A Fuzzy Neural Network for Pattern Classification and Feature Selection

By

Rui-Ping Li*
Masao Mukaidono**
I. Burhan Turksen***

* Caelum Research Corporation, Rockville, Maryland, 20850, USA
** Dept. of Computer Science, Meiji University, Kawasaki-shi, 214, Japan
*** Dept. of Industrial Engineering, University of Toronto, Toronto, M5S 1A4, Canada

Corresponding Address:

Masao Mukaidono, Prof.
Dept. of Computer Science
Meiji University, 1-1-1 Higashimita, Tama-ku
Kawasaki, 214, Japan
Tel: +81_44_934_7450
Fax: +81-44-934-7912
E-mail: masao@cs.meiji.ac.jp

Abstract: A fuzzy neural network with *memory* connections for classification, and *weight* connections for selection is introduced, thereby solving simultaneously two major problems in pattern recognition: pattern classification and feature selection. The proposed network attempts to select important features from among the originally given plausible features, while maintaining the maximum recognition rate. The resulting value of weight connection represents the degree of importance of feature. Moreover, the knowledge acquired by the network can be described as a set of interpretable rules. The effectiveness of this new method has been validated by using Anderson's IRIS data. The results are these: first, the use of two features selected by our method from among the original four in the proposed network results in virtually identical classifier performance; and secondly, the constructed classifier is described by three simple rules that are of if-then form.
Key words: Pattern classification, Feature selection, Neural networks, Fuzzy logic, Linguistic modeling.

1. Introduction

The recognition of patterns is the basis of all science. The aim is to discover a structure in a system consisting of partial subsystems. Usually, something structured refers to the knowledge of the state of a partial subsystem allowing us to easily guess the state of other parts of the same whole system [1]. Techniques of pattern recognition can be generally described as deterministic, statistical, or fuzzy in terms of their axiomatic bases. Traditional statistical classification methods usually try to find a clear cut boundary to divide the pattern space into some classification regions based on some pre-defined criterion, such as maximizing deviation-between-groups divided by deviation-within-groups in the linear discriminant analysis (LDA) [2]. As pointed in [3], it is impossible to provide information of degree of uncertainty for a particular example for LDA method since the error rate estimate is a statistical result of the entire sample set. In fact, pattern recognition systems are systems that automatically identify objects based on their measured properties or features derived from these properties. With this viewpoint, a neural network also is a pattern recognition system. The existing neural networks that can be served as classifiers may be grouped into four categories or their variations: Backpropagation (BP) [4], Adaptive resonance theory (ART) [5], Radial basis functions (RBF) [6], and Probabilistic neural networks (PNN) [8]. The first three are based on the deterministic axiomatics, and the last one is based on the probabilistic-statistical axiomatics. Although these techniques have been proven to be useful tools for pattern classification, the selection of features still is a challenge.

Since fuzzy set theory was suggested in 1965 [9], pattern recognition problems have been intensively studied with fuzzy set [10]. The revolutionary significance of fuzzy set theory is that it provides a mathematical method for describing *intuitive* knowledge of humans. In principle, a mathematical model constructed in accordance with the classical theory must be interpreted in natural language that could be understood intuitively. In contrast to classical methodology, a fuzzy approach to modeling begins with a practical interpretation of concepts, and then generates intuitive logical relations between concepts and constructs a model. A model constructed in accordance with the fuzzy theory, therefore, is certainly interpretable. This methodology is called ‘empirical-semantic’ approach in [11], and this modeling method is called ‘linguistic’ modeling in [12]. In recent years, a great deal of attention has been directed toward using the fusion of fuzzy logic and neural networks to develop intelligent systems. This is because the two technologies are strongly complementary to each other [13], [25]. Keller [14], for example, incorporated fuzzy membership functions into the Perceptron learning algorithm. Archer [3] used fuzzy set representation in neural network classification problems.

There are two main aspects to the effort of pattern recognition: pattern classification and feature selection. Although many efforts have been made, we still do not have a complete and satisfactory technique that can simultaneously deal with the above two problems. Bezdek [10] proposed a measure of feature selection that works only for binary data. Kuncheva proposed a new selection criterion in [15] using the concept of fuzzy rough sets. The latter overcome the limitation of the former, however combinatorial explosion would become a major problem for the cases in which there are more than a small number of features.

