APPLICATION OF INFORMATION FUSION METHODS TO BIOMEDICAL DATA

by

Petr Jilkine

A Thesis Submitted to the Faculty of Graduate Studies in Partial Fulfillment of the Requirements for the Degree of

Ph.D.

Department of Electrical and Computer Engineering,
University of Manitoba Winnipeg, Manitoba

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION PAGE

APPLICATION OF INFORMATION FUSION METHODS TO BIOMEDICAL DATA

BY

PETR JILKINE

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirements of the degree of

DOCTOR OF PHILOSOPHY

TABLE OF CONTENTS

University of Manitoba
Abstract

APPLICATION OF INFORMATION FUSION METHODS TO BIOMEDICAL DATA

by Petr Jilkine

Classification of Magnetic Resonance (MR) and Infrared (IR) spectra promises to become an effective tool for early medical diagnosis of diseases. The proposed thesis project involves the development and comparison of classification strategies and algorithms for the analysis of spectra of healthy and diseased tissue biopsies of various disease states. Several methods of aggregating outcomes of classifiers are considered in order to improve the classification accuracy, and applied to artificial and real-life spectra. Logistic regression, linear combination of classifiers, fuzzy integration, stacked generalization and some other methods of classifier aggregation, as well as different ways of estimating necessary parameters are considered. The results indicate that in many cases aggregation of classifiers improves the classification performance in comparison to that of the classifiers being aggregated. The results on real-life spectra vary. The methods perform well on some data sets and relatively poorly on others. Strategies are recommended to gain from classifier aggregation.

ACKNOWLEDGMENTS

I would like to acknowledge my supervisor Dr. R.L. Somorjai for his invaluable support and training. I also acknowledge the Institute for Biodiagnostics of the National Research Council Canada for financial support, and for the excellent research facilities.

I would like to thank all the great people in the Informatics group who helped me during the completion of this project, and made these years real fun.

I gratefully acknowledge my PhD committee for their support and advice.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

MR	Magnetic Resonance
MSE	Mean Squared Error
LDA	Linear Discriminant Analysis
KNN	*K*-Nearest Neighbor
ANN	Artificial Neural Network
SG	Stacked Generalization
FI	Fuzzy Integral
FM	Fuzzy Measure
LR	Logistic Regression
CF	Confidence Factor
AVE	Simple Averaging (or Arithmetic Averaging)
MV	Majority Voting
QDA	Quadratic Discriminant Analysis
IR	Infrared
PCA	Principal Component Analysis
PC	Principal Component
GA	Genetic Algorithm

1. Introduction and Literature Review

The Magnetic Resonance (MR) spectrum of a biological tissue sample characterizes its chemical composition. In particular, MR spectra are characteristics of the disease state of the tissue, because cells in different states produce biochemical substances in different amounts. Biochemical changes that signal the onset of disease occur earlier than manifest at the morphological (visual) level. Thus, the possibility of early diagnosis of disease (e.g., cancer) via MR spectroscopy is real and very important. For example, 'average' (or centroid)[1] spectra of healthy (or normal) and diseased brain biopsies are presented in Fig. 1.1. One observes quite substantial differences between the spectra.

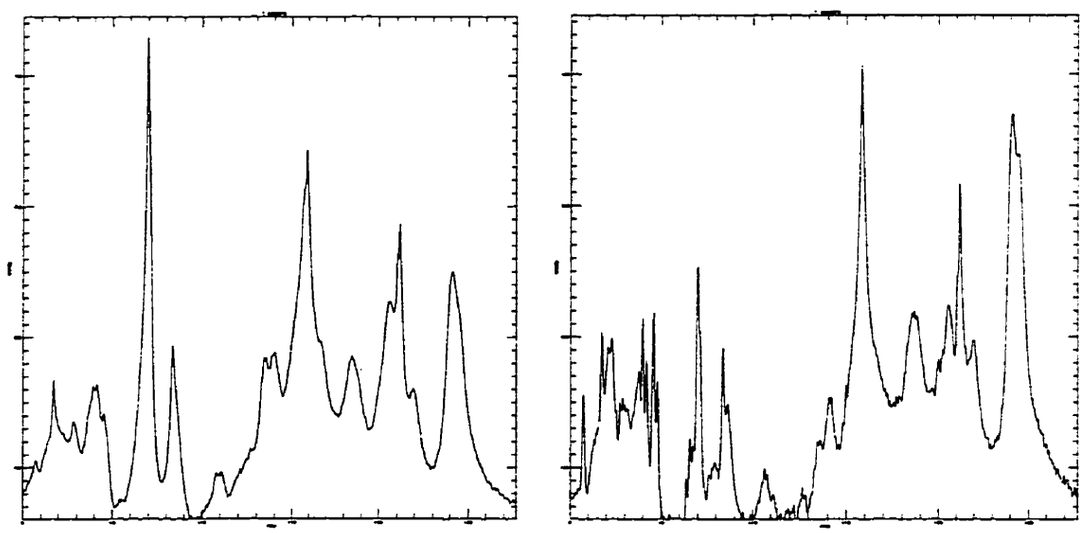

Fig. 1.1 Average spectra of brain samples in normal and diseased states.

[1] The centroid spectra were obtained by averaging all available spectra for the particular disease state, thus the spectral noise is eliminated to a great degree.

The proposed thesis project involves the development and comparison of classification strategies and algorithms for the analysis of MR spectra of healthy and diseased tissue biopsies of various disease states. In particular, combined classification methods based on the aggregation of the outcomes of several classifiers are considered. Different pattern classifiers employ different concepts, have different architectures, adjustable parameters and, thus, behave differently even if trained on the same data set. Ideally, the combining classifiers should take advantage of the strengths of the individual classifiers, avoid their weaknesses, and improve classification accuracy.

Given a complete training set, a consistent[2] classifier is able to approximate Bayesian (optimal) decision boundaries between classes with arbitrary accuracy [8]. In real life we usually operate with finite training sets, often noise corrupted. Trained on these data, different classifiers approximate the decision boundaries differently. In addition, the size of the data set makes the classifier outcome dependent on the particular data set. We can choose the classifier which classifies best and ignore all the others. However, potentially useful information, which the ignored classifiers could access, can be lost. In order to avoid this loss of information one can aggregate the classifier outcomes in some manner in the hope of getting better classification results. This is often the situation for difficult cases, with very limited number of data samples and high noise level (see [10] for example). In particular, this is the case for MR spectra of biological tissues. There is usually a shortage of experimental data. The spectra are noise-corrupted spectra with low signal-to-noise ratio,

[2] The asymptotic convergence of a classifier to the object of classification is called consistency.

overlapping peaks and pronounced baseline distortions. There is also some uncertainty in the *a priori* tissue classification by experts.

The architecture of a combining classifier can be like the ones presented in Fig. 1.2. The original MR spectra are first preprocessed by some technique[3] (or several techniques as in the right-hand scheme). Then the preprocessed data are submitted to several different classifiers (or to a single reliable classifier). The outcomes of these classifiers are aggregated by an aggregation scheme. One can view the group of classifiers to be aggregated as a group

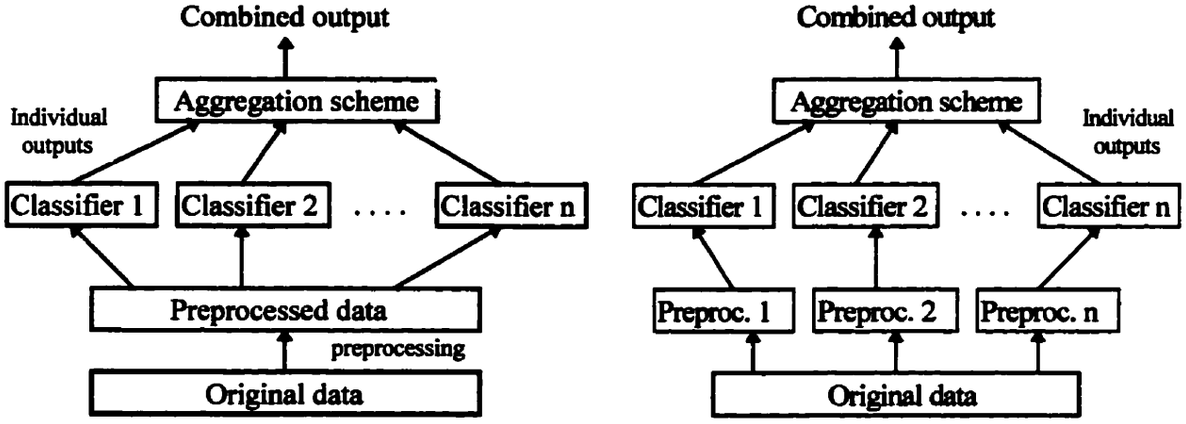

Fig. 1.2 Possible architectures of a combining classifier.

of experts looking into the same problem from their personal points of view and stating their opinions. The aggregation scheme is another expert who generalizes the opinions of the experts in some manner and makes a final decision. The aggregation scheme can be as simple as Majority Voting (MV), where all classifiers are supposed to be equally competent, or more sophisticated, when the aggregation scheme learns the competence of different

[3] Some detail about preprocessing techniques used may be found on page 22.

classifiers during training. The aggregation scheme also can be a classifier such as the one used at the previous stage.

What improvement of classification accuracy can be expected, if we aggregate several classifiers? Some theoretical estimates of error reduction are presented in [7, 9]. It has been shown that given some reasonable assumptions (local monotonicity of the *a posteriori* probabilities about decision boundaries) a linear combination of classifiers reduces variance in the boundary location about the Bayes boundary. If the aggregated classifiers are unbiased, the reduction of added error[4] is proportional to the reduction in variance. If these classifiers make independent errors, the variance becomes smaller by a factor of n, the number of aggregated classifiers. In the presence of classifier bias, the error reduction is smaller since both bias and variance need be reduced. If the bias of classifiers is small, and the error is mainly due to variance, aggregating can be an effective tool. However, in the case of high bias aggregating is effective only if the biases are not highly correlated. Then it is important to keep the biases of the classifiers uncorrelated. This can be achieved by using classifiers based on different principles or by applying the classifiers to differently preprocessed data.

In [9] the authors considered a combining classifier based on order statistics, and demonstrated an improvement in classifier performance. The benefits of aggregating were demonstrated on several data sets. The authors also observed that aggregating compensates

---

[4] Error reduction is applied to the portion of total error (called added error) which occurs because the boundary between classes is not chosen perfectly. Another portion of the total error is Bayes error, which can't be reduced in the problems with overlapping classes.

Another important problem known as the bias/variance dilemma [8] should be taken into account in combined classification. The essence of the dilemma lies in the fact that the error can be decomposed into two components, bias and variance. The bias measures how closely the learning algorithm's guess matches the target. The variance measures how much the learning algorithm's guess "bounces around" for different training sets of given size. Attempts to reduce bias lead to an increase in variance. Keeping the variance small results in bias increase. A compromise is usually reached as a trade-off between bias and variance, which suggests a kind of uncertainty principle. In [9] a possible way of overcoming this difficulty is indicated. The authors noticed that aggregation provides a method for decoupling bias and variance. The bias of aggregated classifiers (also called individual classifiers) should be reduced (e.g., in the case of neural network by using a larger network). The increased variance then can be reduced during the aggregation stage.

Kohavi in [24] investigated bias and variance decomposition and, in particular, gave an example of bias-variance trade-off during classifier aggregation using a UCI data set [25]. The data set was divided into two subsets, one to generate several training sets by uniform random sampling without replacement, the other to evaluate bias and variance in the expected misclassification rate. A decision tree classifier was applied to each generated training set, and the outcomes of 50 classifiers were aggregated by a voting scheme. The results indicate that the reduction of error is almost solely due to reduction in variance. Although the bias increases slightly (especially for smaller training sets) the reduction of variance is significant enough to keep the overall error smaller.

Breiman in [26] analyzed the aggregation of classifiers. In particular, arcing and bagging algorithms were considered. In bagging one forms modified training sets by sampling the original training set, constructs classifiers using these training sets, and has them vote for the classes. Arcing is a more complex procedure, in which the construction of the (k+1)-th classifier depends on the performance of the k previously constructed classifiers. The main effect of both schemes is the reduction of variance. Arcing is more successful in this than bagging. Instability of the classification methods used in the above schemes is essential to improve accuracy. A classifier is called unstable when small perturbations in the training set result in large changes in classifier outcome. Unstable classifiers characteristically have high variance and low bias. Trees and Artificial Neural Networks (ANN) are considered unstable classifiers. Stable classifiers have low variance, but may have high bias. K-Nearest Neighbor (KNN) and Linear Discriminant Analysis (LDA) classifiers are considered stable, so the above techniques have little or no effect on error rates.

Jacobs [28] reviewed two classes of aggregation methods. A Supra Bayesian procedure, in which the decision maker treats the expert opinions as data that may be aggregated with their own *a priori* distribution via Bayes rule, and a linear opinion pool, where the decision maker forms a linear combination of the expert opinions. The first technique is theoretically well-motivated. The disadvantage is that it may be impractical for some real-world tasks. Defining an appropriate likelihood function for the expert opinions can involve much guesswork. Moreover, evaluating this likelihood function can be computationally expensive. The linear opinion pool has the advantage that it is relatively simple, and frequently yields useful results with a moderate amount of computation. The disadvantage is

the lack of a solid theoretical foundation. A high correlation or dependence among expert opinions makes the aggregation difficult. The author suggests that there is a need for training procedures that result in experts with relatively independent opinions, or for aggregation methods that implicitly or explicitly model the dependence among experts. The analysis presented indicates that a smaller number of independent experts are worth the same as more but dependent experts.

In [10] the effect of combining different linear least-square estimators[7] on the performance of linear regression was studied[8]. It was shown that by splitting the data set into several independent parts and training each estimator on a different subset, the performance of the combined estimation can in some cases be significantly improved. In particular, it works for data sets with a small number of noisy samples. The improvement in the quality of the combined estimation occurs because the decrease in variance resulting from the independence of different estimators is larger than the concomitant increase in bias. The author stresses that the general claim that combining experts is always helpful is clearly fallacious. That classifier aggregation can make good classifiers better but can make bad classifiers worse is also noted or observed in [26,31,32].

Perrone in [29] presented a general theoretical framework for ensemble methods of constructing significantly improved regression estimates. The general idea is to generate multiple estimates by subsampling or resampling a finite data set, and then combine them.

[7] The terminology of the author is kept here. A classifier can be considered as an estimator that uses the class memberships as attributes.

[8] Classification can be considered as a special case of regression with zero/one values, and the results of [10] are also applicable to classification.

A hybrid estimator constructed is as good or better in the MSE sense than any of the individual estimators. In particular, two methods were developed: Basic Ensemble Method (BEM) and Generalized Ensemble Method (GEM). GEM was applied to the recognition of characters and numbers. The results indicate that the GEM estimator is better than standard techniques. For instance, the best of ten backpropogation networks with a single hidden layer and 20 hidden units gives 89% performance for lowercase characters, the GEM estimator gives approximately 91.5%. During training individual networks converge to different local minima, thus their error terms are not strongly correlated. This lack of correlation drives the averaging method, allowing to construct an improved estimate. Thus, the averaging method can efficiently utilize the local minima that other techniques try to avoid. It was also found in this paper that for the example considered aggregating more than 6-8 networks doesn't improve the BEM estimator. The authors also state that "training a population of large nets to find the best estimator is computationally much more expensive than training and averaging a population of small nets. In addition, small networks are more desirable since they are less prone to over-fitting than large networks".

David Wolpert [5] introduced the so-called Stacked Generalization (SG) in which different classifier outcomes are aggregated via another classifier. Several classifiers (level 0) applied to preprocessed data (or a single classifier applied to differently preprocessed data) produce class memberships. These memberships form a new set of attributes for another classifier (level 1). Classifiers of level 0 are supposed to behave differently from one another, i.e., their decisions should not synchronized.

Several papers show that aggregating classifiers improves the classification performance in different applications.

In [1] a regression method is used to fuse the decisions of different recognition algorithms. The method computes a weighted sum of the outcomes of individual classifiers (scores) for every class. This sum reflects the confidence of the algorithm that a given sample belongs to a particular class. The class with maximal score is considered as the most likely class. The necessary weights are estimated by logistic regression on the training set. The weights express the relative importance of the aggregated classifiers. Applying this approach to the recognition of machine printed characters (a problem with 48 classes, 6 different classifiers, 19151 training samples, 12000 test samples) yields a 3% increase in accuracy over the best individual classifier. This improvement was achieved when a set of four classifiers out of six used in the study were aggregated. The authors noticed that aggregating two different classifiers trained on the same preprocessed data achieves a higher performance than the individual classifiers do. Aggregating two classifiers trained on differently preprocessed data gives even better improvement in performance. Applying the logistic regression approach to the handwritten digit recognition problem provides additional benefits over individual classifiers. The authors stressed that independence of the classifiers used is a key to better performance.

Tim Kam Ho applied several methods, such as highest rank, Borda count and logistic regression to handwritten digit recognition and degraded multifont machine-printed character and word recognition [6]. The strength of the methods was demonstrated in problems with a large number of classes. In a word recognition experiment four classifiers

were used to discriminate between 1365 classes. An improvement of 7.8% was achieved, from 86.1% accuracy for the best individual classifier to 93.9% for the aggregation by a dynamically selected model. In [4] several combination techniques were considered. The authors found that for the analyzed data a Dempster-Shafer based method obtained high recognition and reliability rates. It is also robust. Application of the method to US zip codes showed significant improvement over the performance of individual classifiers. A performance of 98.9% was achieved, while the performance of the best individual classifier was 93.9%.

In [27] a hybrid system for protein secondary structure prediction was developed. Three different experts based on neural network, memory-based reasoning and statistics learned the mapping between amino acid sequences and secondary structures from the known secondary structures. A combiner (a neural net) took the predictions from the three experts and made a final prediction. The database included 107 protein from the Brookhaven Protein Data Bank. The set of all proteins contain 19,861 amino acids, 113 subunits. There were three possible outcomes (elements of the secondary structure): α-helix, β-strand and coil. The way the system was trained is interesting. The training set was divided into two parts. One part was used to train the experts. The outcomes of the experts on the other part of the training set were used to train the combiner. The reason for dividing the training set into two parts was that the behavior of each expert on the training data can be very different from its behavior on the proteins whose structures were unknown; their performance on the data that they were not trained on (the second half of the training set) reflected their behavior on truly unknown protein structures, which was exactly what the combiner should

know about and be trained on. After the training of the combiner was completed, each expert was trained again with the whole training set. These trained experts together with the trained combiner formed a trained hybrid system. The hybrid system had an overall performance of 66.4%, which was higher than individual experts and all previously reported algorithms. Compared to each expert, the hybrid system produced better results in terms of the number of secondary structures (rather than the number of residues) that were predicted correctly. This was important from the biological point of view.

Joydeep Ghosh in [15] applied a number of aggregation methods to the classification of underwater acoustic signals. It is a difficult problem because of the low signal-to-noise ratio and the high degree of variability in the signals emanating from the same type of sound source. Four approaches to evidence combination were presented and compared using realistic oceanic data. They included an entropy-based weighting of the outcomes of individual classifier, a method based on the combination of confidence factors in a manner similar to that used in MYCIN expert system, majority voting and a simple averaging. A multi-layer perceptron augmented with weight decay strategy and two kernel-based classifiers were among individual classifiers being aggregated. All combining techniques gave better results than those obtained by the best individual classifier.

In [11] Rogova considered an aggregation method based on the Dempster-Shafer theory of evidence. The proposed method leads to a considerable improvement of classification accuracy without complex computations. The method has the useful property of penalizing overoptimistic and overtrained classifiers. Application of this method to hand-printed digits led to the reduction of misclassification error by 15-30%. Experiments showed that a better

result is not necessarily achieved on aggregating classifiers with better individual performance. Independence of the classifiers is a more important factor in aggregation. It was also noticed that classification of differently preprocessed data provide more independent results than different architectures of neural networks.

Hashem considered a combination of different neural networks to achieve better performance [12]. Optimal linear combination of the outcomes of neural networks was proposed to improve the accuracy of a 'combined' model. Accuracy was measured by MSE, optimality was achieved by minimizing this MSE. The method was applied to the problem of approximating a function. The aggregation of six neural networks gives 88% better accuracy (MSE = 0.000017) than the best individual neural network (MSE = 0.000137). Thus, to get the same accuracy one can individually train several "small" networks and aggregate them, rather than train a single "large" network. Although the authors applied their approach to a regression problem, it can also be used in classification. The attractiveness of this approach is its linearity with respect to the unknown weights, which converts the estimation of the weights to a simple matrix inversion problem.

In [13] variants of the majority vote were considered, and combined performances of 7 classifiers on a set of handwritten numerals were analyzed. In particular, a weighted majority vote approach was implemented. The values of the weights were obtained by optimization of an objective function. The objective function was chosen to increase recognition on one hand, and to reduce error rate on the other. Application of the method to a set of 46451 numerals demonstrated 2.2% improvement by the combined classification.

Fuzzy set based methods have recently achieved success in pattern recognition and classification [2,3,14]. Fuzzy methods don't provide solutions to all problems, but they can be useful in situations when features, criteria, etc. are vague. This is often the case in pattern recognition. Fuzzy integration is one of the approaches used in pattern classification. Fuzzy integration is a nonlinear way to combine multiple sources of information. Basically, the Fuzzy Integral (FI) is an aggregation operator. Suppose we use n classifiers to classify an unknown sample. Each classifier provides a confidence value that this sample belongs to a particular class. We aggregate these individual confidence values by FI into a global confidence value. This value represents the likelihood or degree of certainty that the unknown sample belongs to a particular class, taking into account all the evidence available, Fig. 1.3. The so-called Fuzzy Measures (FMs) underlying fuzzy integration play the role of weights for the different classifiers and their subsets. For n classifiers and an m class problem there are $m2^n$ fuzzy measures ($2m$ of them are trivial and are equal to 1 or 0). Classification performance obviously depends on the FMs, thus their accurate estimation is very important. FMs could be obtained by an expert estimating the relative importance of the classifiers and their subsets, or by learning these from a training set. If many classifiers are aggregated by FI, then it is practically impossible to effectively determine FMs by an expert. Estimation of FMs from the training set requires a constrained nonlinear optimization technique.

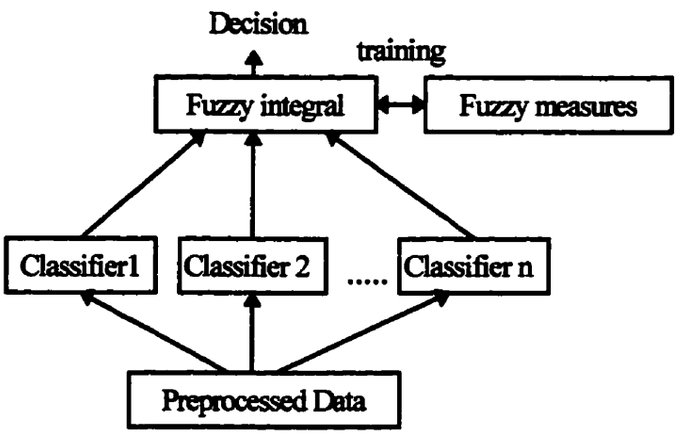

Fig. 1.3 Architecture of a FI-combining classifier.

The process of classification by an FI-combining classifier is as follows. The FMs are estimated for every class from the training set, using some criterion. Given a sample, evidence provided by all individual classifiers is integrated with respect to corresponding class FMs, resulting in an overall confidence value for each class. The sample is assigned to the class with the highest overall confidence value.

Keller in [2] examined the FI as a decision making tool for object recognition. In particular, FI was used to fuse the results of two neural network based classifiers in a handwritten character recognition problem. It was shown that the combined classification achieved 4% higher correct classification rate than the best of the individual classifiers. Application of FI to automatic target recognition gives 92.6% correctness vs. 90.9% by a Bayes classifier and 86.4% by a Dempster-Shafer classifier.

The FIs for classification purposes were also analyzed by Grabisch and Nicolas [3]. In particular, the problem of identifying the FMs was considered. Several methods of learning FMs from a training set were considered. They included a perceptron-like criterion minimizing the number of misclassified samples, a quadratic error-like criterion minimizing the difference between expected and actual outcomes, and a generalized quadratic criterion. The authors also derived the minimal number of training samples necessary to estimate correctly the fuzzy measures. Application of a number of different approaches to simulated and real-life data demonstrated the validity of the methods. The generalized quadratic criterion was found to give the best results. The authors emphasized that the problem of identifying fuzzy measures is crucial to the FI approach.

In [20] outcomes of several neural network classifiers were aggregated by the Sugeno fuzzy integral. Sugeno's λ-fuzzy measure was used. Calculation of FI with respect to λ-measure only requires knowledge of the so-called fuzzy densities (which in essence are FMs for single classifiers). Fuzzy densities can be interpreted as the degrees of importance of corresponding classifiers. FMs for subsets of classifiers can be calculated recursively[9]. The method was applied to handwritten character recognition. Fuzzy densities were assigned, based on how well the corresponding networks performed on the training set. It has been demonstrated that aggregating by FI increases recognition rates in comparison with individual networks, majority voting and Borda count methods applied to the same networks.

Tresp in [19] considered a linear combination of several estimators. The weights were proposed not to be constant but dependent on the input. Several methods of obtaining the weighted functions were considered. The method was applied to the Boston housing data set (13 inputs, one continuous output). The training set consisted of 170 samples and 20 classes obtained by *k*-mean clustering. Application of the proposed methods gave smaller errors than did individual networks.

