Experiments with Classifier Combining Rules

Robert P.W. Duin, David M.J. Tax

Pattern Recognition Group, Department of Applied Physics
Delft University of Technology, The Netherlands[1]

Abstract. A large experiment on combining classifiers is reported and discussed. It includes, both, the combination of different classifiers on the same feature set and the combination of classifiers on different feature sets. Various fixed and trained combining rules are used. It is shown that there is no overall winning combining rule and that bad classifiers as well as bad feature sets may contain valuable information for performance improvement by combining rules. Best performance is achieved by combining both, different feature sets and different classifiers.

1 Introduction

It has become clear that for more complicated data sets the traditional set of classifiers can be improved by various types of combining rules. Often none of the basic set of traditional classifiers, ranging from Bayes-normal to Decision Trees, Neural Networks and Support Vector Classifiers (see section 3) is powerful enough to distinguish the pattern classes optimally as they are represented by the given feature sets. Different classifiers may be desired for different features, or may reveal different possibilities for separating the data. The outputs of the input classifiers can be regarded as a mapping to an intermediate space. A combining classifier applied on this space then makes a final decision for the class of a new object.

Three large groups of combining classifiers will be distinguished here as follows:

- *Parallel combining* of classifiers computed for different feature sets. This may be especially useful if the objects are represented by different feature sets, when they are described in different physical domains (e.g. sound and vision) or when they are processed by different types of analysis (e.g. moments and frequencies).
  The original set of features may also be split into subsets in order to reduce the dimensionality and hopefully the accuracy of a single classifier. Parallel classifiers are often, but not necessarily, of the same type.
- *Stacked combining* of different classifiers computed for the same feature space. Stacked classifiers may be of a different nature, e.g. the combination of a neural network, a nearest neighbour classifier and a parametric decision rule.

1. *Author's address:* Department of Applied Physics, Delft University of Technology, Lorentzweg 1, 2628CJ Delft, The Netherlands,
Tel: +31 (15) 278 6143. *Fax:* +31 (15) 278 6740.
E-mail: duin@ph.tn.tudelft.nl. *WWW:* http://www.ph.tn.tudelft.nl/~duin.

J. Kittler and F. Roli (Eds.): MCS 2000, LNCS 1857, pp. 16−29, 2000.

- Combining weak classifiers. In this case large sets of simple classifiers are trained (e.g. based on decision trees or the nearest mean rule) on modified versions of the original dataset. Three heavily studied modifications are bootstrapping (bagging), reweighting the data (boosting) and using random subspaces.

For all cases the question arises how the input classifiers should be combined. Various possibilities exist, based on fixed rules like maximum selection, product and majority voting. In addition one may also train a classifier, treating the classifier outputs in the intermediate space as feature values for the output classifier. An important condition is that the outputs of the input classifiers are scaled in one way or another such that they constitute the intermediate space in some homogeneous way.

In this paper we illustrate some issues of combining on a large example. This example has partially been published before [8] in the context of a review on the entire field of statistical pattern recognition. Here additional details will be given, together with a more extensive analysis that, due to lack of space, could not be presented in the original paper. In the next sections the data, the input classifiers and the output classifiers will be presented. Next the results are discussed and analysed. We like to emphasize that it is not our purpose to classify the given dataset optimally, in one way or another. It is merely our aim to illustrate various combining possibilities and analyse the effects on the performance. Leading questions in this analysis will be: when are which combining rules useful? How does this depend on the dataset? How do the input classifiers have to be configured? What is the influence of the combining rule on the final result?

2 The data set

The experiments are done on a data set which consists of six different feature sets for the same set of objects. It contains 2000 handwritten numerals extracted from a set of Dutch utility maps. For each of the ten classes'0', ... ,'9' a set of 200 objects is available. In all experiments we assumed that the 10 classes have equal class probabilities P_j = 0.1, j = 1, ..., 10. Each of the classes is split in a fixed set of 100 objects for learning and 100 for testing. Because of computational limitations, we use a fixed subset of only 50 objects per class for training. The six feature sets are:

- Fourier: 76 Fourier coefficients of the character shapes.
- Profiles: 216 profile correlations.
- KL-coef: 64 Karhunen-Loève coefficients.
- Pixel: 240 pixel averages in 2 x 3 windows.
- Zernike: 47 Zernike moments.
- Morph: 6 morphological features.

A slightly different version of this data set has been used in [9]. The presently used data is publicly available under the name 'mfeat' in the Machine Learning Repository [11].

