Original paper Soft Computing 5 (2001) 152–159 © Springer-Verlag 2001

A neural network and fuzzy rule base hybrid for pattern classification

V. Ravi, H.-J. Zimmermann

Abstract This paper presents a novel hybrid of the two complimentary technologies of soft computing viz. neural networks and fuzzy logic to design a fuzzy rule based pattern classifier for problems with higher dimensional feature spaces. The neural network component of the hybrid, which acts as a pre-processor, is designed to take care of the all-important issue of feature selection. To circumvent the disadvantages of the popular back propagation algorithm to train the neural network, a meta-heuristic viz. threshold accepting (TA) has been used instead. Then, a fuzzy rule based classifier takes over the classification task with a reduced feature set. A combinatorial optimisation problem is formulated to minimise the number of rules in the classifier while guaranteeing high classification power. A modified threshold accepting algorithm proposed elsewhere by the authors (Ravi V, Zimmermann H.-J. (2000) Eur J Oper Res 123: 16–28) has been employed to solve this optimization problem. The proposed methodology has been demonstrated for (1) the wine classification problem having 13 features and (2) the Wisconsin breast cancer determination problem having 9 features. On the basis of these examples the results seem to be very interesting, as there is no reduction in the classification power in either of the problems, despite the fact that some of the original features have been completely eliminated from the study. On the contrary, the chosen features in both the problems yielded 100% classification power in some cases.

Keywords Neural networks, Fuzzy rule bases, Feature selection, Modified threshold accepting

V. Ravi, H.-J. Zimmermann
Lehrstuhl für Unternehmensforschung
RWTH, Templergraben 64
D-52056 AACHEN, Germany

V. Ravi (✉)
Deputation from Computer Center
Indian Institute of Chemical Technology
Hyderabad – 500 007 (AP), India
E-mail: vravi@iict.ap.nic.in

The first author wishes to thank Deutscher Akademischer Austauschdienst (DAAD) for providing him the financial support for this work through a research fellowship leading to his Ph.D degree. The authors also express their gratitude to Dr. W. H. Wolberg, University of Wisconsin Hospitals Madison, Wisconsin USA and Prof. O. Mangasarian (mangasarian@cs.wisc.edu) who are principal donors of the breast cancer database to the internet.

1 Introduction

Among the many paradigmatic shifts witnessed in the current century, fuzzy set theory enunciated by Zadeh [28], appears impressive with its applications, making inroads into as diverse fields as computer science, operations research, economics, medicine and all branches of engineering and science during the past two decades. Knowledge based systems are the earliest applications which are predominantly based on fuzzy 'if-then' rules. Most of these systems derive these fuzzy 'if-then' rules from human experts [10]. In the literature, one comes across several methods to generate these fuzzy 'if-then' rules directly from numerical data. Kosko [15] employed neural networks to achieve this goal. Later, Ishibuchi et al. [10], proposed a sound methodology to generate such rules from numerical data and then they went ahead to apply a genetic algorithm to determine a compact rule set with a high classification power [11]. A software by name "WINROSA" [27], which automatically generates fuzzy 'if-then' rules from numerical data using statistical methods, became available in the market. However, all the aforementioned studies differ from that of [10, 11] in several aspects.

Fuzzy logic and neural nets can be integrated in several ways. To mention some of these, fuzzy logic can be incorporated into the nodes of a neural net [20]; neural nets can used to estimate the membership function of a fuzzy set [13]; using a neural network to learn certain parameters of a fuzzy model such as using a self-organizing feature map to find fuzzy rules [21]; viewing a fuzzy model as a special neural network and applying a learning algorithm directly [19]. In the present paper, a novel way of combining the two vital areas of soft computing viz., neural nets and fuzzy logic is proposed. In the present hybrid, neural nets are used only for the feature selection purpose and fuzzy logic is used for 'if-then' rule based pattern classification. Further, to train feedforward neural nets, a new algorithm based on threshold accepting meta-heuristic is proposed, which outperforms the traditional backpropagation algorithm in both speed and quality.