Recently several papers on the bias-variance decomposition of misclassification rate have appeared [26,31,32]. Friedman in [31] investigated how an error in the target probability estimates affects classification error when these estimates are used in a classification rule. The bias/variance trade-off is very different for the classification error from the estimation error on the probabilities themselves. The dependence of the estimation error on bias and

[9] Solving an algebraic equation is required

variance is additive. However, there is a strong interaction effect in the classification error. Friedman introduced the notion of 'boundary bias'. "The effect of boundary bias on classification error can be mitigated by low variance. Similarly, the effect of variance depends on the value (especially the sign) of the boundary bias. Therefore, low variance can be very important for classification but low (estimation) bias is not. All that is required is that the boundary bias be negative. This being the case, one can reduce classification error toward its minimal (Bayes) value by reducing variance alone. In this sense variance tends to dominate the bias." This explains why some methods don't work properly for function estimation because of high bias, but perform well for a classification problem when the biased estimates of probabilities are used in the classification rule. Several examples are provided to demonstrate that misclassification error is not simply related to estimation error. The author concludes that "good probability estimates are not necessary for good classification; similarly, low classification error does not imply that the corresponding class probabilities are being estimated accurately."

1.1 Brief Description of Individual Classifiers

Fuzzy *K*-Nearest Neighbor Classifier

The KNN classifier has often been used in pattern recognition problems. The decision rule provides a simple nonparametric procedure to assign a class label to a sample, based on the labels of the K closest neighbors of the sample in the space of vectors characterizing the samples. A KNN classifier doesn't require training. The crisp KNN classifier assigns a sample to the majority class among K nearest neighbors. The fuzzy KNN classifier assigns class memberships to the sample.

Let $\underline{\mathbf{x}}_k$ be a vector characterizing the k-th sample of the training set, and u_{ck} the sample's membership in class c (c=1,...,m). Denote by $\underline{\mathbf{x}}$ the vector characterizing an unknown sample, and by $u_c(\underline{\mathbf{x}})$ its membership in class c. This membership is calculated as follows:

$$u_c(\underline{\mathbf{x}}) = \frac{\sum_{k=1}^{K} u_{ck} \cdot \left(1 \Big/ \left\| \underline{\mathbf{x}} - \underline{\mathbf{x}}_k \right\|^{2/(s-1)} \right)}{\sum_{k=1}^{K} \left(1 \Big/ \left\| \underline{\mathbf{x}} - \underline{\mathbf{x}}_k \right\|^{2/(s-1)} \right)},$$

A value of s=2 was used. The membership of the k-th sample of the training set in class c $u_{ck} = \begin{cases} 1 & \textit{for the true class of the sample} \\ 0 & \textit{otherwise} \end{cases}$ was chosen. Class label for the sample $\underline{\mathbf{x}}$ is assigned according to the maximal value among the memberships. The fuzzy KNN classifier was implemented in [38] according to [33].

Linear and Quadratic Discriminant Analysis

In discriminant analysis the spectra of the samples are assumed to be normally distributed about the mean (average) spectra of corresponding classes. This distribution is characterized by the mean vector and the covariance matrix:

$$\underline{\mu}_c = E(\underline{\mathbf{x}}) \qquad \Sigma_c = E\left((\underline{\mathbf{x}} - \underline{\mu}_c)(\underline{\mathbf{x}} - \underline{\mu}_c)'\right)$$

$\underline{\mu}_c$ is the mean vector and Σ_c is the covariance matrix for class c, the expectation $E(\cdot)$ is taken over all possible samples of class c, which are characterized by the observation vectors $\underline{\mathbf{x}}$. The mean vector and covariance matrix are usually unknown and have to be estimated from training set during training phase:

$$\underline{\mu}_c = \frac{1}{N_c}\sum_{k=1}^{N_c} \underline{\mathbf{x}}^{(k)} \qquad \Sigma_c = \frac{1}{N_c - 1}\sum_{k=1}^{N_c} (\underline{\mathbf{x}}^{(k)} - \underline{\mu}_c)(\underline{\mathbf{x}}^{(k)} - \underline{\mu}_c)'$$

index k goes through all N_c samples of class c of the training set. After the mean vectors and the covariance matrices are estimated, the probabilities of different classes are calculated[10]:

$$p_c(\underline{\mathbf{x}}) \cong \exp\left(-\tfrac{1}{2}(\underline{\mathbf{x}} - \underline{\mu}_c)\,\Sigma_c^{-1}(\underline{\mathbf{x}} - \underline{\mu}_c)'\right) \qquad (1.1.1)$$

The sample is assigned to the class corresponding to the maximal probability. If different classes have different covariance matrices and these are estimated separately, the method is called Quadratic Discriminant Analysis (QDA).

[10] The *a priori* probabilities for different classes are assumed to be same.

If the covariance matrix is assumed to be the same for all classes, then it is calculated as follows:

$$\Sigma = \sum_{c=1}^{m} \frac{N_c}{N} \Sigma_c$$

In this case the quadratic term $-\frac{1}{2}\underline{x}\Sigma^{-1}\underline{x}$ in the exponent in Eq. (1.1.1) becomes independent of c, and can be ignored, resulting in Linear Discriminant Analysis (LDA). LDA and QDA classifiers are implemented in [35].

Artificial Neural Network

The computer package NeuralWorks Professional II/PLUS [40] was used to create a Artificial Neural Network (ANN) classifier. An ANN is a self-adaptive learning system composed of layers of processing elements or neurons. Every neuron has several inputs and corresponding weights (or input connection strengths) and combines, usually by a summation, the values of the inputs into a value, which is then modified by a transfer function into an output value.

A back-propagation network is an ANN that propagates forward the input through the hidden layers to the output layer, determines the error at the output layer by comparing the actual and desired outcomes, and then propagates the errors back through the network to the input layer. The constructed back-propagation ANN classifier has 1 hidden layer with 5-10 nodes. The number of inputs depends on the dimensionality of the preprocessed spectra and

the number of outputs on the number of classes, Fig 1.4. The hyperbolic tangent function was used as the transfer function.

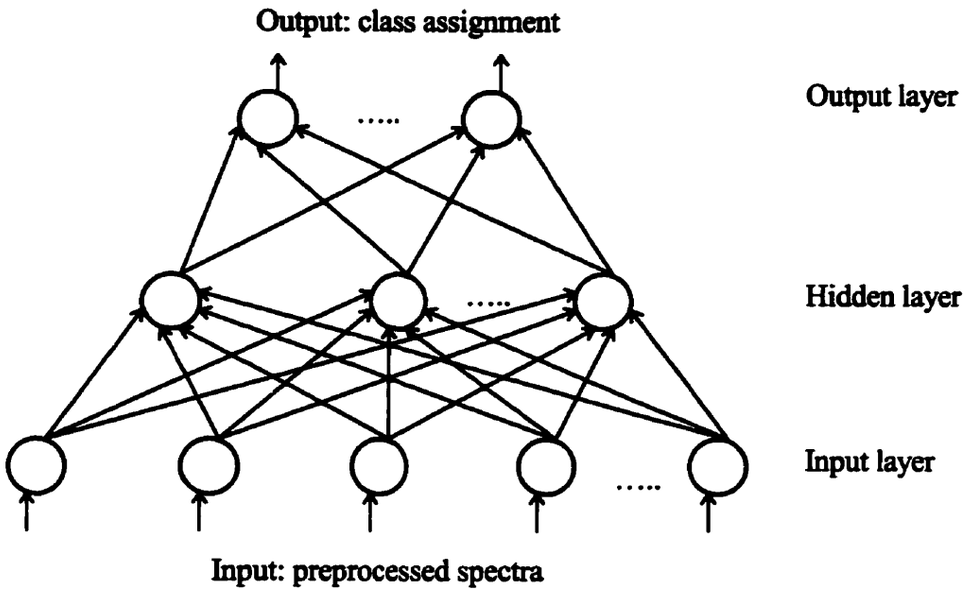

Fig. 1.4 The architecture of the ANN classifier used in this thesis. The classifier has one hidden layer with 5-10 nodes. The number of inputs is equal to the dimensionality of preprocessed spectra, the number of outputs to the number of classes.

A back-propagation ANN classifier learns by examples, i.e., the classifier self-adapts by modifying internal weights when samples with known class identity are presented to it. This process is called learning. The weights were updated after 1 epoch (or training cycle) i.e., after all training samples were presented once to the classifier (so-called cumulative delta learning rule). The error at the output layer was the root mean square error. Back-propagation network assumes no dependency between output values. However, in classification problems there exists such a dependency between outputs. A softmax activation function was used on the output layer in order to solve this problem [40]. The

constructed ANN classifier was trained for 50,000 epochs and then its performance evaluated using a test set.

1.2 Brief Description of Preprocessing Techniques

Spectra usually are sampled at 500-3000 points. The number of spectra available rarely exceeds 100 samples per class. Classifying such high-dimensional data with that small number of samples cannot always be performed or often gives unreliable results. Thus, the dimensionality of the spectra has to be reduced before classification. Several techniques to reduce the dimensionality of spectra were used in this thesis. They are now briefly described.

Condensing spectra. A spectrum is divided into contiguous subregions. The average (or median) amplitude of the spectrum is computed in each subregion. Considering these average/median amplitudes as new attributes reduces the dimensionality of the spectrum to the number of the subregions.

Selecting optimal regions by Dynamic Programming. Even after spectral condensation, the dimensionality may still be too high for some of the classifiers. The next logical step is to choose a subset of those regions that contribute most to discrimination among the classes. Depending on the number of regions one wants to select and the total number of regions in the spectra this can be a very time consuming procedure. The following near-optimal procedure was used to choose such a subset [35]. Suppose for the sake of argument that the total number of regions in each spectrum is 100, and 10 regions are to be selected. First, 100 lists are created, each containing one attribute, the average (or median) amplitude of spectrum in the corresponding region, Fig. 1.5. Then each of 99 unused attributes is added

one at the time and LDA classification of the training set is performed. The attribute giving the best classification is added to the considered list. The same procedure is applied to every list, resulting in 100 lists now containing 2 attributes. The procedure is repeated until the required number of attributes is selected. Then the list giving the best classification accuracy is selected among the 100 lists. Thus, the 100-dimensional condensed spectra are transformed into a set of 10-dimensional attributes.

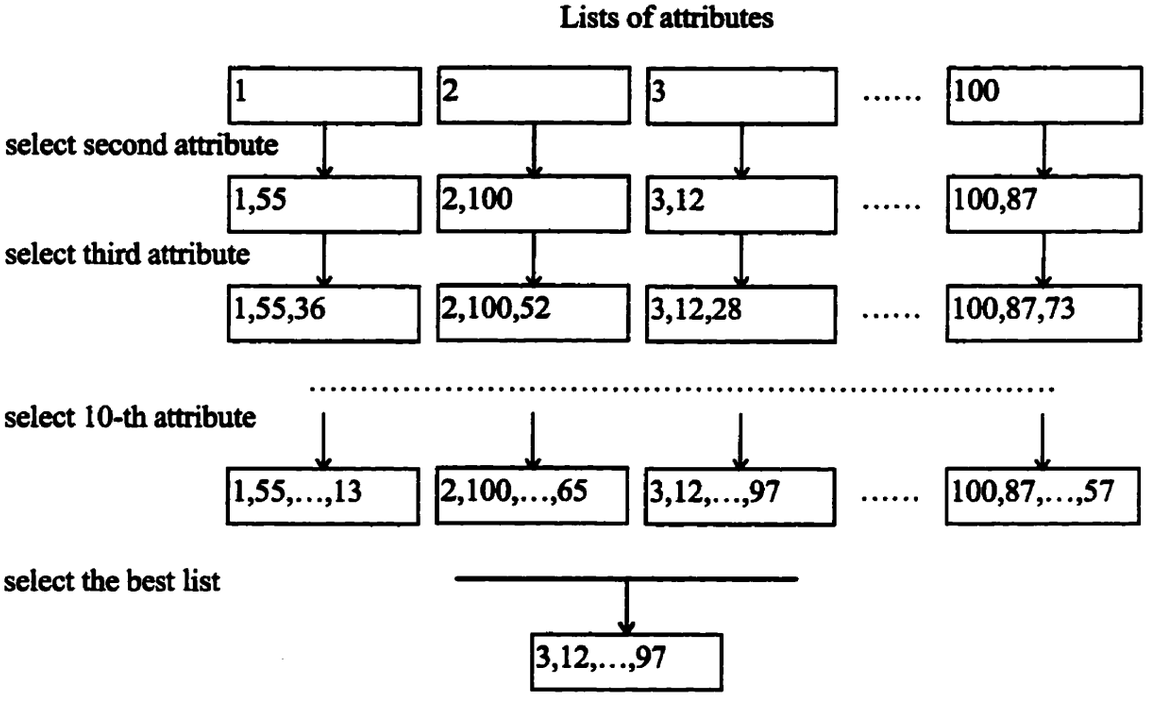

Fig. 1.5 An example of selecting 10 out of 100 regions in spectra contributing optimally to discrimination among classes. (See text for details)

Selecting optimal regions by Genetic Algorithm (GA). Instead of selecting a subset of regions in the list of contiguous subregions one can optimize the boundaries of the desired number of subregions in order to find the regions in spectra which are maximally

discriminatory. The following procedure was used [35]. For a preselected number of regions the boundaries were randomly changed by GA, the average amplitudes of spectra in these regions were calculated and used as attributes in an LDA classifier. This process was repeated until the MSE between the desired membership values (1/0) and the ones obtained by LDA become small for the training set. The best regions found during optimization were saved. An example of region selection by GA is shown in Fig. 1.6. The two centroid spectra of two different classes and the regions selected by GA are shown.

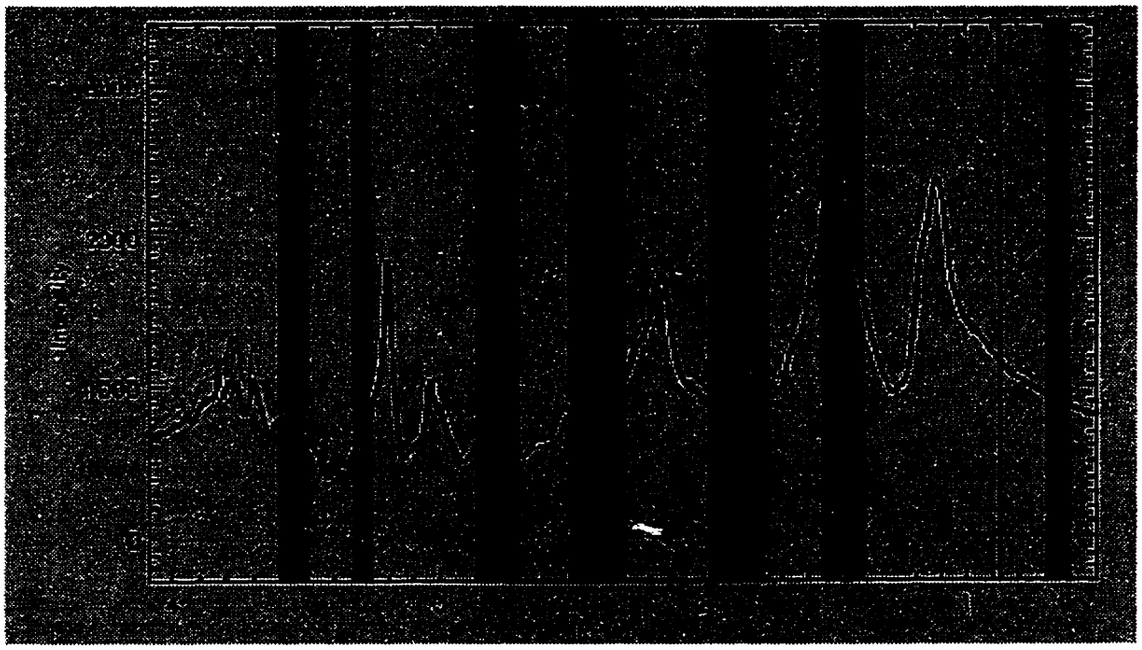

Fig. 1.6 Application of Genetic Algorithm to selecting a given number of regions in the spectra responsible for maximally discrimination. Average amplitudes of spectra in these regions give the best classification accuracy on the training set. In this example the centroid spectra of two classes (low and high grade astrocytomas) as well as the regions selected by GA are shown. The difference between the two centroid spectra is also shown at the bottom. This figure was kindly provided by A. Nikulin.

2. Aggregation Methods

We now describe the aggregation methods used.

2.1 Logistic Regression.

One of the ways to aggregate several classifiers is to calculate a weighted sum of their outcomes. The weights are constant and express the relative importance of the classifiers. The values of the weights are estimated from how well the combining classifier performs on the training set. This approach requires estimating of a small number of parameters (i.e., number of aggregated classifiers plus one) compared to some of the other methods considered later.

Suppose a sample (or an observation) has been submitted to n classifiers C_1, C_2, ... C_n. Their outcomes are the degrees of confidence $\underline{\mathbf{x}}_c=\{x_{1c}, x_{2c}, ..., x_{nc}\}$ that this sample belongs to class c. Stated differently, we obtain a vector of class c memberships, assigned by the classifiers. If m is the total number of classes, we obtain m such vectors for every submitted sample. The following aggregation function is proposed for every $\underline{\mathbf{x}}_c$:

$$f(\underline{\mathbf{x}}_c, \underline{\mathbf{w}}) = w_0 + w_1 x_{1c} + w_2 x_{2c} + \ldots + w_n x_{nc} \qquad (2.1.1)$$

where w_1, ..., w_n are constant weights, w_0 is a bias. Given a sample, the value of $f(\underline{\mathbf{x}}_c,\underline{\mathbf{w}})$ is calculated for each class c. $f(\underline{\mathbf{x}}_c,\underline{\mathbf{w}})$ represents the degree of confidence of the combining classifier that the sample belongs to class c. Notice, that the weights are the same for every class (i.e., independent of c).

Suppose $t(\underline{\mathbf{x}}_c)$ is a binary value associated with each sample. $t(\underline{\mathbf{x}}_c) = 1$ if c is the true class of the sample, and $t(\underline{\mathbf{x}}_c) = 0$ otherwise[11]. The expected value of $t(\underline{\mathbf{x}}_c)$ is

$$E(t(\underline{\mathbf{x}}_c)) = 1 \cdot P(t(\underline{\mathbf{x}}_c) = 1) + 0 \cdot P(t(\underline{\mathbf{x}}_c) = 0) = P(t(\underline{\mathbf{x}}_c) = 1) \equiv \pi(\underline{\mathbf{x}}_c)$$

One expects that the likelihood of class c being the true class (i.e., $\pi(\underline{\mathbf{x}}_c)$) is greater when it is ranked higher by the combining classifier. The relationship between the degree of confidence and the tendency of being the true class is expected to be a monotonic function. Also one expects that $\pi(\underline{\mathbf{x}}_c) \to 0$ when the components of the vector $\underline{\mathbf{x}}_c$ are small, and $\pi(\underline{\mathbf{x}}_c) \to 1$ when they are large. Suppose the function $\pi(\cdot)$ has the following form

$$\pi(\underline{\mathbf{x}}_c, \underline{\mathbf{w}}) = \frac{\exp(f(\underline{\mathbf{x}}_c, \underline{\mathbf{w}}))}{1 + \exp(f(\underline{\mathbf{x}}_c, \underline{\mathbf{w}}))} \qquad (2.1.2)$$

In fact, this transformation converts the degree of confidence (2.1.1) into the range [0,1].

A training set is used to estimate the values of the parameters $w_0, w_1, \ldots, w_n$. The weights should be chosen such that the performance of the combining classifier is optimized. There are several ways to measure the performance. One is to consider the outcomes of the combining classifier as memberships in the corresponding classes, and try to make them as close as possible to the desired one/zero values for all samples of the training set. For the k-th sample[12] of the training set the error is

[11] in fact $t(\underline{\mathbf{x}}_c)$ depends on the class c only, not on $\underline{\mathbf{x}}_c$. For convenience we use the notation $t(\underline{\mathbf{x}}_c)$

[12] index k will appear in vector $\underline{\mathbf{x}}_c$ to denote the k-th sample.

$$\delta^2(k,\underline{\mathbf{w}})=\sum_{c=1}^{m}\left[t(\underline{\mathbf{x}}_c^{(k)})-\pi(\underline{\mathbf{x}}_c^{(k)},\underline{\mathbf{w}})\right]^2$$

Calculating the error over all training samples we get the MSE on the training set, which we are going to minimize with respect to the vector $\underline{\mathbf{w}}$

$$MSE\left(\delta^2(\underline{\mathbf{w}})\right)=\frac{1}{N}\sum_{k=1}^{N}\sum_{c=1}^{m}\left[t(\underline{\mathbf{x}}_c^{(k)})-\pi(\underline{\mathbf{x}}_c^{(k)},\underline{\mathbf{w}})\right]^2 \rightarrow \text{minimize} \qquad (2.1.3)$$

N is the total number of training samples. This is an unweighted MSE, because all classes have the same (unit) weight. There is another way to calculate the MSE, taking into account the number of samples/classes in the training set. It is a weighted MSE :

$$MSE\left(\delta^2(\underline{\mathbf{w}})\right)=\frac{1}{m}\sum_{c=1}^{m}\frac{1}{N_c}\sum_{k=1}^{N_c}\left[t(\underline{\mathbf{x}}_c^{(k)})-\pi(\underline{\mathbf{x}}_c^{(k)},\underline{\mathbf{w}})\right]^2 \rightarrow \text{minimize} \qquad (2.1.4)$$

N_c is the number of training samples of class c.

The MSE on the training set measures the estimation error of target probabilities rather than the classification error. The minimization of the MSE does not necessarily lead to the maximization of the classification performance (in terms of the number of correctly classified samples).

If the performance of the combining classifier is defined as the number of correctly classified samples of the training set (i.e., we minimize the number of misclassified samples), the function to be minimized is not continuous, but piece-wise constant. This

means that many vectors $\underline{\mathbf{w}}$ give the same performance for the combining classifier. The function to minimize is:

$$N_{misc}(\underline{\mathbf{w}}) = \sum_{k=1}^{N} (1 - t(\mathbf{x}_s^{(k)})) \rightarrow \text{minimize}, \tag{2.1.5}$$

where s denotes the class corresponding to $\max_c(\pi(\mathbf{x}_c^{(k)}, \underline{\mathbf{w}}))$. However, the results of such classification are often fuzzy (i.e., such a classifier is unable to distinguish between good and poor solutions). Classification of the sample is considered fuzzy if the membership in the predicted class is smaller than $(m+1)/2m$, where m is the number of classes. Otherwise it is considered crisp.

A slight modification of the objective function (2.1.5) might improve the crispness of classification. Instead of minimizing the number of misclassified samples one can minimize the number of misclassified and fuzzily classified samples together.

One can also combine the previously considered criteria, obtaining the weighted sum of the MSE on the training set (2.1.3) or (2.1.4) and the number of misclassified samples (2.1.5)

$$F(\underline{\mathbf{w}}) = \frac{1}{N} \sum_{k=1}^{N} \sum_{c=1}^{m} \left[t(\underline{\mathbf{x}}_c^{(k)}) - \pi(\underline{\mathbf{x}}_c^{(k)}, \underline{\mathbf{w}}) \right]^2 + \beta \cdot \sum_{k=1}^{N} (1 - t(\mathbf{x}_s^{(k)})) \tag{2.1.6}$$

Here β is a parameter, possibly to be optimized.

As was mentioned earlier, for every observation of the training set m different values of (2.1.2) are computed. The class corresponding to the maximal one is considered as the most likely class for this observation. Now let us not just choose the maximal value, but also maximize the squared difference between the $\pi(\cdot)$ corresponding to the true class of the sample and that of every other class

$$F(\underline{\mathbf{w}}) = \sum_{k=1}^{N} \sum_{c \neq s} \left(\pi(\underline{\mathbf{x}}_s^{(k)}, \underline{\mathbf{w}}) - \pi(\underline{\mathbf{x}}_c^{(k)}, \underline{\mathbf{w}}) \right)^2 \rightarrow \text{maximize} \qquad (2.1.7)$$

Here index s in $\underline{\mathbf{x}}_s$ denotes the true class of the sample. This way one is able not only to choose a solution, but also to discriminate between unequivocal and ambiguous solutions. Taking into account that $0 \leq \pi(\cdot) \leq 1$ and that the desired values are unity for the true class of the sample and zero for all others, this problem can easily be converted into a minimization problem :

$$F(\underline{\mathbf{w}}) = \sum_{k=1}^{N} \sum_{c \neq s} \left(\pi(\underline{\mathbf{x}}_s^{(k)}, \underline{\mathbf{w}}) - \pi(\underline{\mathbf{x}}_c^{(k)}, \underline{\mathbf{w}}) - 1 \right)^2 \rightarrow \text{minimize} \qquad (2.1.8)$$

Consider a generalized version of the above criterion. Suppose one has an increasing function, for instance a sigmoid-type function

$$\Psi(t) = \frac{1 - e^{-Kt}}{1 + e^{-Kt}}, \quad K > 0 \qquad (2.1.9)$$

Applying the function $\Psi(\cdot)$ to the difference in expression (2.1.7) can enhance the discrimination between good and bad classifications. I applied the function $\Psi(\cdot)$ not to the

difference of $\pi(\cdot)$s, but to the difference of $f(\cdot)$s in (2.1.1) instead. The larger the difference, the closer $\Psi(\cdot)$ to unity. Then the function to be minimized takes the following form:

$$F(\underline{\mathbf{w}}) = \frac{1}{N}\sum_{k=1}^{N}\sum_{c\neq s}^{m}\left[\Psi\left(f(\mathbf{x}_s^{(k)},\underline{\mathbf{w}}) - f(\mathbf{x}_c^{(k)},\underline{\mathbf{w}})\right) - 1\right]^2 \rightarrow \text{minimize} \qquad (2.1.10)$$

Notice that for the objective function (2.1.10) the bias w_0 in (2.1.1) is not required.