All characters are originally sampled in a 30*48 binary image. The features are all computed from these images and are therefore not strictly independent. In figure 1 the performance for the Fisher classifier is shown for the first 9 principal directions in the Fisher map (i.e. the subspace that maximizes the between scatter of the classes over the averaged within scatter). In figure 2 the 2-dimensional Fisher maps are shown. It is

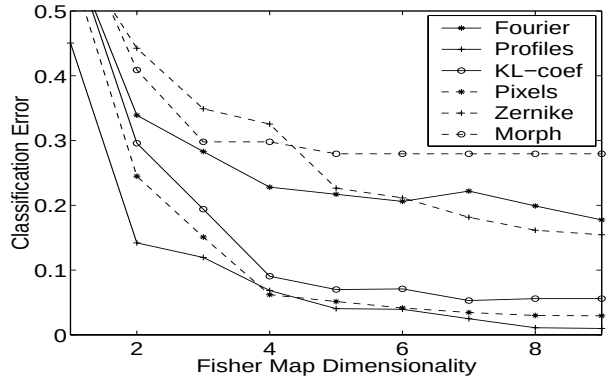

Fig. 1 The Fisher classification error for the six feature sets optimally projected on low-dimensional subspaces.

hoped these mappings find the most separating directions in the data, and thus reveal the clusters in the data. Each class is labelled in these figures by one unique marker in all datasets. In the Morph dataset the features have discrete values. One class can be separated but for other classes (e.g. one with the white circular label) the discrete feature deteriorates the cluster characteristics. In most feature sets nice clusters can be distinguished. The scaling of the features is comparable over all feature sets. This is caused by the fact that as a part of the Fisher mapping the data is prewithened to unit variance.

