Pergamon

Pattern Recognition, Vol. 30, No. 10, pp. 1579–1590, 1997
© 1997 Pattern Recognition Society. Published by Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0031-3203/97 $17.00+.00

PII: S0031-3203(96)00190-2

FEATURE ANALYSIS: NEURAL NETWORK AND FUZZY SET THEORETIC APPROACHES

RAJAT K. DE, NIKHIL R. PAL* and SANKAR K. PAL
Machine Intelligence Unit, Indian Statistical Institute, Calcutta 700035, India

(*Received* 8 *July* 1994; *in revised form* 7 *October* 1996)

Abstract—In this paper a new scheme of feature ranking and hence feature selection using a Multilayer Perceptron (MLP) Network has been proposed. The novelty of the proposed MLP-based scheme and its difference from another MLP-based feature ranking scheme have been analyzed. In addition we have modified an existing feature ranking/selection scheme based on fuzzy entropy. Empirical investigations show that the proposed MLP-based scheme is superior to the other schemes implemented. © 1997 Pattern Recognition Society. Published by Elsevier Science Ltd.

Feature ranking Feature selection Neural networks Fuzziness measure

1. INTRODUCTION

A pattern recognition system may be defined as a function which transforms measurement space into decision space via feature space, i.e. $\mathcal{M} \rightarrow \mathcal{F} \rightarrow \mathcal{D}$, where $\mathcal{M}$ is the measurement space, $\mathcal{F}$ is the feature space and $\mathcal{D}$ is the decision space. The measurement space is defined as the set of measurable quantities; feature space may include a subset of the measurement space and/or some new attributes derived based on two or more measurable quantities. The decision space, on the other hand, consists of the decision(s) made from the feature space. In a recognition problem there are many measurable or detectable quantities based on which the object is recognized. But all of them may not be important or have significant impact on the recognition process. Some features may be redundant and confusing also. Therefore, to reduce space–time complexity and to avoid confusion, one often needs to analyze/extract/select features from $\mathcal{M}$. The transformation $\mathcal{M} \rightarrow \mathcal{F}$ thus constitutes an important integral part of a pattern recognition system. We divide feature analysis task ($\mathcal{M} \rightarrow \mathcal{F}$) into two parts: selection and extraction. Feature selection deals with choosing some of the measurable quantities which are important for discrimination and have a great impact on the decision space, while feature extraction deals with developing some new attributes (features) based on some selected measurable quantities. In this investigation we restrict ourselves to feature selection only. Selection of features can be done by ranking the features first and then ignoring some which are near the bottom or by investigating the combined effect of a set of features on discrimination.

There are many techniques for feature selection. Some of these techniques are based on interclass and intraclass distances,[1] some are based on probabilistic/fuzzy models,[2–4] while others are based on neural networks.[5–8] Each approach has its own advantages and drawbacks.

In this paper we have proposed two schemes for feature analysis. One of the schemes is a modification of the fuzzy set theoretic method of Pal,[4] while the other approach uses a multilayer perceptron. Given a labeled data set we first learn it using an MLP with an adequate architecture. Then, for each training data point we set a feature value to zero (one may call such a vector as corrupted data point) and use it as an input for classification. The deviation of the output vector thus produced from the output generated by the corresponding uncorrupted data point is noted. A feature is considered more important if the average deviation over the entire data set for that feature is more. The basic idea behind this scheme is as follows. After the MLP has learnt the data set, the absence of an important feature is likely to influence the output significantly. On the other hand, for a less important feature, the output is not expected to change much with variation of the value of that feature. Performance of the proposed schemes has been compared with some existing schemes using a few data sets.

The rest part of the paper is organized as follows: Section 2 discusses some of the existing techniques for feature ranking and basics of an MLP, while Section 3 deals with the proposed scheme. Section 4 describes the experimental results and the paper is concluded in Section 5.

2. SOME EXISTING TECHNIQUES OF FEATURE RANKING

In this section, we describe some existing techniques for feature ranking and/or selection.

2.1. *Feature ranking based on criterion function (2)*

For reducing the dimension of the measurement space, we should eliminate those features which are less im-

* Author to whom correspondence should be addressed. E-mail: res9318@isical.ernet.in, nikhil@isical.ernet.in.sankar@-isical.ernit.in.

portant or redundant for *discriminating* the classes. As mentioned earlier, one can achieve this through ranking features according to their importance in discrimination. *We emphasize here the fact that the discriminating ability of a feature is dependent on the type of classifier we use.* For example, the most important feature for training an MLP may be different from the most important feature for a nearest prototype classifier. This issue will be illustrated later. There exist a number of methods for feature ranking, each having its own merits and demerits. We describe here a few of them, which have been adopted in our investigation for comparison of results.

The ability to classify patterns by machines relies on an implicit assumption that classes occupy distinct regions in the feature space. Intuitively, the more the distance between the classes, the better the chance of successful recognition. One approach could, therefore, be to select those features for which the classes are maximally separated.

Let $\mathbf{X} = \{\mathbf{x}_i | \mathbf{x}_i \in \Re^p, i = 1, 2, \ldots, n\}$ be a data set. There are c classes $\mathscr{C}_1, \mathscr{C}_2, \mathscr{C}_3, \ldots, \mathscr{C}_c$ with *a priori* class probability P_i for class $\mathscr{C}_i$, such that $\cup_{i=1}^{c} \mathscr{C}_i = \mathbf{X}$, $\mathscr{C}_i \cap \mathscr{C}_j = \phi \forall i \neq j$ and $|\mathscr{C}_i| = n_i$. Let $\mathbf{Y} = \{\mathbf{y}_i | \mathbf{y}_i \in \Re^{p'}, p' \leq p, i = 1, 2, \ldots, n\}$ be a data set generated from $\mathbf{X}$ by some feature selection technique, where y_{ik}, the kth component of $\mathbf{y}_i$, is equal to x_{il}, some lth component of $\mathbf{x}_i$. In other words, $\mathbf{Y}$ is generated by deleting some $(p - p')$ rows of $\mathbf{X}$, if $\mathbf{X}$ and $\mathbf{Y}$ are represented as matrices of orders $p \times n$ and $p' \times n$, respectively. Now a criterion function for ranking the features is defined as:(2)

$$J(\mathbf{Y}) = \frac{1}{2} \sum_{i=1}^{c} P_i \sum_{j=1}^{c} P_j \frac{1}{n_i n_j} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} d(\mathbf{y}_{ik}, \mathbf{y}_{jl}), \tag{1}$$

where the distance metric $d(\mathbf{y}_{ik}, \mathbf{y}_{jl})$ may be defined in various ways; we use here the Euclidian distance. $J(\mathbf{Y})$ can be calculated for every possible subset of the features. The feature set is so selected that $J(\mathbf{Y})$ is maximum. In calculating the criterion function, an estimate of P_i may be taken as:

$$P_i = \frac{n_i}{n}. \tag{2}$$

Therefore,

$$J(\mathbf{Y}) = \frac{1}{2n^2} \sum_{i=1}^{c} \sum_{j=1}^{c} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} d(\mathbf{y}_{ik}, \mathbf{y}_{jl}) = \frac{1}{2n^2} \sum_{k=1}^{n} \sum_{l=1}^{n} d(\mathbf{y}_k, \mathbf{y}_l). \tag{3}$$

Let $\mathbf{m}_i$ be the sample mean vector of the ith class, i.e.

$$\mathbf{m}_i = \frac{\sum_{k=1}^{n_i} \mathbf{y}_{ik}}{n_i} \tag{4}$$

and the mixture sample mean be designated by $\mathbf{m}$, i.e.

