ELSEVIER

Pattern Recognition Letters 19 (1998) 43–51

Pattern Recognition Letters

Backpropagation learning algorithms for classification with fuzzy mean square error

Manish Sarkar [*], B. Yegnanarayana, Deepak Khemani

Neural Networks Laboratory, Department of Computer Science and Engineering, Indian Institute of Technology, Madras 600 036, India

Received 20 September 1996; revised 3 October 1997

Abstract

Most of the real life classification problems have ill defined, imprecise or fuzzy class boundaries. Feedforward neural networks with conventional backpropagation learning algorithm are not tailored to this kind of classification problem. Hence, in this paper, feedforward neural networks, that use backpropagation learning algorithm with fuzzy objective functions, are investigated. A learning algorithm is proposed that minimizes an error term, which reflects the fuzzy classification from the point of view of possibilistic approach. Since the proposed algorithm has possibilistic classification ability, it can encompass different backpropagation learning algorithm based on crisp and constrained fuzzy classification. The efficacy of the proposed scheme is demonstrated on a vowel classification problem. © 1998 Elsevier Science B.V.

Keywords: Crisp classification; Fuzzy classification; Possibilistic classification; Neural networks and backpropagation

1. Introduction

Feedforward neural networks (FFNN) based on backpropagation (BP) learning algorithm (Haykin, 1994; Yegnanarayana, 1994) are currently used extensively for pattern classification. However, a major drawback of the BP algorithm is that it assigns each input pattern exactly to one of the output classes, assuming well-defined class boundaries. In real life situations, however, boundaries between the classes may be overlapping. There can be certain input patterns that do not completely belong to a single class, but partially belong to the other classes too. This limits the applicability of the BP algorithm on the real life problems. In order to reduce this limitation, fuzzy sets based classification approach inside the basic framework of the BP algorithm, has been recently investigated. The use of fuzzy concept in pattern classification is also supported by the fact that the psycho-physiological process involved in human pattern classification does not employ precise or crisp mathematical formulations. Several interesting feedforward neuro-fuzzy systems have been proposed (Pal and Mitra, 1992; Pedrycz, 1992), and they cover a wide range of applications (Lin and Lee, 1996).

This research work proposes a method of embedding fuzzy classification properties into the conventional BP learning algorithm of FFNNs. Input is assumed to be crisp for these FFNNs, only the classification is fuzzy. A network applied for this kind of classification is needed to be passed through two phases, namely training and testing phase. In the training phase the network is trained with a given set of training patterns as inputs and the corresponding

[*] Corresponding author. E-mail: manish@bronto.iitm.ernet.in.

classes as target outputs. During training, the network adapts certain parameters, e.g. weights and bias, such that the network output reaches the target value. Since the classification is fuzzy, an input pattern may not necessarily belong to a single class; rather it may belong to more than one class with different degrees of belongingness. Consequently, unlike the conventional BP, the number of target classes corresponding to each input training pattern may be more than one. The aim of the proposed learning algorithm during training is to minimize an error term, henceforth termed as *fuzzy mean square error*. The fuzzy mean square error is the overall weighted sum of the square error between the actual network output and all possible target outputs, where the weight signifies the level of belongingness of the input pattern into the corresponding target class. If a new input pattern is presented to the network after training, it yields the output as class membership values of the corresponding input pattern.

The proposed learning algorithm is derived in such a manner that the sum total of the membership values for a particular pattern to all the classes need not necessarily be equal to one. This implies that the membership assignment is not *constrained fuzzy* (Pal and Bezdek, 1995); on the other hand, it is *possibilistic* (Pal and Bezdek, 1995). This idea of possibilistic membership assignment is in the line of possibility theory, which was first proposed by L. Zadeh (1978). The possibilistic membership assignment is desirable to signify the ignorance or different levels of evidence, which are well discussed in belief theory (Shafer, 1976) and possibility theory (Klir and Folger, 1993; Klir and Yuan, 1995). In the case of constrained fuzzy membership assignment, i.e., when the sum total of the membership values of an input pattern to all the classes is one, we show that the attractive learning algorithm, given by Pal and Mitra (1992), is equivalent to the proposed algorithm. In addition to it, when the classification is crisp, the proposed learning algorithm boils down to the conventional BP algorithm. Thus, it turns out that the possibilistic approach of the proposed algorithm leads it to encompass both constrained fuzzy classification and crisp classification. Another aspect of the proposed learning algorithm is that it has the scope for controlling the amount of fuzziness that is involved in the classification process.

2. Background of fuzzy classification

In this section we present different theoretical aspects of fuzzy sets for applying them to pattern classification problems.