Throughout this paper, the fuzzy partition of a pattern space means its granularity. To generate fuzzy if-then rules from numerical data one must (i) find the partition of a pattern space into fuzzy subspaces and (ii) determine the fuzzy if-then rule for each fuzzy partition [10, 11]. Using those fuzzy if-then rules, either the training data or the test data are classified, which is essentially the classification phase. The performance of such a classifier depends very

much on the choice of a fuzzy partition. If a fuzzy partition is too coarse, many patterns may be misclassified. On the other hand, if it is too fine, many fuzzy if-then rules cannot be generated due to the lack of training patterns in the corresponding fuzzy subspaces. To obviate this difficulty of choosing an appropriate partition, Ishibuchi et al. [10] have proposed to consider simultaneously the fuzzy rules corresponding to both coarse and fine partitions of a fuzzy subspace. For example, in the distributed representation of fuzzy if-then rules, the two-dimensional pattern space gives rise to 90 ($=2^2 + 3^2 + 4^2 + 5^2 + 6^2$) rules, assuming that each feature dimension is divided into 6 partitions at the most. In other words, 5 rule tables corresponding to all the partitions are considered simultaneously.

This approach, however, suffers from a disadvantage viz., the number of fuzzy if-then rules increases exponentially for classification problems with high dimensional pattern spaces [12] such as the wine classification problem [8] where 13 feature variables exist. For example, if 5 partitions are used for each of the 13 feature variables, the total number of rules would be $2^{13} + 3^{13} + 4^{13} + 5^{13}$. To overcome this problem, Ishibuchi et al. [12] have proposed a method where the fuzzy if-then rules, with a small number of antecedent conditions, are generated as candidate rules. We believe, however, that it is still not a complete remedy because the method is not general and it is not difficult to find problems where rules with a small number of antecedent conditions are intractable. This motivated us to develop other alternative methods, which concentrate on feature selection or reduction of feature space dimension by transforming it. Thus it is meaningful to look for any unimportant variables (features) and remove them from the classification process. This results in reduced computational time and memory requirement and an easy-to-use classifier with a manageable number of features. Thus, the feature selection is an essential component of any classifier, especially in dealing with problems having a large number of features. Ravi and Zimmermann [22] addressed this problem by resorting to the use of a software plug-in to "DataEngine", viz. "FeatureSelector" [26]. They used it as a pre-processor to select the most salient features from the original set of features and then derived a compact set of fuzzy 'if-then' rules with high classification power. In another study, Ravi et al. [23] employed principal component analysis to reduce the dimension of the feature space.

In the present paper, the authors propose the use of neural networks as a pre-processor to take care of the feature selection. A comparison between the present study and the one in [22] is also carried out. However, such a comparison with our earlier study based on principal components analysis [23], is simply not meaningful because the principal components are nothing but linear combinations of the original feature variables and hence, they do not directly reflect the importance or otherwise of the original variables. The remainder of the paper has been structured as follows. Section 2 gives an overview of the general feature selection algorithms and describes the works related to neural networks as feature selecting tools. Section 3 briefly presents the fuzzy rule based classifier and the formulation of the multi-objective optimization problem. Discussion of the numerical simulations is presented in Sects. 4 and 5 concludes the paper.

2
Neural networks application to feature selection

As early as 1973, it was recognised that feature selection is an integral component of classification tasks. Kittler and Young [14] while reviewing the then existing feature selection procedures, proposed a new feature selection algorithm based on Karhunen–Loeve expansion. Then Chang [5] applied dynamic programming to this problem. The most recent work is that of Bradley et al. [4] who proposed a linear programming formulation of the feature selection problem. All these methods are based on either statistical principles or mathematical programming techniques. Among the neural networks based methods for feature selection the most relevant work is that of Bastian and Gasos [2], who present a recursive identification algorithm to determine the decisive input variables by employing the generalization ability of neural networks. Since this method is recursive in nature and requires several networks to be trained and tested, it is extremely time consuming.

However, the work of Garson [9] requires only one neural network to be trained and tested in determining the important feature variables in classification tasks. Nath et al. [18] conducted numerical simulations to conclude that the method of Garson's method [9] is promising, which depends essentially on the well-known backpropagation algorithm to train the network. However, it is now well-known that the backpropagation algorithm, notwithstanding its immense popularity, takes enormously long time to converge. The second disadvantage of the backpropagation algorithm is that it cannot obtain the global "minimum" of the error surface and thus, can get entrapped in a local minimum, as the weights are updated via the steepest descent method. To overcome the second disadvantage, Dorsey et al. [6] and Sexton et al. [24] proposed a genetic algorithm and Sexton et al. [25] proposed tabu search method to train the feedforward neural networks. In this paper, another global optimization metaheuristic viz., threshold accepting [7], which overcomes both the disadvantages, is proposed to train the neural network. Since it is not the central theme of the paper, a comparison of this method with the backpropagation algorithm will be reported elsewhere.