Finally, similarly to the hybrid criterion (2.1.6), one can combine (2.1.10) and (2.1.5):

$$F(\underline{\mathbf{w}}) = \frac{1}{N}\sum_{k=1}^{N}\sum_{c\neq s}^{m}\left[\Psi\left(f(\mathbf{x}_s^{(k)},\underline{\mathbf{w}}) - f(\mathbf{x}_c^{(k)},\underline{\mathbf{w}})\right) - 1\right]^2 + \beta\cdot\sum_{k=1}^{N}\left(1 - t(\mathbf{x}_s^{(k)})\right) \rightarrow \text{minim.} \quad (2.1.11)$$

By minimizing the MSE, one is trying to raise the degree of confidence to unity that the sample belongs to the true class, and make it zero for all other classes. Actually, if the degree of confidence is 0.9 instead of 1.0, for instance (and others are still close to zero), the sample is still classified correctly and crisply. In other words, the classifier became worse with regard to the estimation error, but in terms of classification error it as good as the original one. This example shows that the classification error responds to the error in the underlying membership estimates differently than does the estimation error. It also helps to understand why the improvement of the latter does not necessarily lead to the improvement of former. Indeed, if the class probability estimates are 0.9 for the true class and about zero for others, their improvement to unity and zero doesn't change classification performance at all. The following objective function is worth trying:

$$F(\underline{\mathbf{w}}) = \frac{1}{N}\sum_{k=1}^{N}\sum_{c=1}^{m}\left[y(\underline{\mathbf{x}}_c^{(k)},\underline{\mathbf{w}})\right]^2 + \beta\cdot\sum_{k=1}^{N}\left(1 - t(\mathbf{x}_s^{(k)})\right) \rightarrow \text{minimize} \qquad (2.1.12)$$

where $$y(\underline{\mathbf{x}}_c^{(k)},\underline{\mathbf{w}}) = \begin{cases} t(\underline{\mathbf{x}}_c^{(k)}) - \pi(\underline{\mathbf{x}}_c^{(k)},\underline{\mathbf{w}}) & \text{if } |t(\cdot) - \pi(\cdot)| \geq \textit{threshold} \\ 0 \quad \textit{otherwise} \end{cases}$$

β is a weight and index s denotes the class corresponding to $\max_c(\pi(\mathbf{x}_c^{(k)},\underline{\mathbf{w}}))$. The threshold is an additional parameter. The first term of (2.1.12) keeps the class memberships in the necessary range and the second term helps minimize the number of misclassified samples.

After the weights $\underline{\mathbf{w}}$ are estimated one can apply the algorithm to a test set. For a sample the algorithm has not seen before the values of the function $\pi(\underline{\mathbf{x}}_c,\underline{\mathbf{w}})$ are calculated for every class c. The sample is assigned to the class with the largest value of $\pi(\underline{\mathbf{x}}_c,\underline{\mathbf{w}})$.

In order to estimate the unknown vector $\underline{\mathbf{w}}$, the different objective functions above were minimized by a simulated annealing simplex procedure [17]. This procedure tries to find the global minimum of a function of many variables. On a number of tests (in particular, functions with enormous number of minima) the above procedure was able to find either a global minimum or one of the deepest minima.

What is the idea of the simulated annealing simplex minimization? The ordinary simplex moves in the space of variables controlled by function values at the simplex vertices. In annealing simplex minimization some positive random values which depend on the control parameter (analog of temperature) are added to the function values. These "temperature-boosted" values are now responsible for the simplex movement and allow the simplex

method to escape from local minima[13]. While traversing the space of variables the simplex algorithm retains the best minima it has found. Then the "temperature" of the system is reduced slightly, and the simplex continues its trip from the best found minimum, etc. Of course there is no guarantee that a global minimum will ever be found. But in general this is not required. The most important thing is to find a 'good enough' minimum, and not get trapped in local minima. Notice, that in order to achieve good results a proper annealing schedule is very important. Furthermore, some internal parameters of the minimization procedure, such as the initial temperature of the system and the initial size of simplex have to be set properly. This minimization procedure can be quite time consuming, depending on the dimensionality of the problem and complexity of the objective function. The Logistic Regression (LR) classifier is implemented in [37].

2.2 Linear Combination of Classifiers

Keeping the previous notation, again consider the task of aggregating the outcomes of n classifiers for an m-class problem. Consider the vector $\underline{\mathbf{x}}_c=\{x_{1c}, x_{2c}, ..., x_{nc}\}$, with element x_{ic}, the membership in class c assigned by the i-th classifier. Compute a linear combination of these elements:

$$f(\underline{\mathbf{x}}_c, \underline{\mathbf{a}}_c) = a_{1c}x_{1c} + a_{2c}x_{2c} + ... + a_{nc}x_{nc} \quad = \quad \underline{\mathbf{a}}_c^t \cdot \underline{\mathbf{x}}_c \qquad c = 1,..., m \qquad (2.2.1)$$

The coefficient a_{ic} reflects the importance of the i-th classifier for class c in the combining classifier. Notice, that unlike LR, where the weights are the same for all classes, in (2.2.1)

[13] the temperature has to be sufficiently high.

they are different for different classes. In order to classify an unknown sample, the optimal values of the parameters a_{ic} have to be estimated first. Consider $f(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)$ as the degree of confidence assigned by the combining classifier that the sample belongs to class *c*. We would like to make this confidence value as close as possible to the desired confidence value $t(\underline{\mathbf{x}}_c)$. Regard $\underline{\mathbf{x}}_c$ as a random variable. The squared error in estimating class *c* membership is[14]

$$\delta^2(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)=\left(t(\underline{\mathbf{x}}_c)-f(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)\right)^2$$

The MSE can be obtained by taking expectation over all possible $\underline{\mathbf{x}}_c$:

$$MSE\left(\delta^2(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)\right)=E\left(\delta^2(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)\right) \text{ or}$$

$$MSE\left(\delta^2(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)\right)=\mathrm{E}\left((t(\underline{\mathbf{x}}_c)-\underline{\mathbf{a}}_c^t\cdot\underline{\mathbf{x}}_c)^2\right)=\mathrm{E}\left(t^2(\underline{\mathbf{x}}_c)-2t(\underline{\mathbf{x}}_c)\underline{\mathbf{a}}_c^t\cdot\underline{\mathbf{x}}_c+(\underline{\mathbf{a}}_c^t\cdot\underline{\mathbf{x}}_c)^2\right)=$$

$$=\mathrm{E}\left(t^2(\underline{\mathbf{x}}_c)\right)-2\underline{\mathbf{a}}_c^t\cdot\underline{\theta}+\underline{\mathbf{a}}_c^t\cdot\Phi\cdot\underline{\mathbf{a}}_c$$

where $\underline{\theta}=E\left(t(\underline{\mathbf{x}}_c)\underline{\mathbf{x}}_c\right)$, $\Phi=E(\underline{\mathbf{x}}_c\cdot\underline{\mathbf{x}}_c^t)$. In order to calculate the optimal values of the weight vector $\underline{\mathbf{a}}_c$ we take the derivative with respect to $\underline{\mathbf{a}}_c$ and set it equal to zero:

$$\nabla_{\underline{\mathbf{a}}_c}\left\{MSE(\delta^2(\underline{\mathbf{x}}_c,\underline{\mathbf{a}}_c)\right\}=\underline{\theta}-\Phi\cdot\underline{\mathbf{a}}_c=0$$

Then $\underline{\mathbf{a}}_c=\Phi^{-1}\cdot\underline{\theta}$. One can perform this procedure for all classes and get *m* different vectors $\underline{\mathbf{a}}_c$. The obtained values of the parameters guarantee that within the considered linear

[14] error over all classes is the sum of the errors for each class

model the combining classifier gives a result not worse than that of the best individual classifier in the sense of the MSE between the obtained and desired outcomes.

In practice, the distribution function of $\underline{\mathbf{x}}_c$ is usually unknown, and one can't calculate the expectations above. Nevertheless, if the samples of the training set are considered independent, one can estimate the vector $\underline{\boldsymbol{\theta}}$ and matrix $\boldsymbol{\Phi}$ as follows:

$$\phi_{ij} = \frac{1}{N}\sum_{k=1}^{N} x_{ic}^{(k)} x_{jc}^{(k)} \qquad\qquad \theta_i = \frac{1}{N}\sum_{k=1}^{N} t(\mathbf{x}^{(k)}) x_{jc}^{(k)}$$

Here indices i and j go through all classifiers, index k goes through all training samples, c is the considered class, and N is the total number of the training samples.

The advantage of this method lies in the simplicity of estimating the values of the parameters $\underline{\mathbf{a}}_c$. It requires only the inversion of a few matrices of small dimension (the number of the individual classifiers), a fast and well developed procedure. We used the singular value decomposition method to invert the matrices [18].

After all coefficients $\underline{\mathbf{a}}_c$ are calculated, for any sample of the test set one can calculate the values of $f(\underline{\mathbf{x}}_c, \underline{\mathbf{a}}_c)$ for every class using the proper vector $\underline{\mathbf{a}}_c$ and the memberships $\underline{\mathbf{x}}_c$ supplied by the individual classifiers. The sample is assigned to the class corresponding to the maximal value of $f(\underline{\mathbf{x}}_c, \underline{\mathbf{a}}_c)$. The Linear Combination (LC) classifier is implemented in [37].

2.3 Entropy Classifier

Again n classifiers, applied to an m-class problem, are to be aggregated. Denote the membership in class c, assigned by the i-th classifier for the sample k, as $x_{ic}^{(k)}$. For every sample calculate the confidence value that the sample belongs to class c:

$$H(c,k) = \frac{1}{n}\sum_{i=1}^{n} \frac{x_{ic}^{(k)}}{-E(i,k)} \qquad (2.3.1)$$

The sum is taken over all individual classifiers. The weight $-1/E(i,k)$ reflects the importance of i-th classifier in the combining classifier. For individual classifiers, strongly suggesting some particular class, this weight is supposed to be larger than for a less favorable classifier.

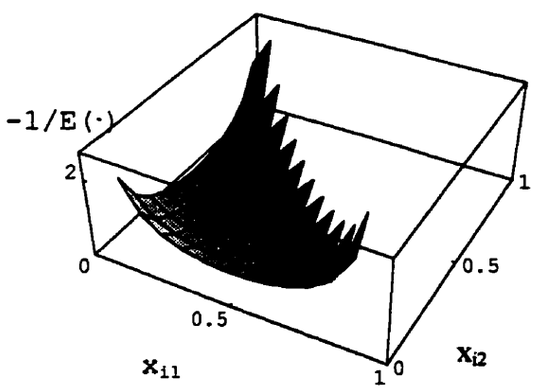

Fig. 2.1 The dependence of the weight $-1/E(\cdot)$ for an individual classifier on the membership values x_{ic}, assigned by this classifier. Case of a 3-class problem. Individual classifier assigns the following membership values to the classes: x_{i1}, x_{i2}, and $(1-x_{i1}-x_{i2})$.

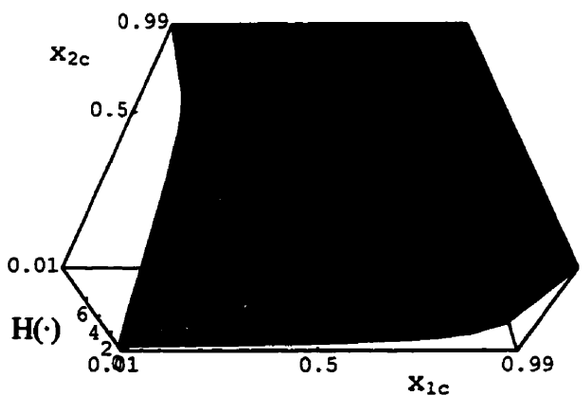

Fig. 2.2 The case of aggregating two classifiers for a 2-class problem. The dependence of the confidence value $H(\cdot)$, assigned by the combining classifier, on the memberships x_{1c} and x_{2c} provided by individual classifiers.

The coefficient $E(i,k)$ is defined as follows:

$$E(i,k) = \sum_{c=1}^{m} x_{ic}^{(k)} \cdot \ln(x_{ic}^{(k)})$$

and is called the entropy. Notice, that the sum is taken over the classes. Several simple examples demonstrate how the confidence value assigned by the entropy based combining classifier, and the weight for individual classifier depend on the memberships supplied by the individual classifier. In Fig. 2.1 the behavior of the weight $-1/E(\cdot)$ in (2.3.1) with respect to the memberships x_{1c} for a 3-class problem is shown. If the outcome of the individual classifier is close to unity for one class (hence close to zero for the other classes) the weight for this classifier in the combining classifier increases exponentially. Some constraints on the value of the weight are introduced in order to avoid such situations. This is done by not allowing the membership values to be greater than $(1-p)$ and smaller than p,

$$\text{i.e., } x_{ic} = \begin{cases} p & \text{if } x_{ic} < p \\ x_{ic} & \text{if } p \le x_{ic} \le (1-p) \\ 1-p & \text{if } x_{ic} > (1-p) \end{cases}$$

In the calculations a value of p=0.05 was chosen.

The dependence of the confidence value $H(\cdot)$ on the memberships x_{ic} is presented in Fig. 2.2 when aggregating two individual classifiers, applied to a 2-class problem. One observes, that the confidence value $H(\cdot)$ doesn't change much for most values of x_{1c} and x_{2c} and increases when at least one individual classifier strongly suggests some particular class.

Any sample is classified by assigning it to the class with the largest confidence value. Notice, that this approach doesn't have any adjustable parameters, i.e., no training is required. The Entropy (ENT) classifier is implemented in [37].

2.4 Confidence Factor Classifier

This approach originates from the techniques used in expert systems. First, the memberships assigned by the individual classifiers are mapped into a Confidence Factor (CF) space by the following transformation:

$$CF_{ic}^{(k)} = \log_n\left((n-\frac{1}{n})x_{ic}^{(k)} + \frac{1}{n}\right)$$

The previous notation is used. For the considered sample the CFs corresponding to the memberships assigned by individual classifiers are aggregated for every class, resulting in c CFs. The class corresponding to the maximal value among these CFs is considered as the most likely class for the considered sample.

The following rule is used to aggregate CFs:

$$CF(a,b) = \begin{cases} 1-(1-a)(1-b) & \text{if } a>0, b>0 \\ -CF(-a,-b) & \text{if } a<0, b<0 \\ a+b, & \text{otherwise} \end{cases}$$

Positive and negative CFs are aggregated separately, the resultant positive and negative CFs are aggregated at the final step.

In Fig. 2.3 the nonlinear mapping of the membership value into CF is presented for the case of 3 classifiers as an example. Another example of how 2 CFs are aggregated

is shown in Fig. 2.4. The CF classifier is implemented in [37].

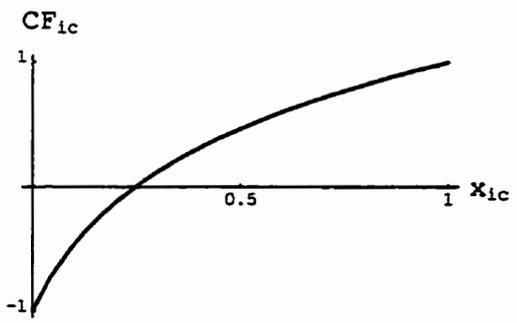

Fig. 2.3 Mapping of the membership x_{ic} into the confidence factor CF_{ic} for the case of 3 classifiers.

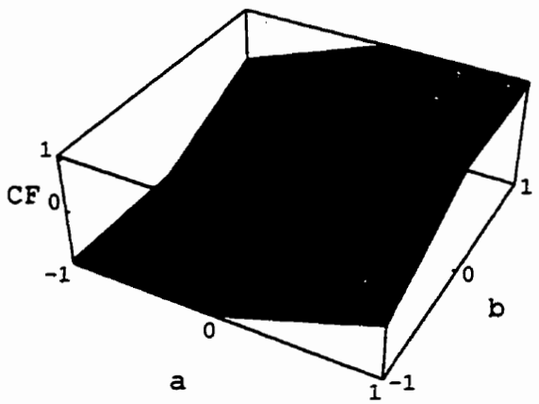

Fig. 2.4 The confidence factor aggregation rule. Confidence factors *a* and *b* are aggregated into confidence factor CF.

2.5 Fuzzy Integral Classifier

The Fuzzy Integral (FI) is a nonlinear approach to aggregating multiple sources of uncertain information. Before considering how the outcomes of different classifiers can be aggregated by FI, some definitions are introduced, following [2,3,16,21]. Consider the case of finite spaces. A Fuzzy Measure (FM) over a set X is a function

$$\mu: 2^X \rightarrow [0,1]$$

such that

- $\mu(\varnothing)=0$, $\mu(X)=1$
- $\mu(B) \geq \mu(A)$ if $B \supseteq A$

(2^X is the family of all subsets of X, including the empty set $\varnothing$)

Let X be a set of n information sources (e.g., classifiers) $X=\{C_1, C_2, ..., C_n\}$. The values $\mu(\{C_i\})$ ($i=1, ..., n$) are called fuzzy densities. These densities can be interpreted to represent

the importance of the individual classifiers toward answering a particular question (such as class membership). The FM of a subset A of classifiers is interpreted as the importance of that subset.

Let $f(\cdot)$ be a function from X to [0,1], and $\mu(\cdot)$ a FM on X. The Sugeno FI of function $f(\cdot)$ with respect to FM $\mu(\cdot)$ is defined by

$$(S)\int f \circ \mu = \bigvee_{i=1}^{n} \left(f(C_i) \wedge \mu(A_i) \right) \tag{2.5.1}$$

where the function values are supposed to be sorted $0 \le f(C_1) \le f(C_2) \le \ldots \le f(C_n) \le 1$, and $A_i \triangleq \{C_i, C_{i+1}, \ldots, C_n\}$. $\vee$ and $\wedge$ denote maximum and minimum operators, respectively. The sorting reduces the number of subsets required to evaluate the FI from 2^n to n.

The Choquet FI of $f(\cdot)$ with respect to $\mu(\cdot)$ is defined by

$$(C)\int f \circ \mu = \sum_{i=1}^{n} \left(f(C_i) - f(C_{i-1}) \right) \mu(A_i) \tag{2.5.2}$$

with the same assumptions as before, and $f(C_0)=0$.

The generalized FI of $f(\cdot)$ with respect to $\mu(\cdot)$ is defined by

$$(G)\int f \circ \mu = \bigvee_{i=1}^{n} \left(f(C_i) \, t \, \mu(A_i) \right) \tag{2.5.3}$$

with the same assumptions as before. Operator t is a t-norm, the function of two arguments

$t: [0,1]^2 \rightarrow [0,1]$

such that

- $x\,t\,w \le y\,t\,w$ for $x \le y$, $w \le z$
- $x\,t\,y = y\,t\,x$
- $(x\,t\,y)\,t\,z = x\,t\,(y\,t\,z)$
- $x\,t\,0 = 0, \quad x\,t\,1 = x$

$x,y,z,w \in [0,1]$

Think of x as the membership in some class provided by the individual classifier, and y as the importance of this classifier in the combining classifier. The job of the t-norm is to calculate the degree of confidence that the sample belongs to the considered class, based on the corresponding outcome of the individual classifier and on the importance of this classifier. Several examples of t-norms used in this thesis are presented below and in Fig 2.5.

$$x\,t\,y = \min(x,y) = x \wedge y \tag{2.5.4}$$

$$x\,t\,y = xy \tag{2.5.5}$$

$$x\,t\,y = 1 - \min\left[1, \left[(1-x)^p + (1-y)^p\right]^{1/p}\right], \quad p > 0 \tag{2.5.6}$$

$$x\,t\,y = \frac{xy}{\gamma + (1-\gamma)(x+y-xy)}, \quad \gamma \ge 0 \tag{2.5.7}$$

$$x\,t\,y = \log_w\left[1 + \frac{(w^x - 1)(w^y - 1)}{w-1}\right], \quad 0 < w < \infty, \quad w \ne 1 \tag{2.5.8}$$

$$x\,t\,y = \max\left[0, (\lambda+1)(x+y-1) - \lambda xy\right], \quad \lambda \ge -1 \tag{2.5.9}$$

Let us return to the combining classifier. The n individual classifiers each classify a sample. For each class c (c=1, ..., m) they provide n memberships $\underline{x}_c = \{x_{1c}, x_{2c}, \ldots, x_{nc}\}$ in this class. Suppose the FMs for all classes for all subsets A_i of the individual classifiers are

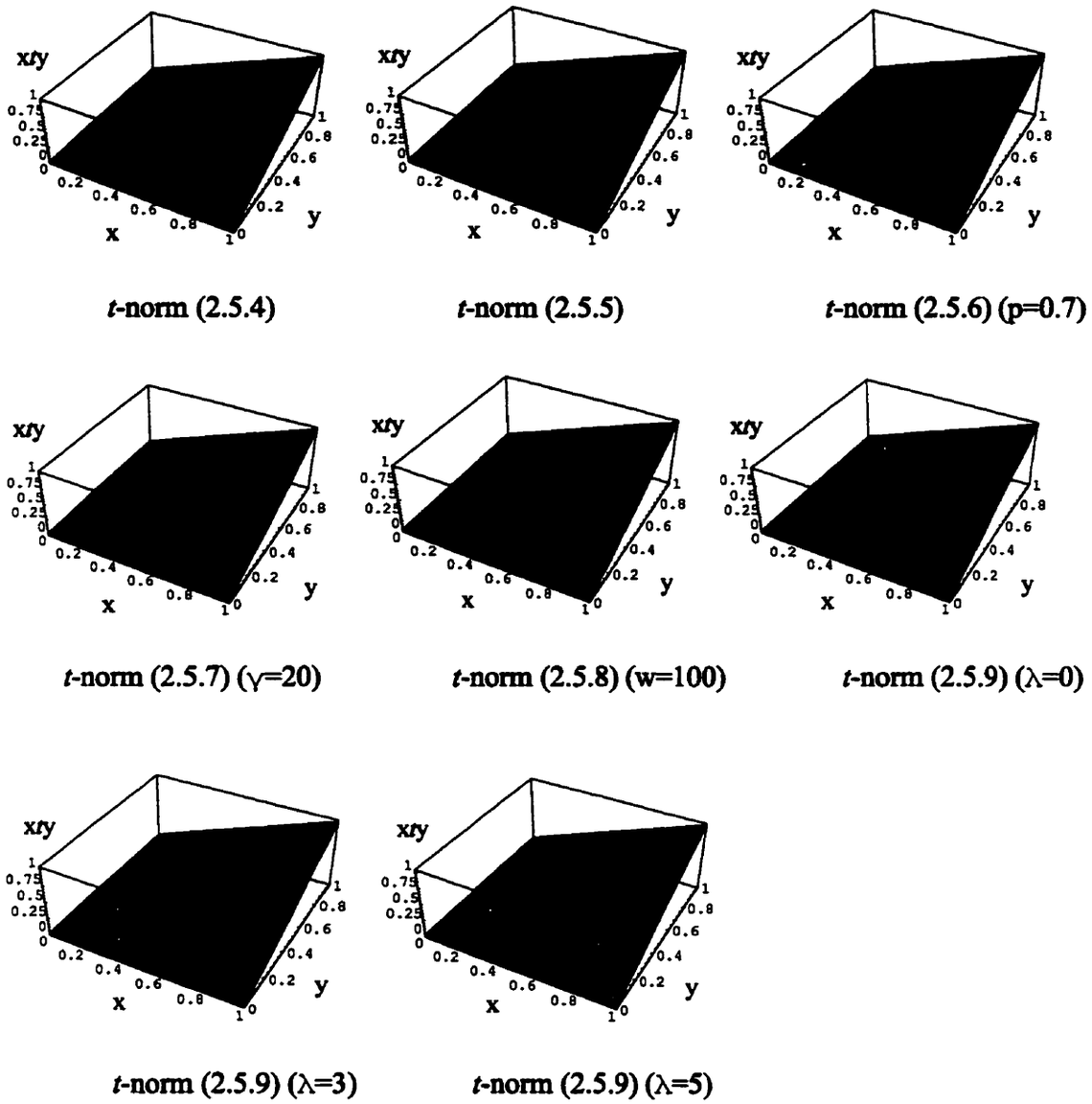

Fig. 2.5 Several examples of *t*-norms used in this thesis. **x** is the membership in some class provided by an individual classifier, **y** is the importance of this classifier, **x*y** is the degree of confidence that the sample belongs to the considered class, taking into account both factors **x** and **y**.

known. In other words, the importance of every individual classifier, their pairs, triplets, etc., are known. The memberships are aggregated by the FI as follows:

$$\delta^2(k,\underline{\mu})=\sum_{c=1}^{m}\left[t(\underline{\mathbf{x}}_c^{(k)})-F(\underline{\mathbf{x}}_c^{(k)},\underline{\mu}_c)\right]^2,$$

where $t(\underline{\mathbf{x}}_c^{(k)})$ is equal to unity for the true class of the sample, and to zero otherwise, m is the number of classes, $\underline{\mu}_c$ is the set of FMs for class c. This error depends on all FMs for all classes, denoted by $\underline{\mu}$. Calculating the error over all training samples results in the MSE on the training set, which should be minimized with respect to the FMs $\underline{\mu}$:

$$MSE\left(\delta^2(\underline{\mu})\right)=\frac{1}{N}\sum_{k=1}^{N}\sum_{c=1}^{m}\left[t(\underline{\mathbf{x}}_c^{(k)})-F(\underline{\mathbf{x}}_c^{(k)},\underline{\mu}_c)\right]^2 \rightarrow \text{minimize} \qquad (2.5.10)$$

where N is the total number of the training samples. Eq. (2.5.10) is similar to the Eq. (2.1.3) for the LR. The difference is in the number of estimated parameters. In the FI approach there are $m(2^n\text{-}2)$ FMs[15] instead of the n+1 weights in LR. This exponentially growing number of parameters to be estimated restricts the FI approach to the aggregation of a relatively small number of the individual classifiers.