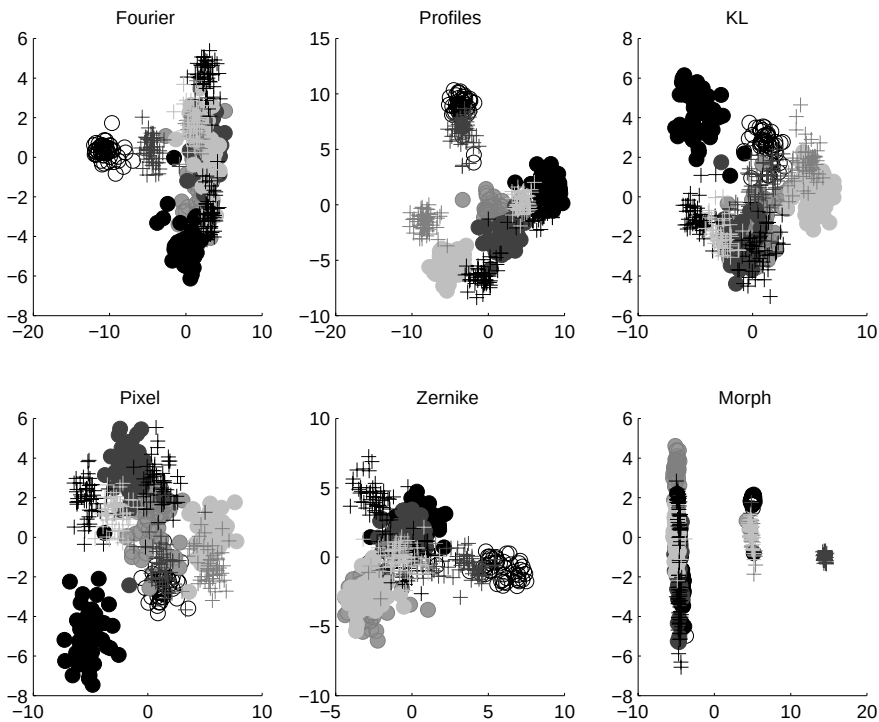

Fig. 2 Scatter plots of all six datasets, mapped on the 2-dimensional Fisher map

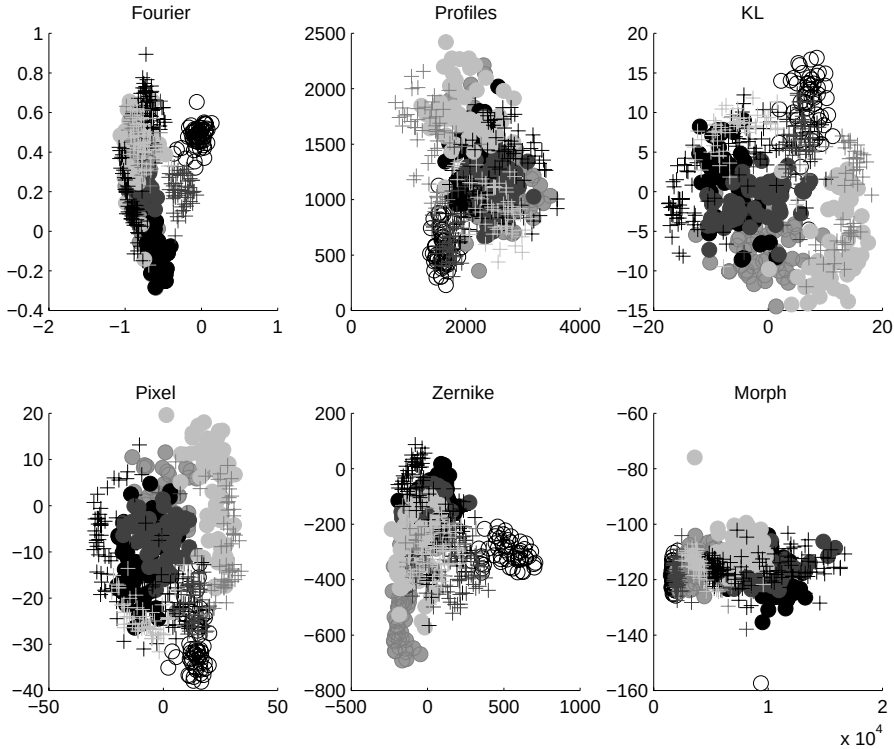

Fig. 3 Scatter plots of all six datasets, mapped on their first 2 principal components

In figure 3 scatter plots of the first two principal components (PCA) of the six datasets are shown. Note the differences in scaling for the various feature sets, which is preserved in these mappings. The PCA plots are focused on the data distributions as a whole, while the Fisher mapping emphasizes the class differences. Although it is not possible to extract quantitative features from these plots, they show that the data sets have quite distinct class distributions.

3 The classifiers

For this experiment we used a set of off-the-shelf classifiers taken from our Matlab toolbox PRTools [12]. They were not optimized for the particular application. In this way they illustrate well the differences between these classifiers, and, moreover, it serves better the aim to study the effects of combining classifiers of various performances. As argued in the introduction, it is important to make the outputs of the classifiers comparable. We use estimates for the posterior probabilities or confidences. This is a number $p_j(\boldsymbol{x})$, bounded between 0 and 1 computed for test objects $\boldsymbol{x}$ for each of the c classes the classifiers are trained on. These numbers are normalized such that:

$$\sum_{j}^{c} p_j(\boldsymbol{x}) \;=\; 1 \tag{4}$$

We will now shortly discuss the set of basic classifiers.

Bayes-normal-2: This is the Bayes rule assuming normal distributions. For each class a separate covariance matrix is estimated, yielding quadratic decision boundaries. Because of the size of the training set (50 objects per class) and the large set of features, regularization is necessary. This was done by estimating the covariance matrix C for the scatter matrix S by

$$C = (1 - \alpha - \beta)S + \alpha \mathrm{diag}(S) + \frac{\beta}{n} \sum \mathrm{diag}(S) \tag{5}$$

in which n is the dimensionality of the feature space. We used $\alpha = \beta = 10^{-6}$. Posterior probabilities are computed from the estimated class densities $f_j(\boldsymbol{x})$:

$$p_j(\boldsymbol{x}) = \frac{P_j f_j(\boldsymbol{x})}{\sum_i P_i f_i(\boldsymbol{x})} \tag{6}$$

Bayes-normal-1: This rule is similar to Bayes-normal-2, except that all classes are assumed to have the same covariance matrix. The decision boundaries are thereby linear.

Nearest Mean: Objects are assigned to the class of the nearest mean. Posterior probabilities are estimated using a sigmoid function over the distance. This is optimized over the training set using the maximum likelihood rule [10].

Nearest Neighbour (1-NN): Objects are assigned to the class of the nearest object in the training set. Posterior probabilities are estimated by comparing the nearest neighbour distances for all classes [10].

k-Nearest Neighbour (k-NN): Objects are assigned to the class having the majority in the k nearest neighbours in the training set. For $k > 2$ posterior probabilities are estimated using the class frequencies in the set of k neighbours. For $k = 1$, see the 1-NN rule. The value of k is optimized for the training set using a leave-one-out error estimation.

Parzen Classifier: Class densities are estimated using Gaussian kernels for each training object. The kernel width is optimized for the training set using a leave-one-out error estimation. Posterior probabilities are computed according to (6).

Fisher's Linear Discriminant (FLD): We computed a FLD between each of the 10 classes and all other classes. For each of these classifiers posterior probabilities are computed using a sigmoid over the distance to the discriminant. These sigmoids are optimized separately over the training set using the maximum likelihood rule [3]. Test objects are assigned to the class with the highest posterior probability. For two-class problems, this classifier is almost equivalent (except for the way posterior probabilities are computed) to Bayes-normal-1. For multi-class problems these rules are essentially different.

Decision Tree: Our algorithm computes a binary decision tree on the multi-class dataset. Thresholds are set such that the impurity is minimized in each step [1]. Early pruning is used in order avoid overtraining [2]. Posterior probabilities are estimated by the class frequencies of the training set in each end node.