Basic concepts of neural networks

Neural networks are biologically inspired, massively parallel computational structures that appear to solve almost all problems in function approximation, pattern classification etc. Their characteristic properties which make them mimic the human brain are (i) learning from experience (examples) (ii) generalization (iii) abstraction and (iv) fault tolerance. Neural networks can usually be trained in two distinct ways: supervised and unsupervised. Since supervised training is used in this paper, the aspect of unsupervised training shall not be discussed any more in this paper. The most popular method of supervised training of neural networks is the backpropagation algorithm for a standard neural network architecture of one input layer, one hidden layer and one output layer [2, 6, 9, 18].

An overview of Garson's method

Garson [9] proposed a simple but powerful feature selection method based on the connection weights of the neural network. He starts by arguing that the black-box image of the neural networks is quite misleading, as the connection weights obtained after training the neural network contain more information than normally expected. The weights along the paths from input layer to the output layer indicate the relative importance of the input nodes (i.e. input variables). That is, the weights can be used to partition the sum effects on the output layer, using the step-by-step method described below. Thus, the hidden-to-output connection weights of each hidden node are partitioned into components associated with each input node. It is to be noted that the weights related to bias node are not used in this procedure, as the structure of the backpropagation algorithm does not allow this. Hence, it is assumed that the partitions of the connection weights other than the bias node reasonably estimate the partition related to the weights of bias node. Nath et al. [18] compared this method with the traditional stepwise variable selection rule in Fisher's linear discriminant analysis. Garson's method and then its use by Nath et al. [18] is restricted to only two-group classification problems, however. In this paper, we further extend it to multi-group classification problems. We modified the Garson's method by replacing the backpropagation algorithm with a threshold accepting based algorithm. We introduced another variation in the present implementation of Garson's method, which enforces attainment of 90% classification rate in the testing phase of the neural net, before actually proceeding ahead with partitioning the connection weights. This enforcement is assumed to ensure better assessment of all the features. This kind of stipulation was not mentioned either in the original algorithm of Garson or its implementation by Nath et al. [18].

Following the notation of Nath et al. [18], Garson's method is described as follows. Consider a neural network with p nodes in the input layer, h nodes in the hidden layer and one node in the output layer. Let w_{ij}, $i = 1, 2, \ldots, p$ and $j = 1, 2, \ldots, h$ denote the weight of the connection from the ith input node to the jth hidden node. Also, let w_k, $k = 1, 2, \ldots, h$ be the weight of the connection from the kth hidden node to the sole output node. It is important to note that in the following procedure, the optimal connection weights i.e. weights obtained at the end of training phase using threshold accepting algorithm, are used to select the most important input variables (features). Garson's method divides the hidden-to-output node connection weights of each hidden node into partitions associated with each input node. The adjusted weight of each input node, obtained as a result of these partitions, acts as an indicator of its importance or ratings in percentage terms.

Step-by-step procedure of Garson to obtain the ratings of the input variables

(i) The absolute value of each hidden-to-output connection weight w_k, $k = 1, 2, \ldots, h$ is incorporated into the input-to-hidden node weights w_{ij} to yield partitions w^*_{ij} using the following expression
$w^*_{ij} = \frac{|w_{ij}|}{S_j} \times |w_k|$; $i = 1, 2, \ldots, p$ and $j = 1, 2, \ldots, h$
where $|\bullet|$ represents the absolute value and
$S_j = \sum_{i=1}^{p} |w_{ij}|$. Hence, for each hidden node, the sum of w^*_{ij} over all input nodes is exactly equal to the absolute value of the hidden-to-output node weight w_k.

(ii) For each input node, the adjusted weights w^*_{ij} are summed over all the hidden nodes and converted into a percentage of the total for all input nodes. This percentage value, serves as a rating of the feature represented by the input node.

3 A classifier with fuzzy if-then rules

Once the ratings of the input features (variables) are obtained, the most important features are chosen from the original set of features depending on their importance. In the feature selection phase, the parameters of the threshold accepting algorithm, which is used to train the neural network, are adjusted such that more than 90% classification rate is yielded by the neural network on the test data, for each of the problems studied in the paper. This, in the opinion of the authors, ensures accurate and appropriate rating for all the features. This is done primarily because some of the original features are going to be dropped from the subsequent part of the study after assessing their relative importance. At this stage, the fuzzy rule based classifier, described briefly below, is invoked with the reduced feature set.