One can minimize the weighted instead of the unweighted MSE on the training set. In this case the objective function is similar Eq. (2.1.4), with the replacement of the function $\pi(\cdot)$ by the FI $F(\cdot)$. Minimizing the number of misclassified samples of the training set (the objective function is similar to Eq. (2.1.5)) is another way of estimating the FMs. The weighted sum of the MSE on the training set (2.5.10) and the number of misclassified samples of the training set is another objective function (similar to Eq. (2.1.6)). One can

[15] Two FMs, for the empty and complete sets of the individual classifiers, are obvious in every class. Their values are equal to zero and unity respectively.

also maximize the difference between the FI for the true class of the sample and that for every other class, as was done for LR in Eq. (2.1.7):

$$G(\underline{\mu}) = \sum_{k=1}^{N} \sum_{c \neq s} \left(F(\underline{x}_s^{(k)}, \underline{\mu}_s) - F(\underline{x}_c^{(k)}, \underline{\mu}_c) \right)^2 \rightarrow \text{maximize}$$

here index s denotes the true class of the sample of the training set.

The above criterion can be generalized the same way as for LR. Consider an increasing function $\Psi(\cdot)$, Eq. (2.1.9). Applying this function to the difference between the FI for the true class of the sample and that for every other class may enhance the difference between good and poor classifications. The larger the difference, the closer the value of $\Psi(\cdot)$ to unity. The following function is minimized:

$$G(\underline{\mu}) = \frac{1}{N} \sum_{k=1}^{N} \sum_{c \neq s} \left[\Psi\left(F(\underline{x}_s^{(k)}, \underline{\mu}_s) - F(\underline{x}_c^{(k)}, \underline{\mu}_c) \right) - 1 \right]^2 \rightarrow \text{minimize} \qquad (2.5.11)$$

Finally, one can minimize the weighted sum of the above function $G(\cdot)$ and the number of misclassified samples of the training set. In other words, all objective functions used for the weight estimation in LR are applicable here.

The estimation of the FMs requires much more computer time than the weight estimation in LR, because of the higher dimensionality. A simulated annealing simplex procedure was used in order to minimize the objective functions.

After all FMs are estimated, the classification of an arbitrary sample may be performed. FI for all classes should be computed using corresponding memberships and FMs. The class

corresponding to the maximal value of the FI is considered as the most likely class for the sample. The FI-combining classifier is implemented in [37].

2.6 Simple Averaging and Majority Voting Classifiers

Another way of aggregating several individual classifiers can be done by simple Averaging (AVE) of their outcomes, i.e., the memberships provided by the individual classifiers for the particular sample are summarized for each class, resulting in m overall memberships. The class label for the sample is assigned according to the maximal value among them. This approach is a special case of the linear combination of classifiers, when the values of all weights are fixed and set equal to unity. This method is very straight-forward and doesn't require any training.

The Majority Voting (MV) scheme is another simple method of aggregating classifiers. For an arbitrary sample one counts how many individual classifiers vote for each of m different classes, resulting in m scores. Class label for the sample is assigned according to the maximal score. In the case of a tie, when two or more classes obtain the same highest score, the average outcomes of individual classifiers for those classes are taken into account. Class label is assigned to the sample according to the maximal average value.

Both methods of aggregating individual classifiers are considered in order to compare the classification accuracy of relatively complex methods (such as, LR, FI, which require nonlinear optimization during the training phase) with that of these simple methods. These combining classifiers are implemented in [38].

2.7 Stacked Generalization Classifier

Aggregating classifiers via another classifier is called Stacked Generalization (SG). There are many different schemes of stacking classifiers. The particular one used in this thesis is shown in Fig 2.6. The original spectra are preprocessed by different methods and these are submitted to level 0 of classifiers: '*classifier 1*', ..., '*classifier n*'. Usually LDA, QDA, ANN or KNN classifiers were used at this stage. The outcomes of these classifiers form a new set of attributes which is submitted to the classifier of level 1. This classifier (also referred as the combining or aggregating classifier) was limited to either LDA or QDA classifiers in this thesis. If the number of the individual classifiers is n, and the number of classes is m, then the dimensionality of the new set of attributes becomes $n \cdot m$. This way of forming new attributes was proposed by Wolpert [5], and it will be called below as Wolpert's method of input generation. When the dimensionality of the classified data becomes comparable with the number of the training samples, the obtained classification often becomes unreliable. For example, in the case of a combining LDA classifier the estimate of the pooled covariance matrix may become unreasonable; this can lead to good performance on the training set and poor performance on the test set. In such situations another way of generating input was used: the median value of the outcomes of the individual classifiers was chosen for the particular class for the particular sample. That is, instead of $n \cdot m$-dimensional input only m-dimensional input ($Median(x_{11},x_{21},\ldots,x_{n1})$, $Median(x_{12},x_{22},\ldots,x_{n2})$,..., $Median(x_{1m},x_{2m},\ldots,x_{nm})$) is used. This method of input generation will be referred to below as 'median'. The SG classifier is implemented in [36].

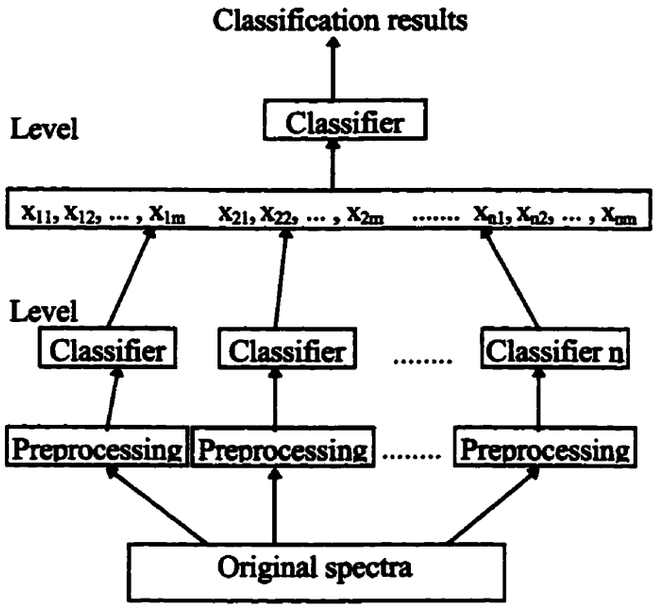

Fig. 2.6 The architecture of the SG classifier, as applied to data.

3. Results on Artificial Magnetic Resonance Spectra

It was mentioned in the introduction that the number of MR spectra available is often limited. As a result, the improvement of classification accuracy due to applying different aggregation techniques to a set of spectra depends on the spectra and on the preprocessing technique used. To objectively analyze the behavior of the aggregation methods discussed above, an artificial set was generated. The description of the artificial set of spectra, the results of applying different aggregation techniques, and the analysis follow.

3.1 Artificial Set of Magnetic Resonance Spectra

In order to make the artificial spectra look similar to real-life spectra the following procedure was performed. A set of real-life proton MR spectra of brain biopsies (531 data points each) that belong to three classes was considered as the starting point for this simulation. Centroid spectra were calculated for all classes. The three centroid spectra were considered as the average representatives of the classes (Fig. 3.1). An MR spectrum can be modeled as a sum of Lorentzians plus noise. The AllFit computer program [34] was used to select a set of 26 Lorentzians (different sets of Lorentzians for different centroid spectra) in such a way, that the sum of these Lorentzians (or peaks) optimally fitted the considered centroid (Fig. 3.2). Each Lorentzian is characterized by its position, width, amplitude and phase. An artificial spectrum was generated by perturbing the position and amplitude of every peak and summing the modified peaks. The width and phase of the peaks were left unchanged. Uniform noise was also added to every generated spectrum.

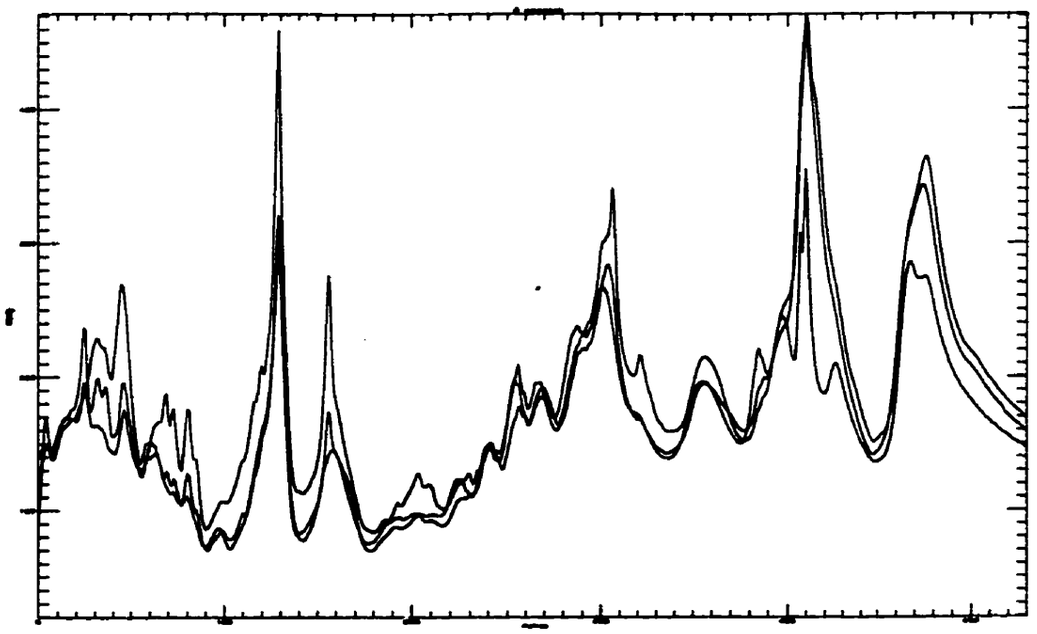

Fig 3.1 The three centroid spectra of real-life spectra of brain biopsies, used to generate a set of artificial spectra.

The positions and the heights of the Lorentzians were randomly perturbed as follows:

$$Pos_i = Pos_i^c + P \cdot (RND - 0.5)$$
$$Height_i = Height_i^c \cdot (1 + H \cdot (RND - 0.5))).$$

Here Pos_i^c and $Height_i^c$ are the position and height of the i-th Lorentzian for class c obtained after fitting with AllFit, RND is a random number in [0,1]. $P/2$ is the maximal shift in the position of the Lorentzian (measured in points), H defines the amplitude variability of the height. An artificial spectrum was computed as follows:

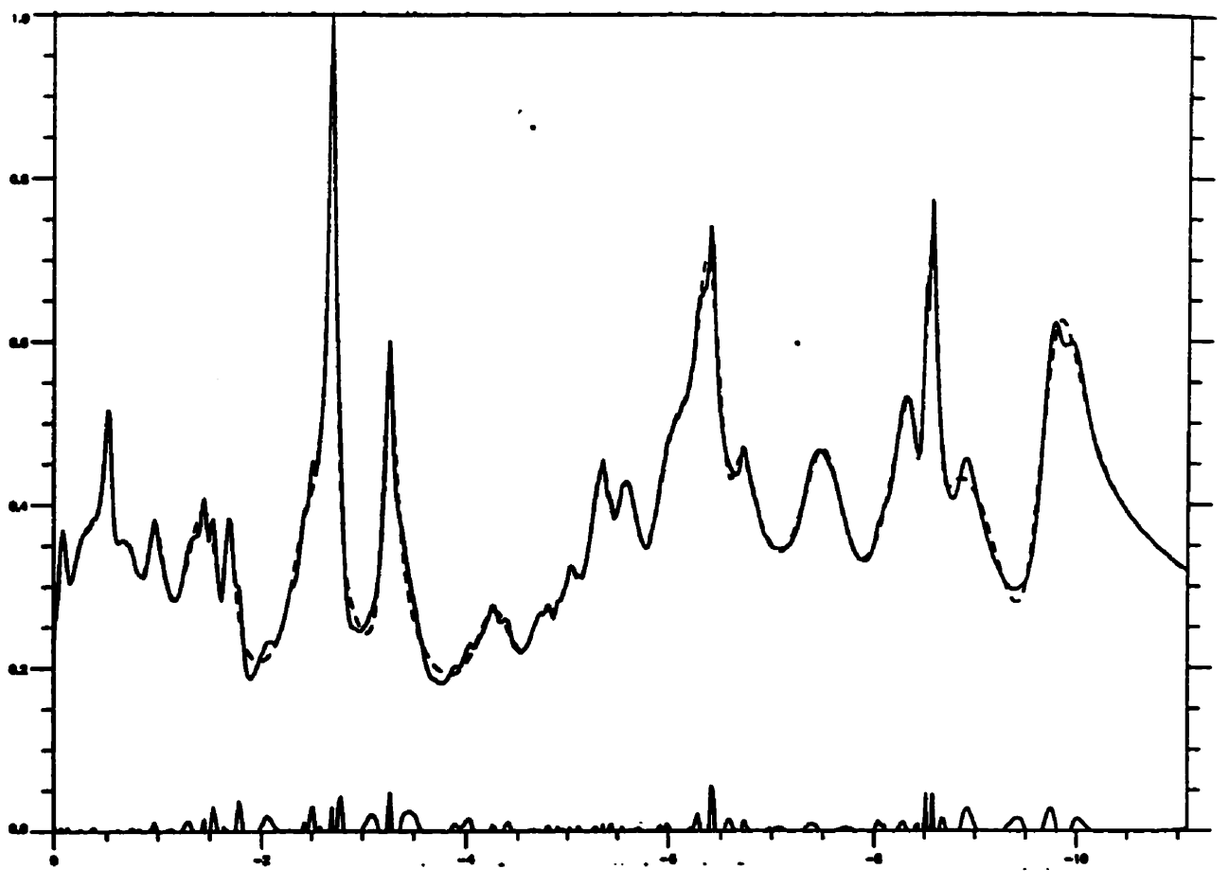

Fig 3.2 Centroid spectrum for class 1 of MR spectrum of brain biopsies (solid line) fitted with a sum of 26 Lorentzians (dotted line). The difference between the original centroid and the one obtained after fitting is shown at the bottom.

$$Lorentzian(x, Pos_i, Height_i, ...) = \frac{Height_i}{1 + \left(\frac{Pos_i - x}{Width_i}\right)^2}\left(\cos(Phase_i) + \frac{Pos_i - x}{Width_i}\sin(Phase_i)\right)$$

$$Spectrum(x) = \sum_{i=1}^{26} Lorentzian(x, Pos_i, Width_i, Height_i, Phase_i) + Q \cdot (RND - 0.5).$$

Here x is the current position (in points) in the spectrum being generated, $Width_i$ and $Phase_i$ are the width and phase of the i-th Lorentzian (they were not perturbed), RND is a uniformly distributed random number in [0,1], Q is the level of the noise added to the spectrum.

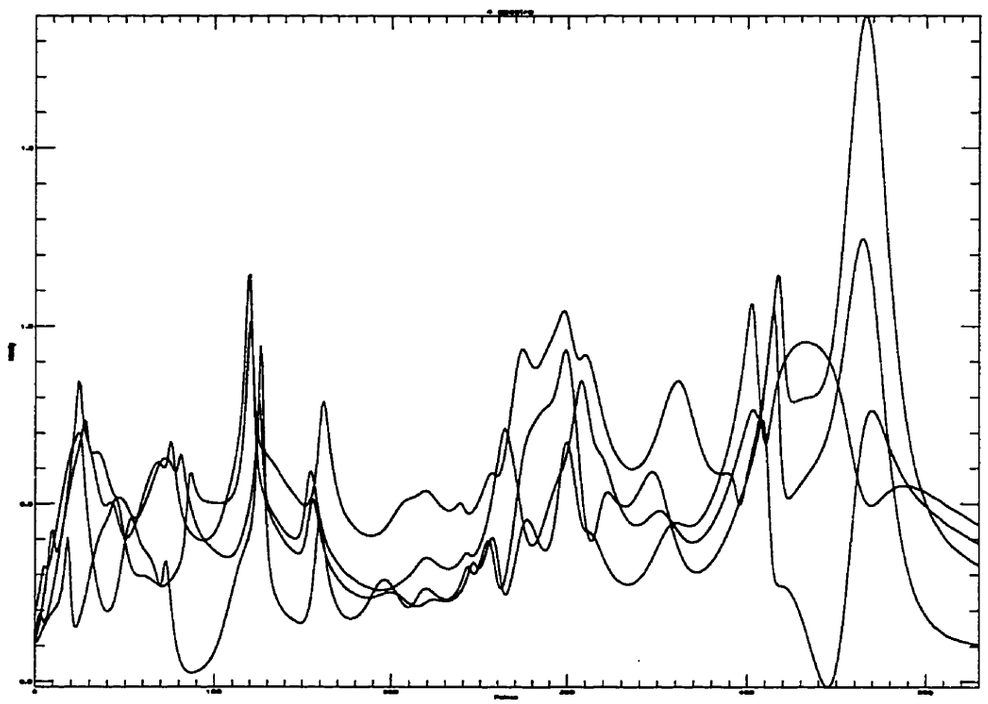

Fig. 3.3 Several examples of the generated artificial spectra from the same class. The uniform noise added to the spectra is not shown.

The spectra were generated in such a way that the three classes overlapped by choosing the values of the parameters P, H and Q. Different values were tried, and the following ones were chosen: P=20, H=1,Q=0.1. A total of 600 spectra (200 per class) were created, half of them were used for training, the second half for testing. Some examples of the generated spectra are presented in Fig. 3.3.

Before any aggregation technique can be applied, the generated spectra must be classified by several classifiers. Some problems arise when a set of 531-dimensional observations is being classified. For instance, discriminant analysis requires knowledge of the covariance matrix, which is usually estimated from the training set. If the number of samples in the training set is smaller than the dimensionality of the sample, the covariance matrix is singular. Even if we have enough samples to get a nonsingular estimate of the covariance matrix, this estimate is reasonable only if the number of the training samples is larger than the dimensionality of the sample (it is desirable to have a few observations per dimension). Since the above conditions almost never hold for real-life spectra, the latter are usually preprocessed first. The same should be done to the artificial spectra. Several preprocessing techniques were employed to reduce the dimensionality of the artificial spectra. They included the following:

- each original spectrum of 531 points was condensed into 53 equal consecutive regions, and the average/median values of the spectrum in each region were calculated
- the same as above for 106 regions
- 12 best regions out of either 53 or 106 above were selected by dynamic programming [35]
- Principal Component Analysis (PCA) was applied to the spectra. The 20 first Principal Components (PCs) were selected [35]

- every original spectrum was split into two segments of 265 left-hand and 266 right-hand points. All the above preprocessing techniques were applied to both segments
- 10 best regions (with average value of the spectrum in the region) were selected in the original spectra by a Genetic Algorithm (GA) [35]

After the preprocessing was done the following classifiers were applied to the preprocessed spectra:

- Linear Discriminant Analysis
- Quadratic Discriminant Analysis
- Artificial Neural Network
- *K*-Nearest Neighbor classifier

Table 3.1 Classification of differently preprocessed artificial spectra by LDA classifier. The following characteristics are listed: preprocessing technique, classification performance and crisp classification performance (in parenthesis) for both training and test sets, MSE for the training set. The definitions of the MSE, classification performance and crisp classification performance are given in chapter 2.1. The best achieved performance is in bold.

No	Classifier and preprocessing technique	Training set (crisp)	Test set (crisp)	MSE on the training set
1	LDA on 12 best regions out of 106; full spectra; average value in a region	0.73 (0.577)	0.70 (0.50)	0.3586
2	LDA on 106 regions; full spectra; average value in a region	**0.833 (0.80)**	0.717 **(0.69)**	0.2705
3	LDA on 106 regions; full spectra; median value in a region	0.783 (0.727)	0.727 (0.683)	0.3523
4	LDA on 12 best regions out of 106; full spectra; median value in a region	0.646 (0.363)	0.636 (0.363)	0.4590
5	LDA on first 20 PCs from the full spectra	0.567 (0.307)	0.583 (0.32)	0.5315
6	LDA on 12 best regions out of 53; full spectra; average value in a region	0.72 (0.42)	0.697 (0.46)	0.3645
7	LDA on 53 regions; full spectra; average value in a region	0.767 (0.683)	**0.736** (0.63)	0.3296
8	LDA on 53 regions; full spectra; median value in a region	0.74 (0.643)	0.723 (0.617)	0.3590
9	LDA on first 20 PCs from the left-hand half of the spectra	0.63 (0.35)	0.636 (0.38)	0.4890
10	LDA on 12 best regions out of 53; left-hand half of spectra; average value in a region	0.683 (0.34)	0.637 (0.373)	0.4703

11	LDA on 12 best regions out of 53; left-hand half of spectra; median value in a region	0.67 (0.34)	0.63 (0.353)	0.4731
12	LDA on 53 regions; left-hand half of spectra; median value in a region	0.67 (0.55)	0.72 (0.597)	0.4636
13	LDA on first 20 PCs from the right-hand half of the spectra	0.603 (0.33)	0.573 (0.347)	0.5019
14	LDA on 12 best regions out of 53; right-hand half of spectra; average value in a region	0.643 (0.38)	0.647 (0.39)	0.4018
15	LDA on 53 regions; right-hand half of spectra; average value in a region	0.707 (0.61)	0.693 (0.59)	0.3998
16	LDA on 12 best regions out of 53; right-hand half of spectra; median value in a region	0.66 (0.327)	0.603 (0.367)	0.4148
17	LDA on 53 regions; right-hand half of spectra; median value in a region	0.683 (0.52)	0.633 (0.52)	0.4734

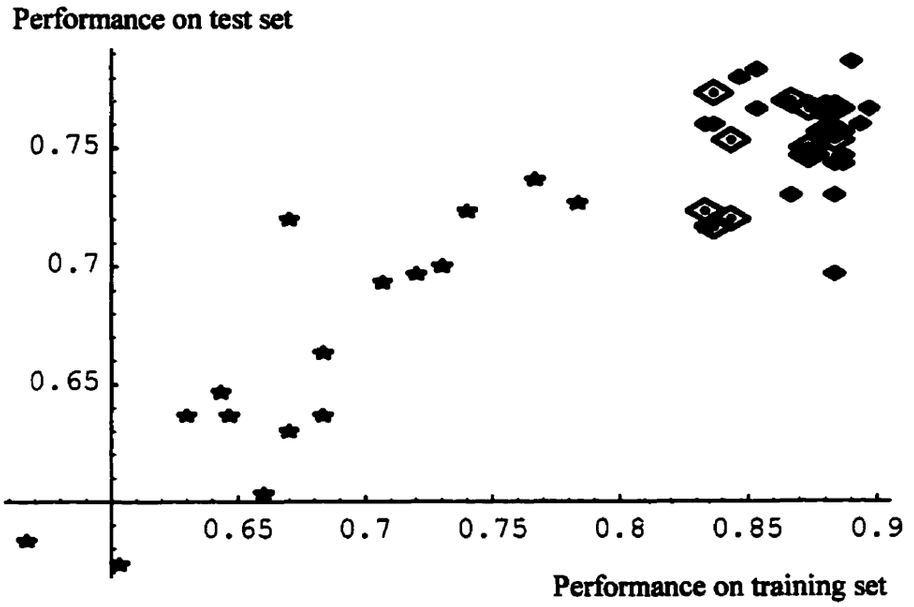

Fig. 3.4 Classification performance of the individual LDAs, and the LR-combining classifier on the training and test sets, artificial spectra. Stars correspond to the LDA classifiers, small diamonds to the aggregation of 17 available LDA classifications, big diamonds to aggregation of a subset of 7 classifications out of 17. Different diamonds of the same type correspond to different objective functions.

We are going to aggregate the outcomes that were obtained from applying the classifiers to the differently preprocessed spectra. For example, the results of applying an LDA classifier to the preprocessed artificial spectra are presented in Table 3.1.

3.2 Performance of Logistic Regression Classifier

The LR-combining classifier was applied to aggregate the individual LDA classifications of Table 3.1. A subset of 7 top performing classifications out of 17, and all 17 classifications were aggregated. Several objective functions were minimized in order to estimate the weights for the individual classifiers. They included:

- unweighted MSE on the training set (2.1.3)
- the number of misclassified samples of the training set (2.1.5)
- the number of misclassified and fuzzily classified samples of the training set
- function (2.1.10)
- unweighted MSE on the training set plus the number of misclassified samples of the training set (2.1.6) (the second term of (2.1.6) had different weights: $\beta=\{1, 5, 0.3\}$)
- function (2.1.11) with different values of the weight β
- unweighted MSE on the training set plus the number of misclassified samples of the training set (2.1.12), with different values of the threshold {0.9, 0.8, 0.7}
- function (2.1.8), etc.