Artificial Neural Network with 20 hidden units (ANN-20): This is a standard feedforward network with one hidden layer of 20 neurons and sigmoid transfer functions.

The network is trained by the back-propagation rule using the Matlab Neural Network toolbox [13]. The 10 output values are normalized and used as posterior probabilities.

Artificial Neural Network with 50 hidden units (ANN-50): The same algorithm as above, but now using 50 neurons in the hidden layer.

Support Vector Classifier with linear kernel (SVC-1): This is the standard SVC using a linear inner product kernel [3]. The computation of the multi-class classifier as well as the posterior probabilities is similar to the procedure described above for the FLD.

Support Vector Classifier with quadratic kernel (SVC-2): In this case the squares of the inner products are used as a kernel, resulting in quadratic decision boundaries.

4 The combining rules

Once a set of posterior probabilities $\{p_{ij}(\boldsymbol{x}), \mathrm{i} = 1,m; \mathrm{j} = 1,c\}$ for m classifiers and c classes is computed for test object $\boldsymbol{x}$, they have to be combined into a new set $q_j(\boldsymbol{x})$ that can be used, by maximum selection, for the final classification. We distinguish two sets of rules, fixed combiners and trained combiners.

4.1 Fixed combining rules

Fixed combiners are heavily studied in the literature on combining classifiers, e.g. see [4], [5] and [6]. The new confidence $q_j(\boldsymbol{x})$ for class j is now computed by:

$$q_j'(\boldsymbol{x}) = \mathrm{rule}_i(p_{ij}(\boldsymbol{x})) \tag{7}$$

$$q_j(\boldsymbol{x}) = \frac{q_j'(\boldsymbol{x})}{\sum_j q_j'(\boldsymbol{x})} \tag{8}$$

The following combiners are used for rule in (7): **Maximum, Median, Mean, Minimum, Product.** Note that the final classification is made by

$$\omega(\boldsymbol{x}) = \mathrm{argmax}_j(q_j(\boldsymbol{x})) \tag{9}$$

The Maximum rule selects the classifier producing the highest estimated confidence, which seems to be noise sensitive. In contrast, the Minimum rule selects by (9) the classifier having the least objection. Median and Mean average the posterior probability estimates thereby reducing estimation errors. This is good, of course, if the individual classifiers are estimating the same quantity. This probably will not hold for some of the classifiers discussed in section 3.

A popular way of combining classifiers is **Majority**: count the votes for each class over the input classifiers and select the majority class. This fits in the above framework if this rule is substituted in (7):

$$q_j'(\boldsymbol{x}) = \sum_i I(\mathrm{argmax}_i(p_{ij}(\boldsymbol{x})) = i) \tag{10}$$

in which $I()$ is the indicator function: $I(y) = 1$ if y is true and $I(y) = 0$ otherwise.

4.2 Trained combining rules

Instead of using fixed combination rules, one can also train an arbitrary classifier using the m x c values of $p_{ij}(\boldsymbol{x})$ (for all i and all j) as features in the intermediate space. The following classifiers are trained as an output classifier, using the same training set as we used for the input classifiers, see section 3: **Bayes-normal-2**, **Bayes-normal-1**, **Nearest Mean** and **1-NN**.

It is point of discussion whether it is wise to use the posterior probabilities directly for building the intermediate feature space. The classes may be far from normally distributed. It might therefore be advantageous to apply some nonlinear rescaling. The use of the Nearest Mean classifier on the posterior probabilities is almost equivalent to the procedure of fuzzy template matching as investigated by Kuncheva et. al. [7].

5 The experiment

In table 1 the obtained test errors x 1000 are listed. The top-left section of this table lists the results for all 12 individual classifiers for all feature sets combined and for the 6 feature sets individually. The combined result is only occasionally somewhat better than the best individual result. The is caused by the high dimensionality of the combined set (649) as well as by differences in scaling of the features. The best results for each feature set separately (column) by an individual classifier are underlined. For instance, an error of 0.037 is obtained, among others, by the 1-NN rule for the Pixels dataset. Because the entire test set contains 10 x 100 = 1000 objects the number 37 is in fact the number of erroneously classified test objects. Due to the finite test set this error estimate has a standard deviation of $\sqrt{0.037 \times (1 - 0.037) / 1000} = 0.006$, which is not insignificant. All error estimates, however, are made by the same test set and are thereby not independent.

The bottom-left section of the table deals with the stacked combining rules applied on all 12 input classifiers for each feature set separately. Again, all best results for each feature set are underlined. It appears that the Majority rule frequently scores a best result. In addition all combined results that are better than the best individual classifier are printed in bold. For instance, for the Zernike feature set, the best individual classifier is Bayes-normal-1 (0.180). Combining all classifiers using the Median rule, however, improves this result (0.174). This combination rule is thereby useful.