Following the notation of [10, 11], let the pattern space be two-dimensional (i.e. there are two features in the feature space) and given by the unit square [0, 1] × [0, 1]. Suppose that $X_p = (x_{p1}, x_{p2})$, $p = 1, 2, \ldots, m$ are given as training patterns from M classes (where $M \ll m$): Class 1 ($C1$), Class 2 ($C2$), ... , Class M (CM). The problem is to generate fuzzy if-then rules that divide the pattern space into M crisp classes. For more details of the extension to the case of higher dimensions, the reader is referred to [11]. Let each axis of the pattern space be partitioned into K fuzzy subsets $\{A_1^K, A_2^K, \ldots, A_K^K\}$, where A_i^K is the ith fuzzy subset and the superscript K indicates the number of fuzzy subsets on each axis. Thus, K denotes the grid size of a fuzzy partition. A symmetric triangular membership function in the unit interval [0, 1] is used for A_i^K, $i = 1 \ldots K$ [10, 11, 22]. For problems involving M classes and 2 features, a fuzzy if-then rule corresponding to K^2 fuzzy subspaces has the following structure.

Rule R_{ij}^K: If x_{p1} is A_i^K and x_{p2} is A_j^K

then X_p belongs to Class C_{ij}^K,

$i = 1, 2, .., K$ and $j = 1, 2, .., K$. (1)

where R_{ij}^K is the label of the fuzzy if-then rule, C_{ij}^K is the consequent (i.e. one of the M classes). C_{ij}^K in Eq. (1) is determined by the procedures, which can be found in detail in [10, 11]. These procedures (i) generate the fuzzy if-then rules and (ii) classify the new patterns. These procedures are slightly modified by the authors in [22]. Let S^K be the set of generated K^2 fuzzy if-then rules given by $S^K = \{R_{ij}^K | i = 1, 2, \ldots, K; j = 1, 2, \ldots, K\}$. Let the set of all

fuzzy if-then rules corresponding to $K = 2, 3, \ldots, L$ partitions be S_{ALL} given by

$$S_{\text{ALL}} = S^2 \cup S^3 \cup \ldots\ldots \cup S^L$$
$$= \{R_{ij}^K | i = 1, 2, \ldots, K;$$
$$j = 1, 2, \ldots, K \text{ and } K = 2, 3, \ldots, L\}$$

where L is an integer to be specified depending on the classification problem. Let S be a subset of S_{ALL}.

The multi-objective combinatorial optimization problem

As in [11, 22], the main objective is to find a compact rule set S with very high classification power by employing a combinatorial optimization algorithm. The two objectives in the present problem are: (i) maximize the number of correctly classified patterns and (ii) minimize the number of fuzzy if-then rules. Accordingly, we have,

$$\text{Maximize NCP}(S) \text{ and Minimize } |S|$$
$$\text{Subject to } S \subseteq S_{\text{ALL}}$$

where NCP(S) is the number of correctly classified patterns by S and $|S|$ is the number of fuzzy if-then rules in S. This is further reformulated as a scalar optimization problem below.

$$\text{Maximize } f(S) = W_{\text{NCP}} \cdot \text{NCP}(S) - W_S \cdot |S| \text{ subject to } S \subseteq S_{\text{ALL}} \quad (2)$$

Since the classificatory power of the system is more important than its compactness [10, 11, 22], the weights have been specified as $W_{\text{NCP}} = 10.0$ and $W_S = 1.0$ in Eq. (2) as suggested in [11]. For the details regarding the coding of the rules to be used in the optimization module, the reader is referred to [22]. We employ a meta-heuristic viz., modified threshold accepting algorithm [22] to solve the combinatorial optimization problem just described in Eq. (2).

4 Numerical simulations and results

The first illustrative example solved using the methodology presented here, is the well-known wine classification problem for which the data are freely available in the Internet [8]. It has 13 features (attributes) which classify 178 patterns into three types of wines. The second numerical example concerns the determination of the breast cancer in humans from Wisconsin University hospital in Madison, USA. This is also freely available in the Internet ("Wisconsin Breast Cancer Database". 1992. Available via anonymous ftp from ics.uci.edu in directory/pub/machine-learning-databases/Wisconsin-breast-cancer). This problem has 9 features or attributes, which determine whether a patient is benign or malign. There are 683 samples or patterns. This data has been used in the past by Aeberhard et al. [1]; Mangasarian et al. [16, 17] and Bennet and Mangasarian [3]. To implement the model, a software, in ANSI C, is developed by the authors on a Pentium 100 MHz machine under Windows 95 platform using the MS-Visual C++ 5.0 compiler.