2.1. Fuzzy sets

In traditional two-state classifiers, where a class $\boldsymbol{A}$ is defined as a subset of a universal set $\boldsymbol{X}$, any input pattern $\boldsymbol{x} \in \boldsymbol{X}$ can either be a member or not be a member of the given class $\boldsymbol{A}$. This property of whether or not a pattern $\boldsymbol{x}$ of the universal set belongs to the class $\boldsymbol{A}$ can be defined by a *characteristic function* $\mu_{\boldsymbol{A}}: \boldsymbol{X} \to \{0,1\}$ as follows:

$$\mu_{\boldsymbol{A}}(\boldsymbol{x}) = \begin{cases} 1 & \text{if and only if } \boldsymbol{x} \in \boldsymbol{A}, \\ 0 & \text{if and only if } \boldsymbol{x} \notin \boldsymbol{A} \end{cases}$$

In real life situations, however, boundaries between the classes may be overlapping. Hence, it is uncertain whether an input pattern belongs totally to the class $\boldsymbol{A}$. To take care of such situations, in fuzzy sets (Bezdek, 1981) the concept of the characteristic function has been modified to *membership function* $\mu_{\boldsymbol{A}}: \boldsymbol{X} \to [0,1]$. This function is called membership function, because larger value of the function denotes more membership of the element to the set under consideration.

A *C*-class classification problem for a set of input patterns $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_P\}$ is basically an assignment of the membership values $\mu_c(\boldsymbol{x}_p)$ on each $\boldsymbol{x}_p \in \boldsymbol{X}$, $\forall c = 1,2,\ldots,C$, $\forall p = 1,2,\ldots,P$. If the membership values are crisp, then $\boldsymbol{X}$ is partitioned into C subgroups during the classification process. In fuzzy context, C partitions of $\boldsymbol{X}$ are the set of values $\{\mu_c(\boldsymbol{x}_p)\}$, that can be conveniently arranged on a $C \times P$ matrix $\boldsymbol{U} = [\mu_c(\boldsymbol{x}_p)]$. Based on the characteristic of $\boldsymbol{U}$, classification can be of the following three types (Pal and Bezdek, 1995):

1. *Crisp classification*:

$$M_{\mathrm{hc}} = \left\{ \boldsymbol{U} \in \mathbb{R}^{CP} \mid \mu_c(\boldsymbol{x}_p) \in \{0,1\} \ \forall c, \forall p; \right.$$
$$\left. \sum_{c=1}^{C} \mu_c(\boldsymbol{x}_p) = 1; 0 < \sum_{p=1}^{P} \mu_c(\boldsymbol{x}_p) < P \ \forall c \right\}. \quad (1)$$

2. *Constrained fuzzy classification*:

$$M_{\mathrm{fc}} = \left\{ \boldsymbol{U} \in \mathbb{R}^{CP} \mid \mu_c(\boldsymbol{x}_p) \in [0,1]\, \forall c, \forall p; \right.$$
$$\left. \sum_{c=1}^{C} \mu_c(\boldsymbol{x}_p) = 1; 0 < \sum_{p=1}^{P} \mu_c(\boldsymbol{x}_p) < P\ \forall c \right\}. \quad (2)$$

3. *Possibilistic classification*:

$$M_{\mathrm{pc}} = \left\{ \boldsymbol{U} \in \mathbb{R}^{CP} \mid \mu_c(\boldsymbol{x}_p) \in [0,1]\, \forall c, \forall p; \right.$$
$$\left. 0 < \sum_{p=1}^{P} \mu_c(\boldsymbol{x}_p) < P\ \forall c \right\}. \quad (3)$$

Fig. 1 shows a situation containing two classes. Here, both the patterns A and B are equidistant from the two classes. In crisp classification, the membership value of A in one class will be 1 and in the other class it will be 0. It is true for the pattern B also. Obviously, this kind of membership assignment does not reflect the intuitive classification situation as A and B partially belong to both the classes. In constrained fuzzy membership assignment, both the patterns A and B will be assigned the membership values equal to 0.5. Although this membership assignment is better than the crisp counterpart, it fails to consider the pattern A as a more typical one than the pattern B. It is because, here the membership assignment is a relative one, and it depends on the membership values to both the classes. In possibilistic membership assignment, the pattern A will receive equal membership values to both the classes. It is true for B also. But, the membership of B in any class will always be less than that of pattern A. Therefore, the possibilistic assignment may not be summed to one, and thus, it can distinguish between equal evidence and ignorance (Krishnapuram and Keller, 1993). This property of the possibilistic assignment makes it attractive compared to the other two assignments. From the relations (1), (2) and (3), it is obvious that $M_{\mathrm{hc}} \subset M_{\mathrm{fc}} \subset M_{\mathrm{pc}}$. We will see later that our proposed learning algorithm is based on the possibilistic classification, and hence as a natural consequence, various BP algorithms based on the constrained fuzzy and crisp classification become particular cases of the proposed algorithm.