On the other hand, in order to solve the initialization problem and the normalization problem with traditional learning vector quantization (LVQ) [7], a proportional learning vector quantization (PLVQ) method was introduced in [16] and [17]. PLVQ is a generalized learning vector quantization based on a fuzzy learning law (FLL). The second section of this article provides an overview of the PLVQ algorithms, since the FLL is employed in the presented network. The third section of this article describes our feature-weighted detector (FWD) network that can fulfill both tasks of feature selection and pattern classification. The fourth section includes two examples to verify the effectiveness of our FWD. For the sake of understanding, an artificial data set is chosen in the first example. In the second example, the data set used is Anderson's IRIS data [18] that has been widely studied, allowing us to easily analyze and compare our new technique with existing methods. Finally, the fifth section contains a summary and conclusions.

2. Proportional Learning Vector Quantization

As well known to all, LVQ is a clustering algorithm for organizing a large of unlabeled vectors into some given clusters. Although some good practical results have been obtained with it, the method still suffers from an initialization problem [19] and normalization problem [17].

Based on Hebb's learning postulate, we assume that a desired learning rule for weights of LVQ network should satisfy the following differential equation in continuous space:

$$\frac{d\mathbf{m}_i}{dt} = a_t u_i(\mathbf{x})(\mathbf{x} - \mathbf{m}_i) \qquad (1a)$$

or, in discrete domains, we have

$$\Delta\mathbf{m}_i = a_t u_i(\mathbf{x})(\mathbf{x} - \mathbf{m}_i) \qquad (1b)$$

Where $\mathbf{x}$ denotes the input vector, $\mathbf{m}_i$ denotes the memory vector of neuron *i*. $u_i(\mathbf{x})$ represents the output value of neuron *i* when $\mathbf{x}$ is presented in input layer. $a_t = a(1 - t/T)$ is referred to as *temporal* learning rate in [19]. To find the mathematical expression of $u_i(\mathbf{x})$ and its physical meaning, consider the following loss function *L* as introduced in [20]-[21]

$$L = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik} \left\|\mathbf{x}_k - \mathbf{m}_i\right\|^2 \qquad (2)$$

Where $u_{ik} \equiv u_i(\mathbf{x}_k)$ represents the degree to which input $\mathbf{x}_k$ ($\mathbf{x}_k = (x_{k1}, x_{k2}, \ldots, x_{kp})$) matches memory vector $\mathbf{m}_i$ ($\mathbf{m}_i = (m_{i1}, m_{i2}, \ldots, m_{ip})$). *N* is the number of data and *c* is the number of clusters. The number of clusters here is equal to the number of output layer neurons. *p* is the number of features, i.e., the number of input layer nodes. Using the method of the maximum-fuzzy-entropy interpretation [21] and the normalization condition ($\sum_{i=1}^{c} u_{ik} = 1$ for each *k*), the following solution that minimizes the loss function *L* was found,

$$u_{ik} = \exp\left[-\frac{\left\|\mathbf{x}_k - \mathbf{m}_i\right\|^2}{2s^2}\right] \Big/ \sum_{j=1}^{c} \exp\left[-\frac{\left\|\mathbf{x}_k - \mathbf{m}_j\right\|^2}{2s^2}\right] \qquad (3)$$

Where s is a nonzero number that can be chosen by user. Physically, s represents the fuzziness in clustering. The smaller s is, the less the fuzziness is. There is no theoretical basis for choosing an optimal s . A heuristic guideline is $s = 0.25\sqrt{p}$ but not limited this. Note that p represents the number of features here. $u_i(\mathbf{x}_k) \in$ [0, 1] is a fuzzy membership function. For a given s , if the closer the input $\mathbf{x}_k$ is to the memory $\mathbf{m}_i$, then the closer the output $u_i(\mathbf{x}_k)$ is to one; if the more the input $\mathbf{x}_k$ is away from the memory $\mathbf{m}_i$, then the closer the output $u_i(\mathbf{x}_k)$ is to zero. From Eqns.(1b) and (3), it is clear that each input updates all the weights (i.e., memory connections $\{\mathbf{m}_i\}$) in proportion as their output values. Eqn.(1) was called fuzzy learning law (FLL) in [17], and $u_i(\mathbf{x}_k)$ also can be called *special* learning rate corresponding with *temporal* learning rate a_t. When $s \to 0$, $u_i(\mathbf{x}_k)$ ={0, 1}, and thus the FLL reduces to competitive learning law (CLL) [7]. The corresponding algorithm is referred to as proportional learning vector quantization (PLVQ). It has been shown that PLVQ avoids above two problems with LVQ.