The results of the feature selection phase, which uses neural networks, are as follows. As regards the wine classification problem, 118 patterns were chosen for training the network and the rest 60 for testing. The TA based training proposed in this paper, yielded 90% classification in this case and the most important features selected by the authors following Garson's method are: Feature no. 13 (13.745%); Feature no. 6 (11.393%); Feature no. 7 (11.245%); Feature no. 3 (10.965%); Feature no. 2 (10.907%). These are treated as the new set of features and the classifier is invoked. As far as the Wisconsin breast cancer problem is concerned, the TA based training produced 99% classification when working with 483 training patterns and 200 testing patterns. The most important features chosen by the authors following the method of Garson are: Feature no. 9 (16.685%); Feature no. 7 (13.912%); Feature no. 3 (13.272%); Feature no. 6 (12.914%). In the next phase, this reduced feature set is fed to the classifier. For the sake of convenience, these features are renumbered from 1 to 5 in Table 5 and 1 to 4 Table 6 respectively.

In the classification phase using fuzzy rule bases, the algorithm is tested in two ways: (i) using the training data itself as the test data (ii) using the leave-one-out technique in the testing phase. The latter method is preferable, as there is the danger of over-fitting in the former method. In each of these methods, all the feature spaces have been divided into a maximum of 5 partitions for both the examples. This is done in order to keep the computational complexity to a reasonable level, as we work with 5 features in example 1 and 4 features in example 2. The earlier study of Ravi and Zimmermann [22] via "FeatureSelector" used 5 features for the wine classification problem (example 1) and 3 features for the Wisconsin breast cancer problem (example 2). Thus the present study as well as that of [22] selected 5 features for example 1 whereas in example 2, only 3 features were selected in [22], but the present study selected 4 features. Hence, the comparison of the present study with that of Ravi and Zimmermann [22] would be meaningful, if we compare only the results of example 1 and not those of example 2. In Tables 1 and 2, the results of the present study are presented in bold-faced numerals, whereas those of [22] are presented without bold face numerals. Further, the study has been conducted for 5 cases each corresponding to different aggregator viz. (1) product operator (2) 'min' operator (3) γ – operator (compensatory 'and') (4) 'fuzzy and' and (5) a convex combination of min and max operators. Thus in the Tables 1 to 4, cases (1) to (5) represent the type of aggregator used in the above order.

Results of the wine classification problem (see Table 1) indicate that the product operator performed consistently well and gave the best solution of 98.88% classification with 21 rules when 5 partitions were considered. The γ – operator (with $\gamma = 0.00001$) came closely behind giving 98.31% classification with 22 rules in the case of 5 partitions. When 4 partitions were considered, the product operator yielded 98.88% classification with 24 rules and the γ – operator (with $\gamma = 0.00001$) resulted in 98.88% classification with 25 rules.

Table 1. Results of Example 1 (training data used as test data)

Case	1		2		3		4		5	
No. Partitions	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$
5	**98.88**	**21**	**98.31**	**30**	**98.31**	**22**	**96.63**	**42**	**96.63**	**42**
	100	14	99.44	13	100	16	98.88	65	98.88	42
4	**98.88**	**24**	**97.75**	**22**	**98.88**	**25**	**95.51**	**15**	**97.19**	**21**
	98.88	13	98.88	13	98.88	13	100	11	98.31	15
3	**95.51**	**15**	**95.51**	**18**	**95.51**	**16**	**94.94**	**16**	**95.51**	**16**
	98.88	15	97.75	12	98.88	15	95.51	12	97.75	12

C.P: Classification power

Table 2. Results of Example 1 (leave-one-out technique)

Case	1		2		3		4		5	
No. Partitions	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$
5	**98.88**	**6.88**	**100**	**6.95**	**98.88**	**6.88**	**98.88**	**23.16**	**91.57**	**21.52**
	100	4.71	100	4.71	100	4.71	100	54.3	95.51	52.2
4	**100**	**3**	**99.44**	**2.98**	**100**	**3**	**99.44**	**2.98**	**99.44**	**2.98**
	100	3	93.26	2.8	93.26	2.8	100	2.96	96.63	2.89
3	**99.44**	**3.24**	**98.88**	**3.24**	**99.44**	**3.26**	**98.88**	**3.29**	**98.88**	**3.29**
	99.44	3	99.44	3.36	93.26	3.2	98.88	3.03	98.88	3.31

Bold-faced numbers indicate present study

Table 3. Results of Example 2 (training data used as test data)

Case	1		2		3		4		5	
No. Partitions	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$
5	97.66	15	96.63	15	97.66	16	97.8	13	97.07	15
4	97.36	13	96.34	16	96.19	10	97.36	13	96.19	7
3	96.34	13	96.05	15	95.9	13	96.93	8	95.9	7

