

Methods for Designing Multiple Classifier Systems

Fabio Roli¹, Giorgio Giacinto¹, and Gianni Vernazza²

¹Department of Electrical and Electronic Engineering, University of Cagliari
Piazza d'Armi, 09123, Cagliari, Italy
{roli,giacinto}@diee.unica.it,

²Dept. of Byophysical and Electronic Eng., University of Genoa
Via all'Opera Pia 11a, 16145, Genoa, Italy
vernazza@dibe.unige.it

Abstract. In the field of pattern recognition, multiple classifier systems based on the combination of outputs of a set of different classifiers have been proposed as a method for the development of high performance classification systems. In this paper, the problem of design of multiple classifier system is discussed. Six design methods based on the so-called “overproduce and choose” paradigm are described and compared by experiments. Although these design methods exhibited some interesting features, they do not guarantee to design the optimal multiple classifier system for the classification task at hand. Accordingly, the main conclusion of this paper is that the problem of the optimal MCS design still remains open.

1. Introduction

In the last decade, quite a lot of papers proposed the combination of multiple classifiers for designing high performance pattern classification systems [1, 2]. The rationale behind the growing interest in multiple classifier systems (MCSs) is the acknowledgement that the classical approach to designing a pattern recognition system that focuses on finding the best individual classifier has some serious drawbacks [3]. The main one being that it is very difficult to determine the best individual classifier for the classification task at hand, except when deep prior knowledge is available. In addition, the use of a single classifier does not allow the exploitation of the complementary discriminatory information that other classifiers may encapsulate.

Roughly speaking, MCS consists of an ensemble of different classification algorithms and a decision function for combining classifier outputs. Therefore, the design of MCSs involves two main phases: the design of the classifier ensemble, and the design of the combination function. Although this formulation of the design problem leads one to think that effective design should address both the phases, most of the design methods described in the literature focus only on the former one. In particular, methods that focus on the design of the classifier ensemble have tended to assume a fixed, simple decision combination function and aim to generate a set of mutually complementary classifiers that can be combined to achieve optimal accuracy

[2]. A common approach to the generation of such classifier ensembles is to use some form of data “sampling” technique, such that each classifier is trained on a different subset of the training data [4]. Alternatively, methods focused on the design of the combination function assume a given set of carefully designed classifiers and aim to find an optimal combination of classifier decisions. In order to perform such optimisation, a large set of combination functions of arbitrary complexity is available to the designer, ranging from simple voting rules through to “trainable” combination functions [2].

Although some design methods have proved to be very effective and some papers have investigated the comparative advantages of different methods [5], clear guidelines are not yet available for choosing the best design method for the classification task at hand. The designer of an MCS therefore has a toolbox containing quite a range of instruments for generating and combining classifiers. She/he may also design a myriad of different MCSs by coupling different techniques for creating classifier ensembles with different combination functions. However, the best MCS can only be determined by performance evaluation. Accordingly, some researchers proposed the so-called “overproduce and choose” paradigm (also called “test and select” approach [6]) in order to design the MCS most appropriate for the task at hand [7, 8, 9]. The basic idea is to produce an initial large set of “candidate” classifier ensembles, and then to select the sub-ensemble of classifiers that can be combined to achieve optimal accuracy. Typically, constraints and heuristic criteria are used in order to limit the computational complexity of the “choice” phase (e.g., the performances of a limited number of candidate ensembles are evaluated by a simple combination function like the majority voting rule [6, 7]).

In this paper, six design methods based on the overproduce and choose paradigm are described (Section 2). Two methods proposed in [7], and four methods developed by the authors (Section 2.3 and 2.4). The measures of classifier “diversity” used for MCS design are discussed in Section 2.2. The performances of such design methods were assessed and compared by experiment. Results are reported in Section 3. Conclusions are drawn in Section 4.

2. Design methods based on the overproduce and choose paradigm

According to the overproduce and choose design paradigm, MCS design cycle can be subdivided into the following phases:

- 1) Ensemble Overproduction
- 2) Ensemble Choice

The overproduction design phase is aimed to produce a large set of candidate classifier ensembles. To this end, techniques like Bagging and Boosting that manipulate the training set can be adopted. Different classifiers can be also designed by using different initialisations of the respective learning parameters, using different classifier types and different classifier architectures. In practical applications, variations of the classifier parameters based on the designer expertise can provide very effective candidate classifiers [1,11].

