Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Fusion 6 (2005) 63–81
INFORMATION FUSION
www.elsevier.com/locate/inffus

Classifier selection for majority voting

Dymitr Ruta [a,*], Bogdan Gabrys [b]

[a] Computational Intelligence Group, BT Exact Technologies, Orion Building 1st floor, pp12, Adastral Park, Martlesham Heath, Ipswich IP5 3RE, UK

[b] Computational Intelligence Research Group, School of Design, Engineering & Computing, Bournemouth University, Poole House, Talbot Campus, Fern Barrow Poole BH12 5BB, UK

Received 31 October 2003; received in revised form 30 March 2004; accepted 18 April 2004
Available online 21 July 2004

Abstract

Individual classification models are recently challenged by combined pattern recognition systems, which often show better performance. In such systems the optimal set of classifiers is first selected and then combined by a specific fusion method. For a small number of classifiers optimal ensembles can be found exhaustively, but the burden of exponential complexity of such search limits its practical applicability for larger systems. As a result, simpler search algorithms and/or selection criteria are needed to reduce the complexity. This work provides a revision of the classifier selection methodology and evaluates the practical applicability of diversity measures in the context of combining classifiers by majority voting. A number of search algorithms are proposed and adjusted to work properly with a number of selection criteria including majority voting error and various diversity measures. Extensive experiments carried out with 15 classifiers on 27 datasets indicate inappropriateness of diversity measures used as selection criteria in favour of the direct combiner error based search. Furthermore, the results prompted a novel design of multiple classifier systems in which selection and fusion are recurrently applied to a population of best combinations of classifiers rather than the individual best. The improvement of the generalisation performance of such system is demonstrated experimentally.

Keywords: Classifier fusion; Classifier selection; Diversity; Search algorithms; Majority voting; Generalisation

1. Introduction

Given a large pool of different classifiers there are a number of possible combining strategies to follow and it is usually not clear which one may be the optimal for a particular problem. The simplest strategy could be to select the single, best performing classifier on the training data and applying it to the previously unseen patterns [26]. Such an approach, although the simplest, does not guarantee the optimal performance [28]. Moreover, there is a possibility that at least some subsets of classifiers could jointly outperform the best classifier if suitably combined. To ensure the optimal performance, a multiple classifier design should be able to select the subset of classifiers that is optimal in the sense that it produces the highest possible performance for a particular combiner. On one hand, it is clear that combining the same classifiers does not contribute to anything but the increased complexity of a system. On the other hand, different but much worse performing classifiers are unlikely to bring any benefits in combined performance. It is believed that the optimal combinations of classifiers should have good individual performances and at the same time sufficient level of diversity [35]. In many recent works it has been shown however that neither individual performances [27,40] nor diversity [29,37] on their own provide a reliable diagnostic tool able to detect when combiner outperforms the individual best classifier. As noted by Rogova [27], individual classifier performances do not relate well to combined performance as they miss out the important information about the team strength of the classifiers. In turn, diversity, due to problems with measuring and

* Corresponding author. Tel.: +44-147-360-5491; fax: +44-1473-623-683.
E-mail addresses: dymitr.ruta@bt.com (D. Ruta), bgabrys@bournemouth.ac.uk (B. Gabrys).

1566-2535/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.inffus.2004.04.008

even perceiving it, also does not provide a reliable selection criterion that would be well correlated with combiner performance [31]. Some attempts at including both components jointly guiding selection proved to be highly complex while offering only relatively small improvements [30,40]. A little more successful have been selection attempts based on specific similarity measures devised in conjunction with the combiner for which the classifiers are selected. The *fault majority* presented in [29] or similarity S_{3h} measure presented in [16] are just two examples that have shown high correlation with majority voting performance. Unlike general statistically driven diversity measures, measures exploiting combiner definition take into account information of what makes a particular combiner work and selection guided by such a combiner naturally have greater chances of being successful. All these findings point to the combined performance as a relevant selection criterion.