In the entire right half of the table the results of parallel combining are shown. Combining rules are applied on the 6 results for a single classification rule (e.g. Bayes-normal-2), obtaining an error of 0.068 for the Product rule. The results of the combined set of all 649 features (first column) are not used here. The best results over the 10 combining rules for each classifier are underlined. For instance, the Median rule yields the best combining result (0.028) of all combiners of the Bayes-normal-2 input classifiers. Again all combining results that are better than the results for individual classifiers (now compared over all feature sets) are printed in bold. All these combined classifiers are also better than those the same input classifier trained by the entire feature set. E.g., the Parzen classifier trained on all features simultaneously yields an error of 0.036 (which is better than the best individual feature set result obtained by Parzen of 0.037).

Table 1: Summary of experimental results (error x 1000)[a]

		Feature Set							Fixed Combiners						Trained Combiners			
	Classifiers	All (649)	Fourier (76)	Profiles (216)	KL-coef (64)	Pixels (240)	Zernike (47)	Morph (6)	Maximum	Median	Mean	Minimum	Product	Majority	Bayes-normal-2	Bayes-normal-1	Nearest Mean	1-NN
Basic Classifiers	Bayes-normal-2	52	257	58	128	62	212	310	200	**28**	63	84	63	68	701	799	67	**50**
	Bayes-normal-1	183	213	34	57	99	180	291	52	37	34	44	**31**	51	75	99	39	42
	Nearest Mean	87	224	181	99	96	278	540	540	**62**	**45**	**80**	**46**	**75**	124	**20**	103	**46**
	1-NN	36	192	90	44	37	197	570	569	**26**	**17**	**35**	**17**	40	46	**29**	113	**30**
	k-NN	36	189	92	44	37	193	510	192	54	60	82	42	51	97	**27**	**36**	**26**
	Parzen	36	171	79	37	37	185	521	37	**29**	**32**	**29**	**27**	51	**36**	37	**31**	**31**
	Fisher	408	248	47	82	153	210	282	**39**	**32**	**33**	65	52	57	48	**45**	**35**	**36**
	Dec. Tree	480	454	403	400	549	598	329	**275**	**134**	**113**	**262**	**110**	**218**	**283**	**104**	**102**	**108**
	ANN-20	896	900	46	146	852	900	328	900	177	784	900	900	327	**45**	**27**	**26**	**21**
	ANN-50	900	245	130	823	810	265	717	692	244	290	805	807	163	**42**	**31**	**55**	**33**
	SVC-1	500	246	66	61	77	294	848	123	108	**59**	190	101	**47**	74	69	**60**	**58**
	SVC-2	96	212	51	40	60	193	811	40	**36**	**37**	42	**38**	**38**	40	40	40	40
Fixed Combiners	Maximum	105	747	42	39	44	839	436	39	**22**	**21**	53	**20**	76	**33**	**27**	**19**	**19**
	Median	**34**	190	43	**36**	45	**174**	287	50	**23**	**25**	39	**25**	50	195	58	**19**	50
	Mean	53	190	35	45	56	**176**	285	**34**	**20**	**21**	37	**20**	46	57	**34**	**20**	**19**
	Minimum	315	790	137	109	200	737	652	138	165	135	131	135	160	**58**	**71**	**72**	**56**
	Product	215	294	131	44	82	401	412	86	234	86	84	86	568	47	860	851	685
	Majority	**33**	175	35	**32**	37	**169**	318	34	**23**	**20**	122	**20**	48	198	**27**	**21**	**20**
Trained Combiners	Bayes-normal-2	104	273	49	44	99	195	289	244	**17**	115	133	115	**28**	822	897	129	64
	Bayes-normal-1	60	427	51	40	53	190	294	160	**24**	41	49	41	**26**	107	656	56	63
	Nearest Mean	**32**	198	37	46	73	181	**266**	133	**20**	**19**	**36**	**19**	51	79	42	**15**	**18**
	1-NN	**33**	186	38	41	72	**170**	328	212	**18**	**18**	38	**18**	41	49	**36**	**19**	**18**

a. ©IEEE

Several combiners of these Parzen classifiers, however, yield even better results, e.g. 0.027 by the Product rule. This shows that combining of classifiers trained on subsets of the feature space can be better than using the entire space directly.

6 Analysis

Our analysis will be focused on the combining of classifiers. The performances of the individual classifiers as shown in the top left section of table 1, however, constitute the basis of this comparison. These classifiers are not optimized to the data sets at hand, but are used in their default configuration. Many classifiers would have performed better if the data would have been rescaled or if other than default parameter values would have been used. Especially the disappointing performances of the Decision Tree (most likely by the way of pruning) and some of the neural networks suffer from this. It is interesting to note that the use of the combined feature sets yields for some classifiers a result better than by using each of the feature sets individually (Bayes-normal-2, Nearest Mean, 1-NN, k-NN) and for other classifiers a result worse than by using each of the feature sets individually (Fisher, ANN-50, SVC-1).