The PLVQ Algorithm:

```
1). Fix $2 \le c << n$, $e > 0$, $s$ >0 and the maximum number of iterations $T$.
2). Initialize $\{\mathbf{m}_i(0)\}$ and learning rate $a_0 \in [0,1]$.
3). For $t$ = 1, 2, ..., $T$;
    For $k$ = 1, 2, ..., $N$;
        a. Calculate $\{u_i(\mathbf{x}_k)\}$ using Eqn.(3).
        b. Update $\{\mathbf{m}_i(t)\}$ based on Eqn.(1b), i.e.,
            $\mathbf{m}_i(t) = \mathbf{m}_i(t-1) + a_0(1 - t/T)u_i(\mathbf{x}_k)(\mathbf{x}_k - \mathbf{m}_i)$        (4)
        c. Next $k$.
4). Calculate $E = \sum_{i=1}^{c}\sum_{j=1}^{p}\left|m_{ij}(t) - m_{ij}(t-1)\right|$.
5). IF $E < e$ or $t>T$ stop; ELSE next $t$.
```

3. Feature-weighted Detector Networks

A feature-weighted detector (FWD) network is shown in Fig.1. The network consists of input (*I*), matching (*M*), detecting (*D*) and output (*O*) layers (Fig.1(b)). Below, we give a description of this network in detail.

A. Input-output Relations

As shown in Fig.1(b), each *M* node can receive inputs from 2 sources: the left-right input; and right-left input from a node of *D* via a *D-M* adaptive connection. *f()* is a comparative function, and the output is the difference of two input values. In detecting layer, there are two types of node: forward and backward nodes. Each forward-node receives *p* inputs from *p* nodes of *M* via pathways with weight connections $\{w_{ij}\}$. *g()* is Gaussian functions. Each backward-node receives an input from a node of *O* via a backward-pathway with 1 connection fixed. *b()* is a linear function. The functional of the output layer nodes is to give final classification score for each input by normalizing output values of all *D* nodes. Each *O* node receives *c*+1 inputs. One of them is called set signal. The other *c* are from *D* via *c* pathways with 1 connection fixed. Set signal occurs before input is presented to the input layer. The role of the set signal is to provide an equal opportunity to match the input for each of *M* nodes. Before input $\mathbf{x}_k$ is coming, set signal $s_i = 1$, and thus $v_i = s_i = 1$ ($i = 1,2,...,c$), since *b()* is a linear function. This guarantees that each of neurons have an equal opportunity to match coming input. When input $\mathbf{x}_k$ is coming, outputs of node of neuron *i* are:

$$y_{ij} = (x_{kj} - m_{ji}) , \qquad j = 1,2,..., p \tag{5}$$

$$z_i = \exp[-\frac{1}{2s^2}\sum_{j=1}^{p} w_{ij}{}^2 (x_{kj} - m_{ji})^2] , \tag{6}$$

$$u_i = z_i / \sum_{j=1}^{c} z_j . \tag{7}$$

B. Learning Laws

In feature-weighted detector (FWD) networks, there are two types of learning when input is presented to the input layer. One is *memory* learning. The other is *weight* learning. $\mathbf{m}_i$ represents the memory of neuron *i*. Memory learning is unsupervised, and the updating rule is based on the FLL, i.e.,

$$\Delta\mathbf{m}_i = a_t u_i(\mathbf{x}_k)(\mathbf{x}_k - \mathbf{m}_i) \tag{8}$$

Where $\mathbf{x}_k$ represents the *k*-th input. On the other hand, in weight learning w_{ij} represents the degree to which feature *j* contributes to the cluster *i*. In order to find the updating rule of $\{ w_{ij} \}$, introduce the following error function

$$E = \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{c}(u_i(\mathbf{x}_k) - d_i)^2 \tag{9}$$