Effectively the most reliable strategy seems to be evaluation of as many different designs as possible and subsequent selection of the best performing model. A difficulty however is that such a wide open scale of evaluation is computationally intractable. To realise this, it is sufficient to note that assuming a chosen combiner, evaluation of all subsets from an ensemble of redundant classifiers is a process growing exponentially with the number of classifiers. On top of that, for large numbers of classifiers the performance based search space becomes increasingly flat which makes selection even more difficult [40]. In the light of such difficulties, a modular decomposition model of combining seems advisable, particularly if only one locally best classifier is to be selected for a particular subtask or local input subspace. A number of dynamic selection models [7,8,10] or *cluster and select* based approaches [18,24] illustrate that advantage and in some cases show even substantial improvement compared with the individual best classifier. However, by analogy to redundant combining, in general, improvement may be also sought in combining many classifiers within each subtask or input subspace, which computationally looks even more intractable than in the redundant combining model. Summarising, a large number of various classifiers and combining methods, rapidly increases the ambiguity, risk and selective complexity of a particular choice, leading ultimately to overfitting and weak generalisation ability. On the other hand, in safety critical systems all potentially useful evidence is precious and cannot be wasted. These requirements impose on classifier selection the need to address the problems of complexity, overfitting and search accuracy at the same time.

There is not much evidence in the literature of selection systems dealing well with all three aspects mentioned above. In an attempt to ensure the accuracy of search, Sharkey et al. [36] proposed an exhaustive search algorithm assuming small number of classifiers. For larger numbers of classifiers, in order to avoid computational burden of an exhaustive search, a number of heuristic selection methods have been proposed. From *choose single best* through *pick n best* to *choose the best in the class* strategies have been investigated by Partridge and Yates [26]. As claimed in [26], such simple strategies are particularly effective in flat search spaces where all the classifiers exhibit similar capabilities including performance and diversity. In general however, the validity of such heuristics is not guaranteed. As mentioned above, there have been some attempts at selecting classifiers based on diversity measures. In [9], Giacinto and Roli used a simple double fault measure for clustering classifier outputs and used this to select a single classifier from each cluster for combination. A similarity measure has been used in [16] to select an optimal classifier triplet from a pool of five classifiers. In both examples, optimality of selections is not guaranteed although in [9] a partial optimality proof is presented. In the feature selection domain, Zenobi and Cunningham [40] present a search based jointly on individual performances and an ambiguity measure and show its superiority to the search guided only by classifier performances. Although many other diversity measures have been presented [22,29,32,37], as concluded in [31], combined performance offers a much more precise search while keeping the complexity at a comparable level. Once settled on the performance based classifier selection, the research effort focussed on reducing the huge complexity of a search which in the exhaustive scenario is of exponential order. Clustering and selection approaches [18,24] offer huge computational savings however they nominate only a single locally best classifier for final classification. The same applies to dynamic classifier selection shown initially by Woods et al. [39] and further extensively investigated by Giacinto and Roli in [7,8,10], where rather than using fixed clusters, a single classifier is picked based on the optimal performance in the closest neighbourhood of the incoming pattern to be classified. Clustering based and dynamic selection methods simplify the selection process considerably, nonetheless they make the whole model more complex and still do not guarantee even locally the optimality of the search. The problem of efficient but optimal or close to optimal selection based on combined performance has been addressed in various types of stochastic and evolutionary search algorithms. Unlike for a small number of classifiers where the selection can be done exhaustively [20,36], for large numbers of classifiers genetic algorithms [3] have been shown to be suitable for dealing with large and rough search spaces [2,19,30] and proved superior to other heuristic selection techniques [13,30]. Other evolutionary based selection approaches including *tabu search* [11,28,30], *population-based incremental learning* [1,30],

showed comparable performance while offering faster convergence of the algorithm. We believe and attempt to prove experimentally that population based evolutionary algorithms fit well to the generally perceived classifier selection problem.