The choice phase is aimed to select the subset of classifiers that can be combined to achieve optimal accuracy. It is easy to see that such optimal subset could be obtained by exhaustive enumeration, that is, by assessing on a validation set the classification accuracy provided by all possible subsets, and then choosing the subset exhibiting the best performance. Such performance evaluation should be performed with respect to a given combination function (e.g., the majority voting rule). Unfortunately, if N is the size of the set produced by the overproduction phase, the number of possible subsets is equal to $\sum_{i=1}^N \binom{N}{i}$. Therefore, different strategies have been proposed in order to limit the computational complexity of the choice phase. Although the choice phase usually assumes a given combination function for evaluating the performances of classifier ensembles, there is a strong interest for techniques that allow choosing effective classifier ensembles without assuming a specific combination rule. This can be seen via the analogy with the feature selection problem, where techniques for choosing those features that are most effective for preserving class separability have been developed. Accordingly, techniques for evaluating the degree of error diversity of classifiers forming an ensemble have been used for classifier selection purposes. We review some of these techniques in Section 2.2.

In the following, we shall assume that a large ensemble C made up of N classifiers was created by the overproduction phase:

$$C = \{c_1, c_2, \dots, c_N\} \quad (1)$$

The goal of the choice phase is to select the subset C^* of classifiers that can be combined to achieve optimal accuracy.

2.1 Methods based on heuristic rules

Partridge and Yates proposed some techniques that exploit heuristic rules for choosing classifier ensembles [7]. One technique can be named “choose the best”. It assumes an a priori fixed size n of the “optimal” subset C^* . Then, selects from the set C the n classifiers with the highest classification accuracy to create the subset C^* . The rationale behind such heuristic choice is that all the classifier subsets exhibit similar degrees of error diversity. Accordingly, the choice is based only on the accuracy value. The other choice technique proposed by Partridge and Yates can be named “choose the best in the class”. For each classifier “class”, it chooses the classifier exhibiting the highest accuracy. Therefore, a subset C^* made up of three classifiers will be created if the initial set C is made up of classifiers belonging to three classifier types (e.g., the multilayer perceptron neural net, the k -nearest neighbours classifier, and the radial basis functions neural net). With respect to the previous rule, this heuristic rule takes also into account that classifiers belonging to different types should be more error independent than classifiers of the same type. It should be noted that the use of heuristic rules allows us to strongly reduce the computational complexity of the choice phase, because the evaluation of different classifier subsets is not required. On the other hand, the general validity of such heuristics is obviously not guaranteed.

2.2 Diversity measures

As previously pointed out, several measures of error diversity for classifier ensembles have been proposed. Partridge and Yates proposed a measure named “within-set generalization diversity”, or simply GD, that is computed as follows [7]:

$$GD = 1 - \frac{p(\text{2both fail})}{p(\text{1fails})} \quad (2)$$

where $p(\text{2 both fail})$ indicates the probability that two randomly selected classifiers from the set C will both fail on a randomly selected input, and $p(\text{1 fails})$ indicates the probability that one randomly selected classifier will fail on a randomly selected input. GD takes values in the range $[0, \dots, 1]$ and provides a measure of the diversity of classifiers forming the ensemble.

Another diversity measure was proposed by Kuncheva et al. [10]. Let $X = \{X_1, X_2, \dots, X_M\}$ be a labelled data set. For each classifier c_i , we can design an M -dimensional output vector $O_i = [O_{1,i}, \dots, O_{M,i}]$, such that $O_{j,i} = 1$, if c_i classifies correctly the pattern X_j , and 0, otherwise. Q statistics allow us to evaluate the diversity of two classifiers c_i and c_k :

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}} \quad (3)$$

where N^{ab} is the number of elements X_j of X for which $O_{j,i} = a$ and $O_{j,k} = b$. ($M = N^{00} + N^{01} + N^{10} + N^{11}$). Q varies between -1 and 1 . Classifiers that tend to classify the same patterns correctly, that is, classifiers that are positively correlated, will have positive values of Q . Classifiers that make errors on different patterns will exhibit negative values of Q . For statistically independent classifiers, $Q_{i,k} = 0$. The average Q computed over all the possible classifier couples is used for evaluating the diversity of a classifier ensemble.