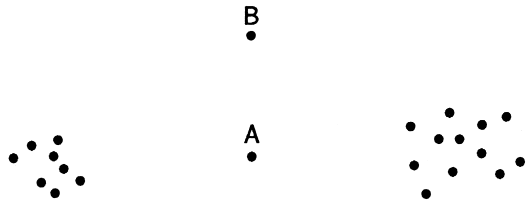

Fig. 1. The crisp membership values of data A and B in both classes are either 0 or 1. The constrained fuzzy membership values of data A and B in both classes are about 0.5, which do not consider the fact that B is much less representative of either class than A.

From the above discussion, it is apparent that any concept that uses fuzzy sets requires the membership function to be defined. This function is usually designed by taking into consideration the requirements and constraints of the problem.

2.2. Computation of membership values for fuzzy classification

This part of the discussion describes how to determine the membership value of each input pattern. The membership value of the pth input pattern to the class c is defined as (Pal and Dutta Majumder, 1986)

$$\mu_c(\boldsymbol{x}_p) = \frac{1}{1 + (z_{\mathrm{pc}}/F_{\mathrm{d}})^{F_{\mathrm{e}}}}, \quad (4)$$

where z_{pc} is the weighted distance, and the positive constants F_{d} and F_{e} are the denominational and exponentional fuzzy generators controlling the amount of fuzziness in this class-membership set. The weighted distance is discussed in detail later. Obviously, $\mu_c(\boldsymbol{x}_p)$ lies in the interval [0,1]. Specifically, the higher the distance of a pattern from a class, the lower is its membership value to that class. In particular, when the distance is zero, the membership value is one (maximum), and on the other hand, when the distance is infinite, the membership value is zero (minimum). The method of calculating the weighted distance is as follows.

Let the N-dimensional vectors $\boldsymbol{m}_c$ and $\boldsymbol{\sigma}_c$ denote the mean and standard deviation, respectively, of the set of training patterns for the cth class. The weighted distance of a training pattern $\boldsymbol{x}_p = [x_{p1}, x_{p2}, \ldots, x_{pN}]^{\mathrm{T}}$ from the cth class is defined as (Pal and Dutta Majumder, 1986)

$$z_{\mathrm{pc}} = \sqrt{\sum_{i=1}^{N} \left[\frac{x_{pi} - m_{ci}}{\sigma_{ci}} \right]^2}, \quad \forall c = 1, \ldots, C. \tag{5}$$

The weight $1/\sigma_{cj}$ is used to take care of the variance of the classes so that a feature with higher variance has less weight (significance) in characterizing a class (Pal and Dutta Majumder, 1986). Note that when the ith feature values of all the patterns from the cth class are the same, then the standard deviation σ_{ci} will be zero. In such a situation, we consider $\sigma_{ci} = \varepsilon$, where ε is a very small positive value. In this work the mean and standard deviation of each class are calculated from the set of input training data corresponding to that particular class. Note that here $\sum_c \mu_c(\boldsymbol{x}_p)\ \forall p$ needs not be equal to one.

However, while choosing the mean as the prototype of a class, we must be careful enough to avoid the outliers present in the input data set. To avoid the effect of the outliers, we first calculate the mean $\boldsymbol{m}_c$ and the standard deviation $\boldsymbol{\sigma}_c$ of the cth class by considering all the training inputs from that class. Then, we recalculate the mean of the cth class by taking the training input patterns of the cth class which lie in between $\boldsymbol{m}_c - 2\boldsymbol{\sigma}_c$ and $\boldsymbol{m}_c + 2\boldsymbol{\sigma}_c$. It can be shown that the speech data from each class form a normal distribution. As the distribution is normal, we basically recalculate the mean based on the 95.45% data from the cth class. Since the outliers lie at the boundary of the class, there are high chances that the outliers will come from the remaining 4.55% patterns, that are lying at the boundary of the class. Thus, the effect of the outliers is reduced. In order to avoid outliers, we could have chosen the median in place of the mean as the class representative. Although the mean and median each provides a single number to represent an entire set of patterns from a class, the mean is usually preferred in the problems of estimation. An intuitive reason is that the median is generally subject to greater chance of fluctuation, i.e., it is apt to vary more from sample to sample. Hence, in this work we decided to choose mean as the class prototype.

3. Back propagation algorithm with fuzzy objective functions

Let, the training set in a C-class problem consists of vector pairs $\{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_P, \boldsymbol{y}_P)\}$, where $\boldsymbol{x}_p \in \mathbb{R}^N$ refers to the pth input pattern and $\boldsymbol{y}_p \in \{\boldsymbol{t}_c |, c = 1,2,\ldots,C;\ \boldsymbol{t}_c \in \mathbb{R}^C\}$ refers to the target output of the network corresponding to this input. Specifically, if $\boldsymbol{x}_p$ is from the kth class, then $\boldsymbol{y}_p = \boldsymbol{t}_k$, where $t_{kk} = 1$ and $t_{ck} = 0\ \forall c,\ c \neq k$.