In some cases the estimated weights were constrained by the absolute value ($|w|<50$) during minimization, in other cases no constraints were applied. The same classifications were aggregated in all aggregation instances[16]. In Fig 3.4 both individual LDA and LR-combined classifications of the artificial spectra are presented. The axes are: horizontal - fraction of correctly classified training samples (or performance on the training set), vertical - same for the test set (or performance on the test set). The LDA classifiers which are being aggregated are shown by stars. The results of combined classification are shown by diamonds. Different diamonds of the same type correspond to different objective functions. The LR-combining

[16] minimization of different objective functions results in different estimates for the weights, which leads to different classification outcomes for the combining classifier.

classifier has an equal or better performance on the training set (resubstitution) than the individual LDA classifiers. The performance on the test set is better in most of the cases, although for some objective functions a degraded performance was obtained. The worst result was obtained when objective function (2.1.5) was minimized during aggregation of 17 individual classifications.

The performance of the LR-combining classifier as a function of the number of aggregated classifiers was also analyzed. For this purpose all available LDA classifications were sorted in decreasing order of the performance on the training set, and then the subsets containing different number of top performing classifications were aggregated. The MSE on the training set (2.1.3) was selected as the objective function. The results are shown in Fig. 3.5. Any combined classification has smaller MSE than any individual LDA classification, as expected. A smaller MSE sometimes leads to an increase of the classification performance on the training and/or test sets. In some situations smaller MSE didn't lead to

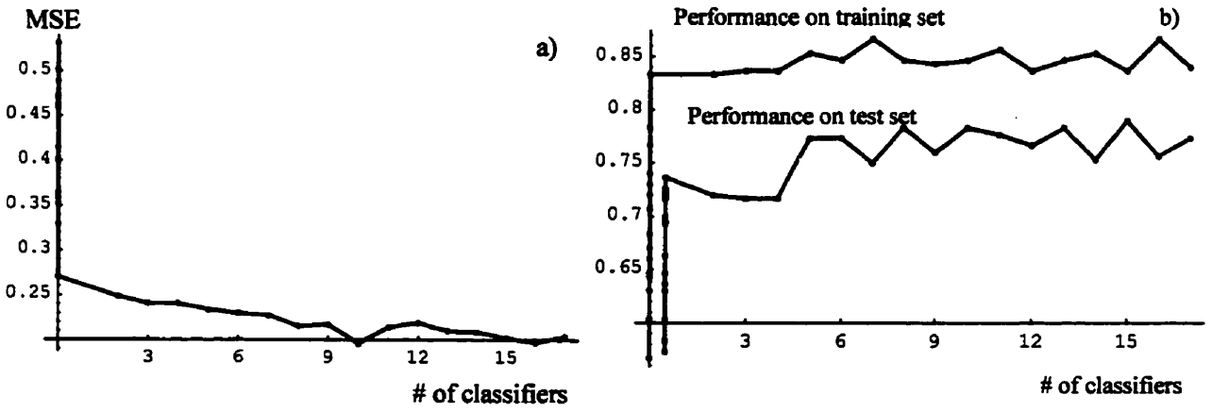

Fig 3.5 LR-combining classifier. MSE on the training set (a), and classification performances on the training and test sets (b) as a function of the number of aggregated classifiers. MSE on the training set (2.1.3) was the objective function. MSE and performances of the individual classifiers which were aggregated are shown as points at the position corresponding to 0 classifiers.

higher classification performance. The analysis of the classification performance on the training set as a function of the number of aggregated classifiers suggested that the optimal number of individual classifiers to aggregate by LR is in the 4-8 range.

It is important that the training set contains sufficient number of samples. To check this, the following procedure was performed. A subset of 7 classifications was selected from Table 3.1 to aggregate using the combining classifier. The combining classifier was trained on subsets of the original training set containing 180, 240 and 300 samples (with equal number of samples in each class) resulting in different estimates of the weights for the individual classifiers. Then it was applied to the test subset consisting of 240 samples. Different aggregation schemes using 5 different objective functions were tried. The results are shown in Fig. 3.6. Stars correspond to the performance on the training set, diamonds on

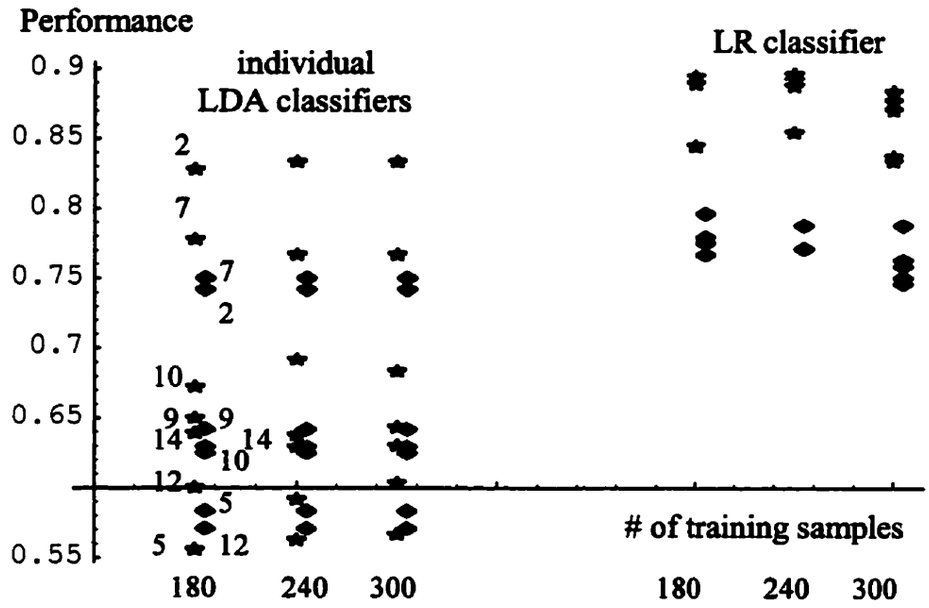

Fig. 3.6 Classification performance of LDA and LR-combining classifiers on the training and test sets as a function of the size of the training set. 7 individual classifications were aggregated. 5 objective functions were minimized, corresponding to 5 different aggregation schemes. Stars correspond to the performance on the training set, diamonds on the test set. The individual classifications are numbered according to Table 3.1

Table 3.2. Six classifications of preprocessed artificial spectra with the best classification performances on the training and test sets. Preprocessing technique, classification performance and crisp classification performance (in parenthesis) for both training and test sets, and the MSE for the training set are presented. The best achieved performance is in bold.

No	Classifier and preprocessing technique	Training set (crisp)	Test set (crisp)	MSE on the training set
1	QDA on 12 best regions out of 106, full spectra, average value in a region	0.87 (0.82)	0.863 (0.81)	0.188
2	QDA on 12 best regions out of 53, full spectra, average value in a region	**0.897** (0.87)	0.913 (0.887)	0.150
3	QDA on 12 best regions out of 53, full spectra, median value in a region	0.893 (0.86)	0.91 (0.877)	0.163
4	QDA on 10 regions selected by GA, full spectra, average value in a region	0.843 (0.773)	0.847 (0.793)	0.223
5	QDA on first 20 PCs from the left-hand half of spectra	0.717 (0.637)	0.90 (0.833)	0.418
6	QDA on 12 best regions out of 53, right-hand half of spectra, average value in a region	0.873 (0.833)	**0.92** (0.87)	0.183

the test set. The results indicate that training the LR-combining classifier on different training sets gives different but close performance on both training and test sets. For all three training sets most of the considered aggregation schemes have better classification performance on the training and test sets than do the individual LDA classifiers. Changing the number of the training samples from 180 to 300 doesn't change much the performance of the individual LDA classifiers on the training set.

Several types of classifiers such as LDA, QDA, ANN, 3-NN were applied to differently preprocessed artificial spectra. Some of the classifiers performed well, some poorly. It was found that QDA had the best performance on both training and test sets. LDA performed worse, ANN performed well on the training set, but not very well on the test set. The 3-NN classifier failed on these data. Six classifications with the best classification performances on the training and test sets were selected among all available classifications. All of them

happened to be QDA classifications, see Table 3.2. These classifications were aggregated by the LR aggregation schemes using the following objective functions to estimate the weights for the individual classifiers:

- unnweighted MSE (2.1.3)
- unnweighted MSE plus the number of misclassified samples of the training set (2.1.6), β=5
- function (2.1.11), β=5
- unnweighted MSE plus the number of misclassified and fuzzily classified samples of the training set, β=5, Eq. similar to (2.1.6)
- function (2.1.12), threshold=0.8 and β=5
- function (2.1.8)
- the number of misclassified and fuzzily classified samples of the training set

The performance and crisp performance of the individual QDA and LR-combining classifiers on the training and test sets are presented in Fig. 3.7. The QDAs are shown by stars, the combined classifications by diamonds. All aggregation schemes demonstrated an improvement in classification accuracy. The classification performance of the LR-combining classifier increased from 90% up to 95% on the training set (resubstitution), and from 92% up to 96% on the test set. Crisp performance also increased on both training and test sets for most of the combining classifiers, Fig. 3.7 (b).

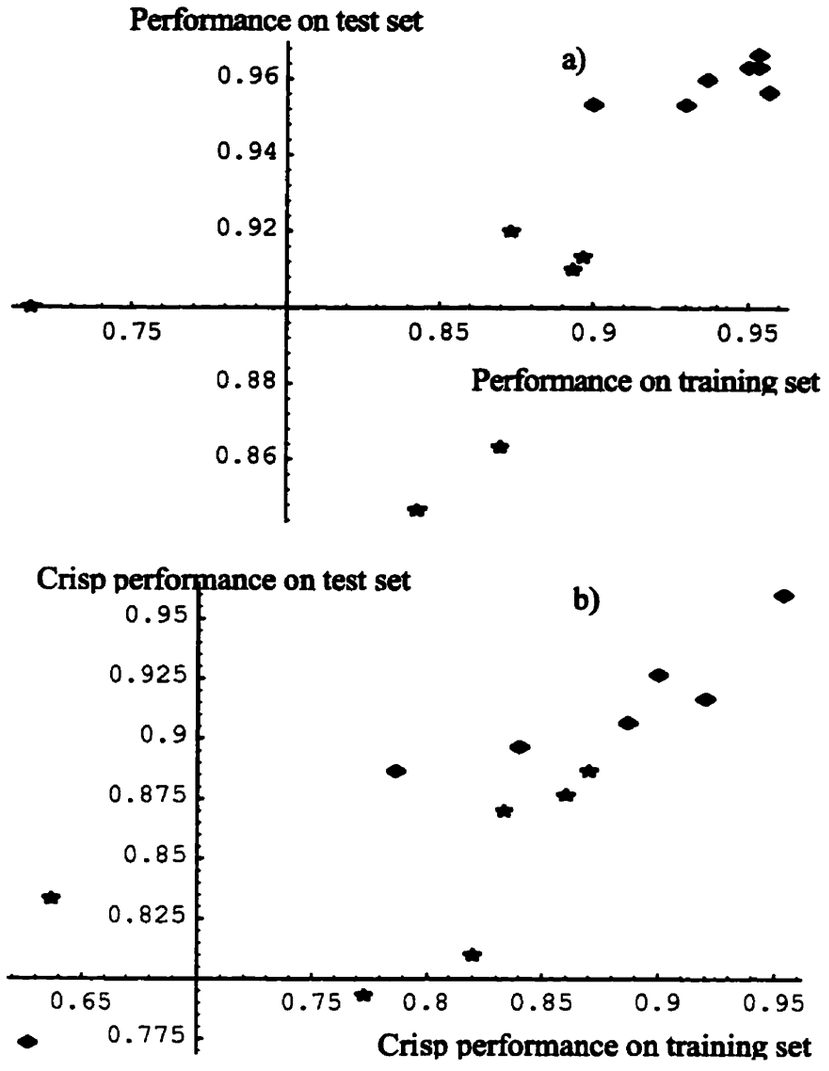

Fig 3.7 Classification performance (a) and crisp classification performance (b) of the individual QDAs, and LR-combining classifier on the training and test sets. Stars correspond to QDA classifiers, diamonds to different aggregation schemes that fuse the QDAs.

3.3 Performance of Linear Combination, Entropy, Confidence Factor, Majority Voting, Simple Averaging, and Stacked Generalization Classifiers

These six aggregating classifiers either do not require any training or the training does not require a nonlinear optimization, i.e., they are fast. This is why they are being considered

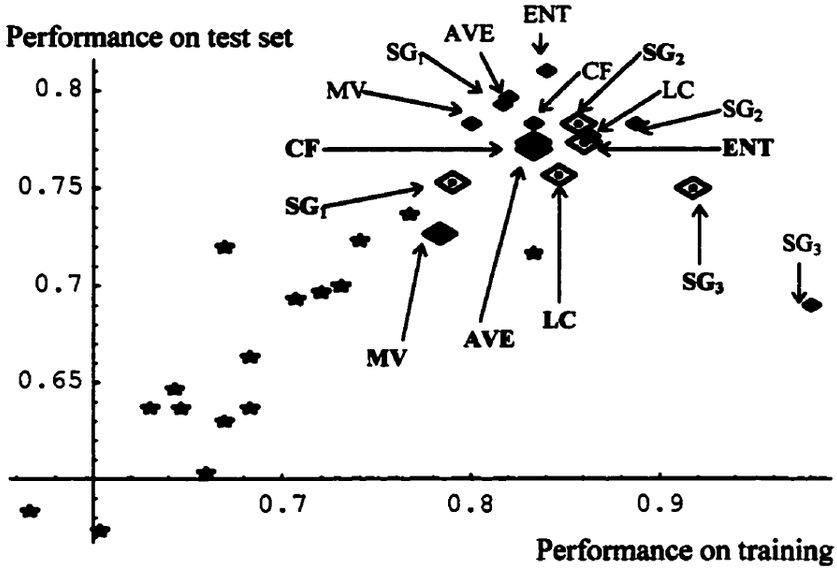

Fig. 3.8 Classification performance of the individual LDAs, and LC, ENT, CF, AVE, MV and SG-combining classifiers on the training and test sets, artificial spectra. Stars correspond to individual classifiers, small diamonds to the aggregation of 17 classifications, big diamonds to aggregation of a subset of 7 classifications out of 17. The subscripts in the SG schemes correspond to different aggregating classifiers and different ways the input was generated for them: 1 - LDA and median-based, 2 - LDA and Wolpert's, 3 - QDA and Wolpert's

together. In order to compare the results of applying these schemes with that of the LR-combined classification the same set of the preprocessed artificial spectra (Table 3.1) was aggregated. The results are presented in Fig. 3.8. The individual classifiers are shown by stars. Small and big diamonds correspond to different subsets of aggregated classifiers: 17

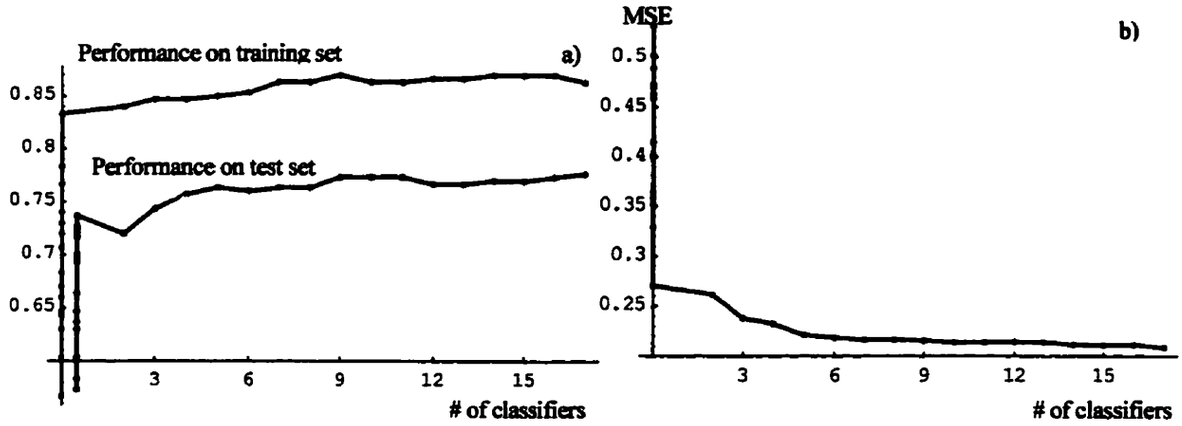

Fig. 3.9 LC-combining classifier. Classification performance on the training and test sets (a), and MSE on the training set (b) as a function of the number of aggregated LDA classifiers. Performances and MSE for the individual classifiers are shown as points at the positions corresponding to 0 classifiers.

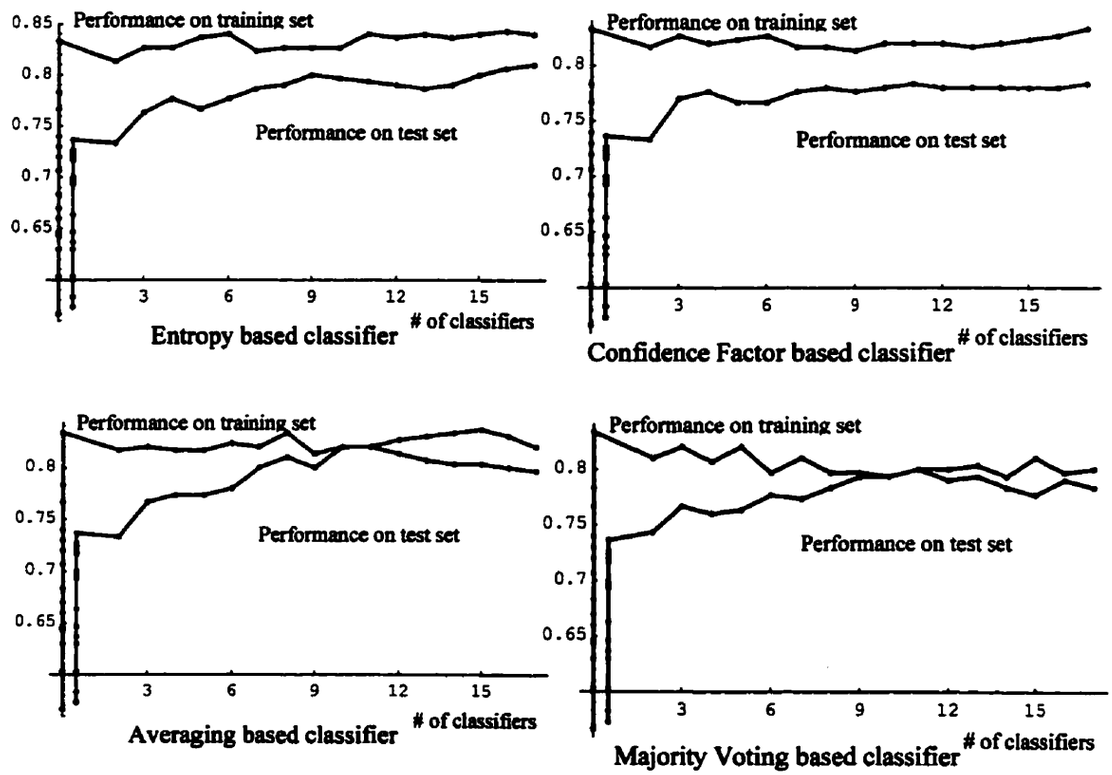

Fig 3.10 The ENT, CF, AVE and MV-combining classifiers. Classification performance on the training and test sets as a function of the number of aggregated classifiers.

different test sets. For one of the test sets the aggregation of the outcomes of the individual classifiers improved the classification performance of the combining classifier. For the other little or no improvement was obtained. This result suggests that the aggregation of classifiers doesn't always lead to better classification performance. Both the training and test sets were generated the same way. The only reasonable explanation why the combining classifiers classify one set of spectra better than the other is that the 'randomly' generated spectra of the test set happen to be distributed more favorably than those of the training set

Similar analysis has been done for the SG-combining classifier. The results for the LDA-based SG classifier with median and Wolpert's methods of input generation are presented in Fig. 3.11. The first classifier showed no improvement in the performance on the training set, Fig 3.11 (a). In fact, some deterioration occurred in comparison to the performance of the best individual classifier. This can be understood, if we look at the dependence of the MSE on the training set as a function of the number of aggregated classifiers, Fig 3.11 (b). None of the combining classifiers has a smaller MSE than the best individual classifier. Thus, it is unlikely to obtain improved performance on the training set[17]. Nevertheless, the performance of this combining classifier on the test set has improved. Wolpert's SG-combining classifier, however, showed an improvement of the classification performance on both training and test sets, Fig 3.11 (c). The MSE on the training set decreases with increasing number of aggregated classifiers, Fig 3.11 (d).

[17] Although there are examples when classifiers have larger MSE, and higher classification performance at the same time.

Six classifiers, which demonstrated top performance on the training and test sets of artificial spectra, were aggregated by all methods considered here, as was done for the LR-combining classifier in chapter 3.2. The results are shown in Fig 3.12. All combining

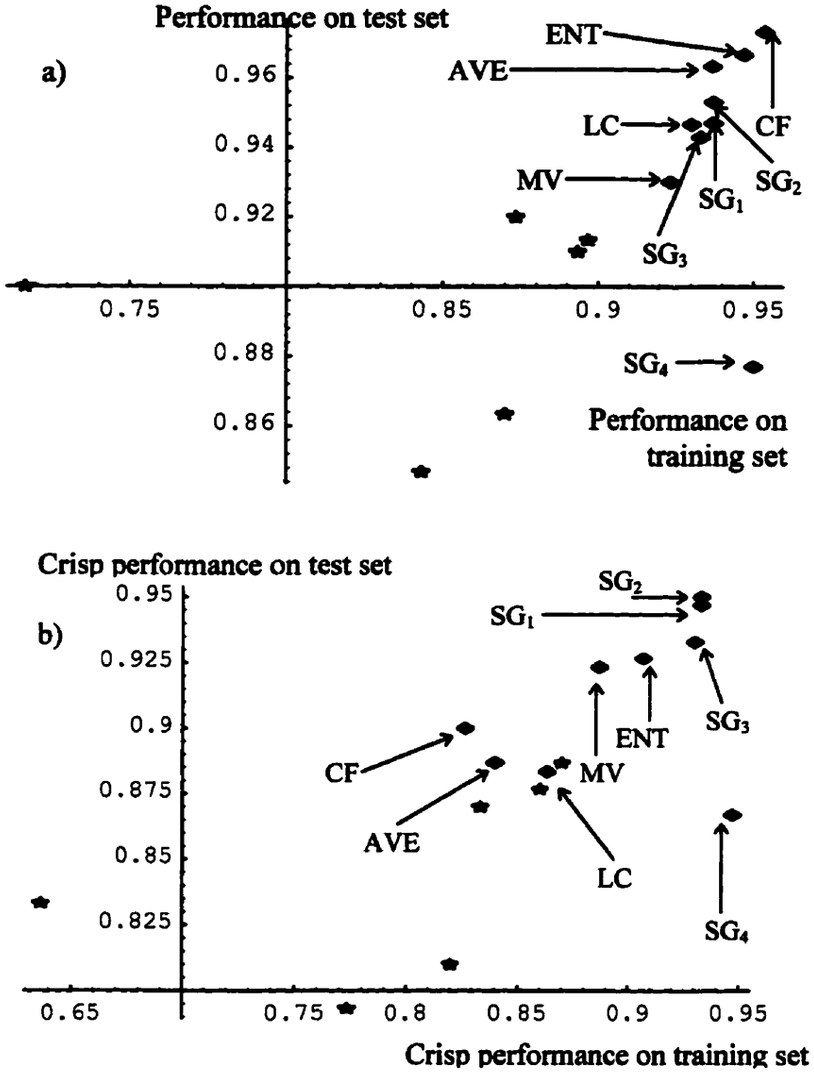

Fig 3.12. Classification performance (a) and crisp classification performance (b) of individual QDAs and different combining classifiers. Stars correspond to the individual classifiers, diamond to their aggregation. Combining classifiers are marked respectively. For the meaning of SG_{1-4} see the text.

classifiers showed improved performance on the training set, and, with the exception of one, on the test set. Abbreviations SG_{1-4} mean SG classifiers with the following aggregating classifiers and input generation methods: LDA and median, LDA and Wolpert's, QDA and median, and QDA and Wolpert's, respectively. Crisp performance also improved for most of the methods. The comparison of these results with those of the LR schemes (Fig. 3.7) indicates that some fast methods produce a similar increase in the classification performance.