Where d_i is the desired value of output layer node *i*. Therefore, unlike memory learning, weight learning is supervised. Based on the chain rule of differential calculus, using Eqns.(9), (7) and (6) the following updating rule is obtained:

$$\Delta w_{ij} = \frac{b}{s^2 s^2}(u_i(\mathbf{x}_k) - d_i)(s - z_i) w_{ij} z_i (x_{kj} - m_{ji})^2 \tag{10}$$

Where $s = \sum_{i=1}^{c} z_i$ and $b > 0$ is learning rate. For the sake of understanding, designing $0 \le w_{ij} \le 1$ for each *i* and *j*. $w_{ij} = 0$ means that feature *j* has no "contribution" to cluster *i*; and $w_{ij} = 1$ means that feature *j* has the most contribution to cluster *i*. The algorithm can be stated as follows.

Feature-weighted Detector (FWD) Network Algorithm

1. Fix $s > 0$, $a \in [0, 1]$, $b > 0$, $e > 0$, and the maximum number of iterations *T*.
2. Initialize $\{ \mathbf{m}_i(0) \}$, using *c* samples randomly chosen from $\{ \mathbf{x}_k \}$ (*k*=1, 2,..., *N*), and $w_{ij}(0) = 1$ for each *i* and *j*.
3. For *t*=1, 2,..., *T*; For *k*=1, 2,..., *N*
   a. Calculate $\{ u_i \}$ using Eqn.(7).
   b. Update $\{ \mathbf{m}_i(t) \}$ using Eqn.(8).

```
        c. Update { $w_{ij}(t)$ } using Eqn.(10).
        d. Next $k$.
4. Calculate $E$ using Eqn.(9).
5. IF $E<e$ , or $t>T$ stop, ELSE next $t$.
```

4. Applications

Two examples have been selected to illustrate the performance of our FWD network. The purpose of selecting Example 1 is to help us to intuitively understand the physical meaning of the method. From the result of Example 2, a potential application value of the presented method should be shown.

Example 1--An artificial data set

For simplicity and intuition, we applied an artificial data set in which each pattern has two plausible features as showed in Fig. 2 to FWD. From the distribution of the data, it is clear that feature x_1 of this example has no contribution to the classification that follows the target outputs of Table 1.

For Data Set of Fig. 2, we run the FWD with e =0.058, a =0.01, b =0.1 and s =0.35. In this example, the number of input layer nodes is two (i.e., *p*=2), and the number of output layer nodes is two too (i.e., *c*=2). After 218 iterations, for each input the actual output is listed in the right two columns of Table 1. This classification is fuzzy. The item “target output” of Table 1 list the desired output when input *k* is presented to the input layer of the FWD. In the Table 1, u_i represents the output of neuron *i*. One neuron corresponds to one cluster here. Each pattern should be assigned to either of two clusters in *hard* classification as shown in the item of “target output” of Table 1. Obviously, if transforming the actual output listed in the right two columns of Table 1 into 0-1 binary value, then the actual classification result of the FWD is identical with the target in this example. Preferably, after learning, the resulting weight connections of the FWD are $\mathbf{w}_1 = $(0.10, 0.99), and $\mathbf{w}_2 = $(0.15, 0.99). The contribution of feature x_1 to the cluster 1 is $w_{11} = 0.10$; the contribution of feature x_1 to the cluster 2 is $w_{21} = 0.15$; the contribution of feature x_2 to the cluster 1 is the same as that to the cluster 2, i.e., $w_{12} = w_{22} = 0.99$. It shows that feature x_1 can be eliminated from the plausible features set selected initially since the degree of the importance of feature x_1 is much less than that of feature x_2. Therefore, the presented FWD enables the classification of pattern, and as well the selection of feature.

Example 2--An application to IRIS data

IRIS data [18] has been used in many papers to illustrate various clustering methods [22], [23]. The motivation of selecting IRIS data here is since we have already known the typical performance of the existing methods applied to it and also we can analyze feature by means of the geometric structure as used in [10]. As well known, the IRIS data is a set of 150 four-dimensional vectors. The plausible features selected initially include sepal length (x_1), sepal width (x_2), petal length (x_3) and petal width (x_4). The 150 IRIS data used come from three subspecies (clusters): sestosa, versicolor, and virginica. Each subspecies owns 50 samples respectively. Anderson measured each feature of 150 plants (samples). For this data, using the existing methods the typical number of mistakes is around 5 for supervised classifiers, and around 15 for unsupervised classifiers [23].