3.1. Architecture

The network used here is a multilayer feedforward network which can have several hidden layers. Without loss of generality, number of the hidden layers can be assumed to be one with H hidden nodes. When an input pattern $\boldsymbol{x}_p = (x_{p1}, x_{p2}, \ldots, x_{pN})^{\mathrm{T}}$ is applied at the input layer of the network, the input units distribute the values to the hidden layer units. The output of the jth hidden unit is

$$o_{pj}^h = f_j^h\left(\mathrm{net}_{pj}^h\right) = \frac{1}{1 + \exp\left(-\mathrm{net}_{pj}^h\right)},$$

where $\mathrm{net}_{pj}^{\mathrm{h}} = \sum_{i=1}^{N} w_{ji}^{\mathrm{h}} x_{pi} + \theta_j^{\mathrm{h}}$. Here, w_{ji}^{h} is the weight of the link from the ith input node to the jth hidden node. θ_j^{h} and f_j^{h} are the bias term and transfer function of the jth hidden node. Similarly, the output of the kth output node is

$$o_{pk}^{\mathrm{o}} = f_k^{\mathrm{o}}\left(\mathrm{net}_{pk}^{\mathrm{o}}\right) = \frac{1}{1 + \exp\left(-\mathrm{net}_{pk}^{\mathrm{o}}\right)},$$

where $\mathrm{net}_{pk}^{\mathrm{o}} = \sum_{j=1}^{H} w_{kj}^{\mathrm{o}} f_j^{\mathrm{o}}(\mathrm{net}_{pj}^{\mathrm{o}}) + \theta_k^{\mathrm{o}}$. The h and o superscripts refer to the quantities in the hidden and output layers, respectively.

3.2. Training

The adaptive parameters of the FFNN consist of all weights and bias terms. The sole purpose of the training phase is to determine the optimum setting of the weights and bias terms so as to minimize the difference between the network output and the target output. This difference is referred to as training error of the network. The error measure can be fuzzy mean square error, which is basically a fuzzy counterpart of the mean square error (Haykin, 1994) used in the conventional BP algorithm.

In the conventional BP algorithm, the mean square error for the pth input pattern is defined as $E_p = \frac{1}{2}\sum_{k=1}^{C}(t_{pk} - o_{pk}^{\mathrm{o}})^2$. However, the use of E_p as an error term is justified when each input pattern belongs to only one class. But, in fuzzy classification the input pattern may belong to more than one class with different degrees of belongingness. It implies that the target value of an input pattern may be more than one. In other words, each input pattern can have all possible target values with different membership values (certain membership values may be zero also). Through training the network attempts to reach those target values weighted by different membership values. In other words, the problem of training can also be conceptually viewed as a fuzzy constraint satisfaction problem. Here, the constraint is that each input pattern should belong to a particular class, and the associated membership value signifies upto what extent this constraint should be satisfied. In the training phase, the proposed network adapts the parameters so that these constraints are resolved optimally. Mathematically, for the pth input pattern the constraints can be expressed as the fuzzy mean square error term, which is defined as

$$E_p^{\mathrm{f}} = \frac{1}{2}\sum_{k=1}^{C}\sum_{c=1}^{C}\mu_c^q(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^{\mathrm{o}}\right)^2. \tag{6}$$

Here, the index of μ, i.e., $q \in [0,\infty)$ controls the amount of fuzziness present into the classification. Different values of q signifies upto what extent, the constraints should be satisfied. When $q = 0$, each input pattern tries to attain all the target outputs with equal importance, and ultimately the network learns the mean of all the class centers. When the value of q is greater than one, the constraints associated with the high membership values get more importance to be resolved. When q tends to be infinity, only the input pattern which belongs to a class completely, i.e. with membership one, is learned. Specifically, the larger the q is, the less fuzzier are the membership assignments. Consequently, we can observe that E_p^{f} decreases strictly towards zero as q increases in $[1,\infty]$ for $0 < \mu_c(\boldsymbol{x}_p) < 1\ \forall c$ (see Appendix A). On the other hand, when q is less than one, the constraints associated with the high membership values get less importance to be resolved. Thus, q controls the extent of the membership sharing between the fuzzy classes. This can be good; on the other hand, one must choose q to actually implement it. In our work q is assumed to be one. The role of q here is quite similar to the index of fuzziness in the concentration and dilation operators found in *fuzzy hedge* (Klir and Folger, 1993), and the index of fuzziness in *fuzzy C-means clustering algorithm* (Bezdek, 1981).