3.4 Performance of Fuzzy Integral Classifier

Since the number of the estimated FMs of the FI-combining classifier increases exponentially with the number of aggregated classifiers, it is practically impossible to apply this classifier to the aggregation of 17 (or even a subset of 7 out of 17) LDA classifications of Table 3.1. The number of the FMs to be estimated is $3\cdot(2^{17}-2)$ in the first case, and $3\cdot(2^{7}-2)$ in the second. In addition, even if such an optimization problem were solved, the results would be very unreliable, because 100 training samples per class is obviously not enough for such a high-dimensional space. For the same reason the analysis of the classification

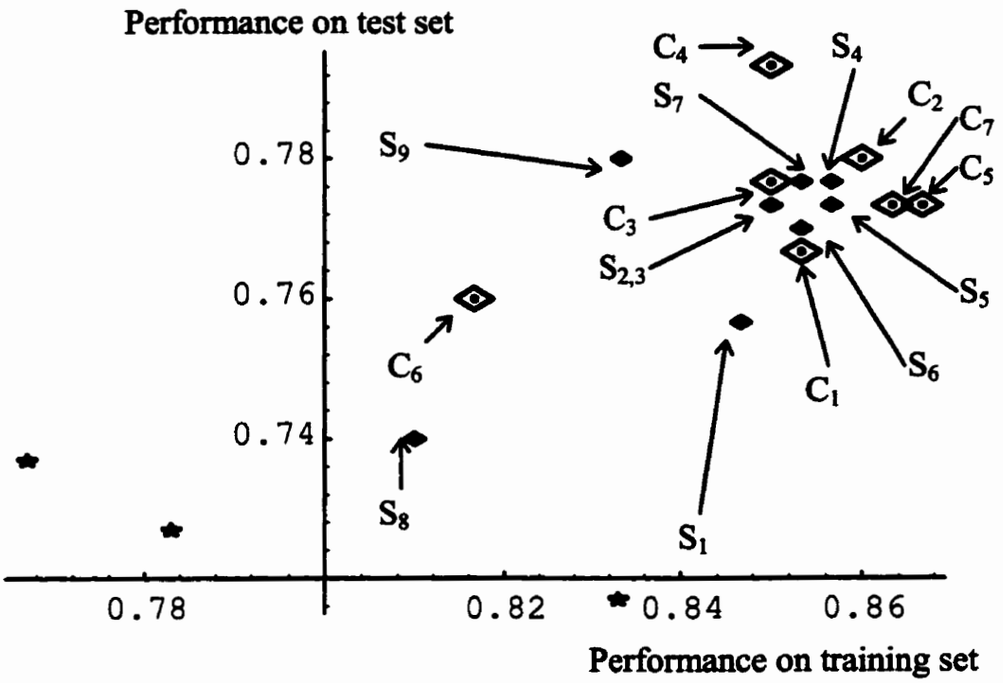

Fig. 3.13 Classification performance of the individual LDAs, and the FI-combining classifiers on the training and test sets. Different objective functions were minimized in order to estimate FMs. Stars correspond to individual classifiers, small diamonds to aggregation by Sugeno FI (S_1-S_9), big diamonds to aggregation by Choquet FI (C_1-C_7). The indices in S_1-S_9 and C_1-C_7 correspond to different objective functions. For their definition see the text.

performance as a function of the number of aggregated classifiers was not performed for the FI-combining classifier.

A subset of 3 individual classifications was selected out of the 17 LDA classifications of Table 3.1 to submit to the FI-combining classifier. The dimensionality of the optimization problem is $3\cdot(2^3-2)=18$ then. These individual classifications were:

- LDA on 106 regions of full spectra, with average values in each region
- LDA on 106 regions of full spectra, with median values in each region
- LDA on 53 regions of full spectra, with average values in each region.

The results of the FI-combining classification based on the Sugeno and Choquet FIs are presented in Fig 3.13. Different objective functions were minimized on the training set in order to estimate the values of the FMs. They were:

<u>for the Sugeno FI-combining classifier:</u>

- S_1 - the unweighted MSE on the training set (2.5.10)
- S_2 - the sigmoid-like function (2.1.9) applied to the difference between the FI value for the true class of the sample and the FI value for any other class (2.5.11)
- S_3 - the sigmoid-like function applied to the difference between the FI value for the true class of the sample and the maximal FI value among those for other classes
- S_4 - the number of misclassified samples of the training set
- S_5 - the weighted sum of the unweighted MSE on the training set (2.5.10) and the number of misclassified samples of the training set. The second term had unit weight.
- S_6 - the same as above but the weight was set to 5.
- S_7 - the weighted sum of the sigmoid-like function applied to the difference between the FI value for the true class of the sample and the FI value for any other class (2.5.11), and the number of misclassified samples of the training set. The second term had unit weight.
- S_8 - the number of misclassified or fuzzily classified samples of the training set
- S_9 - the same as above but the weight was set to 5.

1 - Eq. (2.5.5)
2 - Eq. (2.5.9) with λ=1
3 - Eq. (2.5.6) with p=0.7
4 - Eq. (2.5.7) with γ=2
5 - Eq. (2.5.9) with λ=0
6 - Eq. (2.5.6) with p=2
7 - Eq. (2.5.7) with γ=20

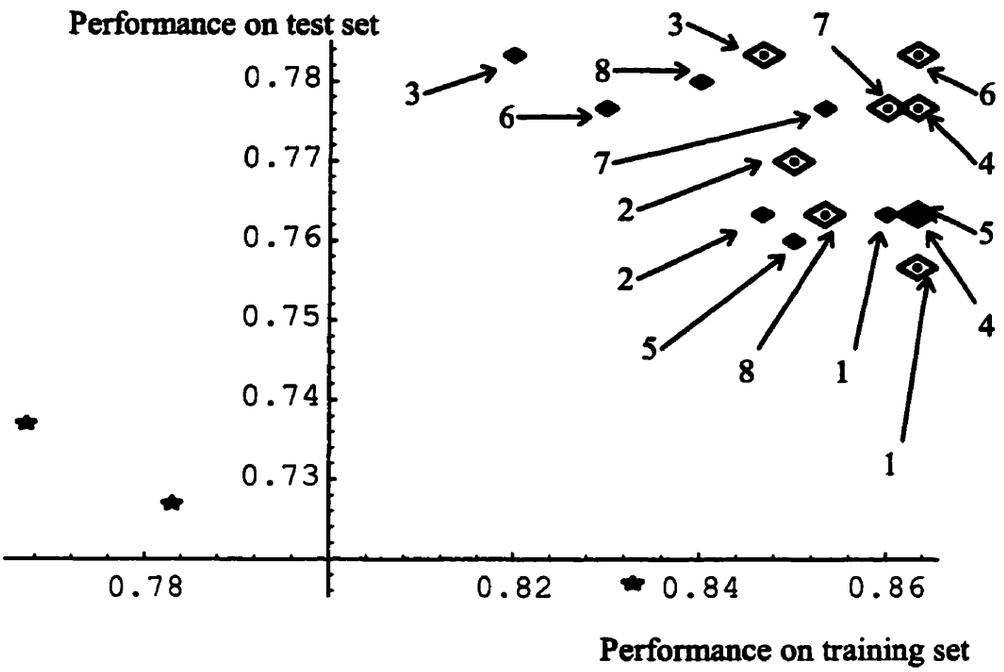

Fig. 3.14 Classification performance of the individual LDAs, and the generalized FI-combining classifiers on the training and test sets. Two objective functions were minimized on the training set in order to estimate FMs. Stars correspond to the individual classifiers, small diamonds to the aggregation scheme using the first objective function, big diamonds to the aggregation scheme using the second one. Notations 1-8 correspond to different *t*-norms used, see text for details.

The results of the combined classification are presented in Fig 3.14. Stars correspond to the individual classifiers, small diamonds to the aggregation scheme using the first objective function, big diamonds to the aggregation scheme using the second one. Notations 1-8 refer to the type of *t*-norm used. The results indicate that all aggregation schemes improved the classification performance on the training and test sets by 1-3% and 2-4% respectively.

Again, the crispness of the classification didn't improve, and stayed near the average of the crisp performances of the aggregated classifiers.

Different FI aggregation schemes, i.e., based on different types of FIs, different objective functions and different *t*-norms in the case of generalized FI, were also applied to the aggregation of four QDA classifications of the artificial set of spectra. These four QDA classifications are:

- QDA on the 12 best regions selected by dynamic programming out of 106 regions of full spectra, with average values in each region
- QDA on the 12 best regions selected by dynamic programming out of 53 regions of full spectra, with average values in each region

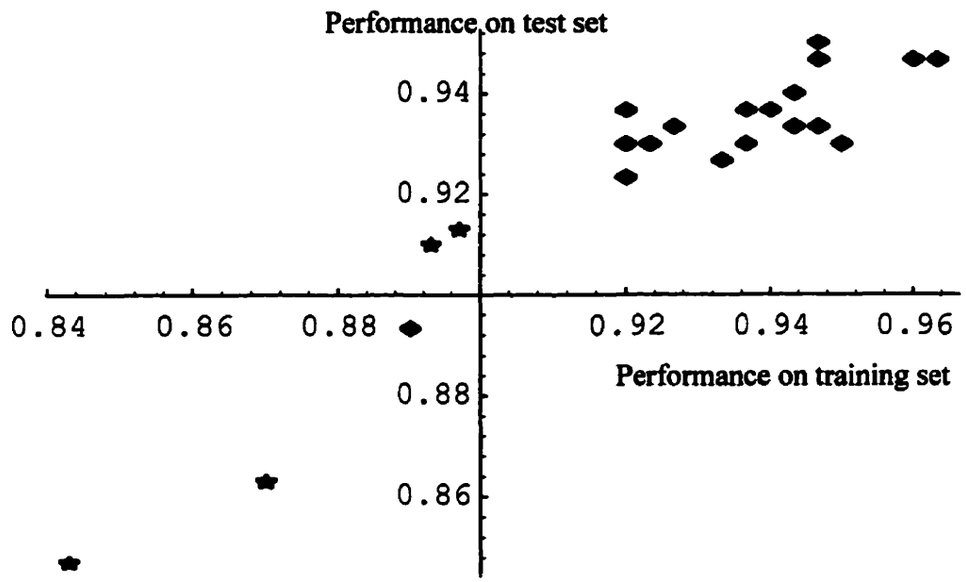

Fig 3.15 Classification performance of the individual QDAs, and different FI-combining classifiers on the training and test sets. Different type of FIs, objective functions, and *t*-norms were used. Stars correspond to the individual classifiers, diamonds to their aggregations.

- QDA on the 12 best regions selected by dynamic programming out of 53 regions of full spectra, with median values in each region
- QDA on the 10 best regions selected by GA on full spectra, with average values in each region

As one can see in Fig 3.15, all FI-combining classifiers, with one exception, showed better classification performance on both training and test sets. This exception happened to be the combining classifier using the generalized FI with *t*-norm (2.5.5), and the objective function 'the number of misclassified samples of the training set'. The crisp performance of the aggregation schemes stayed near the average of the crisp performances of the individual classifiers, Fig. 3.16. However, several aggregation schemes, based on

- generalized FI, MSE on the training set (2.5.10) + the number of misclassified samples of the training set, unit weights for both terms, *t*-norm (2.5.9), λ=1
- generalized FI, MSE on the training set (2.5.10) + the number of misclassified samples of the training set, unit weights for both terms, *t*-norm (2.5.6), p=0.7
- generalized FI, objective function (2.5.11), *t*-norm (2.5.6), p=0.7
- generalized FI, the number of misclassified samples of the training set, *t*-norm (2.5.9), λ=1

gave a better crisp performance on both training and test sets. At the same time the aggregation schemes based on Sugeno FI with 'the number of misclassified samples of the training set', and 'function (2.5.11) + the number of misclassified samples of the training set' objective functions had a worse crisp performance than any of the individual classifiers. However, for other sets of spectra the situation may be different; it is data dependent.

In general, the FI-combining classifier often improves classification performance in comparison to that of the individual classifiers. Crisp performance improves occasionally.

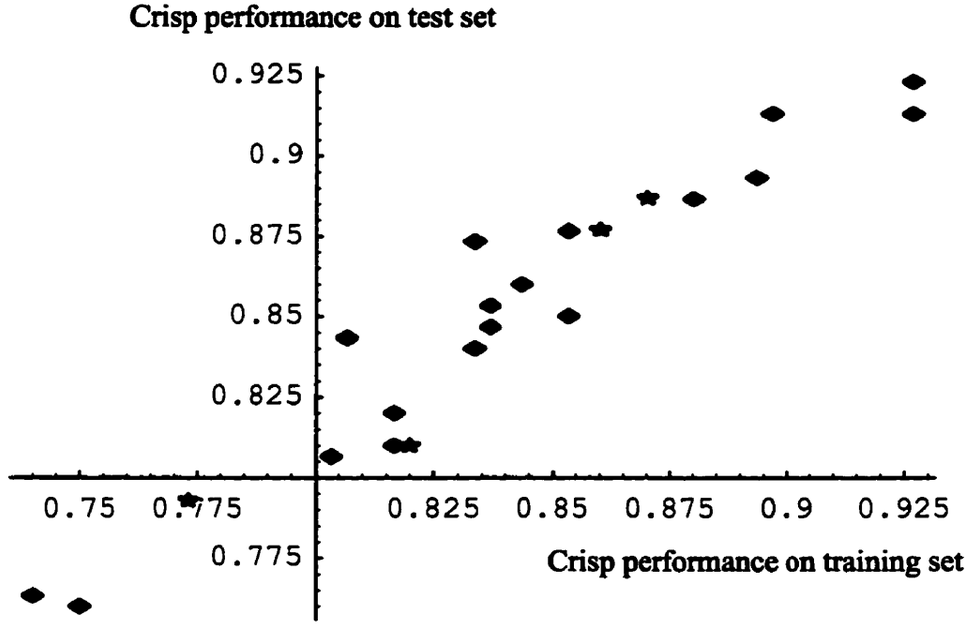

Fig 3.16 Crisp classification performance of four individual QDAs, and FI-combining classifiers on the training and test sets, artificial spectra. FI aggregation schemes are based on Sugeno, Choquet and generalized FIs, different objective functions, which were minimized on the training set in order to estimate FMs, and different *t*-norms (for the generalized FI). Stars correspond to the individual classifiers, diamonds to the aggregating classifiers.

3.5 Comparison of Classification Accuracy

Finally, all aggregation methods were compared, while applied to the set of four QDA classifications, Fig. 3.17. The following schemes of the FI, LR, and SG-combining classifiers were applied:

- FI_1 - Sugeno FI, objective function (2.5.11) + the number of misclassified samples of the training set, unit weights for both terms

- FI_2 - Choquet FI, MSE on the training set (2.5.10) + the number of misclassified samples of the training set, unit weights for both terms
- FI_3 - generalized FI, MSE on the training set (2.5.10) + the number of misclassified samples of the training set, unit weights for both terms, *t*-norm (2.5.9), λ=1
- FI_4 - generalized FI, MSE on the training set (2.5.10) + the number of misclassified samples of the training set, unit weights for both terms, *t*-norm (2.5.6), p=0.7
- FI_5 - generalized FI, objective function (2.5.11), *t*-norm (2.5.6), p=0.7
- LR_1 - MSE on the training set + the number of misclassified samples of the training set (2.1.6), unit weights for both terms
- LR_2 - MSE on the training set (2.1.3)
- LR_3 - objective function (2.1.11), unit weights for both terms
- LR_4 - objective function (2.1.10)
- SG_1 - aggregating LDA classifier, median scheme
- SG_2 - aggregating LDA classifier, Wolpert's scheme
- SG_3 - aggregating QDA classifier, Wolpert's scheme

Practically all methods improved classification accuracy of both training and test sets, Fig. 3.17 (a). The MV-combining classifier had performance close to that of the best individual classifier (slightly better for the training set, and slightly worse for the test set). SG_3 didn't perform well for either of these data. The ENT classifier, one LR, two SG and three FI schemes improved the crisp classification performance for both training and test sets, Fig. 3.17 (b). Most of the others had crisp classification performance slightly worse that that of the best individual classifier. One FI and two LR schemes had worse crisp performance.

Thus, computationally simple and fast aggregation methods can perform as well as complicated and very time consuming aggregation methods.

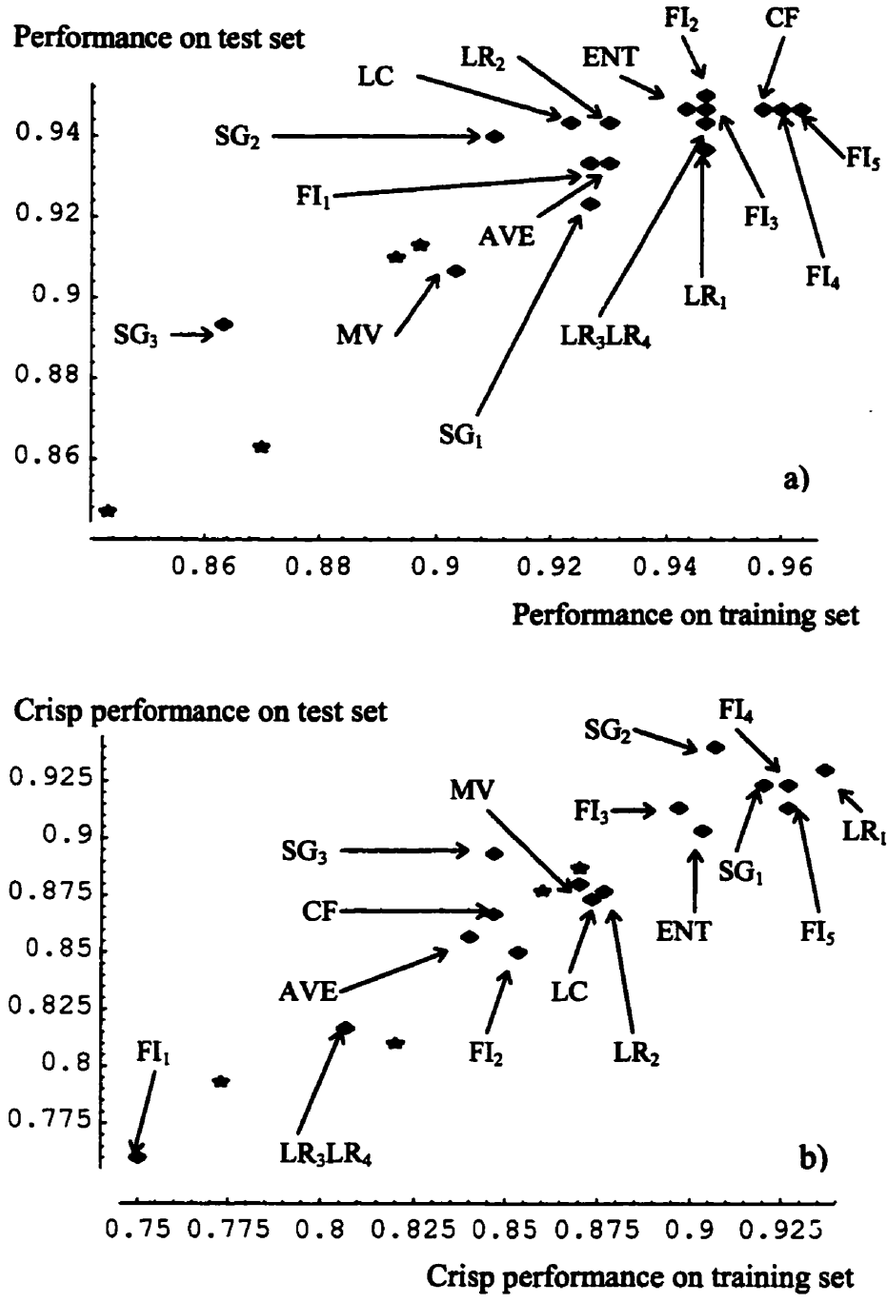

Fig. 3.17 Classification (a) and crisp classification (b) performances of four QDAs, and various aggregation methods. Stars correspond to individual classifications, diamonds to the combined classifications. For the meaning of the indices of the LR, FI, and SG schemes see text.

After analyzing the performance of the different aggregation methods, thirteen were selected to apply to real-life spectra. The description of the methods, parameters if required, and abbreviations are presented in Table 3.3.

Table 3.3 Aggregation techniques found to work well on the artificial MR spectra. These techniques will be applied to classifying real-life spectra in subsequent chapters. The abbreviations below will be used to distinguish among the methods.

#	Aggregation technique, parameters	Abbreviation
1	Confidence Factor	CF
2	Entropy	ENT
3	Majority Voting	MV
4	Logistic Regression; objective function: the MSE on the training set + the number of misclassified training samples, Eq. (2.1.6), β=1., the weights are constrained by absolute values $\|w\|<50$ during minimization	LR_1
5	Logistic Regression; objective function: the MSE on the training set + the number of misclassified training samples, Eq. (2.1.6), β=5., the weights are constrained by absolute values $\|w\|<50$ during minimization	LR_2
6	Logistic Regression; objective function: the MSE on the training set + the number of misclassified and fuzzily classified training samples, Eq. similar to Eq. (2.1.6), β=1., the weights are constrained by absolute values $\|w\|<50$ during minimization	LR_3
7	Logistic Regression; objective function: Eq. (2.1.12), threshold=0.8, the weights are constrained by absolute values $\|w\|<50$ during minimization	LR_4
8	Generalized Fuzzy Integral; objective function: the MSE on the training set + the number of misclassified training samples, Eq. similar to Eq. (2.1.6), β=1., t-norm (2.5.6), p=0.7	FI_1
9	Generalized Fuzzy Integral; objective function: Eq. (2.5.11), t-norm (2.5.6), p=0.7	FI_2
10	Generalized Fuzzy Integral; objective function: the MSE on the training set + the number of misclassified training samples, Eq. similar to Eq. (2.1.6), β=1., t-norm (2.5.9), λ=1.	FI_3
11	Generalized Fuzzy Integral; objective function: the number of misclassified training samples, Eq. similar to Eq. (2.1.5), t-norm (2.5.9), λ=1.	FI_4
12	Stacked Generalization, aggregating classifier LDA, median scheme	SG_1
13	Stacked Generalization, aggregating classifier LDA, Wolpert's scheme	SG_2

3.6 Combining randomized classifiers

In chapter 3.3 we have analyzed the dependence of classification performance of the combining classifiers on the number of aggregated classifiers. The performances of the individual classifiers ranged from less than 50% up to 85%, i.e., were very different. It is interesting to see what improvement in classification accuracy one can obtain if the aggregated classifiers had similar performances (but of course different classifications). In other words, what would happen if the outcomes of individual classifiers are distributed about some 'average' outcome.

The following procedure was performed. Sets of preprocessed real-life spectra were classified by an LDA classifier. The classification outcomes (i.e., memberships in different classes) were perturbed by adding 20% noise, and then normalized. Twelve

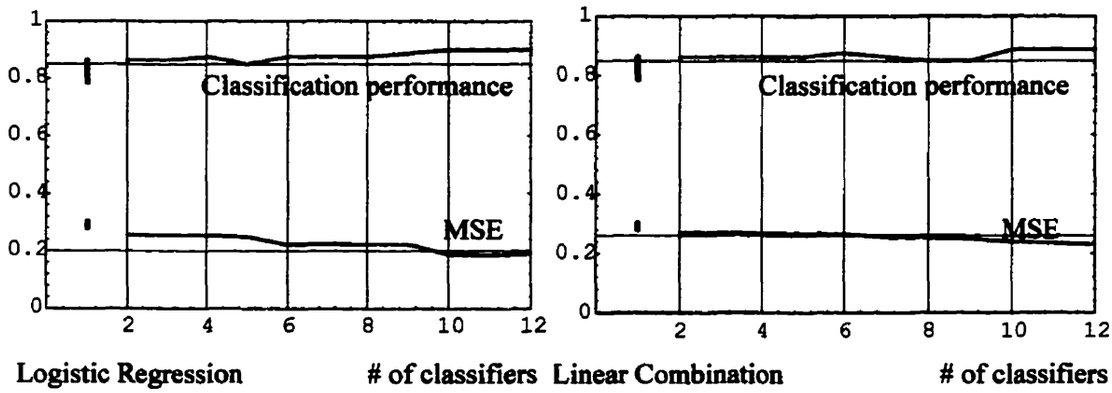

Fig. 3.18 Classification performance of LR & LC-combining classifiers, and the MSE on the training set as a function of the number of aggregated classifiers with similar performances. Classification performances and MSE of the aggregated classifiers are shown as points at the position corresponding to 1 classifier.

'classifications' were obtained this way. These classifications were aggregated by the LR and LC-combining classifiers. The weights for the individual classifiers were obtained by minimizing MSE on the training set. The results are shown on Fig. 3.18. The MSE of the combining classifiers decreases monotonically with increasing number of aggregated classifiers. Classification performance generally increases with increasing number of aggregated classifiers. Crisp classification performance also improves, Fig. 3.19.

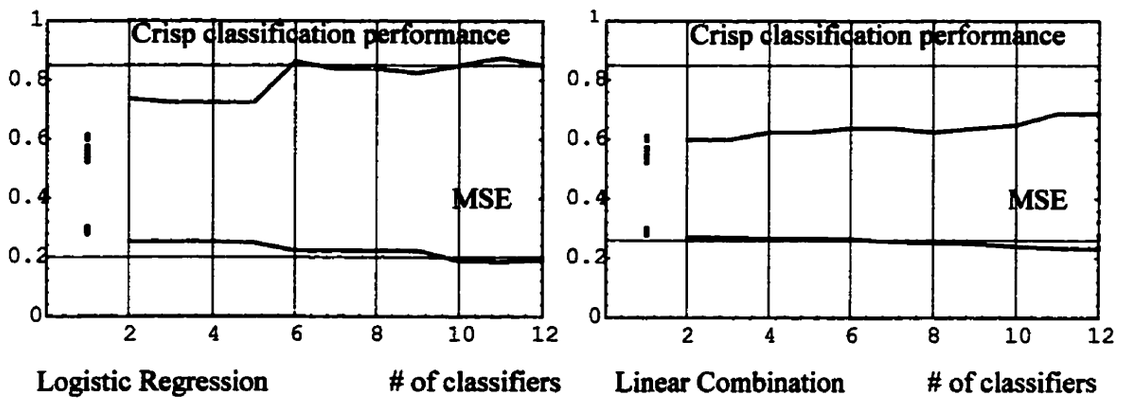

Fig. 3.19 Crisp classification performance of LR & LC-combining classifiers, and the MSE on the training set as a function of the number of aggregated classifiers with similar performances. Crisp classification performances and MSE of the individual classifiers are shown as points at the position corresponding to 1 classifier.