Giacinto and Roli proposed a simple diversity measure, named “compound diversity”, or simply CD, based on the compound error probability for two classifiers c_i and c_j [8]:

$$CD = 1 - \text{prob}(c_i \text{ fails}, c_j \text{ fails}) \quad (4)$$

As for Q , the average CD computed over all the possible classifier couples is used for evaluating the diversity of a classifier ensemble.

It should be noted that the GD and CD measures are based on similar concepts.

As none of the above measures can be claimed to be the best, we used all of them in our design methods (Sections 2.3 and 2.4) and compared their performances (Section 3).

2.3 Methods based on search algorithms

It is easy to see that search algorithms are the most natural way of implementing the choice phase required by the overproduce and choose design paradigm. Sharkey et al. proposed an exhaustive search algorithm based on the assumption that the number

of candidate classifier ensembles is small [6]. In order to avoid the problem of exhaustive search, we developed three choice techniques based on search algorithms previously used for feature selection purposes (Sections 2.3.1 and 2.3.2), and for the solution of complex optimisation tasks (Section 2.3.3). All these search algorithms use an evaluation function for assessing the effectiveness of candidate ensembles. The above diversity measures and the accuracy value assessed by the majority voting rule have been used as evaluation functions. It should be remarked that the following search algorithms avoid exhaustive enumeration, but the selection of the optimal classifier ensemble is not guaranteed. It is worth noting that evaluation functions are computed with respect to a validation set in order to avoid “overfitting” problems

2.3.1 Forward search

The choice phase based on the forward search algorithm starts by creating an ensemble made up of a single classifier (e.g., the classifier c_2). This initial classifier is usually chosen randomly. Alternatively, the classifier with the highest accuracy can be used. Then, single classifiers are added to c_2 to form subsets of two classifiers. If the subset made up of c_2 and c_3 exhibits the highest value of the evaluation function, one more classifier is added to such subset to form the subsets of three classifiers. Such an iterative process stops when all the subsets of size $k+1$ exhibit values of the evaluation function lower than the ones of size k . In this case, the subset of size k that exhibited the highest value of the evaluation function is selected.

2.3.2 Backward search

In order to explain the developed search algorithms, let us use a simple example in which the set C created by the overproduction phase is made up of four classifiers. The backward search starts from the full classifier set. Then, eliminating one classifier from four, all possible subsets of three classifiers are created and their evaluation function values are assessed. If the subset made up of c_1 , c_3 , and c_4 exhibit the highest value, then it is selected and the subsets of two classifiers are obtained from this set by again eliminating one classifier. The iterative process stops when all the subsets of size k exhibit values of the evaluation function lower than the ones of size $k+1$. In such case, the subset of size $k+1$ that exhibited the highest value of the evaluation function is selected.

2.3.3 Tabu search

The two previous algorithms stop the search process if the evaluation function decreases with respect to the previous step. As the evaluation function can exhibit a non-monotonic behaviour, it can be effective to continue the search process even if the evaluation function is decreasing. Tabu search is based on this concept. In addition, it implements both a forward and backward search strategy. The search starts from the full classifier set. At each step, adding and eliminating one classifier creates new subsets. Then, the subset that exhibits the highest value of the evaluation function is selected to create new subsets. It should be remarked that such subsets are selected even if the evaluation function is decreased with respect to the previous step. In order to avoid the creation of the same subsets in different search steps (i.e., in order to avoid “cycles” in the search process), a classifier added or eliminated cannot

be selected for insertion/deletion for a certain number of search steps. Different stop criteria can be used. For example, the search can stop after a certain number of steps, and the best subset created during the search process is returned.

2.4 A method based on clustering of classifiers

We developed an approach to the choice phase that allows the identification of an effective subset of classifiers with limited computational effort. It is based on the hypothesis that set C created by the overproduction phase is made up of the union of M disjoint subsets C_i . In addition, we assumed that the compound error probability between any two classifiers belonging to the same subset is greater than the one between any two classifiers belonging to different subsets. It is easy to see that effective MCS members can be extracted from different subsets C_i , the more highly error-correlated the classifiers belonging to the same subset, the classifiers belonging to different subsets being error-independent. Therefore, under the hypotheses above, we defined a choice phase made up of two phases, namely the identification of the subsets C_i by clustering of classifiers, and the extraction of classifiers from different clusters in order to create an effective classifier ensemble C^* . Classifiers have been clustered according to the CD measure (eq. 4, section 2.2) so that classifiers that make a large number of coincident errors are assigned to the same cluster, and classifiers that make few coincident errors are assigned to different clusters. At each iteration of the clustering algorithm, one candidate ensembles C^* is created by taking from each cluster the classifier that exhibits the maximum average distance from all other clusters. For each candidate ensemble C^* , the classifiers are then combined by majority voting, and the ensemble with the highest performance is chosen. Further details on this design method can be found in [8,9].