Next, we derive the learning laws for the network following the same method as followed in the conventional BP algorithm (Haykin, 1994). Here, we assume that the weight updation, Δw, takes place after the presentation of each input pattern. Assuming the use of same learning-rate parameter η for all the weight changes made in the network, the weight changes applied to the weights w_{kj} and w_{ji} are calculated, respectively, in accordance to the gradient-descent rules:

$$\Delta w_{kj}^{\mathrm{o}} = -\eta\frac{\partial E_p^{\mathrm{f}}}{\partial w_{kj}^{\mathrm{o}}} \quad\text{and}\quad \Delta w_{ji}^{\mathrm{h}} = -\eta\frac{\partial E_p^{\mathrm{f}}}{\partial w_{ji}^{\mathrm{h}}}.$$

From Appendix B we can write

$$\Delta w_{kj}^{\mathrm{o}} = \eta\left[\mu_k^q(\boldsymbol{x}_p) - \sum_{c=1}^{C}\mu_c^q(\boldsymbol{x}_p)\,o_{pk}^{\mathrm{o}}\right] \times o_{pk}^{\mathrm{o}}\left(1 - o_{pk}^{\mathrm{o}}\right)o_{pj}^{\mathrm{h}} \tag{7}$$

$$= \eta\delta_{pk}^{\mathrm{o}}o_{pj}^{\mathrm{h}}, \tag{8}$$

where

$$\delta_{pk}^{\mathrm{o}} = \left[\mu_k^q(\boldsymbol{x}_p) - \sum_{c=1}^{C}\mu_c^q(\boldsymbol{x}_p)\,o_{pk}^{\mathrm{o}}\right]o_{pk}^{\mathrm{o}}\left(1 - o_{pk}^{\mathrm{o}}\right).$$

Again, from Appendix C,

$$\Delta w_{ji}^{\mathrm{h}} = \eta f_j^{\prime\mathrm{h}}\left(\mathrm{net}_{pj}^{\mathrm{h}}\right)x_{pi}\sum_{k=1}^{C}\left[\mu_k^q(\boldsymbol{x}_p) - \sum_{c=1}^{C}\mu_c^q(\boldsymbol{x}_p)\,o_{pk}^{\mathrm{o}}\right]o_{pk}^{\mathrm{o}}\left(1 - o_{pk}^{\mathrm{o}}\right)w_{kj}^{\mathrm{o}} \tag{9}$$

$$= \eta f_j^{\prime\mathrm{h}}\left(\mathrm{net}_{pj}^{\mathrm{h}}\right)x_{pi}\sum_{k=1}^{C}\delta_{pk}^{\mathrm{o}}w_{kj}^{\mathrm{o}} \tag{10}$$

$$= \eta\delta_{pj}^{\mathrm{h}}x_{pi}, \tag{11}$$

where

$$\delta_{pj}^{\mathrm{h}} = f_j^{\prime\mathrm{h}}\left(\mathrm{net}_{pj}^{\mathrm{h}}\right) \sum_{k=1}^{C} \delta_{pk}^{\mathrm{o}} w_{kj}^{\mathrm{o}}.$$

In the learning equations, for faster learning momentum (Haykin, 1994) term can be used. Moreover, to make the whole learning faster, learning-rate can be adaptive, i.e., its value can increase or decrease dynamically as the learning algorithm progresses.

The conventional BP algorithm may not converge quickly in many cases, especially, when the classes are overlapping (Pal and Mitra, 1992). It is because the ambiguous vectors are given full weightage in one class. In the proposed version, the error to be backpropagated has more weightage in case of the nodes with higher membership values, and hence, can induce greater weight corrections in favour of that class for an input data that demands such adjustment. Thus, the contribution of the ambiguous or uncertain vectors to the weight correction is reduced and as a result the convergence becomes easier. It is evident that we can obtain this advantage fully only when the membership assignment is not any way constrained, i.e., possibilistic.

Now we illustrate the following particular cases of the proposed learning algorithm.

(1) *Crisp Classification*: In case of crisp classification only one component of $\boldsymbol{\mu}_c^q(\boldsymbol{x}_p)\ \forall c = 1,\dots,C$, is one and the remaining components are zero. Thus, the expression for E_p^{f} boils down to the following expression:

$$E_p^{\mathrm{f}} = -\frac{1}{2} \sum_{k=1}^{C} \sum_{c=1}^{C} \left(t_{cp} - o_{pk}\right)^2 \tag{12}$$

which is the mean square error term found in the conventional BP algorithm. Consequently, in a crisp case the learning algorithm based on mean square error and fuzzy mean square error become identical. This can be easily verified by making the membership assignments in (6) and (9) crisp.