The first thing to notice from table 1 is that combining the results of one classifier on different feature sets is far more effective than combining the results of different classifiers on one feature set. Clearly the combination of independent information from the different feature sets is more useful than the different approaches of the classifiers on the same data. This is also visible in the performances of the different combining rules. For combining the results of classifiers on independent feature sets the product combination rule is expected to work very well. Kittler [4] showed that a product combination rule especially improves the estimate of the posterior probability when posterior probabilities with independent errors are combined. Unfortunately this product rule is sensitive to badly estimated posterior probabilities. One erroneous probability estimation of $p = 0$ overrules all other (perhaps more sensible) estimates. On the other hand for combining posterior probabilities with highly correlated errors, the product combination rule will not improve the final estimate. Instead a more robust mean, median rule or even a majority vote rule is expected to work better. These rules are not very sensitive to very poor estimates.

The results of combining the feature sets show that the product rule gives good results. The product combination of the Bayes-normal-1, 1-NN and Parzen results gives (one of) the best performances over all combination rules. The posterior probabilities of these classifiers on these feature sets appear to have independent errors. The combination results for classifiers trained on the same feature set reflect the fact that here the errors are very correlated. From the fixed combination rules only the majority vote and the mean/median combination improve the classification performance. The product rule never exceeds the best performance obtained by the best individual classifier.

When the different classifiers from one feature set are combined, performance is only improved in case of the Zernike and KL feature sets. For the other feature sets the combining performances are worse than the best individual classifier. For the Morph feature set all combining rules fail except for the Nearest Mean rule which is slightly better than the best individual performance. The maximum, product and especially the

minimum rule perform very poorly in combining the different classifiers. These rules are extra sensitive to poorly estimated posterior probabilities, and suffer therefore by the poor performance of the Decision Trees and ANNs.

The Bayes-normal-1 and Bayes-normal-2 combinations probably suffer from the fact that they are trained on the (normalized) posterior probabilities of the first level classifiers. The distributions in the new 120 dimensional space (12 classifiers times 10 classes) are bounded in an unit hypercube, between 0 and 1. Moreover many of these estimated probabilities are 1 or 0 such that most objects are in the corners of the cube. The model of a normal distribution is therefore violated. Remapping the probability to a distance (e.g. by the inverse sigmoid) might remedy the situation.

The best overall performance is obtained by combining both, all classifiers and all feature sets. Although combining the classifiers trained on one feature set does not improve classification performance very much (only the majority and mean combining rules give improvement), combining again over all feature sets show the best performances. Both the product and the median rules work well and give the best overall performances, in the order of 2% error. Only the results obtained by the minimum and product combination is very poor. These outputs are too contaminated by bad posterior probability estimates. Finally the trained combination rules on the results of the fixed combinations of the different classifiers work very well, while trained combiners on the trained Bayes-normal combination of the classifiers seems to be overtrained. Combining the very simple classifier Nearest Mean with a Nearest Mean gives the overall lowest error of 1.5%. To obtain this performance, all classifiers have to be trained on all feature sets. Good performance can already be obtained when an 1-NN classifier is trained on all feature sets and the results are combined by mean or product rule. This gives an error of 1.7%, slightly but not significantly worse than the best 1.5% error.

Combining the estimates of the Parzen classification consistently gives good results, for almost all combining rules. The Parzen density estimation is expected to give reasonable estimates for the posterior probabilities, and is therefore very suitable to be combined by the fixed combining rules. The 1-NN classifier on the other hand gives very rough posterior probability estimates. The fact that these probabilities are estimated in independent feature spaces, cause independent estimation errors, which is corrected very well by the combination rules. Only the maximum combination rule still suffers from the poor density estimates in case of the 1-NN classifiers, while for the Parzen classifiers performance of the maximum rule is very acceptable. In some situations the combining rules are not able to improve anything in the classification. For instance all fixed combination rules perform worse on the ANN combination than the best individual classifier. Also combining SVC-1 and Bayes-normal-2 by fixed combining rules hardly give any performance improvements. For other situations all combination rules improve results, for instance the combination of the Decision Trees. Individual trees perform very poorly, but combining significantly improves them (although performance is still worse than most of the other combination performances). Performances tend to improve when the results of the Decision Trees, Nearest Mean, 1-NN and Parzen classifier are combined.