Shown in Table 2 are results of two experiments. In the first experiment (Experiment 1), all of four plausible features were used. The result is that: the number of mistakes, $e = 5$; three weight vectors, $\mathbf{w}_1 =$(1.00, 1.00, 0.95, 1.00), $\mathbf{w}_2 =$(0.00, 0.00, 1.00, 1.00), and $\mathbf{w}_3 =$(0.00, 0.00, 0.82, 0.97). Based on this result, it has been clearly shown that, 1) feature x_1 and feature x_2 have no contribution to cluster s_2 and cluster s_3, and 2) the role of feature x_1 and feature x_2 in cluster s_1 also can be jointly played by feature x_3 and feature x_4. For this, feature x_1 and feature x_2 is supposed being meaningless. In order to prove that, in the second experiment (Experiment 2) only feature x_3 and feature x_4 were used. The results of the second experiment are demonstrated in the right column of Table 2: the number of mistakes of the second experiment is the same as that of the first experiment, i.e., $e = 5$; three weight vectors are $\mathbf{w}_1 =$(1.00, 1.00), $\mathbf{w}_2 =$(1.00, 1.00), $\mathbf{w}_3 =$(0.82, 0.97). Obviously, the use of two feature x_3 and feature x_4 results in virtually identical classifier performance. After feature selection, therefore, only feature x_3 (petal length) and feature x_4 (petal width) are chosen.

For the sake of description, above selected two feature variables are renamed: x_1 now represents petal length, and x_2 petal width. When a new input $\mathbf{x} = (x_1, x_2)$ is coming, using the obtained $\{w_{ij}\}$, $\{m_{ji}\}$ and given s , based on Eqns.(6)-(7), we have the following:

$$\begin{aligned} z_i &= \exp[-\frac{1}{2s^2} {w_{i1}}^2 (x_1 - m_{1i})^2] \times \exp[-\frac{1}{2s^2} {w_{i2}}^2 (x_2 - m_{2i})^2] \\ &= U_{A_{i1}}(x_1) \times U_{A_{i2}}(x_2) \qquad i = 1,2,3 \end{aligned} \tag{11}$$

$$u_{c_i} = U_{A_{i1}}(x_1) U_{A_{i2}}(x_2) \sum_{j=1}^{3} U_{A_{j1}}(x_1) U_{A_{j2}}(x_2) \qquad i = 1,2,3 \tag{12}$$

Note that z_1, z_2 and z_3 are outputs that correspond respectively *class* sestosa (c_1), *class* versicolo (c_2), and *class* virginica (c_3). Normalized $u_{c_i}(\mathbf{x})$ represents the degree to which input $\mathbf{x}$ belongs to *class* c_i ($i = 1,2,3$). Further, we have

$$U_{A_{11}}(x_1) = \exp[-2(x_1 - 1.46)^2], \tag{13}$$

$$U_{A_{12}}(x_2) = \exp[-2(x_2 - 0.25)^2], \tag{14}$$

$$U_{A_{21}}(x_1) = \exp[-2(x_1 - 4.29)^2], \tag{15}$$

$$U_{A_{22}}(x_2) = \exp[-2(x_2 - 1.36)^2], \tag{16}$$

$$U_{A_{31}}(x_1) = \exp[-1.34(x_1 - 5.54)^2], \tag{17}$$

$$U_{A_{32}}(x_2) = \exp[-1.88(x_2 - 2.0)^2]. \tag{18}$$

The following three fuzzy rules, which correspond respectively to i =1, 2, 3 in Eqn.(11), thus are constructed

R_1 IF petal-length is *nearly* 1.46 and petal-width is *nearly* 0.25,
THEN it is *sestosa*.

R_2 IF petal-length is *nearly* 4.29 and petal-width is *nearly* 1.36,
THEN it is *versicolor*.

R_3 IF petal-length is *nearly* 5.54 and petal-width is *nearly* 2.00,
THEN it is *virginica*.