The core element of classifier selection is a selection criterion. Probably the most natural choice is the combined performance as it is also the criterion for evaluation of the combiner. An immediate drawback of using performance as a selection criterion is its exponential complexity if all combinations of classifiers are to be evaluated. Among other problems, as some authors claim, is that selecting classifiers according to combiner performance is at high risk of overfitting phenomenon [40]. Diversity measures together with all other team strength measures represent an alternative to performance based selection criterion [21,25,29]. In view of the very weak correlation with combined performance [29,37] it seems that the performance should be used instead of diversity as a selection criterion or a measure strongly exploiting the phenomena making a particular combiner work. Nevertheless there are situations where diversity guided search could be invaluable. The quadratic complexity of pairwise measures offers direct complexity payoffs if only well performing combinations could be found. In other scenarios, performance based selection may run into problems if the search space turns out to be almost flat. This could happen if the classifiers are similar and performing at comparable recognition rate individually. Various diversity measures could then potentially provide additional criteria for selection, which may be particularly appreciated in relation to generalisation ability. This case becomes even more apparent if a performance driven selection algorithm repeatedly results in perfect performances as in boosting [4,34]. Diversity, which can be perceived in various more or less combiner specific forms, becomes then the only criterion for selection aimed at reducing generalisation error. A comprehensive experimental work with 27 datasets and 15 classifiers is carried out and is aimed at ultimate comparison of the quality of direct combiner-error-based search and selection based on diversity measures.

In relation to classifiers, selection should provide the answers to which ones and/or how many classifiers to select in order to obtain an optimal combined performance of the selected subset. Important is the fact that unless resulting in a single classifier, selection on its own does not complete the system design since the selected classifiers need to be further combined. To avoid dissonance between selection model and the combiner, selection method should be tuned to the combiner by exploiting its characteristics. By analogy with the myth of a universal classifier, an individual selection model may be insufficient to extract the optimal combination of classifiers, especially in the presence of large number of classifiers to choose from. Following this analogy, combining multiple selection algorithms or many results from a single selector could address the overfitting problem of individual selection and thus improve the generalisation ability of the system. An example of such system in which selection and fusion are recurrently applied to a population of best combinations of classifiers rather than the individual best, is proposed and its properties investigated thoroughly.

The remainder of this paper is organised as follows. Section 2 provides the overview of the selection model and presents various selection criteria that can be applied. The following section covers search algorithms used in classifier selection and provides a experimental results evaluating the performance of search and the appropriateness of selection criteria introduced. Section 4 presents a multistage selection–fusion model that has been developed on the grounds of experimental findings from previous section. Finally concluding remarks are drawn in Section 5.

2. Selection model

2.1. Static vs dynamic selection

Classifier selection techniques fall into two general methodologies. According to the first type called *static classifier selection* (SCS), the optimal selection solution found for the validation set is fixed and used for the classification of unseen patterns. The whole analytical effort is thus focussed on the extraction of the best combination for the labelled validation set, which can also be used for evaluation. The selection of classifiers in SCS is fully based on the average performances obtained for the labelled validation set, and thus complies with the redundant ensembles type of combination.

In a more aggressive approach called *dynamic classifier selection* (DCS), the selection is done online, during classification, based on training performances and also various parameters of the actual unlabelled pattern to be classified. In [39], Woods et al. proposed to select the single classifier that shows the best performance in the closest neighbourhood defined by an arbitrarily set number of neighbouring training samples. Giacinto and Roli [7,10] enriched the selection criterion by incorporating classifier outputs produced during classification. In a weaker version of DCS, referred to as *cluster and select* (CS), the input space is initially partitioned into disjoint regions obtained by clustering the training data. Then the best classifier for each cluster is identified and selected to classify the new pattern if it falls into its region [18,24]. The CS approach is somewhat in between SCS and DCS as the classifiers are selected dynamically depending on the input space region into which the new sample falls, but the regions are themselves static, set in

advance during the training process. DCS and CS are an integral part of combining by modular decomposition.