$$\mathbf{m} = \sum_{i=1}^{c} P_i \mathbf{m}_i \tag{5}$$

After some algebraic manipulation, equation (1) becomes:

$$J(\mathbf{Y}) = tr(\mathbf{S}_w) + tr(\mathbf{S}_b) \tag{6}$$

where

$$\mathbf{S}_w = \sum_{i=1}^{c} P_i \frac{1}{n_i} \sum_{k=1}^{n_i} (\mathbf{y}_{ik} - \mathbf{m}_i)(\mathbf{y}_{ik} - \mathbf{m}_i)^t \tag{7}$$

and

$$\mathbf{S}_b = \sum_{i=1}^{c} P_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^t. \tag{8}$$

Intuitively, for the feature selection task we like to maximize $tr(\mathbf{S}_b)$ and at the same time minimize $tr(\mathbf{S}_w)$. Hence, $J(\mathbf{Y})$, as given by equation (6) may not be a good criterion function for feature selection.(2) A better approach would be to maximize

$$J(\mathbf{Y}) = \frac{tr(\mathbf{S}_b)}{tr(\mathbf{S}_w)}. \tag{9}$$

The main drawback of the above criterion function (equation (9)) is, if for a particular feature subset a class $\mathscr{C}_i$ is well scattered and a portion of $\mathscr{C}_i$ is overlapped with another class $\mathscr{C}_j$ such that their centroids are far away, then $J(\mathbf{Y})$ may be greater than that for another feature subset, which separates the two classes (Fig. 2) in such a fashion that a single hyperplane may pass between them, but their centroids are not so apart. Intuitively, the second feature subset is better than the first one although the criterion function may indicate the reverse. This will be further elaborated in Section 4 using an example.

2.2. *Feature ranking based on fuzzy entropy*

Let $X = \{x_1, x_2, \ldots, x_m\}$ be a universe of discourse and a fuzzy set $\mathscr{A} = \{\mu_{\mathscr{A}}(x_i)/x_i | x_i \in X; i = 1, 2, \ldots, m; \mu_{\mathscr{A}} \in [0, 1]\}$ be defined on X where $\mu_{\mathscr{A}}(x_i)$ denotes the membership of x_i to $\mathscr{A}$. A measure of fuzziness for $\mathscr{A}$ can be defined as:(9)

$$H(\mathscr{A}) = \alpha \sum_{i=1}^{n} f(\mu_{\mathscr{A}}(x_i)), \tag{10}$$

where α is a constant and the function $f(\cdot)$ can be defined in various ways.(9) $H(\mathscr{A})$ is also called entropy of the fuzzy set. One can obtain entropy of Deluca–Termini using

$$f(\mu_{\mathscr{A}}(x_i)) = -\mu_{\mathscr{A}}(x_i) \ln(\mu_{\mathscr{A}}(x_i)) - (1 - \mu_{\mathscr{A}}(x_i)) \ln(1 - \mu_{\mathscr{A}}(x_i)) \tag{11}$$

in equation (10). Thus, the entropy becomes:

$$H(\mathscr{A}) = \frac{1}{n \ln 2} \sum_{i=1}^{n} \{-\mu_{\mathscr{A}}(x_i) \ln(\mu_{\mathscr{A}}(x_i)) - (1 - \mu_{\mathscr{A}}(x_i)) \ln(1 - \mu_{\mathscr{A}}(x_i))\} \tag{12}$$

where $\alpha = 1/(n \ln 2)$ is the normalization factor. Pal and Chakraborty used equation (12) for feature ranking.(3) $H(\mathscr{A})$ attains the maximum value when $\mathscr{A}$ is most fuzzy, i.e. when $\mu_{\mathscr{A}}(x_i) = 0.5 \ \forall\, i$ and it attains the minimum value when $\mu_{\mathscr{A}}(x_i) = 0$ or $1 \ \forall\, i$. Pal and Chakraborty[3] used S-type and π-type[10] membership functions for modeling μ. Here we have considered only the standard S-function. The S-function is defined as:

$$\mu_{\mathscr{A}}(x_i,;\ a,b,c) = \begin{cases} 0, & x_i \le a \\ 2[(x_i - a)/(c-a)]^2, & a \le x_i \le b \\ 1 - 2[(x_i - c)/(c-a)]^2, & b \le x_i \le c \\ 1, & x_i \ge c \end{cases} \tag{13}$$

in the interval $[a,c]$ with $b=(a+c)/2$. The parameter b is known as the crossover point for which $\mu_{\mathscr{A}}(b;\ a,b,c) = 0.5$.

For computing H of the class $\mathscr{C}_j$ along the qth feature, the parameters of the S-function can be computed as:[3]

$$b = (x_{qj})_{\mathrm{av}}, \tag{14}$$

$$c = b + \max\{|(x_{qj})_{\mathrm{av}} - (x_{qj})_{\max}|, |(x_{qj})_{\mathrm{av}} - (x_{qj})_{\min}|\}, \tag{15}$$

and

$$a = 2b - c. \tag{16}$$

Here av, max, and min are used to denote the average, maximum, and the minimum value of x_{qj}, respectively. If each x_{qj} is equal to b, H will be maximum and equal to 1. H tends to zero as x_{qj} moves away from b towards either c or a. The higher the value of H, the greater would be the number of samples having $\mu(x)\approx 0.5$ and hence greater would be the tendency of the samples to cluster around its mean value, resulting in less (internal) scatter within the class. If we pool together the classes $\mathscr{C}_j$ and $\mathscr{C}_k$ and compute the mean, maximum, and minimum values of the qth feature over all $(n_j + n_k)$ samples where n_r $(r = j,k)$ is the number of samples of class $\mathscr{C}_r$, H for the pooled sample would decrease as the goodness of feature increases. This is because, for a good feature, the samples from both classes should be away from the overall mean, i.e. most of the points will have $\mu(x)\approx 0$ or 1. The feature evaluation index for feature $q(\mathrm{FEI}_q)$ can thus be defined as:[3]

$$\mathrm{FEI}_q = \frac{H_{qjk}}{H_{qj} + H_{qk}} \tag{17}$$

where H_{qjk} is the value of the entropy for feature q after pooling the classes $\mathscr{C}_j$ and $\mathscr{C}_k$; and H_{qj} and H_{qk} are those for the feature q computed for $\mathscr{C}_j$ or $\mathscr{C}_k$, respectively. The lower the value of FEI_q, the higher is, therefore, the quality of the qth feature in characterizing and discriminating classes $\mathscr{C}_j$ and $\mathscr{C}_k$. Instead of using only one feature q, FEI can be calculated even for a set of features.[4] In this case, we need to use the multidimensional membership function.[11]

Note that, instead of H any measure of fuzziness can be used. Pal and Chakraborty,[3] in addition to the entropy, used the index of fuzziness. We mention here that in a similar manner, the feature evaluation index can be calculated in terms of π-type functions.[3,4]

A drawback of this approach is, it can be used to assess features for a pair of classes only. It may happen that a feature p is good for discriminating $\mathscr{C}_i$ and $\mathscr{C}_j$, while feature q may be a better discriminator for classes $\mathscr{C}_k$ and $\mathscr{C}_l$. Further, it may happen that some other feature r is, on average, a better discriminator for all the classes $\mathscr{C}_i$, $\mathscr{C}_j$, $\mathscr{C}_k$, and $\mathscr{C}_l$ taken together. Thus, using FEI it is difficult to assess the goodness of a feature with respect to all classes taken together.