3. Experimental results

The Feltwell data set was used for our experiments. It consists of a set of multisensor remote-sensing images related to an agricultural area near the village of Feltwell (UK). Our experiments were carried out characterizing each pixel by a fifteen-element feature vector containing the brightness values in six optical bands and over nine radar channels. We selected 10944 pixels belonging to five agricultural classes and randomly subdivided them into a training set (5124 pixels), a validation set (528 pixels), and a test set (5238 pixels). We used a small validation set in order to simulate real cases where validation data are difficult to be obtained. A detailed description of this data set can be found in [11].

Our experiments mainly aimed to assess the performances of the proposed design methods (Sections 2.3 and 2.4) and to compare our methods with other design methods proposed in the literature (Section 2.1).

To this end, we performed different overproduction phases, thus creating different initial ensembles C (see equation 1). Such sets were created using different classifier types, namely, Multilayer Perceptrons (MLPs), Radial Basis Functions (RBF) neural

networks, Probabilistic Neural Networks (PNNs), and the k-nearest neighbour classifier (k-nn). For each classifier type, ensembles were created by varying some design parameters (e.g., the network architecture, the initial random weights, the value of the “k” parameter for the k-nn classifier, and so on). In the following, we report the results relating to three initial sets C, here referred to as sets C^1 , C^2 , and C^3 , generated by overproduction phases:

- set C^1 contains fifty MLPs. Five architectures with one or two hidden layers and different numbers of neurons per layer were used. For each architecture, ten training phases with different initial weights were performed. All the networks had fifteen input units and five output units as input features and data classes, respectively;
- set C^2 contains the same MLPs belonging to C^1 and fourteen k-nn classifiers. The k-nn classifiers were obtained by varying the value of the “k” parameter in the following two ranges: (15, 17, 19, 21, 23, 25, 27) and (75, 77, 79, 81, 83, 85, 87);
- set C^3 contains thirty MLPs, three k-nn classifiers, three RBF neural networks, and one PNN. For the RBF neural network, three different architectures were used.

3.1 Experiments with set C^1

First of all, we evaluated the performances of the whole of set C^1 , the best classifier in the ensemble, and those ensembles designed by the two methods based on heuristic rules (see Section 2.1). Such performances are reported in Table 1 in terms of percentage accuracy values, percentage rejection rates, and differences between accuracy and rejection values. The sizes of the selected ensembles are shown. The classifiers were always combined by the majority-voting rule. A pattern was rejected when a majority of classifiers assigning it to the same data class was not present. All values reported in Table 1 referred to the test set. For the method named “choose the best” (indicated with the term “Best” in Table 1), the performances of ensembles of size ranging from 3 through 15 were assessed. The size of the ensemble designed by the method named “choose the best in the class” method (indicated with the term “Best-class”) is five, because five types of classifiers (namely, five types of net architectures) were used to create the ensemble C^1 (Section 2.1). For each ensemble, the value of the Generalisation Diversity measure (GD) is reported in order to show the degree of error diversity among the classifiers.

Table 1 shows that the design methods based on heuristic rules can provide some improvements with respect to the accuracy of the initial ensemble C^1 and the best classifier. However, such improvements are small. It should be noted that these design methods do not provide improvements in terms of error diversity as assessed by the GD measure. This can be explained by observing that such methods select classifiers on the basis of accuracy, and they do not take explicitly into account error diversity.

Table 2 reports results obtained by our design methods based on search algorithms (Section 2.3). The classifiers were always combined by the majority-voting rule. A pattern was rejected when a majority of classifiers assigning it to the same data class was not present. All values reported in Table 2 referred to the Feltwell test set. It should be noted that these design methods improve the error diversity, that is, the ensembles are characterised by GD values higher than the ones reported in Table 1.