(2) *Constrained fuzzy classification*: When $\sum_c \mu_c(\boldsymbol{x}_p) = 1\ \forall p$ and $q = 1$, the learning equations (6) and (9) achieve simpler forms as follows:

$$\Delta w_{kj}^{\mathrm{o}} = \eta \delta_{pk}^{\mathrm{o}} o_{pj}^{\mathrm{h}} \tag{13}$$

$$\Delta w_{ji}^{\mathrm{h}} = \eta \delta_{pj}^{\mathrm{h}} x_{pi} \tag{14}$$

where

$$\delta_{pk}^{\mathrm{o}} = \left[\, \mu_k(\boldsymbol{x}_p) - o_{pk}^{\mathrm{o}} \right] o_{pk}^{\mathrm{o}} \left(1 - o_{pk}^{\mathrm{o}}\right)$$

and

$$\delta_{pj}^{\mathrm{h}} = f_j^{\prime\mathrm{h}}\left(\mathrm{net}_{pj}^{\mathrm{h}}\right) \sum_{k=1}^{C} \delta_{pk}^{\mathrm{o}} w_{kj}^{\mathrm{o}}.$$

We can note down that this particular version of the proposed algorithm is available as the learning algorithm proposed by Pal et al. in (Pal and Mitra, 1992). It is also important to note that we are not considering Pal et al.'s algorithm with fuzzy linguistic input; rather we are considering it with crisp inputs. In the future correspondence, the proposed algorithm will be extended to take care of the fuzzy linguistic input.

Thus, being possibilistic in nature, the proposed algorithm encapsulates various BP algorithms based on crisp as well as constrained fuzzy classification.

3.3. Testing

The network learns the fuzzy boundaries between the classes after training. In this stage, a separate set of test patterns is given as the inputs to the network. Generated outputs are the class membership values corresponding to the respective test inputs.

4. Results and discussion

We consider the task of vowel recognition (Rabiner and Juang, 1993) to demonstrate the efficiency of the proposed scheme. For our study we consider the vowels ''a'', ''e'', ''i'', ''o'' and ''u''. The data required for training is collected from vowel part of the utterances of consonant vowel pairs of three different speakers. The raw speech signal cannot be used directly for training the networks because the features are deep hidden. Therefore, we extract features from the speech signal and use them for training the network. For this purpose we consider first three formants obtained from the utterances of speech as the extracted features. The formants are extracted by taking the LPC and finding the frequencies at which the spectrum reaches peaks.

We use different BP learning paradigms to train a feedforward neural network with 3 input nodes, 5

Table 1
Classification results of the conventional BP and the proposed BP on a set of formant data with two different network architectures

No. of hidden nodes	Conventional BP	Proposed BP
10	74.87%	80.21%
15	77.05%	82.11%

output nodes and variable number of hidden nodes. The target patterns are 5-dimensional vectors containing 1 in one location and 0 in all others. Here, we adopt the strategy of picking the output node with the highest activation value as the output class corresponding to an input. The learning-rate is adaptively changed for the learning algorithms in the following way: If the error decreases during training, then the learning-rate is increased by a predefined amount. In contrast, if the error increases, then the learning-rate is decreased and the new weights and errors are discarded. As a result, the error always decreases or stays as it is. The momentum is kept constant throughout the process. The values of F_e, F_d and q are taken as 2, 5 and 1, respectively.

In the first experiment, we use two different network architectures to compare the performance of the proposed learning algorithm and the conventional BP learning algorithm. The two architectures are based on ten and fifteen hidden nodes, respectively. The classification performance of the proposed algorithm and the conventional algorithm over these two architectures are shown in the first and second rows of Table 1. We take a training set of 200 examples and evaluate their performances on 1000 samples of test data. From Table 1, we can observe that the proposed algorithm performs well in both the cases.

In the second experiment, we use different learning techniques on an another training set of 300 examples and evaluate their performances on 1800 samples of test data. We use Bayes classifier for multivariate normal patterns with the a priori probabilities $p_i = P_i/P$, where P_i denotes (Pal and Dutta Majumder, 1986) the number of patterns in the ith class and P is the total number of training patterns. The covariance matrices for each class is determined from the training patterns of that particular class. Classification performance of the Bayes classifier on the test set is shown in the second column of Table 2. Next we use three different BP learning paradigms on the data set to train a feedforward neural network with 3 input nodes, 5 hidden nodes and 5 output nodes. Classification efficiency of the network on the test set is demonstrated in the third column of Table 2. The fourth and fifth columns of Table 2 show the classification performance of the same network trained with Pal et al.'s algorithm and the proposed algorithm, respectively.