To get around this problem, Pal[4] extended his earlier work to define the average importance of a set of features $\mathscr{S}$ as:

$$(\mathrm{FEI})^{\mathrm{av}}_{\mathscr{S}} = \sum_j \sum_k W_j W_k\, (\mathrm{FEI})^{(jk)}_{\mathscr{S}}, \tag{18}$$

where $W_j = n_j/n_t$, $W_k = n_k/n_t$ with $n_t = \sum_j n_j, j,k = 1,2,\ldots,c;\ k \ne j$, are weight factors. Here the weights are nothing but the *a priori* probabilities of different classes. Hence, $(\mathrm{FEI})^{\mathrm{av}}$ depends on the number of points in a class and this may not be desirable. Preferably, $(\mathrm{FEI})^{\mathrm{av}}$ should depend only on the structure of the classes but not on the number of points in a class. Moreover, in equation (18), even when $n_j + n_k = \psi$ (a constant) for two different pairs of classes, W_jW_k could be different for the two pairs. W_jW_k attains the maximum value when $n_j = n_k = \psi/2$. Hence, $(\mathrm{FEI})^{\mathrm{av}}$ is biased towards equiprobable classes. This motivated us to define a new index, called *overall feature evaluation index* or OFEI, described in Section 3.1.

Next we describe the MLP-based feature selection method of Ruck *et al.*,[5] but before that we describe the basics of an MLP.

2.3. *Description of neural network*

An MLP[12] is a classifier network, capable of learning an input–output relation. An MLP consists of several layers of processing elements called nodes or neurons. There is no connection between nodes within a layer, but complete connections exist between nodes of successive layers. The layer of nodes which receives inputs from outside is called the input layer and the layer that produces output is called the output layer. In between input and output layers there are several layers called hidden layers. The number of nodes in the input layer is the same as the dimension of input data, and that in the output layer is the same as the number of pattern classes. The nodes in the hidden layers receive inputs from its preceding layer and produce outputs which become inputs to the nodes of the next layer. There is no computation in the input layer. Nodes in other layers receive inputs, which are functions of the outputs of nodes in the previous layer and the connection weights between the two layers, and apply a nonlinear transformation (activation function) to produce the output.

The total input to the ith unit (node) of layer $h+1$ $(h \ge 0)$ is:

$$u_i = \sum_j \omega^{(h)}_{ij} v_j. \tag{19}$$

Here v_j is the output of the jth unit in layer h (h=0 corresponds to the input layer) and $\omega_{ij}^{(h)}$ is the connection weight between the ith node in layer h+1 and the jth node of layer h. The output of a node i in layer $h > 0$ is $v_i = g(u_i)$, g is the activation function. Mostly sigmoidal activation functions are used.

In the learning phase (training) of such a network we present the pattern $\mathbf{x} = \{x_i\}$, where x_i is the ith component of the input vector $\mathbf{x}$, as input and adjust the set of weights in the connecting links such that the desired output $\mathbf{t} = \{t_k\}$ is obtained at the output nodes. The process is repeated until the weights stabilize.

In general, for the output layer the actual outputs $\{v_k\}$ will not be the same as the target or desired values $\{t_k\}$. Thus, for a pattern vector, the error is:

$$e_{\mathbf{x}} = \sum_k (t_k - o_k)^2 \tag{20}$$

and the total error is:

$$E = \sum_{\mathbf{x}} e_{\mathbf{x}}. \tag{21}$$

An MLP attempts to minimize E by moving in the direction of negative gradient of the instantaneous error $e_{\mathbf{x}}$. In other words, the incremental change $\Delta\omega_{kj}^{(h)}$ is taken as proportional to $[-\partial e_{\mathbf{x}}/\partial\omega_{kj}^{(h)}]$ for a particular pattern $\mathbf{x}$, i.e. $\Delta\omega_{kj}^{(h)} = [-\eta(\partial e_{\mathbf{x}})/(\partial\omega_{kj}^{(h)})]$, where η is the learning rate. After some algebraic manipulation the learning rule becomes:

$$\Delta\omega_{kj}^{(h)} = \begin{cases} -\eta(\partial e_{\mathbf{x}}/\partial v_k)g'(u_k)v_j \\ \quad \text{if layer } (h+1) \text{ is output layer,} \\ -\eta\left(\sum_m (\partial e/\partial u_m)\omega_{mk}^{(h)}\right)g'(u_k)v_j \\ \quad \text{for other layers.} \end{cases} \tag{22}$$

The incremental changes $\Delta\omega_{ji}^{(h)}$ may be summed up for all the patterns in the training set and then the weights $\omega_{ji}^{(h)}$ are updated with the resulting increments. This process is called *batch mode* training. In this strategy the learning process remains independent of the sequence in which the training data are fed. Normally, a complete pass through the training data is known as "epoch." On the other hand, in *on-line* training weights are updated with each pattern. In this case, learning depends upon the sequence of data feeding. We have adopted the batch mode learning in our experiment. Thus, the expression for the updated weight after t epochs is given by:

$$\omega_{ji}^{(h)}(t+1) = \omega_{ji}^{(h)}(t) + \sum \Delta\omega_{ji}^{(h)}. \tag{23}$$

2.4. *Neural network based feature selection*

Ruck *et al.*[5] developed an algorithm for feature ranking using an MLP. The sensitivity of output of the network to its input is used to rank the input features. An expression for feature *saliency* (as proposed by them), used for feature ranking, is defined as:

$$\Lambda_j = \sum_{\mathbf{x}\in\mathcal{S}} \sum_k \sum_{x_j \in D_j} \left|\frac{\partial o_k(\mathbf{x}, \mathbf{W})}{\partial x_j}\right|. \tag{24}$$

where D_j is the domain of the jth feature and $\mathcal{S}$ is the training set. The matrix $\mathbf{W}$ is an array of all connection weights in the network arranged in some suitable form. They used the derivative as a sensitivity indicator of the network output with respect to the input feature. Therefore, $\Lambda_j > \Lambda_i$ is assumed to indicate that the importance of the jth feature is higher than that of the ith feature.

For evaluating $(\partial o_k(\mathbf{x},\mathbf{W})/\partial x_j)$ in equation (24) we use the chain rule, and thus for an MLP with one hidden layer we have

$$\begin{aligned} \frac{\partial o_k}{\partial x_j} &= o_k(1-o_k)\frac{\partial}{\partial x_j}\left(\sum_i \omega_{ki}^{(1)} v_i + \theta_k\right) \\ &= o_k(1-o_k)\sum_i \omega_{ki}^{(1)} \frac{\partial v_i}{\partial x_j} \\ &= o_k(1-o_k)\sum_i \omega_{ki}^{(1)} v_i(1-v_i)\omega_{ij}^{(0)}. \end{aligned} \tag{25}$$

Here θ_k is the threshold for the sigmoidal activation function, g. From equation (25) we find that the derivative depends on the current input to the network as well as its weights. To calculate Λ_j, ideally, each input should be independently sampled over its expected range of values. For example, if R points are used for each input feature, the total number of points that the derivatives have to be evaluated at would be R^p, where p is the total number of features. Therefore, the problem is of exponential complexity.

To reduce the computational load, Ruck *et al.* suggested to sample it at the most important points. The points of greatest importance in the input space are those in which training data exist; hence, the training vectors are used as starting points to sample the input space. For every training vector, each feature is sampled over its range. Thus for n training vectors, the number of derivative evaluations is npR.