It is interesting to remark that, similar to the Parzen classifier, all results of the combined Decision Trees, although on a low performance level, improve those of the

separate trees. Obviously the posterior probabilities, estimated in the cells of the Decision Trees can be combined well in almost any way. This may be related to the successes reported in combining large sets of weak classifiers often based on Decision Trees.

The trained combination rules work very well for combining classifiers which are not primarily constructed to give posterior probabilities, for instance the ANN, Nearest Mean and SVC. This can also be observed in the combinations of the maximum, minimum and mean combination of the different classifiers. Especially for the minimum rule the trained combination can in some way 'invert' the minimum label to find good classification performance.

Table 2: Combining classifiers for good feature sets only (error x 1000)

	Feature Set			Fixed Combiner						Trained Combiner			
	Profiles (216)	KL-coef (64)	Pixels (240)	Maximum	Median	Mean	Minimum	Product	Majority	Bayes-normal-2	Bayes-normal-1	Nearest Mean	1-NN
Bayes-normal-2	58	128	62	60	59	**48**	64	**47**	**55**	889	897	**51**	59
Bayes-normal-1	34	57	99	75	51	64	74	75	46	103	99	86	93
Nearest Mean	181	99	96	165	96	**91**	102	**91**	98	**92**	**47**	179	**75**
1-NN	90	44	37	80	**36**	**36**	**33**	**36**	37	61	65	70	37
k-NN	92	44	37	92	37	66	41	66	38	67	66	90	72
Parzen	79	37	37	39	**34**	**36**	40	39	**33**	46	37	**36**	37

From the left upper part in table 1 it is clear that the data in the Profiles, KL-coefficients and Pixel feature sets is better clustered and easier to classify than the Fourier, Zernike and Morph features. Therefore one might expect that for these datasets the posterior probabilities are estimated well. In table 2 six classifiers are trained on only the good feature sets and then combined. In all cases the performances of the combination rules are significantly lower than the individual best classifier. On the other hand, the results are worse than the combination of all six original feature sets. This indicates that although the individual classification performances on the 'difficult' datasets are poor, they still contain valuable information for the combination rules.

Surprisingly the best performance is now obtained by applying the minimum rule on the 1-NN outputs or the majority vote on the Parzen outputs. Only in one case the product combination rule is best: in the combination of the Bayes-normal-2. There is no combining rule which gives consistently good results. The overall performance improvement is far less than in the case of the combination of the six feature sets.

It appears to be important to have independent estimation errors. The different feature sets describe independent characteristics of the original objects. Within the feature sets the features have a common, comparable meaning but between the sets the features

are hardly comparable (see also the differences in the PCA scatter plots in figure 3). The classifiers which are used in this experiment, do not use prior knowledge about the distribution in the feature sets. When all features of the six feature sets are redistributed into six new sets, again the classifiers can be trained and combined. The results are shown in table 3. The first notable fact is that the performances of the individual classifiers over the different feature sets are far more comparable. This indicates that the distribution characteristics of the sets do not differ very much. Furthermore the Bayes-normal-1 works very well in all feature sets. This indicates that data described with the large set of (mostly) independent features (more than 100) tends to become normally distributed.

Table 3: Randomized feature sets (error x 1000)

	Feature Sets						Fixed Combiner						Trained Combiner			
	Set 1(108)	Set 2(108)	Set 3(108)	Set4(108)	Set 5(108)	Set 6(109)	Maximum	Median	Mean	Minimum	Product	Majority	Bayes-normal-2	Bayes-normal-1	Nearest Mean	1-NN
Bayes-normal-2	104	134	83	123	110	122	**55**	**57**	**55**	97	**57**	**56**	897	608	**57**	**62**
Bayes-normal-1	26	35	25	49	44	30	28	**19**	**18**	**24**	**19**	**18**	82	**18**	**18**	**19**
Nearest Mean	164	181	109	142	149	163	117	**92**	**89**	124	**89**	**99**	396	**46**	158	**63**
1-NN	87	93	69	86	101	83	**67**	**34**	**31**	**40**	**31**	**42**	**50**	**57**	**48**	**34**
k-NN	83	93	68	86	94	83	**59**	**38**	**46**	**43**	**45**	**40**	133	**50**	**47**	**56**
Parzen	75	83	65	74	95	76	**37**	**31**	**31**	**37**	**31**	**40**	119	67	**31**	**31**

The results of combining these six new random sets, are comparable with the old, well defined feature sets. Results of combining 1-NN and Parzen are again good, but also combining k-NN and Bayes-normal-1 works well. Combining the Bayes-normal-1 classifiers works very well and even gives similar results as the best combining rules on the original feature sets. This may be interesting as this method is fast, in training as well as in testing. This good performance of combining classifiers trained on randomly selected feature sets corresponds with the use of random subspaces in combining weak classifiers. The results of the 1-NN and Parzen combinations are quite acceptable, but are not as good as in the original feature sets. Probably these classifiers suffer from the fact that distances within one feature set are not very well defined (by the differences in scale in the original feature sets, which are now mixed). The combined performance is therefore not much better than the 1-NN and Parzen on the combined feature set with 649 features (left column in table 1).

