above five measurements, age, wgt, fpg, fpi, and gdr, were used as physiological coordinates in our expert system. 479 observations were used in our training algorithm. All of the data entries contain five parameters and the corresponding diabetes phase classification identified by experts. Because there are low numbers of data points per phases, it was decided to partition the complete data set in a 90%/10% ratio for training and testing, respectively. This partitioning was felt to be good compromise between having training patterns to fully described the decision regions during training and having enough patterns in the testing set to yield meaningful generalization tests.

Phases	Train	Test	# of rules	Compression Rate
1	100%	91.7%	7	9 : 479
2	100%	90.0%	38	38 : 479
3	100%	88.1%	21	21 : 479
4	100%	94.9%	17	17 : 479
5	100%	84.7%	17	17 : 479
6	100%	91.7%	11	11 : 479
7	100%	83.1%	17	17 : 479
8	100%	94.9%	11	11 : 479
9	100%	96.7%	11	11 : 479
average	100%	90.6%	16.7	1 : 28.5

Table 1 Performance of the hyperspherical composite neural networks as diagnosis expert systems for diabetes mellitus: 479 is the total number of observations and total set (100%) = training set (90%) + testing set (10%) .

Generalization performance is evaluated by presenting testing patterns to the trained networks, and comparing the networks classification with the desired classification. Table 1 depicts the successful generalization performance of the hyperspherical composite neural networks as a diabetes mellitus diagnosis expert system. The average successful performance shown in the last row of Table1 is around 90% which is very encouraging. The number of extracted intermediate diagnosis rules for each phase is not as small as we expected, however, the average compression rate (defined as the number of extracted intermediate rules divided by the total data) shown in the last column of Table1 is acceptable. The large number of hidden nodes is due to the following two reasons: 1) The diabetes mellitus data do not cluster in the form of hyperspherical subspaces. and 2)During training procedure, each hypersphere, at its initial creation, migrates in the input space and generally tends toward the most near mode of the data distribution. As hyperspheres move toward modes, and latter positive patterns create hyperspheres at position from which older hyperspheres have migrated, there will always be hyperspheres which cover few positive patterns [16].

The training procedure was implemented as a C program and run on a SUN-IPX. It took less than 20 minutes for all phases to converge. This is significantly faster than running C programs based on the backpropagation approach, which took several hours to converge. However the important advantage of the hyperspherical composite neural networks is the fact that the classification knowledge can be represented

Phase	age	wgt	fpg	fpi	gpr	d
1	6.4	7.7	63.8	45.2	3.8	0.7
2	7.6	9.6	67.9	31.8	2.7	0.4
3	6.9	9.3	67.3	23.7	6.0	0.3
4	9.2	10.7	63.7	60.0	5.0	0.2
5	10.7	9.5	65.3	23.0	3.3	0.2
6	8.1	10.4	70.1	67.7	4.5	0.1
7	8.9	11.4	69.2	51.5	3.7	0.1
deviation	5.7	4.8	62.9	105.3	1.3	—

Table 2 The seven extracted intermediate rules for phase 1

as production rules in the following form:

$$IF\ (\left(\frac{C_{ge}-age}{5.7}\right)^2+\left(\frac{C_{gt}-wgt}{4.8}\right)^2+\left(\frac{C_{pg}-fpg}{62.9}\right)^2 + \left(\frac{C_{pi}-fpi}{105.3}\right)^2+\left(\frac{C_{pr}-gpr}{1.3}\right)^2 \leq d^2)$$

$$THEN\ \ the\ monkey\ is\ on\ phase\ 1$$
$$\ldots$$
$$ELSE\ IF\ (\ldots)$$
$$THEN\ the\ monkey\ is\ on\ \ldots \tag{13}$$

where d is the radius of the hypersphere and each denominator represents the deviation of each variable, respectively. Table 2 depivts all seven extracted intermediate rules for phase 1.

V. DISCUSSIONS AND CONCLUSIONS

In this paper, we provide a new approach that brings together two distinct methodologies: rule-based expert systems and neural networks. The most important characteristic of this approach is that classification knowledge embedded in numerical weights of networks is extracted and represented as sets of production rules for human users or automated by algorithmic digital computers. It promised to be of high promise in medical diagnosis requiring pathophysiological explanations.

REFERENCES

[1] R. S. Michalski and R. L. Chilausky, Learning by being told and learning from examples: an experimental comparison. International Journal of Policy Analysis and Information Systems, vol. 4, no. 2, pp. 125–160, 1980.

[2] J. W. Shavlik and G. G. Towell, An approach to combining explanation-based and neural learning algorithm. In N. G. Bourbakis, Ed. Applications of Learning & Planning Methods, pp. 71–98, 1991.

[3] L. M. Fu, "Rule learning by searching on adapted nets," AAAI'91, pp. 590–595, 1991.

[4] G. G. Towell and J. W. Shavlik, "Using symbolic learning to improve knowledge-based neural networks," AAAI'9, pp. 177–182, 1992.

[5] R. C. Lacher, S. I. Hruska, and D. C. Kuncicky, "Back-propagation learning in expert networks," IEEE Trans. on Neural Networks, vol. 13, pp. 62–72, 1992.

[6] S. Sestito and T. Dillon, "Machine learning using single-layered and multi-layered neural networks," IEEE 2nd Int. Conf. on Tools for Art. Intell., pp. 269–275, 1990.

[7] R. M. Goodman, C. M. Higgins, and J. W. Miller, "Rule-based neural networks for classification and probability estimation," Neural Computation, vol. 4, pp. 781–804, 1992.

[8] D. Rumelhart, J. McClelland, and PDP Research Group, Explanation in the Microstructure of Cognition, in Parallel Distributed Processing, MIT Press, 1986.

[9] N. DeClaris and M.-C. Su, A novel class of neural networks with quadratic junctions, in IEEE Int. Conf. on Systems, Man, and Cybernectics, pp. 1557–1562, 1991 (Received the 1992 Franklin V. Taylor Award).

[10] D. L. Reilly, L. N. Cooper, and C. Elbaum, "A neural network for category learning," Biol. Cybern. vol. 45, pp. 35–41, 1982.

[11] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," Neural Computation, vol. 1, pp. 281–293, 1989.

[12] R. O. Duda and P. H. Hart, *Pattern Recognition and Scene Analysis. Wiley. New York, NY, 1973.*

[13] M. B. Davidson, Diabetes Mellitus Diagnosis and Treatment. New York, Wiley & Sons, 1986.

[14] B. C. Hansen and N. L. Bodkin, "Heterogeneity of insulin responses: phases leading to Type 2 (non-insulin-dependent) diabetes mellitus in the rhesus monkey," Diabetologia, vol. 29, pp. 713–719, 1986.

[15] S. C. Lin, "Implementation of a novel neural network design for a demanding medical problem," Master's thesis, Univ. of Maryland, College Park, 1991.

[16] G. Sebestyen and J. Edie, "An algorithm for non-parametric pattern recognition," IEEE Trans. on Electronic Computers, vol. 15, pp. 908–915, Dec. 1966.