Limitations of this algorithm are described in Section 3.3.

3. PROPOSED SCHEMES

In this section we propose two schemes for feature analysis. The first feature ranking scheme is based on fuzzy set theoretic concepts. It is an extension of earlier work by Pal and Chakraborty[3] and Pal.[4] The other scheme is based on the Multilayer Perceptron, and it can be used for both feature ranking and selection.

3.1. *Feature evaluation using fuzziness*

Here, we modify the FEI of Pal[4] and define an overall feature evaluation index (OFEI) based on fuzzy set theoretic concepts. This OFEI takes care of the limitations of the earlier approach.[3,4] Feature q will be good if it can discriminate every pair of the c classes. Therefore, the goodness of a feature q increases as H_{qjk} ($j,k = 1,2,\ldots,c$ and $j\neq k$) decreases and H_{qj} ($j=1,2,\ldots,c$) increases; i.e. $\sum_{j,k=1,j\neq k}^{c} H_{qjk}$ decreases and $\sum_{j=1}^{c} H_{qj}$ increases. Thus, the overall feature evaluation index for feature $q(\text{OFEI}_q)$ can be defined as:

$$\text{OFEI}_q = \frac{\sum_{j,k=1\, j\neq k}^{c} H_{qjk}}{\sum_{j=1}^{c} H_{qj}}. \tag{26}$$

We use only the fuzzy entropy of Deluca–Termini. The lower the value of OFEI, the better will be the performance of the feature with respect to discriminating all the classes. In equation (26), it may happen that $H_{qij} < H_{rij}$ but $H_{qkl} > H_{rkl}$ i.e. feature q is more important to discriminate classes i and j than feature r but the converse is true for classes k and l. Since equation (26) considers all possible pairs of classes, OFEI_q will reflect the overall (average) discriminating power of the feature q. Note that OFEI_q does not depend on the size of a class.

3.2. *Feature evaluation using an MLP*

After an MLP successfully learns a data set, the weights of the links will be so adjusted that the value of a redundant (less important) feature will not influence the output vector much. The lower the importance of a feature in discriminating between classes, the lower would be the influence of its value on the output of the network. The proposed scheme banks on this concept.

For every feature q we compute a feature quality index, FQI_q and then rank the features according to FQI_q. To compute FQI_q we proceed as follows: For each training data point $\mathbf{x}_i$, $i = 1, 2, \ldots, n$, we set x_{iq} to zero. Let this modified data point be denoted by $\mathbf{x}_i^{(q)}$; i.e. $x_{ij}^{(q)} = x_{ij}$, $\forall j \neq q$ and $x_{iq}^{(q)} = 0$. Setting the qth component to zero is equivalent to delinking the qth input node and hence delinking all connections associated directly from the qth input node. Thus, the impact of the qth feature will not reach any node of the network. Let the output vectors obtained by $\mathbf{x}_i$ and $\mathbf{x}_i^{(q)}$ be $\mathbf{o}_i$ and $\mathbf{o}_i^{(q)}$, respectively. Note that $\mathbf{o}_i$ is *not* the target output corresponding to $\mathbf{x}_i$, but the *actual* output that is obtained for $\mathbf{x}_i$ from the trained net. For a less important feature, the output vectors $\mathbf{o}_i$ and $\mathbf{o}_i^{(q)}$ are not expected to differ much. Any function of $\mathbf{o}_i$ and $\mathbf{o}_i^{(q)}$ that can measure this variation between the two can be used as an index for feature ranking. A very simple choice would be to define

$$\text{FQI}_q = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{o}_i - \mathbf{o}_i^{(q)}\|^2. \tag{27}$$

After computing FQI_qs for all the p features, they can be ranked according to their importance as $q_1, q_2, \ldots, q_p$ when $\text{FQI}_{q_1} \geq \text{FQI}_{q_2} \geq \cdots \geq \text{FQI}_{q_p}$.

Instead of feature ranking, if the problem is to select $p'(p' < p)$ best features (feature selection), best from the point of view of discrimination between classes, the feature set $\{q_1, q_2, \ldots, q_{p'}\}$ may not be the optimal set. But $q_1, q_2, \ldots, q_{p'}$ will definitely give a very good subset of features. To get the best p' features we proceed as follows: There are $\binom{p}{p'}$ possible subsets of features. Let the lth subset be denoted by $\mathcal{S}_l$. Now we define $\text{FQI}_l^{(p')}$ as:

$$\text{FQI}_l^{(p')} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{o}_i - \mathbf{o}_i^{l}\|^2, \tag{28}$$

where $\mathbf{o}_i^l$ is the output from the net with $\mathbf{x}_i^l$ as input. Note that $\mathbf{x}_i^l$ is derived from $\mathbf{x}_i$ as follows:

$$x_{ij}^{l} = \begin{cases} 0, & j \in \mathcal{S}_\ell \\ x_{ij}, & \text{otherwise.} \end{cases} \tag{29}$$

We choose $\mathcal{S}_j$ as the optimal set of features, when $\text{FQI}_j^{(p')} \geq \text{FQI}_l^{(p')}$, $\forall l;\ l \neq j$.

Instead of equations (27) and (28), one can also compute *feature devaluation indices* (FDIs) as:

$$\text{FDI}_q = \frac{1}{n}\sum_{i=1}^{n} \frac{\mathbf{o}_i'\mathbf{o}_i^{(q)}}{\|\mathbf{o}_i\|\|\mathbf{o}_i^{(q)}\|} \tag{30}$$

and

$$\text{FDI}_l^{(p')} = \frac{1}{n}\sum_{i=1}^{n} \frac{\mathbf{o}_i'\mathbf{o}_i^{l}}{\|\mathbf{o}_i\|\|\mathbf{o}_i^{l}\|}, \tag{31}$$

where $\mathbf{o}_i'$ is the transpose of $\mathbf{o}_i$. Here, lower the value of FDI, the more is the importance of the feature or the feature subset.

3.3. *The proposed scheme vs. scheme of Ruck* et al.

We now compare the proposed neural network-based algorithm with that of Ruck *et al.* Both their method and our scheme are in a sense based on the same concept—sensitivity of network output with respect to its input. In our approach we find the output of the net after removing a feature and then measure the deviation of this output from the learnt output, but not from the target output. We have not considered the target output, because the network might not have been able to learn the target output to a desirable level. It is more logical to consider the sensitivity with respect to what has been learnt by the network. Moreover, in our approach, setting a feature value to zero is equivalent to assuming the absence of that feature. Thus, it is a conservative approach. On the other hand, Ruck *et al.* calculated the rate of change of network output with respect to the input.

Let us assume that a system has p features for characterizing two classes. Among these features, we consider two features F_1 and F_2. Also assume that $(\partial o_k/\partial F_1) > (\partial o_k/\partial F_2)$, but the variance of F_1 is less than that of F_2. Such a situation is always possible as the function approximated by the network may not be adequately represented by the data. In this case the algorithm of Ruck *et al.* will usually show that F_1 is more important than F_2. Since the values of F_2 are more disperse compared to F_1, two patterns from different classes may have well apart F_2 values but close F_1 values. If we sample the domains of F_1 and F_2 uniformly, i.e. into the same number of intervals, then ΔF_2 will be greater than ΔF_1. ΔF_i is the separation between successive F_i values for points considered to compute the *saliency*. It may then happen that the product $(\partial o_k/\partial F_2)\times \Delta F_2 > (\partial o_k/\partial F_1) \times \Delta F_1$, i.e. F_2 is effectively more sensitive than F_1. The algorithm of Ruck *et al.* may fail here.