Of interest is the fact that both static and dynamic selection approaches do not have to be exclusively applied for the multiple classifier system. In fact, knowledge of the location of a new sample in the feature space, that is ignored by SCS, can be used to localise the domain of the system analysis within which SCS can be applied again. Effectively what this means is that rather than selecting the locally best classifier the objective may be reformulated to find the locally best combination of classifiers. On this basis it seems that the strength of the classifier selection component of MCS is independent of its design and regardless of whether it is applied in SCS or DCS combination model, its definition remains the same: select the best subset of classifiers from the complete ensemble. Based on the above considerations, classifier selection investigated in this work will be considered according to a static approach, which does not restrict applying it to dynamic classifier selection models.

2.2. Representation

In the context of classifier fusion, selection is in general perceived as including only some classifiers for further processing ultimately leading to the combination output. More specifically, in most cases classifier selection simply means validation of selected classifier outputs to be combined, which could be also interpreted as a binary weighting procedure. Weighting seems thus a good generalisation of classifier selection although its binary version looks somewhat distinct due to the exclusion of unselected classifiers from further consideration. Weighting proves beneficial when dealing with a number of different combinations. Apart from consistency and identifiability of individual classifiers within different combinations it provides also a uniform platform for any comparisons and joint processing that may be required by the combiner or even the selection algorithm. It will be shown that binary weighted representation of classifier selection combined with binary classifier outputs offer further advantages for multilayer combining systems.

Given a system of M classifiers: $D = \{D_1, \ldots, D_M\}$, let $\mathbf{y}_i = [y_{i1}, \ldots, y_{iM}]^{\mathrm{T}}$ denote the joint output of a system for the ith multidimensional input sample $\mathbf{x}_i$, where y_{ij} denotes the output of the jth classifier for the ith input sample. Assuming that oracle type (binary) outputs are available, the meaning of the outputs takes the form $y_{ij} = 0$ for correct and $y_{ij} = 1$ for error. Let $\boldsymbol{\omega}_i = [\omega_{i1}, \ldots, \omega_{iM}]^{\mathrm{T}}$ represent a weighting vector where each ω_{ij} indicates inclusion ($\omega_{ij} = 1$) or exclusion ($\omega_{ij} = 0$) of the jth classifier in the decision fusion. Note that a weighting vector $\boldsymbol{\omega}_i$ represents in fact a specific combination of classifiers which could vary from sample to sample reflecting a truly general approach to classifier selection complying with both SCS and DCS.

2.3. Selection criterion

The quality of the combined system based on the selected classifiers relies mostly on the goodness of the selection criterion. It is used for evaluation of various joint properties of classifiers, in particular those relating to or deciding directly about the combined performance of the selected team of classifiers.

Individual best performance was always a universal indicator for selection of the individual best classifier, which although called into question recently is still the simplest, yet reliable and robust option preferred in industrial applications. The applicability of the individual performance criterion for selection of classifiers is very limited. The major problem is evaluation inconsistency as adding more and more poorer classifiers could only produce worse combinations in the individual mean sense. In fact the only sensible option is using individual performances for selection evaluated further by the performance of the combiner. A simple example of such a strategy is a selection of n best classifiers, where combined performance could be used for evaluation to determine the optimal value of n. A number of these and other *choose the best* heuristic selection strategies have been proposed by Partridge and Yates [26], who concluded that although the computational complexity of the classifier selection is greatly reduced, the optimality of such heuristics is far from being guaranteed.

An obvious candidate for a selection criterion is a measure of diversity. However in the light of the extensive experimental evidence presented in [21,22,29, 32,37] showing very weak correlation between diversity measures and combined performance, the major risk of using them as selection criteria is simply picking the most diverse and not best performing combinations. Some intermediate measures aimed at modelling the combined performance shown in [16,29] have naturally greater chances for guiding better search. In fact, perfect correlation with combined performance is not a necessity, and taking generalisation aspects into account could even correspond to overfitting. As a result diversity measures that show at least some clear correlation trends have the potential to become suitable selection criteria. This is particularly the case when the objective is to find a population of best classifier combinations when the precise rank of individual combinations is not of crucial importance.