The Bayes classifier gives optimal classification performance, provided the parameters of the input distribution are estimated from the inputs collected over the whole input space. In practice, the distribution parameters are estimated based only on a finite number of training data. As a result, the performance of the Bayes classifier is no longer optimal, but its performances approaches the optimal one as the number of input data is made very large (theoretically, it is infinity). Nevertheless, in Table 2, Bayes classifier, based on a finite number of training samples, is used to compare the performance of the proposed method. From Table 2, we can observe that the performance of the conventional BP algorithm is comparable to that of the Bayes classifier. However,

Table 2
Results of vowel classification for different types of classification algorithms

Class	Bayes classifier	Conventional BP	Pal et al.'s algorithm	Proposed algorithm
‘‘a’’	84.89%	80.36%	87.09%	90.27%
‘‘e’’	82.87%	80.59%	85.66%	86.60%
‘‘i’’	87.34%	82.31%	87.21%	89.41%
‘‘o’’	80.29%	86.89%	83.75%	86.45%
‘‘u’’	84.75%	87.91%	92.73%	93.78%
overall	84.28%	83.61%	87.28%	89.39%

the classification results are significantly improved when Pal et al.'s algorithm is used. This improvement is possible because of the consideration of the fuzziness involved in the classification process. The overall classification result is further enhanced when the proposed algorithms are used. The proposed algorithms can find the fuzzy decision boundary more accurately as some input patterns (especially, at the borders or away from the classes) may not satisfy the condition $\sum_c \mu_c(\boldsymbol{x}_p) = 1$. Thus, the possibilistic approach is more appropriate here. In this work, we did not attempt to optimize the number of hidden nodes of the FFNN. This can be an attractive research issue for the future study. For further reading, see (Kandle, 1986; Rumelhart et al., 1986).

Acknowledgements

This work was carried out while Manish Sarkar held a Dr K.S. Krishnan Research Fellowship from the Department of Atomic Energy, Government of India.

Appendix A. E_p^f is a monotonically decreasing function

For $0 < \mu_c(\boldsymbol{x}_p) < 1$, one might suspect that E_p^f decreases monotonically with q, when $q > 1$. To see that this is indeed the case, differentiating E_p^f with respect to q we get

$$\frac{\partial E_p^f}{\partial q} = \frac{1}{2} \sum_{k=1}^{C} \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p) \ln\left(\mu_c^q(\boldsymbol{x}_p)\right)\left(t_{ck} - o_{pk}^o\right)^2 \tag{A.1}$$

$$= \frac{1}{2} \sum_{k=1}^{C} \sum_{c=1}^{C} \left[\mu_c(\boldsymbol{x}_p) \ln\left(\mu_c(\boldsymbol{x}_p)\right)\right] \times \left[\mu_c^{q-1}(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^o\right)^2\right]. \tag{A.2}$$

With the usual convention that $x \ln(x) = 0$ if $x = 0$, we have $[\mu_c(\boldsymbol{x}_p) \ln(\mu_c(\boldsymbol{x}_p))] \leqslant 0$ and $[\mu_c^{q-1}(\boldsymbol{x}_p)(t_{ck} - o_{pk}^o)^2] \geqslant 0 \ \forall c,k$. Both the inequalities being strict whenever $0 < \mu_c(\boldsymbol{x}_p) < 1$. Hence, when $0 < \mu_c(\boldsymbol{x}_p) < 1$, E_p^f strictly decreases (Bezdek, 1981) on every finite interval of the form $[1,b]$ with $1 < b$.

Appendix B. The expression for $\partial E_p^f / \partial w_{kj}^o$

The expression for $\partial E_p^f / \partial w_{kj}^o$ can be derived as

$$\frac{\partial E_p^f}{\partial w_{kj}^o} = -\sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^o\right) \frac{\partial f_k^o\left(\text{net}_{pk}^o\right)}{\partial\left(\text{net}_{pk}^o\right)} \frac{\partial\left(\text{net}_{pk}^o\right)}{\partial w_{kj}^o} \tag{B.1}$$

$$= -\sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^o\right) o_{pk}^o\left(1 - o_{pk}^o\right) o_{pj}^h \tag{B.2}$$

$$= -\left[\mu_k^q(\boldsymbol{x}_p)\left(t_{kk} - o_{pk}^o\right) + \sum_{c=1_{c \neq k}}^{C} \mu_c^q(\boldsymbol{x}_p) \times \left(t_{ck} - o_{pk}^o\right)\right] o_{pk}^o\left(1 - o_{pk}^o\right) o_{pj}^h. \tag{B.3}$$

Since $t_{kk} = 1$ and $t_{ck} = 0 \ \forall c \neq k$,

$$\frac{\partial E_p^f}{\partial w_{kj}^o} = -\left[\mu_k^q(\boldsymbol{x}_p)\left(1 - o_{pk}^o\right) - \sum_{c=1_{c \neq k}}^{C} \mu_c^q(\boldsymbol{x}_p) o_{pk}^o\right] \times o_{pk}^o\left(1 - o_{pk}^o\right) o_{pj}^h \tag{B.4}$$

$$= -\left[\mu_k^q(\boldsymbol{x}_p) - \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p) o_{pk}^o\right] o_{pk}^o\left(1 - o_{pk}^o\right) o_{pj}^h. \tag{B.5}$$