The same example when studied with leave-one-out technique (see Table 2), produced different results. 'Product' operator and γ – operator (with $\gamma = 0.00001$) both provided the overall best solution with 100% classification with just 3 rules on average, when 4 partitions were considered, whereas the min operator, fuzzy and (with $\gamma = 0.999$, where γ is the weight defined in fuzzy and operator) and the convex combination of min and max operators (with $\gamma = 0.000001$, where γ is the weight defined in the convex combination of min and max operators) produced 99.44% classification with 2.98 rules on average. In the case of 5 partitions, the min operator produced the best solution of 100% classification with 6.95 rules on average. In the case of 3 partitions, all operators yielded high classification powers.

For the wine classification problem, the results of the present study are compared with those of Ravi and Zimmermann [22]. When training data is used as test data, the results of [22] outperformed those of the present study. However, when the leave-one-out technique, which is more authentic, is employed, the difference in both studies is only marginal. In the case of 3 partitions, both the approaches yielded almost similar solutions. In the case of 4 partitions, results of the present study outperformed those of [22] in many cases. More important aspect is that the best solution produced in the leave-one-out technique is almost identical in both the studies (100% classification with 3 rules on average). When 5 partitions were considered, results of [22] turned out to be superior to those of the present study.

In the current study, the authors stopped the feature selection phase, when 90% or more classification was obtained in either of the problems. Hence, the authors feel that by training the neural network to get more classification power in the feature selection phase, would have a positive impact on the final results in the classification phase and would lead to better assessment of the features. This can be achieved by suitably changing the parameters of the TA algorithm and size of the neighbourhood.

Results of the Wisconsin breast cancer problem (see Table 3) show that the fuzzy and operator (with $\gamma = 0.999$, where γ is the weight used in defining fuzzy and operator) yielded the best solution of the table, of 97.8% classification power with 13 rules, when 5 partitions were considered. The product operator and the γ – operator and (with $\gamma = 0.0000001$) came closely behind with 97.66% classifi-

cation power with 15 and 16 rules respectively, whereas the convex combination of min and max operators performed slightly better than the min operator. The product and the fuzzy and operators (with $\gamma = 0.999$, where γ is the weight used in defining fuzzy and operator) gave rise to a maximum 97.36% classification power with 13 rules when 4 partitions were considered. The fuzzy and operator (with $\gamma = 0.000001$, where γ is the weight used in defining fuzzy and operator) yielded a solution of 96.93% classification power with 8 rules when 3 partitions were considered. Thus the fuzzy and operator performed consistently well for all the partitions.

When the leave-one-out technique was applied to the same problem (see Table 4), the best solution of the table viz. 100% classification power with just 3 rules on the average, was achieved by the fuzzy and and the convex combination of min and max when either 3 or 4 partitions were considered, irrespective of whether the γ value, used in defining these operators is taken as 10^{-6} or 0.999. The next best solution of 100% classification power with 6.26 rules on the average was achieved by the convex combination of min and max operators (with $\gamma = 10^{-6}$, where γ is the weight used in defining this operator), when 5 partitions were considered. The product operator at best produced 98.54% classification power and 2.96 rules on the average; the min operator yielded at best 99.12% classification power with just 2.97 rules on the average and the γ – operator (with $\gamma = 10^{-6}$) produced at best 95.75% classification power with 2.88 rules on the average. Thus for this problem, the fuzzy and and the convex combination of min and max operators yielded consistently very good solutions.

The optimal rule-base, for the wine classification problem (case 1 with 5 partitions), obtained after solving the combinatorial optimization problem in Eq. (2), is presented in Table 5. Similarly, Table 6, presents the optimal rule-base for the Wisconsin breast cancer problem (case 4, with 5 partitions). In both tables, the symbols "low-2" and "high-2" indicate that the features are divided into 2 fuzzy partitions. Thus, "low-2" is different from "low-3", "low-4" and "low-5" as they correspond to fuzzy set "low" when the feature is divided into 3, 4 and 5 partitions respectively. Thus it would be confusing, if we use just "low", without mentioning the context (i.e. the number of partitions used for the feature) in which it is used. To avoid this confusion, the above kind of notation is used in the paper.