Note that IF-part in above rules corresponds to the right side of Eqn.(11), and THEN-part the left side of Eqn.(11). Where, fuzzy numbers, *nearly* 1.46, *nearly* 0.25, *nearly* 4.29, *nearly* 1.36, *nearly* 5.54, and *nearly* 2.00 are respectively represented by fuzzy sets A_{11}, A_{12}, A_{21}, A_{22}, A_{31}, and A_{33} in Eqn.(11). The numbers, 1.46, 0.25, 4.29, 1.36, 5.54, and 2.00 are derived from Eqns. (13)-(18).

5. Conclusions

A fuzzy neural network that enables the classification of patterns and the selection of features is introduced. This algorithm includes two types of learning, i.e., unsupervised learning for memory connection and supervised learning for weight connection. Examples that are provided has demonstrated the ability of the feature-weighted detector (FWD) network to classify pattern and select feature. Moreover, distinct to traditional neural networks for which it is usually difficult to interpret the obtained knowledge [24], [26], our FWD provides interpretable rules that are of if-then form. These properties of the FWD suggest that it will be a promising method for pattern recognition.

References

[1] S. Watanabe, *Pattern Recognition*, Wiley Interscience, New York, 1985.

[2] P. A. Lachenbruch, *Discriminant Analysis*, New York: Hafner, 1975.

[3] N. P. Archer and S. Wang, “Fuzzy set representation of neural network classification boundaries”, *IEEE Trans. on SMC*, vol. 21, pp. 735-742, 1991.

[4] D. E. Rumelhart and J. L. McCleland, *Parallel Distributed Processing*, Cambridge, MA: MIT Press, 1986.

[5] G. A. Carpenter and S. Grossberg, *Pattern Recognition by Self-organizing Neural Networks*, Cambridge: MIT Press, 1991.

[6] S. Lee and R.M. Kil, “A Gaussian potential function network with hierarchically self-organizing learning”, *Neural Networks*, 4(2), pp. 207-224, 1991.

[7] T. Kohonen, *Self-organization and Associative Memory*, Berlin: Springer-verlag, 1989.

[8] D. F. Specht, "Probabilistic neural networks", *Neu. Net.*, vol. 3, pp. 109-118, 1990.

[9] L. A. Zadeh, “Fuzzy sets”, *Inform. and Contro.*, vol. 8, pp. 338-353, 1965.

[10] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York, Plenum, 1981.

[11] I. B. Turksen, “Measurement of membership functions and their acquisition”, *Fuzzy sets and systems* 40, pp. 5-38, 1991.

[12] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling", *IEEE Trans., Fuzzy Systems*, vol. 1, pp. 7-31, 1993.

[13] B. Kosko, *Neural networks and fuzzy systems*, Prentice Hall, 1991.

[14] J. M. Keller and D. T. Hunt, “Incorporating fuzzy membership functions into the perceptron algorithm”, *IEEE Trans. PAMI*, vol. PAMI-7, No. 6, pp. 693-699, 1985.

[15] L.I. Kuncheva, “Fuzzy rough sets: Application to feature selection”, *Fuzzy sets and Systems*, 51, pp. 147-153, 1992.
[16] R.-P. Li and M. Mukaidono, “Proportional learning law and local minimum escape in clustering networks”, *International Conference on Neural Information processing ICONIP*’95-Beijing, pp. 684-686, 1995.
[17] R.-P. Li and M. Mukaidono, “Proportional learning vector quantization”, *Journal of Japan Society for Fuzzy Theory and Systems*, vol. 10, No. 6, pp. 1129-1134, 1998.
[18] E. Anderson, "The IRISes of the Gaspe Peninsula", *bulletin of the American IRIS society*, vol. 59, pp. 2-5, 1939.
[19] N. R. Pal, J. C. Bezdek and E. C.-K. Tsao, “Generalized clustering networks and Kohonen’s self-organizing scheme”, IEEE Trans. On Neural Networks, vol. 4, pp. 549-557, 1993.
[20] R.-P. Li and M. Mukaidono, “A maximum-entropy approach to fuzzy clustering”, *International Joint Conference of FUZZ-IEEE/IFES*’95, pp. 2227-2233, 1995.
[21] R.-P. Li and M. Mukaidono, “Gaussian clustering method based on maximum-fuzzy entropy interpretation”, *Fuzzy Sets and Systems*, (102), pp. 153-258, 1999.
[22] P. K. Simpson, “Fuzzy min-max neural networks-part 1: Classification”, *IEEE Trans On Neural Networks*, vol. 3, No. 5, pp. 776-786, 1992.
[23] S. C. Newton, S. Pemmaraju and S. Mitra, “Adaptive fuzzy leader clustering of complex data sets in patterns recognition”, *IEEE Trans On Neural Networks*, vol. 3, No. 5, pp. 794-800, 1992.
[24] L. Fu, “Rule generation from neural networks”, *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 24, No. 8, pp. 1114-1124, 1994.
[25] S. K. Pal, S. Mitra, *Neuro-Fuzzy Pattern Recognition : Methods in Soft Computing*,
John Wiley, NY, 1999.
[26] S. Mitra, Y. Hayashi, “Neuro-fuzzy rule generation: Survey in soft computing framework”, *IEEE Trans. Neural Networks*, Vol. 11, No. 3, pp. 748-768, 2000.