The most natural interpretation of team optimality with respect to a particular combiner is thus its best performance possible. Using combiner performance directly as a selection criterion is precise, meaningful, and allows for consistent comparisons of different classifier

subsets regardless of the number of classifiers and their individual performances. There are, though, two issues that selection based on combined performance has to deal with. First note that even if the selection algorithm is capable of picking the optimal combinations, their optimality can be only assessed for the training set for which the labels are known and hence the performance can be obtained. This means that once the optimal combination is selected it may not necessarily remain optimal for unseen data. As for individual classifiers, the selector runs into the risk of performance degradation due to the generalisation problem. To limit its effect the true performance of the selector should be estimated on a part of the training data which has not been used for selecting the optimal combinations nor for training the individual classifiers. Maintaining reasonable reliability when dealing with static classifier selection requires therefore larger than usual training sets. Another problem that performance driven selection has to tackle is the complexity which is of exponential order in the case of exhaustive evaluation of all possible subsets of classifiers. If, on top of that, the combiner is itself complex, it could drastically slow down the search algorithm or even lead to intractability for larger numbers of classifiers.

For the generality of our investigations, examples of all aforementioned selection criteria will be evaluated in the experimental section. As most of the measures are based on the simple binary algebra the following simplifications are introduced. Let N^{ab} $a,b=\{0,1,*\}$ denote the number of input samples, for which the considered pair of classifiers produce the sequence of outputs: $\{a,b\}$. The star denotes any of the outputs: $*=0$ or 1. Note that $N=N^{**}$. Furthermore let $m(\mathbf{x}_i)$ denote the number of classifiers producing error for the input sample $\mathbf{x}_i$. It can be expressed by

$$m(\mathbf{x}_i)=\sum_{j=1}^{M} y_{ij}, \tag{1}$$

where y_{ij} is the binary output from the jth classifier for the ith input sample. Finally let $e_j=\frac{1}{N}\sum_{i=1}^{N} y_{ij}$ denote the error rate of jth classifier and accordingly the ensemble mean error rate be defined by

$$\bar{e}=\frac{1}{M}\sum_{j=1}^{M} e_j. \tag{2}$$

2.3.1. Minimum individual error MIE

This measure represents the minimum error rate of the individual classifier and promotes individual best classifier selection strategy. Using the above denotations the definition of MIE takes the simple form of

$$\mathrm{MIE}=\min(e_j). \tag{3}$$

2.3.2. Mean error ME

This measure takes the average from individual classifier error rates within the ensemble and formally is equal to mean error rate already defined in (2).

2.3.3. Majority voting error MVE

This measure is simply a measure of majority voting error rate. Assuming that the majority voting applied for individual sample returns the following outputs:

$$y_i^{\mathrm{MV}}=\begin{cases}1, & \text{if } \sum_{j=1}^{M} y_{ij}\geqslant \frac{M}{2},\\ 0, & \text{if } \sum_{j=1}^{M} y_{ij}<\frac{M}{2},\end{cases} \tag{4}$$

its error rate can then be formulated as

$$\mathrm{MVE}=\frac{1}{N}\sum_{i=1}^{N} y_i^{\mathrm{MV}}. \tag{5}$$

2.3.4. Majority voting improvement MVI

Quite often a goodness of ensemble is measured in the form of improvement of the combiner performance compared to the individual mean classifier performance. This measure can be quickly obtained from the following rule:

$$\mathrm{MVI}=\mathrm{MVE}-\mathrm{ME}. \tag{6}$$

In such form negative values of MVI correspond to improved performance of the MV compared to individual mean error.

2.3.5. The correlation coefficient C2

Correlation is a well known statistical measure most often applied to continuous variables [12]. For binary classifier outputs its definition takes the form

$$\begin{aligned}\mathrm{C2}_{ij}&=\frac{N^{11}N^{00}-N^{01}N^{10}}{\sqrt{N^{1*}N^{0*}N^{*1}N^{*0}}},\\ \overline{\mathrm{C2}}&=\frac{2}{M(M-1)}\sum_{\substack{i,j=1,\ldots,M\\ i\neq j}}\mathrm{C2}_{ij}.\end{aligned} \tag{7}$$