The method of sampling data points in equation (5) has another drawback. Let us take a pattern set in two

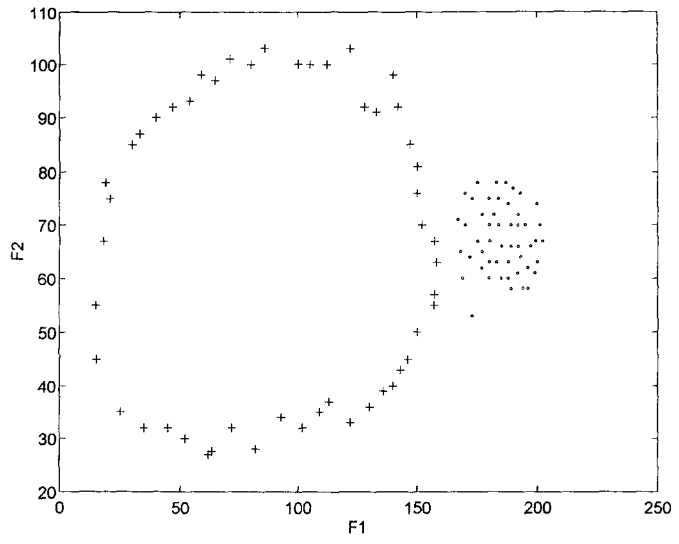

Fig. 1. Scatterplot of a two-dimensional synthetic data set. "+" indicates a pattern belonging to class 1 and "." indicates a pattern belonging to class 2.

Table 1. Values of FQI and *saliency* for the features of pattern set

Feature used	FQI	Rank	*Saliency*	Rank
1	0.678548	1	855.036690	2
2	0.676471	2	2329.231636	1

dimension as in Fig. 1. The pattern set has two classes *viz.* class 1 and class 2. Consider a pattern vector **x** in the training set from class 1. If the value of feature 1 (F_1) is kept fixed and that of F_2 is varied over its range, some of the points may be generated outside of both classes 1 and 2. The network is neither trained with these pattern points nor do these points belong to any of the two classes. Therefore, incorporation of these points in calculating the feature saliency may mislead the process of ranking.

We illustrate the above two observations with an example. Table 1 depicts the ranking of the two features of the pattern set given in Fig. 1. It is found that according to the index FQI (equation (27)), the feature F_1 is more important than the feature F_2 which is also desirable as the feature F_1 alone can separate the two pattern classes, whereas F_2 cannot do the same. On the other hand, the *saliency* measure (in equation (24)) of Ruck *et al.*[5] strongly recommends that feature F_2 is more important than F_1.

Finally, the algorithm proposed by Ruck *et al.* ranks individual features but cannot select the best subset of $p' < p$ features. However, our algorithm ranks the features individually as being able to select the best subset of $p' < p$ features.

4. EXPERIMENTAL RESULTS

In the present investigation we have used three data sets: **Iris**[13] with four features and three classes, **Crude-oil**[14] with five features and three classes, and **Mango-leaf**[15] with 18 features and three classes.

Anderson's **Iris**[13] data set contains three classes, i.e. three varieties of Iris flowers, namely, Iris Setosa, Iris Versicolor, and Iris Virginica consisting of 50 samples each. Each sample has four features, namely, Sepal length, Sepal width, Petal length, and Petal width. **Iris** data has been used in many research investigations related to pattern recognition and has become a sort of benchmark data.

Crude-oil[14] is a five-dimensional data set consisting of 56 patterns divided into three classes. Three classes correspond to three types of oil. Three classes 1, 2, and 3 consist of 7, 11, and 38 patterns, respectively.

Mango-leaf,[15] on the other hand, is a $p = 18$ dimensional data set with 166 patterns. It has three classes representing three kinds of mango. The number of samples in classes 1, 2, and 3 are 100, 35, and 31, respectively. The feature set consists of measurements like area (A), perimeter (Pe), maximum length (L), maximum breadth (B), petiole (P), length+petiole (L+P), length/petiole (L/P), length/maximum breadth (L/B), (L+P)/B, A/L, A/B, A/Pe, upper midrib/lower midrib, upper Pe/lower Pe and so on. The terms upper and lower are used with respect to maximum breadth position.

Since each feature has a different range of values, i.e. some have quite large values while others have very low even fractional values, all features are normalized to the same scale so that the differences in their ranges are reduced. In other words, we applied the following transformation on each feature x',

$$x = \frac{x' - k_1}{k_2 - k_1}, \tag{32}$$

where $k_1 = \min_i \min_j \{x'_{ij}\}$ and $k_2 = \max_i \max_j \{x'_{ij}\}$. Note that this transformation does not change the structure of the classes as it is only a change of scale and origin of the entire data.

Table 2. Values of different indices obtained by MLP for **Iris**

Feature made zero	Misclassification	FQI	Rank	FDI	Rank	*Saliency* measure	Rank
1	50	0.596329	3	0.683017	4	1663.891834	4
2	98	0.804521	2	0.531856	2	2498.589697	3
3	100	0.926212	1	0.335803	1	4041.534737	1
4	50	0.495294	4	0.675122	3	3580.965868	2

Table 3. Values of $J(\mathbf{Y})$ and different entropy-based measures using one of the features of **Iris**

Feature used	$J(\mathbf{Y})$	Rank	$(\mathrm{FEI})^{\mathrm{av}}$	Rank	OFEI	Rank
1	1.622646	3	0.166672	3	0.998300	3
2	0.668844	4	0.169757	4	1.018069	4
3	16.056615	1	0.109842	2	0.656602	2
4	13.061322	2	0.106668	1	0.634167	1

For **Iris**, the ranking of features obtained by the MLP-based scheme is shown in Table 2. In this investigation we considered different network architectures and different initializations. Table 2 presents a typical result. We have obtained mostly the same relative ranking of the features (as shown in the Table 2), although the absolute values of the indices were different in different runs. The ith entry in the *misclassification* column indicates the number of data points that are wrongly classified after the ith feature is made zero. The other columns are self-explanatory. The ranks obtained by FQI and FDI are different, although the first and second most important, features are the same. Here feature 3 is found to be the most important, while feature 2 is the next most important one. On the other hand, the entropy-based method, *saliency*, and $J(\mathbf{Y})$ indicate features 3 and 4 as most important (Tables 2 and 3). Several authors[16,17] also believe that features 3 and 4 are more important for **Iris**. Why does the proposed MLP-based method show a different result? To get an answer to this, let us consider a four-dimensional synthetic data set. The data set has 20 data points, 10 points for each of two classes. Scatterplots of the first two components are shown in Fig. 2, while Fig. 3 shows the scatterplot of the third and fourth components. In the scatterplots class 1 is indicated by "1" and class 2 is represented by "2". Clearly, scatterplot of features one and two (Fig. 2) can be easily separated by a straight line to discriminate the classes, but their centroids are very close. While, for Fig. 3 although the centroids are widely separated, it requires a combination of lines to separate the two classes. Thus,

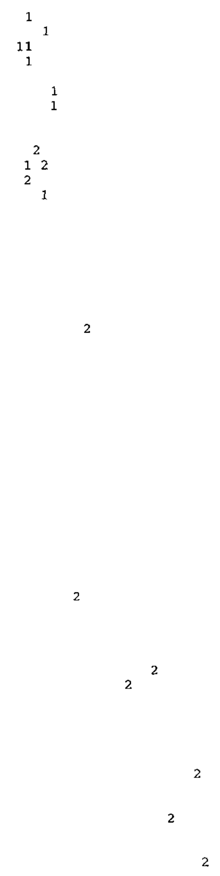

Fig. 3. Scatterplot, using features 3 and 4, of the four-dimensional synthetic data. "1" and "2" indicate patterns belonging to classes 1 and 2, respectively.