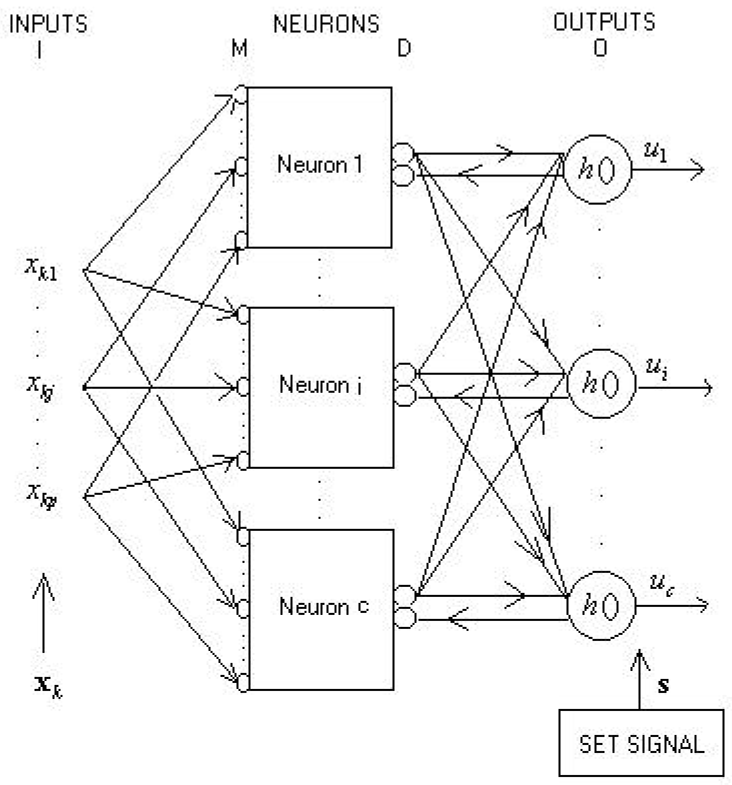
INPUTS
I
NEURONS
M
D
OUTPUTS
O
Neuron 1
Neuron i
Neuron c
x_{k1}
x_{kj}
x_{kp}
$\mathbf{x}_k$
$h()$
u_1
u_i
u_c
$\mathbf{s}$
SET SIGNAL

Fig. 1 (a)