2.3.6. Product-moment correlation measure PM2

This measure was used by Sharkey and Sharkey [35] as a guidance for selection of the most diverse neural network classifiers. Adapting this measure to the binary representation of outputs it can be defined as

$$\begin{aligned}\mathrm{PM2}_{ij}&=\frac{N^{00}}{\sqrt{N^{*0}N^{0*}}},\\ \overline{\mathrm{PM2}}&=\frac{2}{M(M-1)}\sum_{\substack{i,j=1,\ldots,M\\ i\neq j}}\mathrm{PM2}_{ij}.\end{aligned} \tag{8}$$

2.3.7. The Q statistics Q2

Q statistics was used by Kuncheva et al. [23] for assessing the level and sign of dependency between a pair of classifiers with binary outputs, where −1 means full negative dependence, +1 full positive dependence. The measure is defined by

$$\mathrm{Q2}_{ij} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}},$$
$$\overline{\mathrm{Q2}} = \frac{2}{M(M-1)} \sum_{\substack{i,j=1,\ldots,M \\ i \neq sj}} \mathrm{Q2}_{ij}. \tag{9}$$

2.3.8. The disagreement measure D2

The disagreement measure was used by Skalak [38] to determine the diversity between two classifiers. It takes the form of a ratio between the number of samples for which the classifiers disagreed, to the total number of observations. This can be written as

$$\mathrm{D2}_{ij} = \frac{N^{01} + N^{10}}{N},$$
$$\overline{\mathrm{D2}} = \frac{2}{M(M-1)} \sum_{\substack{i,j=1,\ldots,M \\ i \neq j}} \mathrm{D2}_{ij}. \tag{10}$$

2.3.9. The double-fault measure F2

This measure was used by Giacinto and Roli [9] to create a matrix of pairwise dependencies used for selecting the least related classifiers. The measure estimates the probability of coincident errors for a pair of classifiers, which is

$$\mathrm{F2}_{ij} = \frac{N^{00}}{N},$$
$$\overline{\mathrm{F2}} = \frac{2}{M(M-1)} \sum_{\substack{i,j=1,\ldots,M \\ i \neq j}} \mathrm{F2}_{ij}. \tag{11}$$

2.3.10. The entropy measure EN

This measure was used by Kuncheva and Whitaker [22] and shows the level of disagreement among the outputs from a set of classifiers

$$\mathrm{EN} = \frac{1}{N} \sum_{i=1}^{N} \frac{\min\{m(\mathbf{x}_i), M - m(\mathbf{x}_i)\}}{M - \lceil M/2 \rceil}. \tag{12}$$

The entropy measure reaches its maximum (EN = 1) for the highest disagreement, which is the case of observing $\lfloor M/2 \rfloor$ votes with identical value (0 or 1) and $M - \lfloor M/2 \rfloor$ with the alternative value. The lowest entropy (EN = 0) is observed if all classifier outputs are identical.

2.3.11. The measure of difficulty DI

This measure originates from a study of Hansen and Salomon [14] and was developed by Kuncheva and Whitaker [22] for the case of binary classification outputs. Given the set of M classifiers, the measure can be built on the basis of discrete error distribution $Z = [p_z(0), \ldots, p_z(M)]$ defined in [33]. Namely, it measures the variance of the $p_z(j)$ components corresponding to the normalised number of cases for which exactly j errors were observed. This is defined as

$$\mathrm{DI} = \frac{1}{M} \sum_{j=0}^{M} (p_z(j) - \bar{p}_z)^2. \tag{13}$$

2.3.12. Kohavi–Wolpert variance KW

This measure follows a similar strategy to the measure of difficulty. Namely, it measures the average variance from binomial distributions of the outputs for each classifier [17]. The measure can be simply calculated by

$$\mathrm{KW} = \frac{1}{NM^2} \sum_{i=1}^{N} [m(\mathbf{x}_i)(M - m(\mathbf{x}_i))]. \tag{14}$$

It can be shown [22] that, for independent classifiers $\mathrm{KW} = M \times \mathrm{DI}$, which derives from the definition of variance of the joint binomial distribution.