However, the improvements in accuracy with respect to the initial ensemble C^1 and the best classifier are similar to the ones provided by the methods based on heuristic rules.

Table 1. Performances of the whole set C^1 , the best classifier in the ensemble, and the ones of the ensembles designed by two methods based on heuristic rules.

<i>Ensemble</i>	<i>Size</i>	<i>Accuracy</i>	<i>Rejection</i>	<i>Accuracy – Rejection</i>	<i>GD</i>
Initial set C^1	50	89.8357	1.2027	88.6330	0.2948
Best classifier	1	89.2516	0.0000	89.2516	N/A
Best	3	90.2565	0.2673	89.9892	0.2399
Best	5	90.4278	0.4773	89.9505	0.1937
Best	7	90.0134	0.4009	89.6125	0.1801
Best	9	90.0459	0.2673	89.7786	0.1783
Best	11	90.0747	0.3627	89.7120	0.1935
Best	13	89.9732	0.2291	89.7441	0.2008
Best	15	89.9712	0.4391	89.5321	0.2063
Best-class	5	89.9847	0.4964	89.4883	0.2617

Table 2. Performances of the ensembles generated by design methods based on search algorithms. The evaluation function used to guide the search is indicated within brackets.

<i>Choice Method</i>	<i>Size</i>	<i>Accuracy</i>	<i>Rejection</i>	<i>Acc. - Rej.</i>	<i>GD</i>
Initial set C^1	50	89.8357	1.2027	88.6330	0.2948
Best classifier	1	89.2516	0.0000	89.2516	N/A
Backward(GD)	3	89.9981	1.6991	88.2990	0.4752
Backward(CD)	3	90.4890	0.8400	89.6490	0.3573
Backward(Accuracy)	45	89.8517	0.8591	88.9926	0.2950
Backward(Q)	3	88.6901	0.7446	87.9455	0.4129
Forward_from_best(GD)	3	89.5669	0.8209	88.7460	0.4402
Forward_from_best(CD)	3	90.0499	0.6109	89.4390	0.3454
Forward_from_best(Accuracy)	11	90.3965	0.8018	89.5947	0.3274
Forward_from_best(Q)	3	88.2387	1.1455	87.0932	0.3942
Forward_random(GD)	3	89.0993	1.2218	87.8775	0.3958
Forward_random (CD)	7	90.2866	0.7446	89.5420	0.3346
Forward_random (Accuracy)	7	90.2420	0.6109	89.6311	0.2589
Forward_random (Q)	3	87.0609	0.5536	86.5073	0.3845
Tabu(GD)	3	89.9459	1.2600	88.6859	0.4613
Tabu(CD)	3	90.1425	0.8400	89.3025	0.3806
Tabu(Accuracy)	9	90.1156	0.9164	89.1992	0.3416
Tabu(Q)	3	89.8180	1.3746	88.4434	0.4826

Table 3 reports results obtained by our design method based on clustering of classifiers (Section 4.4). Conclusions similar to those for the design methods based on search algorithms can be drawn.

Table 3. Performances of the ensembles generated by design method based on clustering of classifiers. The evaluation function used to guide the search is indicated within brackets.

<i>Choice Method</i>	<i>Size</i>	<i>Accuracy</i>	<i>Rejection</i>	<i>Acc. – Rej.</i>	<i>GD</i>
Initial set C^1	50	89.8357	1.2027	88.6330	0.2948
Best classifier	1	89.2516	0.0000	89.2516	N/A
Cluster(CD)	7	90.5294	0.8209	89.7085	0.3193
Cluster(Q)	49	89.6592	0.8591	88.8001	0.2962
Cluster(GD)	9	89.6179	1.0691	88.5488	0.3788

It is worth noting that the performances of various design methods are slightly better than the ones of initial ensemble C^1 and the best classifier, but the differences are small. However, it should be noted that methods based on search algorithms and clustering of classifiers improve the error diversity of classifiers.

3.2 Experiments with set C^2 and C^3

The same experiments previously described for set C^1 were performed for sets C^2 and C^3 . For the sake of brevity, for each design method we report the average performances in terms of accuracy and error diversity values. Table 4 shows the average accuracy values and the average error diversity values (in terms of the GD measure) of different design methods applied to sets C^2 and C^3 .