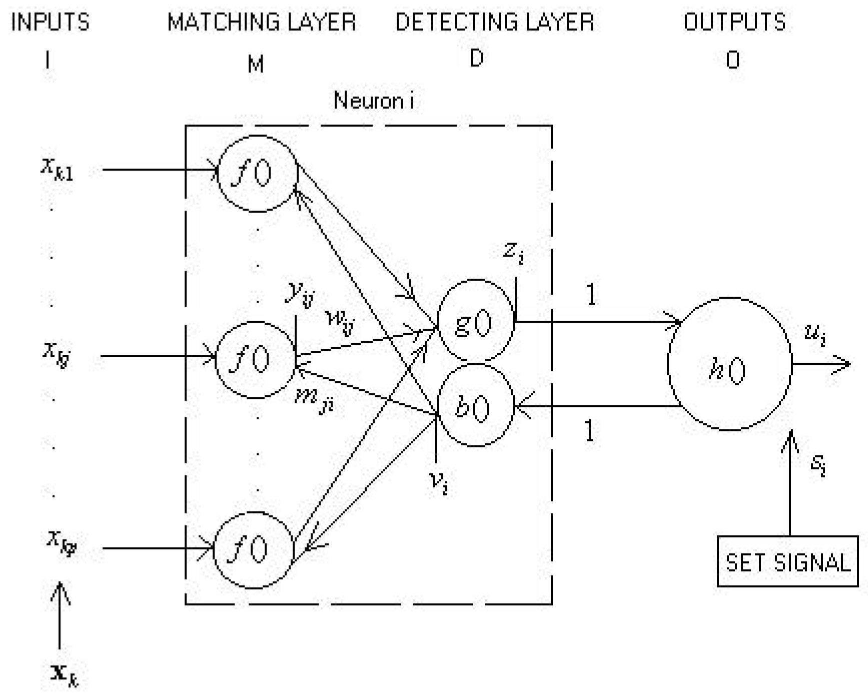

Fig. 1 (b)

Fig. 1. (a) A schematic diagram of the FWD network model. (b) Structure and interconnection of neuron *i*.

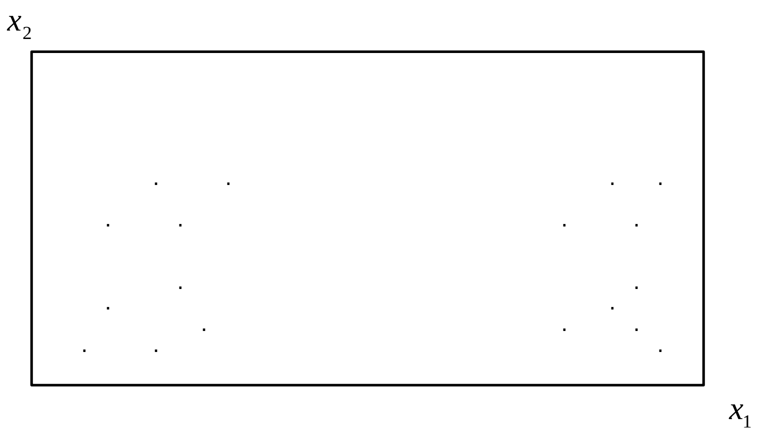

Fig. 2. Data set of Example 1.

Table 1. Classification results of data set of Fig. 2.

Data		Target Outputs		Outputs of Neurons	
k	**x**	u_1	u_2	u_1	u_2
1	(.05, .12)	1.00	0.00	0.989	0.011
2	(.10, .20)	1.00	0.00	0.901	0.099
3	(.15, .05)	1.00	0.00	0.998	0.002
4	(.20, .10)	1.00	0.00	0.993	0.007
5	(.20, .20)	1.00	0.00	0.899	0.101
6	(.70, .35)	0.00	1.00	0.118	0.882
7	(.75, .45)	0.00	1.00	0.008	0.992
8	(.80, .40)	0.00	1.00	0.033	0.967
9	(.85, .50)	0.00	1.00	0.002	0.998
10	(.80, .10)	1.00	0.00	0.993	0.007
11	(.83, .15)	1.00	0.00	0.974	0.026
12	(.85, .19)	1.00	0.00	0.925	0.075
13	(.90, .07)	1.00	0.00	0.997	0.003
14	(.87, .13)	1.00	0.00	0.985	0.015
15	(.10, .35)	0.00	1.00	0.122	0.872
16	(.15, .45)	0.00	1.00	0.008	0.992
17	(.20, .40)	0.00	1.00	0.032	0.968
18	(.25, .50)	0.00	1.00	0.002	0.998

Table 2. Experimental results for IRIS data set. Features x_1, x_2, x_3, and x_4 represent sepal length, sepal width, petal length, and petal width, respectively. e represents the number of mistakes in classification, and $\{\mathbf{w}_i\}$ represent weight vectors.

	Experiment 1 $\{x_1, x_2, x_3, x_4\}$	Experiment 2 $\{x_3, x_4\}$
s	0.50	0.50
a	0.01	0.01
b	0.10	0.10
e	2.00	2.00
T	1000	1000
e	5	5
$\mathbf{w}_1$	(1.00, 1.00, 0.95, 1.00)	(1.00, 1.00)
$\mathbf{w}_2$	(0.00, 0.00, 1.00, 1.00)	(1.00, 1.00)
$\mathbf{w}_3$	(0.00, 0.00, 0.82, 0.97)	(0.82, 0.97)