Conclusions

This paper presents a novel neural network and fuzzy rule based hybrid wherein, the neural network module is especially used for feature selection in classification

Table 4. Results of Example 2 (leave-one-out technique)

Case	1		2		3		4		5	
No. Partitions	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$	C.P.	$\|S\|$
5	98.54	2.96	99.12	2.97	95.75	2.88	99.71	6.01	100	6.26
4	80.53	2.42	79.94	2.4	80.53	2.42	100	3	100	3
3	89.31	2.68	79.94	2.4	89.31	2.68	100	3	100	3

Average number of rules is presented here

Table 5. Rule Table for Wine problem (Case 1 with 5 partitions in Table 1)

Rule No.	Antecedents					Consequent–Class Code
	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	
1	Low-2	Low-2	Low-2	High-2	Low-2	3
2	Low-2	Low-2	High-2	Low-2	Low-2	2
3	Low-2	Low-2	High-2	Low-2	High-2	2
4	Low-2	Low-2	High-2	High-2	Low-2	2
5	High-2	Low-2	Low-2	Low-2	High-2	3
6	High-2	High-2	High-2	Low-2	High-2	1
7	High-2	High-2	High-2	High-2	Low-2	1
8	Low-3	Medium-3	Medium-3	Medium-3	Low-3	2
9	Low-3	High-3	Medium-3	Medium-3	Low-3	2
10	Low-3	High-3	High-3	Low-3	Low-3	2
11	Medium-3	Medium-3	High-3	Low-3	Low-3	1
12	Very-low-4	Very-low-4	Low-4	Low-4	High-4	3
13	Low-4	High-4	Very-low-4	High-4	Low-4	3
14	Low-4	High-4	High-4	Very-low-4	High-4	2
15	Very-low-5	Medium-5	Low-5	High-5	Low-5	3
16	Low-5	Low-5	Low-5	High-5	Low-5	2
17	Low-5	High-5	High-5	Medium-5	High-5	1
18	Medium-5	Low-5	Low-5	High-5	Very-high-5	3
19	Medium-5	Medium-5	Very-low-5	Medium-5	Very-low-5	2
20	Medium-5	High-5	Very-high-5	High-5	Low-5	1
21	High-5	Low-5	Medium-5	Low-5	Low-5	1

Table 6. Rules for Wisconsin breast cancer problem (Case 4 with 5 partitions in Table 3)

Rule #	Antecedents				Consequent–Class Code
	Feature 1	Feature 2	Feature 3	Feature 4	
1	High-2	Low-2	High-2	Low-2	1
2	Low-3	Medium-3	Medium-3	High-3	2
3	Low-3	Medium-3	High-3	Low-3	1
4	Medium-3	Low-3	Medium-3	Medium-3	1
5	Very-low-4	Very-low-4	Very-low-4	Very-low-4	1
6	Low-4	Low-4	Very-low-4	High-4	1
7	Very-low-5	Low-5	Low-5	High-5	2
8	High-5	Medium-5	Very-high-5	Low-5	2
9	High-5	High-5	Very-high-5	Low-5	2
10	High-5	Very-high-5	Medium-5	Very-high-5	2
11	Very-high-5	Very-low-5	Very-low-5	Medium-5	1
12	Very-high-5	Low-5	High-5	Medium-5	2
13	Very-high-5	Medium-5	High-5	Low-5	2

problems involving a large number of dimensions. Threshold accepting has been used for the first time to train the neural network instead of the traditional and more popular backpropagation algorithm, owing to the latter's disadvantages such as long training times and getting trapped in local minima. The threshold accepting proved to be very robust in obtaining the near global optima for the connection weights of the neural network. Low convergence time required by it is a remarkable achievement of this study. The features selected by the neural network are fed as the new set of features to the modified fuzzy rule based classifier [22]. The chosen features in both the examples resulted in a very high classification power of 100% in the leave-one-out technique with a few rules in the case of some aggregators. Comparison of the present results with those of a similar study (Ravi and Zimmermann [22]) show that they outperform each other in different cases, in the leave-one-out form of testing. Hence, the authors conclude that the neural networks, trained by a meta-heuristic such as threshold accepting can be used as a useful alternative to other methods of feature selection existing in literature, while solving classification problems with higher dimensional feature spaces, because this method of training saves the longer learning (training) times, normally associated with neural networks trained with backpropagation algorithm.