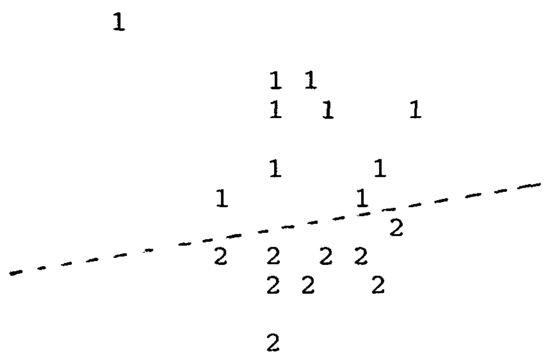

Fig. 2. Scatterplot, using features 1 and 2, of the four-dimensional synthetic data. "1" and "2" indicate patterns belonging to classes 1 and 2, respectively.

Table 4. Values of FQI and $J(\mathbf{Y})$ associated with pairs of features for the synthetic data. Features mentioned in column 1 are made zero for FQI and while they are used for $J(\mathbf{Y})$

Features made zero/used	FQI	Misclassification	Relative rank	$J(\mathbf{Y})$	Relative rank
1,2	0.268825	4	1	0.354866	2
3,4	0.191828	3	2	1.119447	1

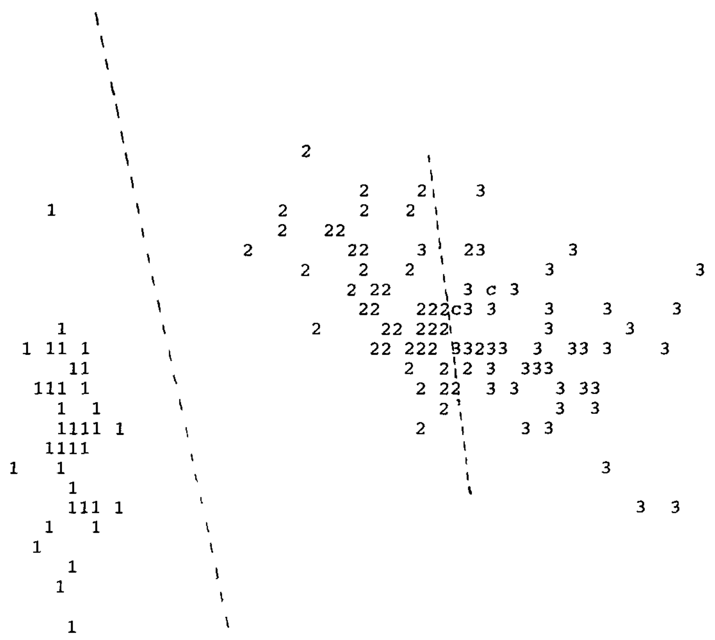

Fig. 4. Scatterplot, using features 2 and 3, of **Iris** data. "1," "2," and "3" indicate patterns belonging to classes 1, 2, and 3, respectively.

for an MLP based method features 1 and 2 may turn out to be more important (Table 4) (of course, depending on the initial condition, it may not necessarily be true) than features 3 and 4. However, for the method based on $J(\mathbf{Y})$ features 3 and 4 will be important. Table 4 shows that it is indeed the case.

Let us now look at the scatterplot of features 2, 3 (Fig. 4) and scatterplot of features 3, 4 (Fig. 5) of **Iris**. In Fig. 4 and Fig. 5 different classes are separated by dotted lines. In both the cases classes can be separated by two lines with only 2–3 errors, but the centroids of the classes as represented by features 3 and 4 are much more separated than those represented by features 2 and 3. This explains why the proposed MLP based method shows features 2 and 3 as more important while others indicate features 3 and 4.

For feature ranking we considered the effect of only one feature at a time on the performance of the network, whereas for the feature selection problem (when we want to select the most important, say, p' features) we need to consider the combined effect of feature subsets. Thus, the set of features with rank $\leq p'$ may not necessarily be the optimal set of p' features, in fact, in most of the cases they will be. Table 5 depicts the results obtained by setting two of the features to zero, i.e. the combined effect of two features on the performance of the network. Table 5 reveals that for **Iris**, feature pair (2,3) (based on FQI) and feature pair (1,3) (based on FDI) are found to be important. Based on individual rank also these two features (3,2) are found to be most important.

Table 6 depicts the ranking obtained by the FQI/FDI based method for **Crude-oil**. In this case the ranks

Table 5. Values of different indices obtained by MLP when a pair of features is set to zero for **Iris**

Feature made zero	Misclassification	FQI	Rank	FDI	Rank
1,2	100	0.903465	3	0.351944	3
1,3	100	0.756021	4	0.335055	1
1,4	51	0.564501	6	0.684518	5
2,3	100	1.035633	1	0.452740	4
2,4	61	0.593168	5	0.806045	6
3,4	100	0.923468	2	0.343355	2

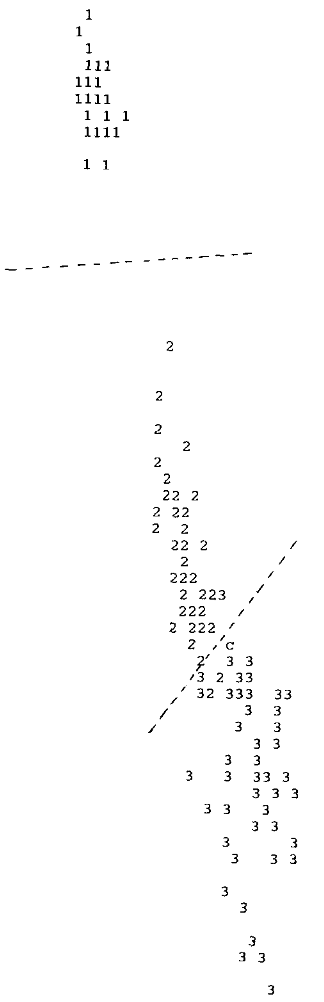

Fig. 5. Scatterplot, using features 3 and 4, of **Iris** data. "1," "2," and "3" indicate patterns belonging to classes 1, 2, and 3, respectively.

obtained by FQI and FDI are identical. The most important feature is found to be the same for all three indices FQI, FDI, and *saliency*. It is interesting to note that the number of misclassifications is consistent with the ranks. These rankings are also almost in accordance with the result (not included in this article) of the experiments during which the network was trained with a pair of features. That is, the network was able to recognize better with features 1 and 4 than that with any other pair of features. Table 7 represents the ranks obtained by $J(\mathbf{Y})$, OFEI, and $(\text{FEI})^{\text{av}}$ for the same data set, **Crude-oil**. We find that the ranks obtained by $J(\mathbf{Y})$ and OFEI/$(\text{FEI})^{\text{av}}$ are different and they are also different from the ranks calculated based on MLP. This difference may be attributed to the following facts: the MLP-based approach is influenced, to a great extent, by how easily classes can be separated using hyper-planes. On the other hand, $J(\mathbf{Y})$ strongly depends on the separation of the centroids, while OFEI depends on both the separation of centroids and overlap of classes. Table 7 also shows that the ranks obtained using $(\text{FEI})^{\text{av}}$ and OFEI are identical. We report the combined effect of feature pairs for this data set in Table 8. It shows that feature pair (1,5) is the most important. Like **Iris**, for this data set also the results were found to be consistent with scatterplots (not included here) of feature pairs.