Table 4. For each design method, the average percentage accuracy value and the average error diversity value (in terms of the GD measure) are reported for the experiment with the set C^2 and for the experiment with the set C^3 .

<i>Choice Method</i>	<i>Set C^2</i>		<i>Set C^3</i>	
	<i>Accuracy</i>	<i>GD</i>	<i>Accuracy</i>	<i>GD</i>
Initial set	90.4918	0.3170	89.4645	0.3819
Best classifier	90.0916	-	88.2016	-
Choose the best	90.1090	0.1989	91.1097	0.3279
Choose the best in the class	89.9847	0.2617	92.0613	0.4905
Backward	89.8945	0.3488	92.3871	0.5851
Forward from the best	89.9024	0.3400	93.2471	0.5917
Forward from random	89.7408	0.3270	91.5023	0.5969
Tabu	90.0931	0.3631	93.5092	0.6225
Clustering	89.9013	0.3410	92.1911	0.5383

With regard to the experiments performed on sets C^2 and C^3 , the performances of various design methods are close to or better than those of the initial ensembles and the best classifier. Significant improvements were obtained for some of the experiments performed on set C^2 .

4. Discussion and conclusions

Although definitive conclusions cannot be drawn on the basis of the limited set of experiments above, some preliminary conclusions can be drawn. The overproduce and choose paradigm does guarantee optimal MCS design for the classification task at hand. In particular, no choice method can be claimed to be the best, because the superiority of one over the other depends on the classification task at hand. Accordingly, optimal MCS design is still an open issue. The main motivation behind the use of the overproduce and choose paradigm is that at present clear guidelines to choose the best design method for the classification task at hand are lacking. Thanks to this design paradigm it is possible to exploit the large set of tools developed to generate and combine classifiers. The designer can create a myriad of different MCSs by coupling different techniques to create classifier ensembles with different combination functions. Then, the most appropriate MCS can be selected by performance evaluation. It is worth noting that this approach is commonly used in engineering fields where optimal design methods are not available (e.g., software engineering). In addition, the overproduce and choose paradigm allows to create MCSs made up of small sets of classifiers. This is a very important feature for practical applications.

References

- [1] Xu, L., Krzyzak A., Suen C.Y.: Methods for combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on Systems, Man, and Cyb.* **22** (1992) 418-435
- [2] Kittler J., Roli F. (eds.): *Multiple Classifier Systems. Proc. of the First Int. Workshop MCS 2000. Lecture Notes in Computer Science, Vol. 1857. Springer-Verlag (2000)*
- [3] Kittler J.: A framework for classifier fusion: is it still needed? In: Ferri F.J., Inesta J.M., Amin A., Pudil P. (eds.): *Advances in Pattern Recognition. Proc. Int. Workshop SSPR&SPR 2000. Lectures Notes in Computer Science, Vol. 1876. Springer-Verlag (2000)* 45-56
- [4] Breiman, L.: Bagging Predictors. *Machine Learning* **24** (1996) 123-140
- [5] Ho T.K.: Complexity of classification problems and comparative advantages of combined classifiers. In: Kittler J., Roli F. (eds.): *Multiple Classifier Systems. Proc. of the First Int. Workshop MCS 2000. LNCS Vol. 1857 Springer-Verlag (2000)* 97-106
- [6] Sharkey A.J.C., et al.: The "test and select" approach to ensemble combination. In: Kittler J., Roli F. (eds.): *Multiple Classifier Systems. LNCS 1857. Springer-Verlag, (2000)* 30-44
- [7] Partridge D., Yates W.B.: Engineering multiversion neural-net systems. *Neural Computation* **8** (1996) 869-893
- [8] Giacinto G., Roli F.: An approach to the automatic design of multiple classifier systems. *Pattern Recognition Letters* **22** (2001) 25-33
- [9] Giacinto G., Roli F.: Design of effective neural network ensembles for image classification purposes. to appear in *Image and Vision Computing Journal* (2001)
- [10] Kuncheva, L.I., et al.: Is independence good for combining classifiers?. *Proc. of ICPR2000, 15th Int. Conf. on Pattern Recognition, Barcelona, Spain (2000), Vol. 2, 168-171*
- [11] Giacinto G., Roli F., Bruzzone L.: Combination of Neural and Statistical Algorithms for Supervised Classification of Remote-Sensing Images. *Pattern Recognition Letters* **21** (2000) 385-397