Appendix C. The expression for $\partial E_p^f / \partial w_{ji}^h$

The expression for $\partial E_p^f / \partial w_{ji}^h$ can be found as follows:

$$\frac{\partial E_p^f}{\partial w_{ji}^h} = -\sum_{k=1}^{C} \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^o\right) \times \frac{\partial f_k^o\left(\text{net}_{pk}^o\right)}{\partial\left(\text{net}_{pk}^o\right)} \frac{\partial\left(\text{net}_{pk}^o\right)}{\partial o_{pj}^o} \frac{\partial o_{pj}^o}{\partial\left(\text{net}_{pj}^h\right)} \frac{\partial\left(\text{net}_{pj}^h\right)}{\partial w_{ji}^h} \tag{C.1}$$

$$= -\sum_{k=1}^{C} \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p)\left(t_{ck} o_{pk}^o\right) o_{pk}^o\left(1 - o_{pk}^o\right) w_{kj}^o \acute{f}_j^h \times \left(\text{net}_{pj}^h\right) x_{pi} \tag{C.2}$$

$$= -f_j^{\prime \mathrm{h}}\left(\mathrm{net}_{pj}^{\mathrm{h}}\right) x_{pi} \sum_{k=1}^{C} \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^{\mathrm{o}}\right) o_{pk}^{\mathrm{o}} \times \left(1 - o_{pk}^{\mathrm{o}}\right) w_{kj}^{\mathrm{o}}. \qquad \text{(C.3)}$$

Following the steps involved while deriving (B.5) from (B.2), we can write

$$\mu_k^q(\boldsymbol{x}_p) - \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p) o_{pk}^{\mathrm{o}} \equiv \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p)\left(t_{ck} - o_{pk}^{\mathrm{o}}\right). \qquad \text{(C.4)}$$

Hence,

$$\frac{\partial E_p^{\mathrm{f}}}{\partial w_{ji}^{\mathrm{h}}} = -f_j^{\prime \mathrm{h}}\left(\mathrm{net}_{pj}^{\mathrm{h}}\right) x_{pi} \times \sum_{k=1}^{C} \left[\mu_k^q(\boldsymbol{x}_p) - \sum_{c=1}^{C} \mu_c^q(\boldsymbol{x}_p) o_{pk}^{\mathrm{o}} \right] \times o_{pk}^{\mathrm{o}}\left(1 - o_{pk}^{\mathrm{o}}\right) w_{kj}^{\mathrm{o}}. \qquad \text{(C.5)}$$

References

Bezdek, J.C., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Haykin, S., 1994. Neural Networks – A Comprehensive Foundation. Macmillan College Publishing Company, New York.

Kandle, A., 1986. Fuzzy Mathematical Techniques with Applications. Addison-Wesley, Reading, MA.

Klir, G.S., Folger, T.A., 1993. Fuzzy Sets, Uncertainty and Information. Prentice-Hall, Englewood Cliffs, NJ.

Klir, G.S., Yuan, B., 1995. Fuzzy sets and Fuzzy Logic – Theory and Applications. Prentice-Hall, Englewood Cliffs, NJ.

Krishnapuram, R., Keller, J.M., 1993. A possibilistic approach to clustering. IEEE Trans. Fuzzy Systems 1 (2), 98–110.

Lin, C.T., Lee, C.S.G., 1996. Neural Fuzzy Systems. Prentice Hall, Englewood Cliffs, NJ.

Pal, N.R., Bezdek, J.C., 1995. On cluster validity for the fuzzy C-means model. IEEE Trans. Fuzzy Systems 3 (3), 330–379.

Pal, S.K., Dutta Majumder, D., 1986. Fuzzy Mathematical Approach to Pattern Recognition. Wiley (Halsted Press), New York.

Pal, S.K., Mitra, S., 1992. Multilayer perceptron, fuzzy sets and classification. IEEE Trans. Neural Networks 3 (5), 683–697.

Pedrycz, W., 1992. Fuzzy neural networks with reference neurons as pattern classifiers. IEEE Trans. Neural Networks 3 (5), 770–775.

Rabiner, L.R., Juang, B.H., 1993. Fundamentals of Speech Recognition. Prentice Hall, Englewood Cliff, NJ.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning internal representations by error propagation. In: D.E. Rumelhart and McClelland (Eds.), Parallel and Distributed Processing. MIT Press, Cambridge, MA.

Shafer, G., 1976. A Mathematical Theory of Evidence. Princeton University Press, Princeton.

Yegnanarayana, B., 1994. Artificial neural networks for pattern recognition. Sadhana 19 (1), 147–169.

Zadeh, L.A., 1978. Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets and Systems 1, 3–28.