To establish the effectiveness of the proposed scheme we also considered a data set (**Mango-leaf**) with 18 features and three classes. Ranks obtained by FQI, FDI, and *saliency* are shown in Table 9; the ranks by the three indices are not exactly identical but almost the same. The difference in the ranking by *saliency* with that by FQI/FDI may be due to the complex class structure of the data set. To assess the validity of the ranks (based on FQI), we trained the network with: (a) top three features, (b) top five features, (c) top six features, (d) the features with ranks 4, 5, 6, 7, 8, and 9, and (e) all 18 features. In

Table 6. Values of different indices obtained by MLP for **Crude-oil**

Feature made zero	Misclassification	FQI	Rank	FDI	Rank	*Saliency* measure	Rank
1	40	0.953513	1	0.299117	1	3167.896934	1
2	11	0.238619	4	0.861617	4	314.656966	5
3	06	0.079707	5	0.946434	5	1557.797274	3
4	13	0.302492	2	0.826327	2	1863.405141	2
5	10	0.242237	3	0.846303	3	1099.465980	4

Table 7. Values of $J(\mathbf{Y})$ and different entropy-based measures using one of the features of **Crude-oil**

Feature used	$J(\mathbf{Y})$	Rank	$(\text{FEI})^{\text{av}}$	Rank	OFEI	Rank
1	0.723300	3	0.132087	5	1.194165	5
2	0.754931	2	0.129089	3	1.121799	3
3	0.221933	5	0.141318	4	1.160894	4
4	0.855593	1	0.125434	2	1.023158	2
5	0.619172	4	0.121439	1	0.864039	1

Table 8. Values of different indices obtained by MLP when a pair of features is set to zero for **Crude-oil**

Features made zero	Misclassification	FQI	Rank	FDI	Rank
1,2	38	0.856033	3	0.377745	3
1,3	40	0.981285	2	0.296769	2
1,4	29	0.703605	4	0.533612	4
1,5	45	1.182584	1	0.156550	1
2,3	12	0.243394	10	0.850202	10
2,4	17	0.385110	6	0.735543	6
2,5	17	0.370540	7	0.742812	7
3,4	13	0.293036	8	0.827088	9
3,5	12	0.276577	9	0.812103	8
4,5	18	0.407598	5	0.712725	5

Table 9. Values of different indices obtained by MLP for **Mango-leaf**

Feature made zero	Misclassification	FQI	Rank	FDI	Rank	*Saliency* measure	Rank
1	25	0.125161	9	0.970644	9	10259.124586	14
2	66	0.526458	5	0.737481	4	13607.626761	12
3	44	0.353275	6	0.883209	6	2685.724258	16
4	130	0.981794	1	0.406196	1	43191.505706	4
5	23	0.089942	13	0.990154	12	19161.967354	10
6	46	0.335267	7	0.889027	7	82910.026417	1
7	18	0.000717	17	0.999999	17	830.137520	17
8	18	0.000160	18	1.000000	18	531.974334	18
9	66	0.888633	2	0.723810	3	61851.395600	2
10	59	0.553753	4	0.770063	5	23367.307882	8
11	30	0.181140	8	0.956033	8	28306.887051	7
12	24	0.108732	10	0.988997	11	21500.352227	9
13	23	0.092504	11	0.991579	13	10867.592324	13
14	20	0.047141	15	0.997323	15	14402.327757	11
15	82	0.683353	3	0.678076	2	42520.772601	5
16	19	0.012551	16	0.999815	16	7677.518133	15
17	23	0.090810	12	0.986581	10	57658.592076	3
18	18	0.058416	14	0.995781	14	35721.791323	6

Table 10. Misclassifications obtained by a trained MLP (for 60,000 epochs) with different feature subsets of **Mango-leaf**

Cases	Features used	Misclassification
(a)	4,9,15	49
(b)	4,9,15,10, 2	25
(c)	4,9,15,10,2,3	40
(d)	10,2,3,6,11,1	41
(e)	All	31

each case the network was trained for 60,000 epochs. The results (number of misclassifications) are presented in Table 10. Table 10 clearly reveals the effectiveness of the ranks obtained by the MLP-based scheme. Here, case (a) just signifies the inadequacy of the features. The top six features (case (c)) obtained by FQI clearly show better discriminating ability than the six other features with rank 4, 5, 6, 7, 8, and 9 (case (d)), respectively. Table 10 also shows that the top five features together (case (b)) are more effective than taking the top six or all 18 features together. This may be due to the redundancy of the system while taking the top six or all the features together. The ranking obtained by $J(\mathbf{Y})$, OFEI, and $(\text{FEI})^{\text{av}}$ for the data set are found to be quite different from that based on FQI and FDI (Tables 9 and 11). Comparing the features with ranks ≤ 5, we find that the ranks obtained by the entropy-based methods are closer to those based on FQI.

In order to establish the superiority of the proposed MLP-based scheme over the others discussed earlier, we have trained the same network (the same architecture and initialization) separately with a few good features obtained by (i) FQI, (ii) fuzzy entropies, (iii) J-function, and (iv) *saliency* for **Mango-leaf** and **Crude-oil**. Table 12 reports the results for **Mango-leaf** with the top five features when the networks are trained for 60,000 epochs. The top five features selected by FQI are found to produce the least number of *misclassifications* (Table 12). For **Crude-oil**, we have trained the networks for 30,000 epochs using the top three features. Here also the proposed scheme (FQI) outperforms the others (Table 13).

Table 11. Values of $J(\mathrm{Y})$ and different entropy-based measures using one of the features of **Mango-leaf**

Feature used	J(Y)	Rank	$(\mathrm{FEI})^{\mathrm{av}}$	Rank	OFEI	Rank
1	0.116022	11	0.146951	7	1.053797	7
2	0.424932	5	0.152861	10	1.082116	9
3	0.097274	13	0.155923	16	1.133981	16
4	0.351634	6	0.144551	3	1.018112	1
5	0.458305	3	0.142962	1	1.026228	2
6	0.023515	16	0.154936	13	1.092866	13
7	0.009207	18	0.146140	6	1.036342	5
8	0.014633	17	0.146018	5	1.035661	4
9	0.793117	1	0.151614	9	1.092510	12
10	0.242699	8	0.155043	12	1.088604	11
11	0.042566	14	0.166411	18	1.203172	17
12	0.188824	9	0.155414	14	1.130467	15
13	0.034397	15	0.160234	17	1.169015	18
14	0.482680	2	0.146503	4	1.051984	6
15	0.342572	7	0.143206	2	1.033192	3
16	0.433072	4	0.152190	11	1.082951	10
17	0.107068	12	0.154561	15	1.123898	14
18	0.173945	10	0.148626	8	1.059308	8

Table 12. Misclassifications obtained by MLP (after training for 60,000 epochs) with the top five features of **Mango-leaf** ranked by different ranking schemes described

Ranking based on	Features used	Misclassification
FQI	4,9,15,10,2	25
Entropy	4,5,15,8,7	32
J-function	9,14,5,16,2	33
Saliency	6,9,17,4,15	43

Table 13. Misclassifications obtained by MLP (after training for 30,000 epochs) with the top three features of **Crude-oil** ranked by different ranking schemes described

Ranking based on	Features used	Misclassification
FQI	1,4,5	6
Entropy	5,4,2	18
J-function	4,2,1	18
Saliency	1,4,3	10

CONCLUSION AND DISCUSSION

We have proposed a scheme for both *feature ranking* and *feature selection* based on a multilayer perceptron network. The scheme is based on the idea that the effect of a missing feature (setting the feature value to zero) on the output of a trained network will depend heavily on the importance of the feature. In fact, the more important a feature is, the more will be its impact on the output of the network. We have also provided a scheme for selecting a feature subset based on the same idea. A feature subset may be regarded as good if the network outputs are heavily affected by assuming the absence of these features (in the subset), i.e. setting the values of the features to zero. In addition to this, we have modified an existing fuzzy entropy-based method. Both schemes are tested on three different data sets. The results have been compared with three existing approaches. The superiority of the proposed MLP-based scheme has been established empirically with several data sets. The novelty of the proposed MLP-based scheme and its difference to the method of Ruck *et al.*, which is also based on a similar concept, have been analyzed.