3.7 Comparison of Speed of Combining Classifiers

Aggregation methods such as the LR, LC, and FI classifiers require estimating unknown parameters (weights for the LR and LC, and FMs for FI classifier) during the training stage. These parameters are calculated by minimizing an objective function on the training set. In general, a nonlinear constrained optimization technique is required for the LR and FI classifiers. The weights in the LC-combining classifier can be estimated by inverting several matrices of small dimension. The number of the parameters to be estimated increases

3.8 Choosing Individual Classifiers

When many individual classifications are available, the problem of how to choose the classifications to aggregate arises. As was mentioned in several papers, aggregating the best individual classifications does not necessarily lead to the best performance of the combining classifier. The independence of classifications is a more important requirement. Suppose two classifiers have high classification performances on some data set. This means that most of the samples of the data set are classified correctly by both classifiers, and just a few of them are misclassified. If the classifiers misclassify these samples into the same class, they are correlated in making errors, otherwise the classifiers are uncorrelated. The uncorrelated classifiers are of interest to combining classifiers. This lack of correlation helps to improve the performance by aggregating these classifiers. Obviously, the better the performance of the individual classifiers, the smaller the correlation between them in making errors, just because fewer samples are misclassified. It is less likely that aggregating such classifiers will improve much the classification performance. On the other hand, classifiers with low performance may be less correlated in making errors, and although their aggregation may improve classification accuracy to a higher degree, this improved accuracy can be worse than the accuracy of a high performing individual classifier. Thus, a trade-off between the classification performance of classifiers and the correlation among them in making errors has to be considered selecting classifiers for aggregation.

The following procedure of selecting classifiers is proposed. Denote the performance of the i-th classifier by P_i, and the correlation in making errors between the i-th and j-th classifiers by C_{ij}. Calculate the correlation in making errors between all pairs of classifiers.

in both lists (lines 1 and 3), but the order is different. For instance, if one selects 3 classifications for aggregation, the proposed procedure suggests selecting the two classifications with the best performances (43 and 12), but classification 32 instead of classification 11 despite the latter's better performance. The value of parameter β was set to 10.

The aggregation methods of Table 3.4 were applied to the aggregation of classifications {43,12,11} (set 1) and {43,12,32} (set 2), and the results are compared, Fig 3.20.

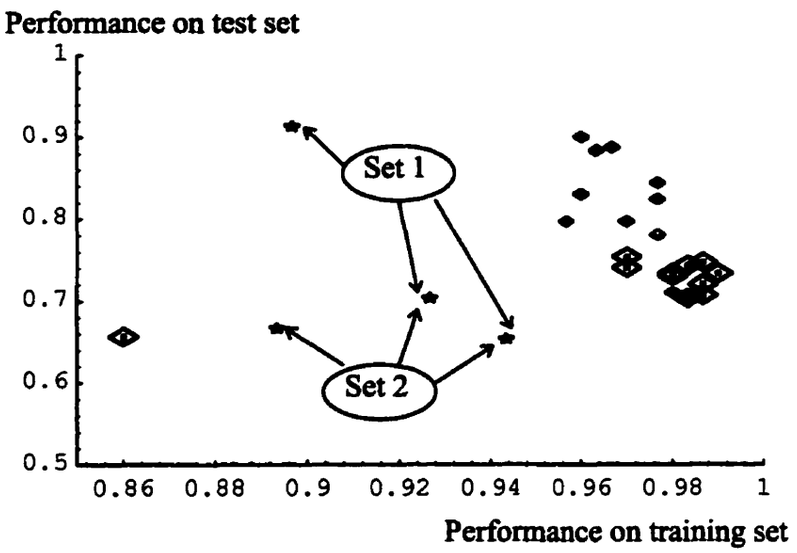

Fig. 3.20 Comparison of the classification performances of the combining classifiers aggregating 3 classifications with the best classification performances on the training set (set 1), and 3 classification with high performances and low correlation in making errors on the training set (set 2). Aggregation methods of Table 3.4 were applied. The individual classification are shown by stars, aggregations of set 1 classifiers by small diamonds, aggregations of set 2 classifiers by big diamonds.

The individual classifications are shown as stars, small diamonds correspond to the aggregation of the classifiers of set 1, big diamonds to the aggregation of the classifiers of set 2. The results are quite interesting. Most of the combining classifiers aggregating the

individual classifiers of set 2 have better classification performance on the training set than for the classifiers of set 1. Thus, taking into account the independence in making errors among individual classifications while choosing what individual classifications to aggregate can indeed improve the classification performance on the training set.

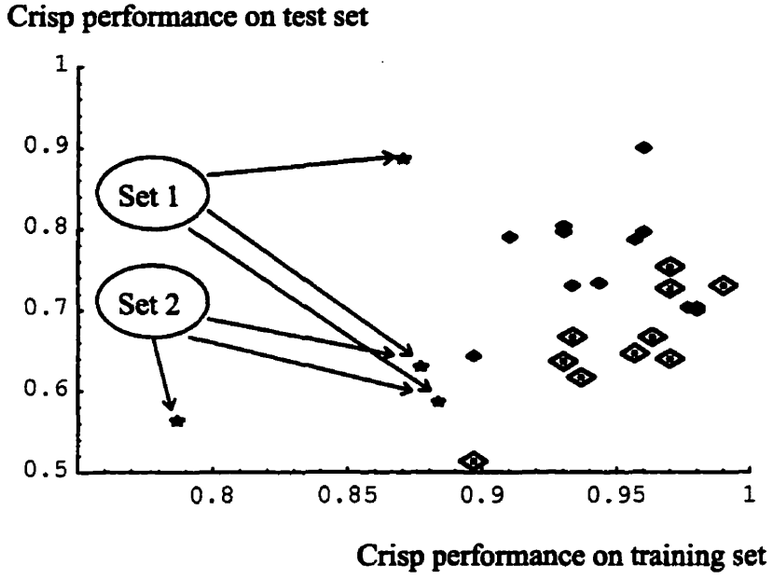

Fig. 3.21 Comparison of the crisp classification performances of the combining classifiers aggregating 3 classifications with the best classification performances on the training set (set 1), and 3 classifications with high performances and low correlation in making errors on the training set (set 2). Aggregation methods of Table 3.4 were applied. The individual classification are shown by stars, aggregations of set 1 by small diamonds, aggregations of set 2 by big diamonds.

Now look at the results on the test set. For some reason the individual classification 11 of the set 1 has a significantly higher performance on the test set than all others. This raises the performances of the combining classifiers on the test set. None of the individual classifications of set 2 has comparable performance. As a result, the classification performances of the combining classifiers aggregating the classifiers of the set 2 are worse,

although in general they are better than the corresponding performances of the aggregated classifiers. In the case of aggregating classifiers of set 1, the performances on the test set are near the average of that of the aggregated classifiers.

The comparison of crisp classification performances is shown in Fig 3.21. The performances on the training set of the combining classifiers aggregating the classifiers of set 2 are not significantly better than that of the combining classifiers aggregating the classifiers of set 1. The crisp performances on the test set of the combining classifiers behave similarly to the performances on the test set.

4. Results on Real-Life Magnetic Resonance Spectra

The thirteen aggregation techniques of Table 3.3, which were found to perform well on the set of artificial MR spectra, were applied to real-life spectra. The description of the analyzed spectra, the preprocessing and classification methods used, the results of the combined classification, and some analysis follow.

A set of 215 proton MR spectra of brain tissue samples was classified. The spectra belong to three classes: high grade astrocytoma, meningioma and epilepsy. 84 samples constituted the training set, the rest were used for testing. Each spectrum consists of 550 points. Several preprocessing techniques were employed in order to prepare the spectra for classification. These techniques were applied to the original unnormalized spectra, as well as to normalized spectra. They included:

- the first 18 PCs, which explain most of the variance in the spectra, were selected by PCA for the unnormalized spectra
- the first 11 PCs were selected for the normalized spectra
- each unnormalized spectrum was condensed into 55 consecutive regions, and the mean value of the spectrum in each region was calculated. Then 23 regions were selected by dynamic programming
- similarly, 25 regions were selected for the normalized spectra

The preprocessed spectra were classified by ANN, LDA, QDA and KNN classifiers, resulting in approximately two dozen classifications. The best classifications were the LDA and ANN classifications in general. The QDA classifier performed worse than the LDA classifier. Thus, it seemed to be more beneficial to apply the LDA classifier to differently

preprocessed spectra rather than to apply the QDA classifier at all. The KNN classifier performed very poorly on these spectra. 8 classifications were selected to submit to the combining classifiers of Table 3.3. The results of the individual and combined classifications are presented in Fig 4.1 (a). The individual classifications are shown by stars and big diamonds. The combined classifications are shown by small diamonds. Most of the aggregating classifiers were applied to the 8 classifications. The FI-combining classifier was applied to four individual classifications shown by big diamonds.

Most of the aggregation methods improved the classification performance on the training set, however only a few of them improved the performance on the test set. The crisp classification performance on the training set was improved by a few methods (mainly, by the LR and FI classifiers), but in general remained near the average on the test set, Fig 4.1 (b). Interestingly, classification performance on the training set was improved mainly by the LR and FI classifiers, the ones which were trained on this set. At the same time these classifiers perform relatively poorly on the test set. Other combining classifiers which don't require training (except the SG classifier) didn't perform as well as the LR and FI classifiers on the training set, but performed much better on the test set.

A set of proton spectra, spectra of cervical biopsies, was classified by the aggregating classifiers. Of the 98 spectra available, 40 were used for training the combining classifiers, the rest for testing. Each spectrum consists of 650 points. The spectra belong to 2 classes. The spectra were preprocessed by

- condensing each spectrum into either 65 or 130 consecutive regions, and calculating the mean value of the spectrum in each region. Then the 12 best regions were selected by dynamic programming
- selecting the 10 regions in the original spectra by GA

The preprocessed spectra were classified by an LDA classifier. The QDA classifier was also applied, but the obtained results were poor. The KNN classifier was not applied because of insufficient number of training samples. After the individual classifications were obtained the combining classifiers were applied to 3 selected classifications. The results of classification are presented in Fig. 4.2. Stars correspond to the individual classifications, diamonds to the combining classifiers. The LR-combining classifier performed well on this set of spectra. Both classification and crisp classification performances were improved by aggregation. This data set is difficult to classify because of the insufficient number of the spectra available. A very high classification accuracy can be achieved on the training set (97.5% in the case of the LDA classifier applied to 10 regions selected by GA), yet applying the same classifier to the test set barely achieves 64% accuracy. Aggregating individual classifiers improved the classification accuracy on the test set from 67.2% up to 70.7%, and crisp classification accuracy on the test set from 62% up to 67.2% by aggregating the individual classifiers.

A set of proton brain spectra was also classified. This set has 215 spectra, 84 of them constitute the training set, the rest the test set. The spectra belong to 3 classes. Each spectrum consists of 550 points. The spectra were preprocessed as follows:

- the first 10 PCs were selected by applying PCA to normalized and unnormalized spectra

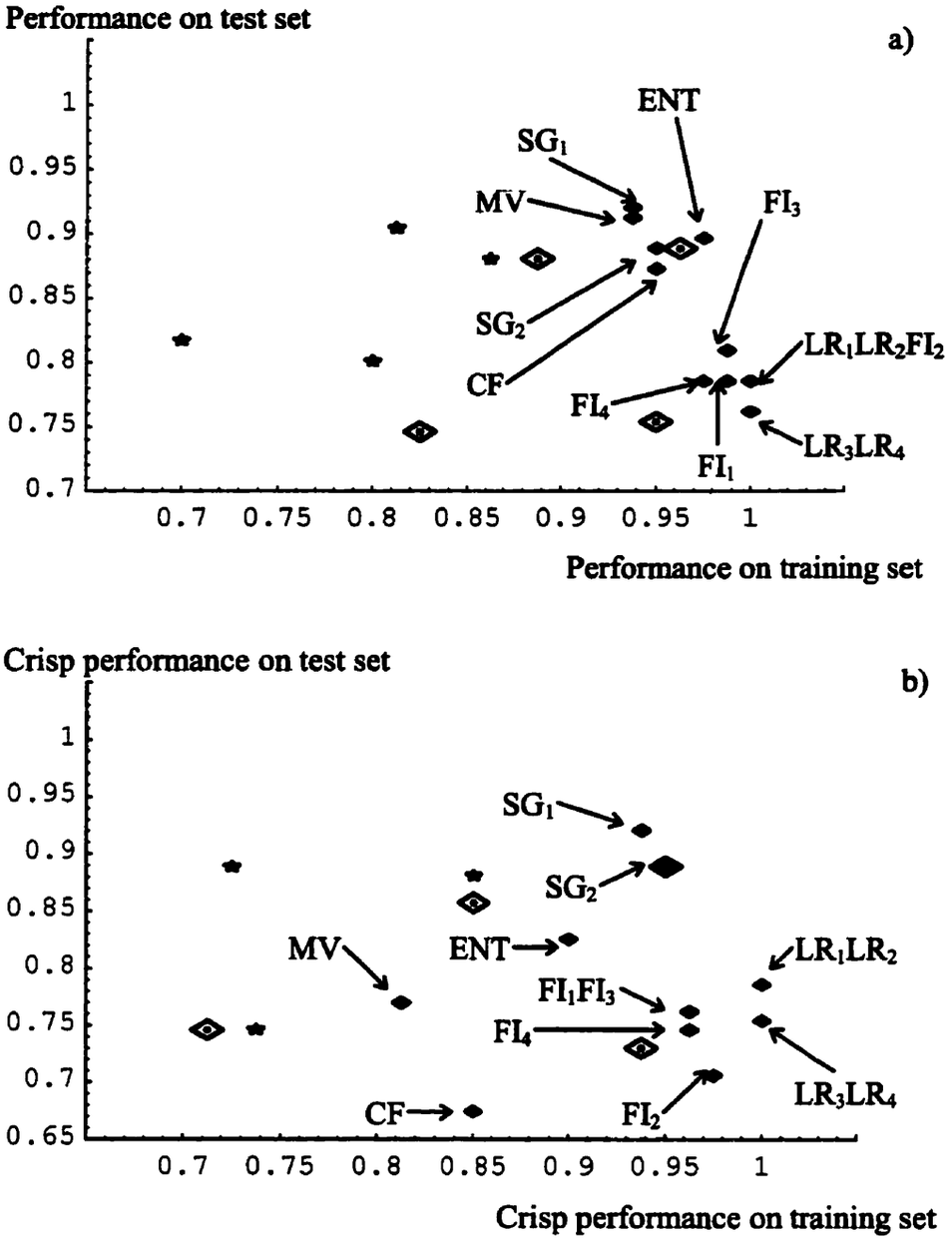

Fig. 4.1 Classification performance (a) and crisp performance (b) of the individual and combining classifiers on the training and test sets, real-life brain spectra. The individual classifiers are shown by stars and big diamonds. Small diamonds correspond to the combining classifiers. All aggregations were performed on the 8 individual classifiers, except for the FI classifiers, which were applied to the 4 individual classifiers shown as big diamonds. The definitions of the abbreviations are described in Table 3.3

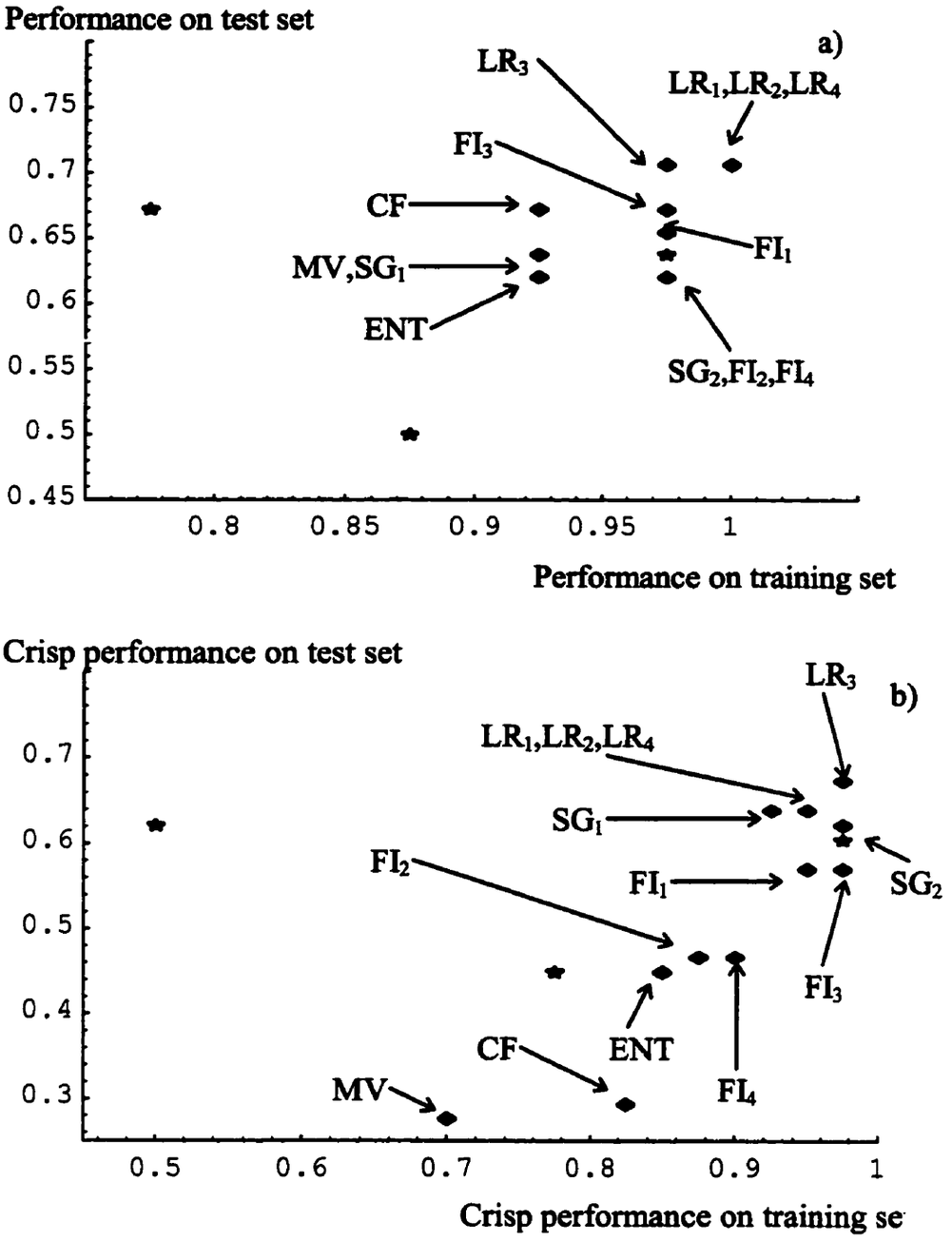

Fig. 4.2 Classification performance (a) and crisp performance (b) of the individual and combining classifiers on the training and test sets, cervical spectra. Individual classifiers are shown by stars. Small diamonds correspond to combining classifiers. The definitions of the abbreviations are described in Table 3.3

- each normalized and unnormalized spectrum was condensed into 55 consecutive regions, and the mean value of the spectrum in each region was calculated. The best 8 regions were selected by dynamic programming

The preprocessed spectra were classified by the ANN, LDA and QDA classifiers, and 8 classifications were selected from all obtained classifications. The combining classifiers of Table 3.3 were applied to these classifications, and the results are shown in Fig. 4.3. Again, stars and big diamonds represent the individual classifications, small diamonds their aggregation. The FI classifiers were applied to the 4 individual classifiers (shown as big diamonds).

Most of the combining classifiers improved the classification accuracy on the training set. Only the LR and SG-combining classifiers improved the classification and crisp classification performance on the test set. The classification performance on the training set improved from 92.8% for the best individual classifier up to 98.8%, on the test set from 84.7% up to 87.8%. The crisp classification performance improved from 89.3% up to 98.8% on the training set, and from 83.2 for the best individual classifier up to 87.7% on the test set. The most successful combining classifier was the LR classifier, minimizing the MSE on the training set and the number of misclassified or fuzzily classified training samples together.

Three classifiers of different architecture (LDA, QDA and ANN) were applied to the same set of preprocessed spectra (the 8 best regions of the spectra selected from 55 regions by dynamic programming). These classifications were aggregated by the same set of combining classifiers. The results are presented in Fig. 4.4. Practically all aggregation

methods improved both classification and crisp classification accuracy on both training and test sets. Notice, that aggregating a fewer number of the individual classifications allows

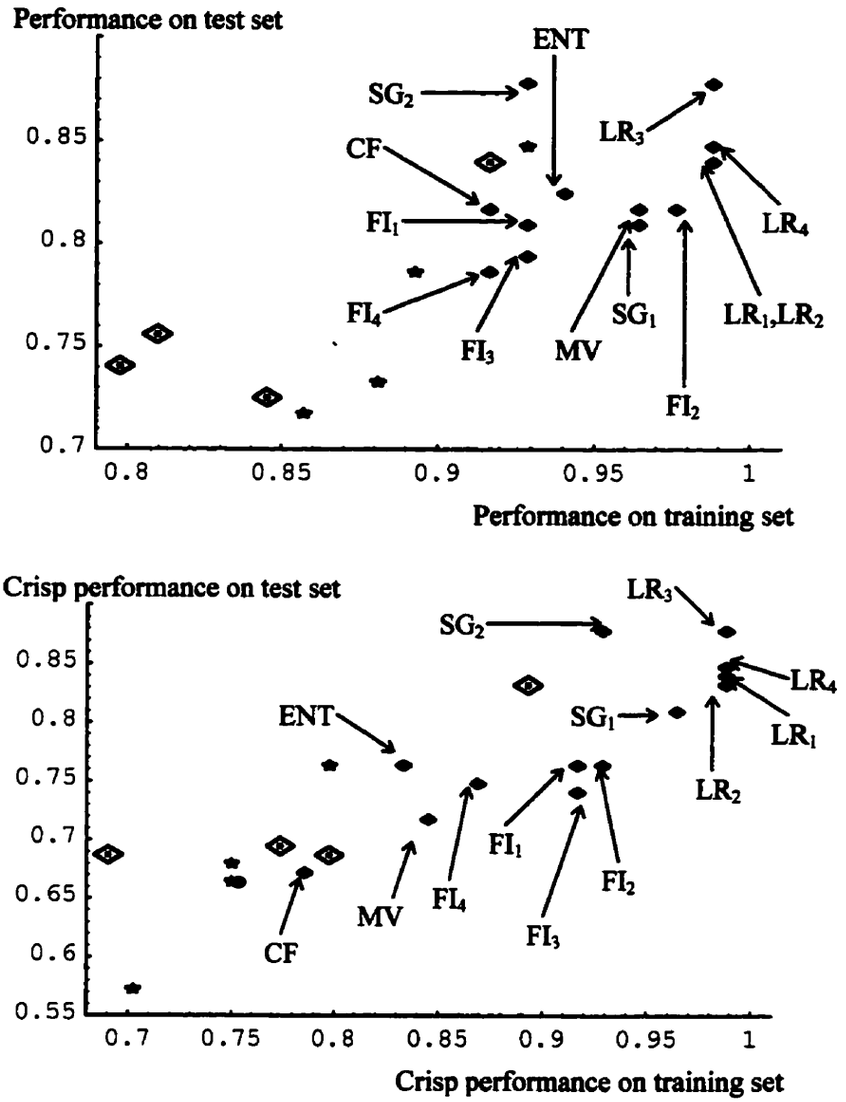

Fig. 4.3 Classification performance (a) and crisp performance (b) of the individual and combining classifiers on the training and test sets, real-life brain spectra. Individual classifiers are shown as stars and big diamonds. Small diamonds correspond to the combining classifiers. All aggregating techniques were applied to the 8 individual classifications, except for the FI classifiers, that were applied to 4 individual classifiers shown as big diamonds. The definitions of the abbreviations are described in Table 3.3

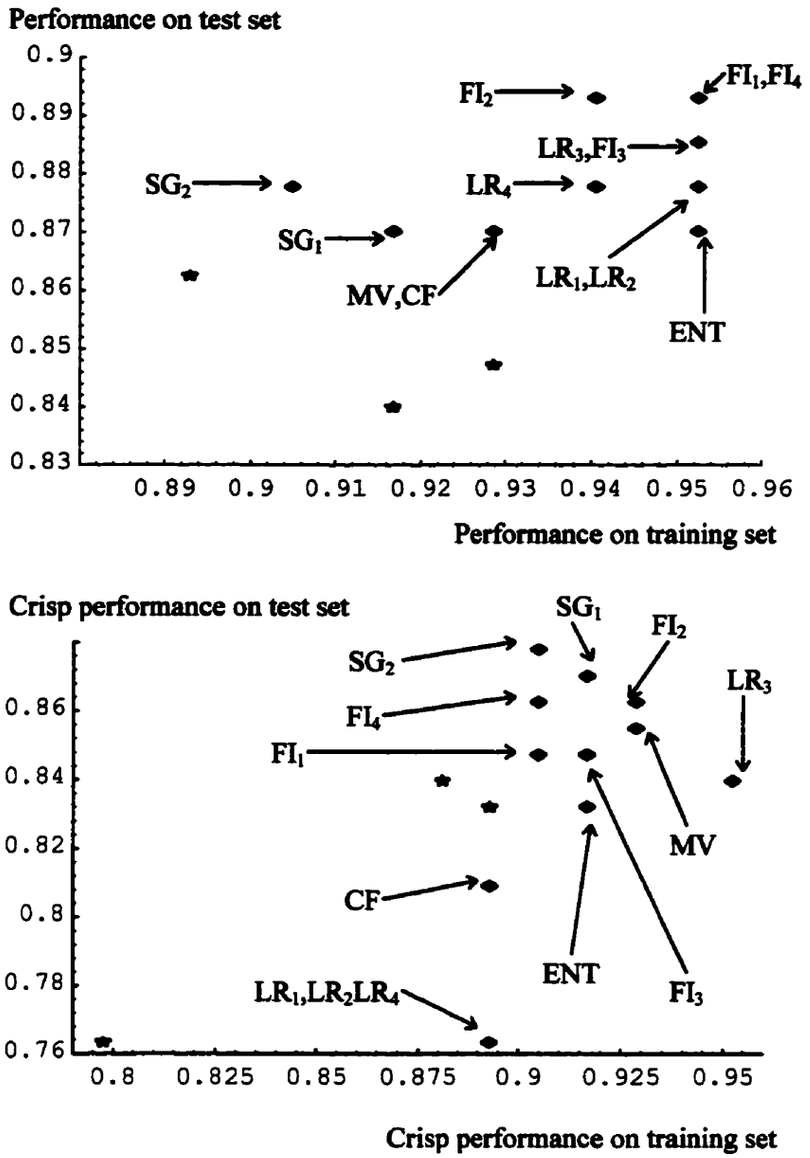

Fig. 4.4 Classification performance (a) and crisp performance (b) of the individual and combining classifiers on the training and test sets, real-life brain spectra. Three individual classifiers are shown by stars. Diamonds correspond to the combining classifiers. The definitions of the abbreviations are given in Table 3.3

for higher classification performance on the test set, than in the case of aggregating 8 classifications (89.3% vs 87.7%). Crisp classification performance on the test set remained the same.