So we can conclude that here, instead of carefully distinguishing the six separate feature sets, we can train Bayes-normal-1 on random (disjunct) subsets of all features

and combine the results using (almost) any combining rule. This gives comparable results as combining the results from the 1-NN on the original feature sets with a mean or product rule.

Table 4: Combining the best classifiers (error x 1000)

	Fixed Combiner						Trained Combiner			
	Maximum	Median	Mean	Minimum	Product	Majority	Bayes-normal-2	Bayes-normal-1	Nearest Mean	1-NN
The best classifier for each feature set	37	24	29	28	26	40	44	52	31	28

Finally in table 4 the best individual classifiers are selected and combined (for the Pixel feature set the Parzen classifier is chosen). All combining rules perform very good, although the best performance does not match the best results in the original combination of the 1-NN classifier. This results might even be somewhat biased, because the best classifiers are selected by their performance on the independent test set. The best performance is reached using the median combination, while the product combination is also very good. The Bayes-normal-1 combination rule now shows the worst performance, although it is still very acceptable. Combining the best classifiers seems to cause overall good performance for all rules, but it might remove some of the independent errors in the intermediate space, such that somewhat less classification errors can be corrected.

7 Conclusions

It should be emphasized that our analysis is based on a single experiment for a single dataset. Conclusions will thereby at most point in a possible direction. They can be summarized as follows:

- Combining classifiers trained on different feature sets is very useful, especially when in these feature set probabilities are well estimated by the classifier. Combining different classifiers trained on the same classifier on the other hand may also improve, but is generally far less useful.
- There is no overall winning combining rule. Mean, median, majority in case of correlated errors, product for independent errors perform roughly as expected, but others may be good as well.
- The divide and conquer strategy works well: the independent use of separate feature sets works well. Difficult datasets should not be thrown away: they contain important information! The use of randomly selected feature sets appears to give very good results in our example, especially for the Bayes-normal-1 classifier.
- The Nearest Mean and the Nearest Neighbour classifiers appear to be very useful and stable when used as combiner.

In retrospection, our experiments may be extended as follows:

- A rescaling of all feature sets to unit variance, which might improve the performance of a number of classifiers.
- Remapping the posterior probabilities to distances for the trained combiners.
- Combining results of combination rules on the different feature-sets (instead of the different classifiers).

8 Acknowledgement

This work is supported by the Foundation for Applied Sciences (STW) and the Dutch Organization for Scientific Research (NWO). The authors thank the IEEE for the permission to reprint table 1, which appears here in an extended version, and which is published before in [8].

9 References

[1] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, *Classification and regression trees*, Wadsworth, California, 1984.

[2] J. R. Quinlan, Simplifying decision trees, *International Journal of Man-Machine Studies*, vol. 27, 1987, 221--234.

[3] V.N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, 1998.

[4] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas, On Combining Classifiers, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, 1998, 226-239.

[5] L. Xu, A. Krzyzak, and C.Y. Suen, Methods of combining multiple classifiers and their application to handwriting recognition, *IEEE Trans. SMC*, vol. 22, 1992, 418--435.

[6] R.E. Schapire, The strenght of weak learnability, *Machine Learning*, vol. 5, pp. 197-227, 1990.

[7] L.I. Kuncheva, J.C. Bezdek, and R.P.W. Duin, Decision Templates for Multiple Classifier Fusion: An Experimental Comparison, *Pattern Recognition*, 2000, in press.

[8] A.K. Jain, R.P.W. Duin, and J. Mao, Statistical Pattern Recognition: A Review, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, 2000, 4-37.

[9] M. van Breukelen, R.P.W. Duin, D.M.J. Tax, and J.E. den Hartog, Handwritten digit recognition by combined classifiers, *Kybernetika*, vol. 34, no. 4, 1998, 381-386.

[10] R.P.W. Duin and D.M.J. Tax, Classifier conditional posterior probabilities, in: A. Amin, D. Dori, P. Pudil, H. Freeman (eds.), *Advances in Pattern Recognition*, Lecture Notes in Computer Science, vol. 1451, Springer, Berlin, 1998, 611-619.

[11] Machine Learning Repository, UCI, //www.ics.uci.edu/~mlearn/MLRepository.html

[12] R.P.W. Duin, PRTools 3.0, *A Matlab Toolbox for Pattern Recognition*, Delft University of Technology, 2000.

[13] H. Demuth and M. Beale, *Neural Network TOOLBOX for use with Matlab*, version 3 Mathworks, Natick, MA, USA, 1998.