References

1. **Aeberhard S, Coomans D, de Vel O** (1992) Comparison of Classifiers in High dimensional Settings, Technical Report No. 92–02, Dept. of Computer Science and Dept. of Mathematics and Statistics, 1992, James Cook University of North Queensland
2. **Bastian A, Gasos J** (1996) Selection of Input variables for model identification of static non-linear systems, J Int Robot Syst **16**: 185–207
3. **Bennett KP, Mangasarian OL** (1992) Robust linear programming discrimination of two linearly inseparable sets, Optimiz Meth Software **1**: 23–34
4. **Bradley PS, Mangasarian OL, Street WN** (1998) Feature selection via mathematical programming, INFORMS J Comput, **10**: 209–217
5. **Chang C** (1973) Dynamic programming as applied to feature subset selection in a pattern recognition system, IEEE Trans Syst Man Cyber **3**: 166–171
6. **Dorsey RE, Johnson JD, Mayer WJ** (1994) A genetic algorithm for training of feedforward neural networks, Advances in Artificial Intelligence in Economics, Finance Management, **1**: 93–111
7. **Dueck G, Scheuer T** (1990) Threshold Accepting: a general purpose optimization algorithm appearing superior to simulated annealing, J Comput Phys **90**: 161–175
8. **Forina M**, et al (1992) "Wine Recognition Database" Available via anonymous ftp from ics.uci.edu in directory/pub/machine-learning-databases/wine
9. **Garson GD** (1991) Interpreting neural network connection weights, AI Expert, 47–51
10. **Ishibuchi H, Nozaki K, Tanaka H** (1992) Distributed representation of fuzzy rules and its application to pattern classification", Fuzzy Sets Syst **52**: 21–32
11. **Ishibuchi H, Nozaki K, Yamamoto N, Tanaka H** (1995) Selecting fuzzy if-then rules for classification problems using genetic algorithms, IEEE Trans Fuzzy Syst **3**: 260–270
12. **Ishibuchi H, Tadahiko M** (1997) Minimizing the Fuzzy Rule base and maximizing its performance by a Multi-objective Genetic Algorithm, Sixth FUZZ-IEEE Conference Barcelona, Spain, 259–264
13. **Kim J, Krishnapuram R** (1993) Membership function generation methods for pattern recognition and computer vision, Proc. International Fuzzy Systems Association Congress, Seoul, 115–118
14. **Kittler J, Young PC** (1973) A new approach to feature selection based on the Karhunen–Loeve expansion, Pattern Recog **5**: 336–352
15. **Kosko B** (1992) Neural Networks and Fuzzy Systems, Prentice-Hall, Englewood Cliffs
16. **Mangasarian OL, Wolberg WH** (1990) Cancer diagnosis via linear programming, SIAM News **23**: 1–18
17. **Mangasarian OL, Setiono R, Wolberg WH** (1990) Pattern recognition via linear programming: theory and application to medical diagnosis, in Coleman TF, Li Y (Eds), Large-Scale Numerical Optimization, SIAM Publications, 22–30
18. **Nath R, Rajagopalan B, Ryker R** (1997) Determining the saliency of input variables in neural network classifiers, Computers and Operations Research, **24**: 767–773
19. **Nauck D, Kruse R** (1996) Designing neuro-fuzzy systems through backpropagation, Fuzzy Modelling: Paradigms and Practice, W. Pedrycz (Ed), Kluwer Academic Publishers, Boston, 203–228
20. **Pal SK, Mitra S** (1992) Multi-layer perceptron, fuzzy sets and classification, IEEE Trans Neural Networks **3**:
21. **Pedrycz W, Card HC** (1992) Linguistic interpretation of self-organizing maps, Proc. IEEE Int. Conference on Fuzzy Systems, Fuzz-IEEE'92, San Diego, 371–378

22. **Ravi V, Zimmermann H-J** (2000) Fuzzy rule based classification with FeatureSelector and modified threshold accepting, Eur J Oper Res **123**: 16–28
23. **Ravi V, Reddy PJ, Zimmermann H-J** (2000) Pattern classification with principal components analysis and fuzzy rule bases, Eur J Oper Res **126**:
24. **Sexton RS, Dorsey RE, Johnson JD** (1998) Toward global optimization of neural networks: a comparison of the genetic algorithms and backpropagation, Decision Support Systems, **22:** 171–185
25. **Sexton RS, Alidaee B, Dorsey RE, Johnson JD** (1998) Global optimization for artificial neural networks: a tabu search application, Eur J Oper Res **106**: 570–584
26. **Strackeljan J, Behr D, Detro F** (1997) FeatureSelector: a Plug-In for Feature Selection with DataEngine, First International Data Analysis Symposium, Aachen, Germany
27. **WINROSA** (1997), Manual, MIT GmbH, Aachen, Germany
28. **Zadeh L** (1965) Fuzzy sets, Inform Control **8**: 338–353