2.3.13. Interrater agreement measure IA

This measure was developed in [6] to measure the level of agreement while correcting the chance (see [6] for details). Using the notation presented above it can be expressed as

$$\mathrm{IA} = 1 - \frac{\sum_{i=1}^{N} m(\mathbf{x}_i)(M - m(\mathbf{x}_i))}{NM(M-1)\bar{e}(1-\bar{e})}. \tag{15}$$

2.3.14. Fault majority measure FM

This measure was proposed by Ruta and Gabrys [29] and uses partial error distributions (PDED) expressing for each classifier the degree it contributes to different levels of ensemble error coincidences. The measure sums up only those PDED components that could contribute to majority voting error for a considered subset of classifiers i.e. for error coincidences with at least $\lceil M/2 \rceil$ errors and coming from $\lceil L/2 \rceil$ locally best classifiers. Formally, the measure is defined as follows:

$$\mathrm{FM} = \sum_{j=\lceil L/2 \rceil}^{L} \sum_{i^*=1}^{\lceil L/2 \rceil} z_{i^* j}, \tag{16}$$

where index i^* refers to the classifiers sorted according to their values of z_{ij} for the fixed coincidence level j. Further details on FM measure can be found in [29].

2.3.15. Generalized diversity GD

Looking at multiversion software failures, Partridge and Krzanowski [25] introduced the probability of k random versions failing simultaneously $p(k)$. Denoting further by p_k probability of exactly k among M versions failing on a random input, they showed that

$$p(k) = \sum_{j=1}^{M} \frac{j}{M} \frac{j-1}{M-1} \cdots \frac{j-k+1}{M-k+1} p_j. \tag{17}$$

Partridge and Krzanowski claimed that maximum diversity is observed when for a random pair of versions a failure of one version is always accompanied by a correct output of the other ($p(2) = 0$), whereas the minimum diversity reflects identical failures for any two versions ($p(2) = p(1)$). Based on these assumptions they proposed a simple *generalised diversity* (GD) measure:

$$\mathrm{GD} = \frac{p(1) - p(2)}{p(1)} = 1 - \frac{p(2)}{p(1)}, \tag{18}$$

ranging between 0 (minimum diversity) and 1 (maximum diversity).

2.3.16. Coincident failure diversity CFD

To account for higher order failure coincidences, Partridge and Krzanowski [25] developed *coincident failure diversity* (CFD) defined by

$$\mathrm{CFD} = \begin{cases} \frac{1}{1-p_0} \sum_{j=1}^{M} \frac{M-j}{M-1} p_j, & p_0 < 1, \\ 0, & p_0 = 1, \end{cases} \tag{19}$$

which again gives maximum diversity ($\mathrm{CFD} = 1$) when at most one version fails on any input and minimum diversity when all versions are always either correct or wrong.

3. Search algorithms

3.1. Heuristic techniques

As mentioned above, one of the simplest yet most reliable and therefore industrially preferred, selection strategies is called *single best* (SB). Investigated formally in [26] but simple enough to be considered elsewhere SB has a number of different variations. In the standard version the classifier showing best performance over a validation set is selected to classify any new patterns to be classified. In a more extended implementation called *N best* (NB) an arbitrarily large number of best performing classifiers are selected. To validate the best N the combinations are evaluated by the combined performance and the optimal N is selected.

3.1.1. Single best (SB)

Selection of the best performing classifier is the simplest and quite often justified choice. Assuming that O(1) represent an evaluation of a single combination SB is undoubtedly the simplest linearly complex search, $O(M)$, with the lack of lack of subsequent combining. In such an approach the combiner is not applied and hence the performance of the SB method is often presented as a reference level that combining methods are trying to challenge. In the experimental section, the SB selection will be applied exactly for the same purpose, which is to evaluate the usefulness of combining classifiers.