These four examples demonstrated that in many cases aggregating individual classifications improves classification accuracy. Some combining classifiers perform well on one data set, and poorly on another. Thus, in order to achieve better performance different methods need be tried.

5. Results on Real-Life Infrared Spectra

A set of infrared (IR) spectra of blood serum samples was analyzed. The IR spectra have some advantages over proton MR spectra. There is no huge water peak, which always exists in the proton MR spectra. This peak is usually removed by water suppression from MR spectra. The IR spectra also have a better signal-to-noise ratio. They are simpler to acquire and less costly, so quite a few are available for the analysis. The 1362 analyzed spectra belong to 3 different classes (396 of them are of class 1 (normal), 326 of class 2 (hyperglycemia), and 640 of class 3 (hypertriglyceridemia, hypercholesterolemia, and lipometabolism)). Half of the spectra in each class constitute the training set, another half the test set. First the spectra were normalized. The centroid spectra of the three classes are presented in Fig. 5.1. The spectra within the classes are distributed about these centroid spectra. For example, the distribution of class 1 spectra is shown in Fig. 5.2. The centroid spectrum is shown in white on the black background of the spectra of individual samples. Before the aggregation methods were applied, the spectra were preprocessed:

- each original spectrum of 1816 points was condensed into either 182 or 91 consecutive regions, and the average/median values of the spectrum in each region were calculated
- the best 12 regions out of the 182/91 above were selected by dynamic programming
- PCA was applied to the spectra. The first 20 PCs were selected
- 10 regions which contribute mostly to the discrimination among the classes were selected in the normalized spectra by GA

and classified by the LDA, QDA, and KNN classifiers. A total of 26 classifications were obtained, four of them were selected for aggregating. They include: 3-NN classification

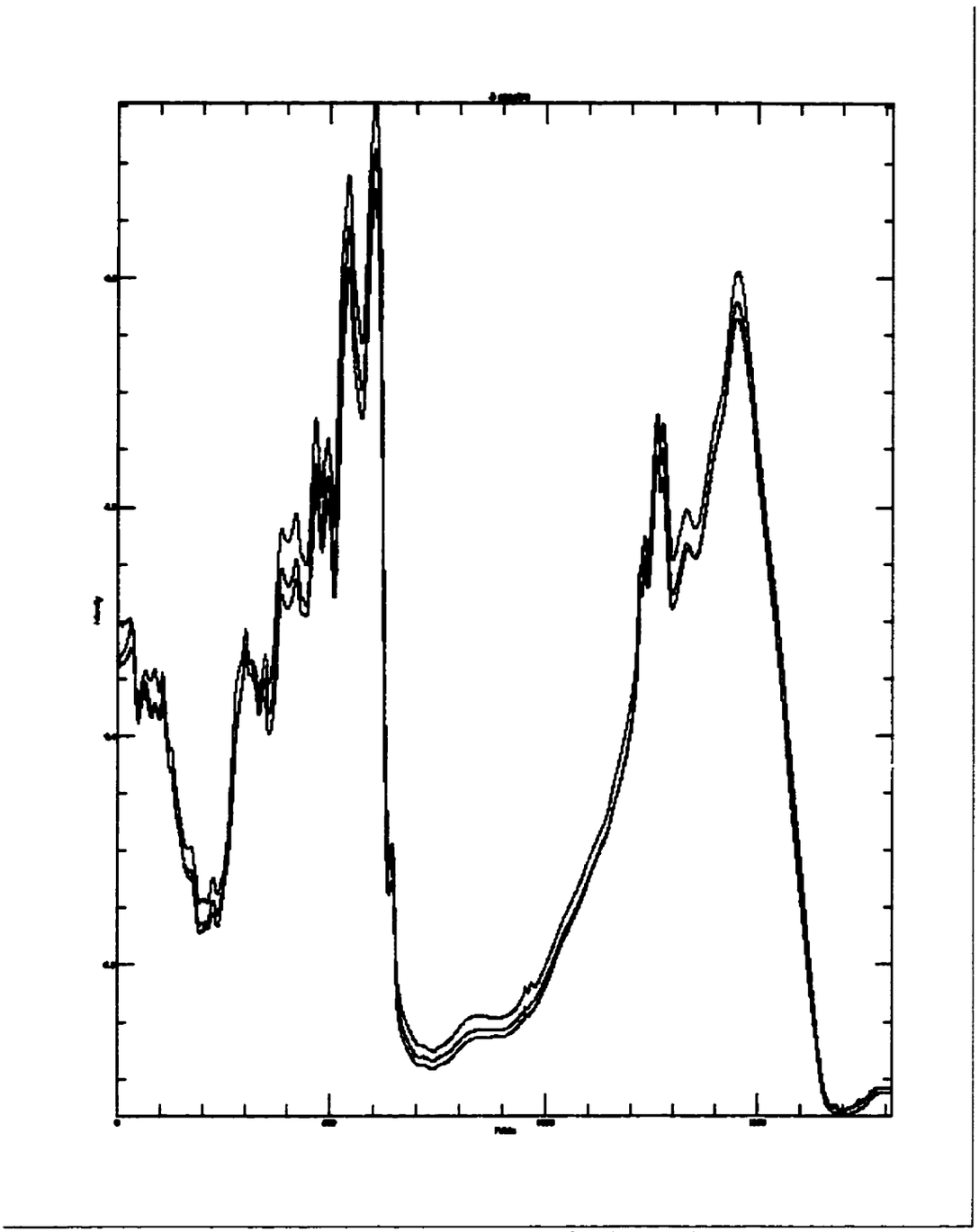

Fig. 5.1 The centroid spectra of 3 classes of IR spectra.

of normalized full-size spectra, the LDA applied to the spectra condensed into 182 regions with the mean value of the spectra in the regions, the LDA applied to the spectra condensed

into 182 regions with the median value of the spectra in the regions (this is the best individual classification), and the LDA applied to the 10 regions selected by GA.

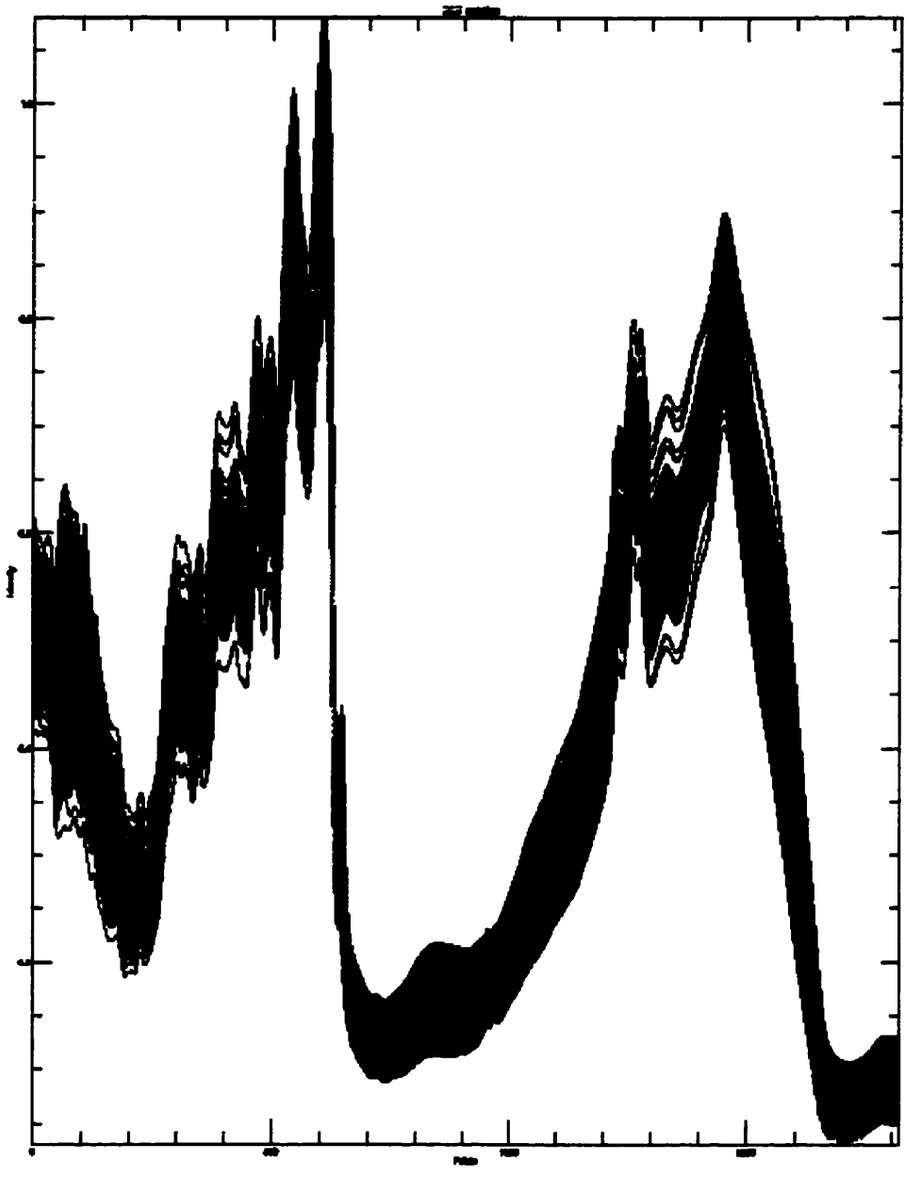

Fig 5.2 The spectra of class 1 distributed about the centroid spectrum, IR spectra. The centroid spectrum is shown in white on the black background of the individual spectra.

The aggregation methods of Table 3.3, which were found to work well in the case of artificial MR spectra were used for aggregating these classifications. The results of individual and combined classifications are presented in Fig. 5.3. The individual classifications are shown as stars, their aggregation as diamonds. All aggregation methods showed improved performance over that of the individual classifiers on both training and test sets. The best individual classifier gives 91.6% and 88.8% accuracy on the training and test sets respectively. The best performance achieved by the combining classifiers is 94.9% on the training set, and 91.8% on the test set. The SG, FI and some LR schemes also improved crisp classification performance. The best achieved crisp performance is 94.6% and 89.7% on the training and test sets respectively, vs. 90.2% and 87.4% for the best individual classifier. Other aggregation methods had crisp performance close to that of the best individual classifier.

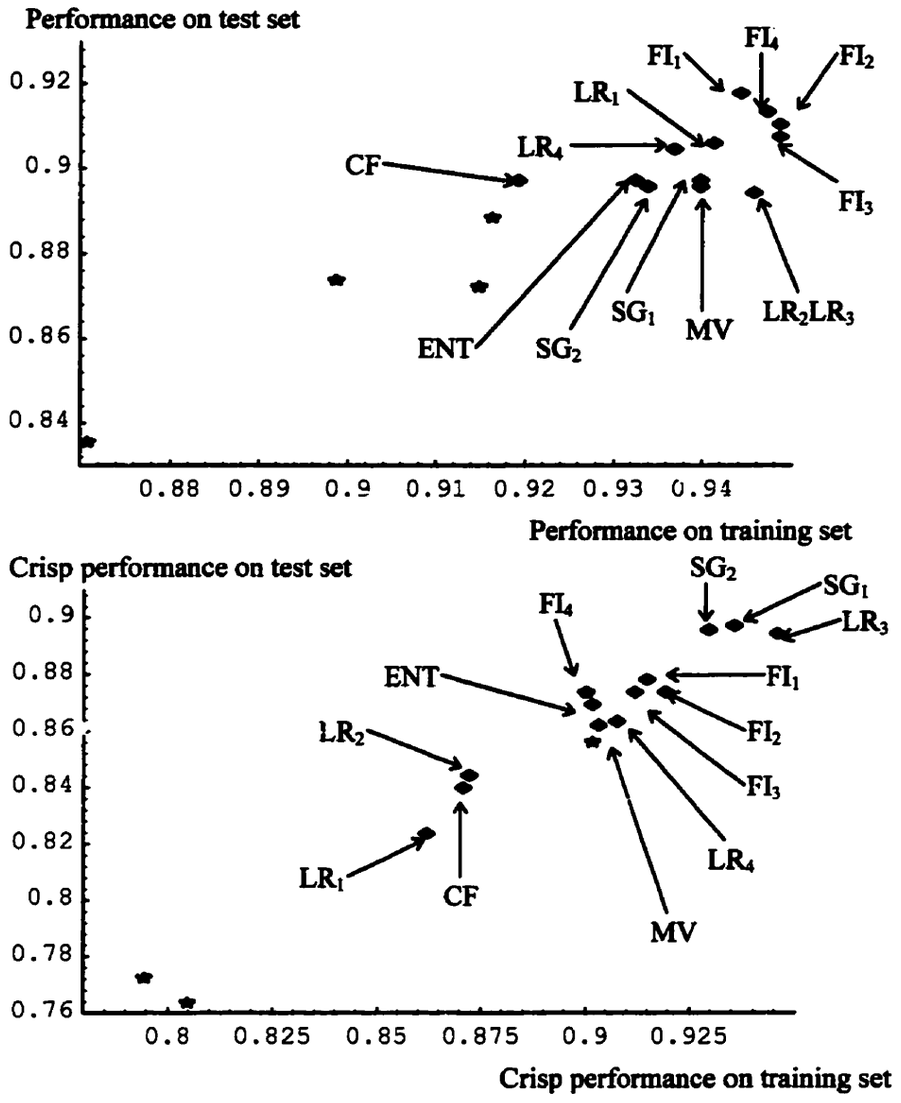

Fig. 5.3 Classification performance (a) and crisp classification performance (b) of the individual and combining classifiers, IR spectra. Stars correspond to the individual classifications, diamonds to their aggregation.

6. Conclusions

Different methods of aggregating classifiers were considered and applied to both artificial and real-life MR and IR spectra in this thesis. The author has developed, adapted and implemented[18] several aggregation methods, such as the LC of classifiers, LR and FI classifiers, entropy based, CF based and MV classifiers. Some aggregation methods require estimating parameters: weights for the LR and LC, and FMs for FI-combining classifiers. An objective function is minimized in order to do this. This process is called training the combining classifier. Several objective functions were considered in order to improve classification accuracy of the combining classifier on the training set. An aggregation method together with an objective function is referred to as an aggregation scheme. Several aggregation schemes (published in the literature and constructed by the author) were implemented for the LR and FI aggregation methods. Some of the suggested schemes showed an improvement in classification accuracy compared to schemes in the literature. In order to minimize the objective functions a nonlinear constrained optimization problem must be solved in general. A simplex minimization procedure with simulated annealing was used for this purpose. All aggregation methods and schemes were also compared among themselves.

The number of MR spectra available for analysis is often limited. The results of applying the aggregation methods to such data is strongly data dependent. In order to test the aggregating methods more objectively a set of artificial MR spectra was generated. The

[18] as C++ classes on SGI UNIX workstations

artificial spectra look similar to real-life ones. The set has sufficient number of spectra for a reliable analysis.

Different aggregating classifiers require different amounts of time to train. Training the LR and FI-combining classifiers may be very time consuming. Training the LC-combining classifier is much faster since it can be done by a matrix inversion. The time required to train the SG classifier depends on the aggregating classifier. The entropy, CF and MV-combining classifiers don't require any training. It has been found that simple and fast aggregating classifiers can perform as well as complicated and slow classifiers. However, in some cases simple classifiers perform poorly in comparison with the complicated classifiers applied to the same data. After the training is completed a new sample is classified by any of the classifiers in negligible time.

In order to get better performance from a combining classifier one has to aggregate individual classifiers that make errors in an uncorrelated manner i.e., different classifiers misclassify the same samples into different classes. One strategy is to apply classifiers of different architecture to the same data. Different classifiers look at the data from different points of view, resulting in different classifications. Another strategy is to preprocess the spectra differently and submit these to a single reliable classifier. Different preprocessing techniques may select different features which distinguish the samples among the classes. A method of selecting classifications for aggregating is proposed in this thesis. This method takes into account both classification performance of individual classifiers and correlation among them in making errors in such a way that high performing classifiers with minimal correlation are selected for aggregation.

Combining classifiers were applied to different sets of MR and IR spectra. The results indicate that these classifiers may in many cases lead to better classification performance than that of the individual classifiers. Combining classifiers may also result in more crisp classification. Just as it is difficult to choose the best classifier among different classifiers, it is also difficult to choose the best aggregation method. Different methods perform well on some data and poorly on others. In order to get high performance out of the combined classification it appears that both different preprocessing techniques and different aggregation methods must be tried.

BIBLIOGRAPHY

1. Tin Kam Ho, Jonathan J. Hull, Sargur N. Srihari. A Regression Approach to Combination of Decisions by Multiple Character Recognition Algorithms. *SPIE*, vol. 1661, pp. 137-145, 1992
2. James M. Keller, Paul Gader, Hossein Tahani, Jung-Hsien Chiang, Magdi Mohamed. Advances in Fuzzy Integration for Pattern Recognition. *Fuzzy sets and Systems*, vol. 65, pp. 273-283, 1994
3. Michael Grabisch, Jean-Marie Nicolas. Classification by Fuzzy Integrals: Performance and Tests. *Fuzzy Sets and Systems*, vol. 65, pp. 255-271, 1994
4. Lei Xu, Adam Krzyzak, Ching Y. Suen. Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 418-435, 1992
5. Wolpert D.H. Stacked Generalization, *Neural Networks*, vol. 5, pp. 241-259, 1992
6. Tin Kam Ho, Jonathan J. Hull, Sargur N. Srihari. Decision Combination in Multiple Classifier Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 1, pp. 66-75, 1994
7. Kagan Tumer and Joydeep Ghosh. Analysis of Decision Boundaries in Linearly Combined Neural Classifiers. *Pattern Recognition*, vo. 29, no. 2, pp. 341-348, 1996
8. S. Geman, E. Bienenstock, R. Doursat. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, vol. 4, no. 1, pp. 1-58, 1992
9. Kagan Tumer and Joydeep Ghosh. Theoretical Foundations of Linear and Order Statistics Combiners for Neural Pattern Classifiers. *Technical Report* 95-02-98, Computer and Vision Research Center, the University of Texas at Austin, 1995
10. Ronny Meir. Bias, Variance and The Combination of Estimators; The Case of Linear least Squares. *To appear in NIPS 7*, Morgan Kauffman, 1995
11. Galina Rogova. Combining the Results of Several Neural Networks Classifiers. *Neural Networks*, vol. 7, no. 5, pp. 777-781, 1994
12. Sherif Hashem and Bruce Schmeiser. Improving Model Accuracy Using Optimal Linear Combinations of Trained Neural Networks. *IEEE Transactions on Neural*

Networks, vol. 6, no. 3, pp. 792-794, 1995

13. Louisa Lam, Ching Y. Suen. Optimal Combinations of Pattern classifiers. *Pattern Recognition Letters*, vol. 16, pp. 945-954, 1995
14. W. Pedrycz. Fuzzy Sets in Pattern Recognition: Methodology and Methods. *Pattern Recognition*, vol. 23, n. 1/2, pp. 121-146, 1990
15. Joydeep Ghosh, Steven Beck, Chen-Chau Chu. Evidence Combination Techniques for Robust Classification of Short-Duration Oceanic Signals. SPIE Conf. on Adaptive and Learning Systems. *SPIE Proc.*, vol. 1706, pp. 266-276, Apr. 1992
16. Michel Grabisch, Hung T. Nguyen and Elbert A. Walker. Fundamentals of Uncertainty Calculi with Application to Fuzzy Inference. *Series B: Mathematical and Statistical Methods*, vol. 30, Kluwer Academic Publishers, 1995
17. William H. Press and Saul A. Teukolsky. Simulated Annealing Optimization over Continuous Spaces. *Computers in Physics*, pp. 426-429, Jul./Aug. 1991
18. William H. Press, et all. Numerical Recipes. The Art of Scientific Computing. *Cambridge University Press*, Cambridge, 1986
19. Volker Tresp and Michiaki Taniguchi. Combining Estimators Using Non-Constant Weighting Functions. In G. Tesauro, D. S. Touretzky and T. K. Leen, eds., "*Advances in Neural Information Processing Systems 7*", MIT Press, Cambridge MA, 1995
20. Sung-Bae Cho, Jin H. Kim. Combining Multiple Neural Networks by Fuzzy Integral for Robust Classification. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, no. 2, pp. 380-384, 1995
21. Witold Pedrycz. Fuzzy Control and Fuzzy systems. *Research Studies Press Ltd.* 1989
22. Kamal M. Ali, Michael J. Pazzani. Error Reduction through Learning Multiple Descriptions. *Technical Report 95-39*, University of California, Irvine, 1995
23. Kamal Ali and Michael Pazzani. On the Link between Error Correlation and Error Reduction in Decision Tree Ensembles. *Technical Report 95-38*, University of California, Irvine, 1995
24. Ron Kohavi and David H. Wolpert. Bias Plus Variance Decompositon fir Zero-One

loss Function. To appear in *Machine Learning: Proceedings of the Thirteen International Conference*, 1996

25. Murphy P.M. & Aha D.W. UCI repositary of machine learning databases. http://www.ics.uci.edu/~mlearn/
26. Leo Breiman. Bias, Variance, and Arcing Classifiers. *Technical Report,* Statistics Department, University of California at Berkeley, 1996
27. Xiru Zhang, Jill P. Mesirov and David L. Waltz. Hybrid System for Protein Secondary Structure Prediction. *J. Mol. Biol.*, vol. 225, pp. 1049-106, 1992
28. Robert A. Jacobs. Methods For Combining Experts' Probability Assessments. *Neural Computation*, vol. 7, pp. 867-888, 1995
29. Michael P. Perrone and Leon N. Cooper. When Networks Disagree: Ensemble Methods for Hybrid Neural Networks. To appear in *"Neural Networks for Speech and Image Processing"*, R.J. Mammone, ed., Chapman-Hall, 1993
30. Jurgen Schurmann. "Pattern Classification: A Unified View of Statistical and Neural Approaches", Willey-Interscience publication, 1996
31. Jerome H. Friedman. On Bias, Variance, 0/1 - loss, and the Curse-of-Dimensionality. *Technical Report*, Department of Statistics, Stanford University, 1996, ftp://playfair.stanford.edu/pub/friedman/curse.ps.Z
32. Robert Tibshirani. Bias, variance and prediction error for classification rules. *Technical Report*, Department of Statistics, University of Toronto, 1996
33. James M. Keller, Michael R. Gray and James A. Givens Jr. A Fuzzy *K*-nearest Neighbor Classifier. *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-15, no. 4, pp. 580-585
34 AllFit. Unpublished in-house computer package developed at the Institute for Biodiagnostics, National Research Council Canada.
35 Nikulin A., Briere K.M., Friesen L., Smith I.C.P. and Somorjai R.L. Genetic Algorithm - Guided Optimal Attribute Selection: A Novel Preprocessor for Classifying MR Spectra. *Proceedings of the Society of Magnetic Resonance*, Third Meeting, p. 1940, 1995
36 Somorjai R.L., Dolenko B., Nikulin A., Pizzi N., Scarth G., Zhilkin P., Briere K.M.

Donnelly S.M., Kuesel A.C., Halliday W., Fewer D., Hill N., Ross I., West M. & Smith I.C.P. A classification strategy for the robust analysis of MR spectra: application to diagnosis of brain neoplasms. *Proceedings of the Society of Magnetic Resonance*, p. 572, 1994

37 Zhilkin P.A. & Somorjai R.L. Application of Several Methods of Classification Fusion to Magnetic Resonance Spectra. *Connection Science*, vol. 8, No 3-4, p. 427, 1996

38 P.A. Zhilkin and R.L. Somorjai. Combined Classification of Artificial and Real Life Magnetic resonance Spectra. *Second International Conference on Computational Intelligence & Neurosciences*, Research Triangular Park NC, March 2-5, 1997

39 Peter A. Zhilkin & Ray L. Somorjai. Classifier Aggregation: Tests of both Artificial and Real-Life Magnetic Resonance Spectra. *Information Sciences, An International Journal*, submitted, 1997

40 NeuralWorks Professional II/PLUS. Computer package by NeuralWare, Inc.