Acknowledgements—The authors gratefully acknowledge the anonymous referee for his/her valuable suggestions and for pointing out the work of Ruck *et al.* which was not known to the authors. This work was carried out while Prof. Sankar K. Pal held the Jawarharlal Nehru Fellowship. Rajat K. De is grateful to the Department of Atomic Energy, Government of India, for providing him with a Dr. K. S. Krishnan Senior Research Fellowship.

REFERENCES

1. J. T. Tou and R. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, Reading, Massachusetts (1974).
2. P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, Prentice-Hall, London (1982).
3. S.K. Pal and B. Chakraborty, Fuzzy set theoretic measures for automatic feature evaluation, *IEEE Trans. Systems Man Cybernet.* **16**, 754–760 (1986).
4. S.K. Pal, Fuzzy set theoretic measures for automatic feature evaluation: II, *Information Sci.* **64**, 165–179 (1992).
5. D. W. Ruck, S. K. Rogers and M. Kabrisky, Feature selection using a multilayer perceptron, *J. Neural Network Computing*, 40–48 (Fall, 1990).
6. K.L. Priddy, S.K. Rogers, D.W. Ruck, G.L. Tarr and M. Kabrisky, Bayesian selection of important features for feedforward neural networks, *Neurocomputing* **5**, 91–103 (1993).
7. A. Kowalczyk and H.L. Ferra, Developing higher-order neural networks with empirically selected units, *IEEE Trans. Neural Networks* **5**, 698–711 (1994).

8. L.M. Belue and J.K.W. Bauer, Determining input features for multilayer perceptrons, *Neurocomputing* **7**, 111–121 (1995).
9. N.R. Pal and J.C. Bezdek, Measuring fuzzy uncertainty, *IEEE Trans. Fuzzy Systems* **2**, 107–118 (1994).
10. S. K. Pal and D. Dutta Majumder, *Fuzzy Mathematical Approach to Pattern Recognition*, John Wiley, New York (Halsted Press) (1986).
11. S.K. Pal and P.K. Pramanik, Fuzzy measures in determining seed points in clustering, *Pattern Recognition Lett.* **4**, 159–164 (1986).
12. Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, New York (1989).
13. R.A. Fisher, The use of multiple measurements in taxonomic problems, *Annals of Eugenics* **7**, 179–188 (1936).
14. R. G. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, Prentice-Hall, New Jersey (1982).
15. A. Bhattacharjee, Some Aspects of Mango (*Mangifora indica L*) Leaf Growth Features in Varietal Recognition. Master's thesis, University of Calcutta, Calcutta, India (1986).
16. J.M. Keller and D.J. Hunt, Incorporating fuzzy membership functions into the perceptron algorithm, *IEEE Trans. Pattern Analysis and Machine Intell.* **7**, 693–699 (1985).
17. B. Chakraborty, On some fuzzy set theoretic measures and knowledge based approach for feature selection in a Pattern Recognition system. Ph.D. thesis, Calcutta University, Calcutta, India (1994).

About the Author—RAJAT K. DE obtained his B. Tech. degree in Computer Science and Engineering from the University of Calcutta, India in 1991, and Master of Computer Science and Engineering from the Jadavpur University, India, in 1993. Since October 1993 he has been working as a research scholar in the Indian Statistical Institute, Calcutta, India, towards his Ph.D. thesis, with a Dr K.S. Krishnan Senior Research Fellowship, sponsored by the Department of Atomic Energy, Government of India. His research interest includes pattern recognition, fuzzy sets, and neural networks.

About the Author—NIKHIL R. PAL obtained his M.B.M. degree from the University of Calcutta, India in 1982 and M.Tech. and Ph.D. degrees in Computer Science from the Indian Statistical Institute, Calcutta, India in 1984 and 1991, respectively. Currently he is an Associate Professor in the Machine Intelligence unit of the Indian Statistical Institute. He was with the Hindusthan Motors Ltd., W.B., India, from 1984 to 1985 and with the Dunlop India Ltd., W.B., India, from 1985 to 1987. In 1987 he joined the Computer Science unit of the Indian Statistical Institute, Calcutta. During August 1991–February 1993 and July 1994–December 1994 he visited the University of West Florida, Pensacola. He was a guest lecturer at the University of Calcutta. His research interests include image processing, fuzzy sets and systems, quantification of uncertainties, neural networks and genetic algorithms. He is an Associate Editor of the *International Journal of Approximate Reasoning* and *IEEE Transactions on Fuzzy Systems.*

About the Author—SANKAR K. PAL is a Professor and Founding Head of Machine Intelligence unit at the Indian Statistical Institute, Calcutta, India. He obtained M.Tech. and Ph.D. degrees in Radiophysics and Electronics in 1974 and 1979, respectively, from the University of Calcutta, India. In 1982 he received another Ph.D. in Electrical Engineering along with a DIC from Imperial College, University of London. In 1986 he was awarded a Fulbright Post-doctoral Visiting Fellowship to work at the University of California, Berkeley and the University of Maryland, College Park, U.S.A. In 1989 he received an NRC-NASA Senior Research Award to work at the NASA Johnson Space Center, Houston, Texas, U.S A. He received the 1990 Shanti Swarup Bhatnagar Prize in Engineering Sciences, 1993 Jawaharlal Nehru Fellowship, 1993 Vikram Sarabhai Research Award, 1993 NASA Tech Brief Award, 1994 *IEEE Transactions on Neural Networks* outstanding paper Award and 1995 NASA Patent Application Award. His research interests mainly include pattern recognition, image processing, neural nets, genetic algorithms, and fuzzy sets and systems. He is a co-author of the book *Fuzzy Mathematical Approach to Pattern Recognition*, John Wiley and Sons (Halsted), N.Y., 1986 and a co-editor of two books *Fuzzy Models for Pattern Recognition*, IEEE Press, N.Y., 1992 and *Genetic Algorithms for Pattern Recognition*, CRC Press, Boca Raton, Florida, 1996. Professor Pal is a Fellow of the IEEE, USA, Indian National Science Academy, Indian Academy of Sciences, National Academy of Sciences, India, Indian National Academy of Engineering, Institute of Engineers, India, and the IETE. He is also a member of the Executive Advisory Editorial Board, *IEEE Trans. Fuzzy Systems* and *International Journal of Approximate Reasoning*, North Holland, and an Associate Editor of *IEEE Trans. Neural Networks*, *Pattern Recognition Letters*, *Neurocomputing*, *Applied Intelligence*, *Information Sciences: Applications*, and *Far-East Journal of Mathematical Sciences*.