3.1.2. N best (NB)

Selection of N best classifiers is also very cheap computationally. It requires to sort classifiers according to their performance and then to check how many best performing classifiers form the optimal ensemble. Effectively it requires examining single best classifier, a pair of best classifiers, best three classifiers and so on, up to complete ensemble of M classifiers. The complexity of such process (associated with a number of performance evaluations) would retain a linear order of $O(M)$, although in this case combiner performance has to be checked for $2M - 1$ combinations (M singletons and $M - 1$ ensembles). In our case, due to the requirement of having an odd number of classifiers, the NB selection is even simpler as only combinations of odd numbers of classifiers $(1, 3, \ldots, M)$ have to be evaluated.

3.2. Greedy approaches

In NB selection consecutive combinations are built according to the information about individual classifier performances. There is a possibility though, that adding a pair of best performing classifiers is not the optimal choice. Greedy approaches concentrate on adding or removing a specific classifier so that the improvement in the combiner performance is maximal. In the majority voting case, a pair of classifiers has to be considered as a minimum additive element such that the odd number of classifiers in the combination is preserved.

3.2.1. Forward search (FS)

Forward search is the most intuitive greedy algorithm. Starting from a single best classifier at each iteration a pair of classifiers, that maximally reduces the majority voting error, is sought. If MVE cannot be reduced for any pair of classifiers the algorithm stops with the combination built so far. The MVE is the most intuitive criterion for selection but in general any measures of optimality can be incorporated as selection criteria. Note that at each iteration, evaluation of the selection criterion for all possible pairs of remaining classifiers imposes quadratic complexity. The overall complexity of the FS algorithm is therefore of $O(M^3)$.

Despite its relatively high complexity, FS does not guarantee the optimality of the combination found.

3.2.2. Backward search (BS)

Backward search represents a symmetrical to FS greedy approach to classifier selection. The algorithm starts with the whole ensemble of M classifiers and each iteration involves searching for a pair of classifiers from the combination that if removed, causes maximum improvement in the combiner performance or any other measure used as a selection criterion. BS shares the cubic complexity $O(M^3)$ of the FS algorithm.

3.3. Evolutionary algorithms

Greedy approaches are often reported to get caught in local maxima [5]. In classifier selection the problem of local maxima could be even more apparent due to the large and quite rough search spaces. Evolutionary algorithms working on the basis of the population of solutions have been shown to deal well with such cases [1–3,11,13,19,30]. The three algorithms are presented in this domain: genetic algorithm [3], stochastic hill-climbing search [11] and population-based incremental learning [1]. All of them use binary representation of selection solutions introduced in Section 2.2 and produce a number of best classifier combinations found as an output. These algorithms have been adjusted to accommodate the constraint of the odd number of classifiers and to control the uniqueness of solutions within the population.

3.3.1. Genetic algorithm (GA)

The genetic algorithm was developed in 1970s by Holland [15] as an effective evolutionary optimisation method. Since that time, intensive research has been dedicated to GAs, bringing lots of applications in the machine learning domain [2,3,13,19,30]. Despite the tremendous number of varieties of GAs, its underlying principles remain unchanged. Chromosomes—the core GA units have been used as a binary encoded representation of the solutions to the optimisation problem. A randomly initialised population of chromosomes is then evaluated according to the required fitness function and assigned a probability of survival proportional to their fitness. The best chromosomes are most likely to survive and are allowed to reproduce themselves by recombining their genotype and passing it on to the next generation. This is followed by a random mutation of some bits, which was designed to avoid premature convergence and enables the search to access different regions of search space. The whole process is repeated until the population converges to a satisfactory solution or after a fixed number of generations. The GA is inspired by and takes strength from an explicit imitation of biological life, in which the strongest (fittest) units survive and reproduce further constantly adjusting to the variable conditions of living.

There are several problems in adopting GAs for classifier selection. The major problem stems from the constraint of the odd number of classifiers that has to be imposed. To keep the number of selected classifiers odd throughout the searching process, we propose a specific design of the crossover and mutation operators. Mutation is rather easy to implement as assuming an existing odd number of classifiers set randomly during initialisation, this odd number of selected classifiers can be preserved by mutating a pair of bits or in general any even number of bits. Crossover is much more difficult to control that way. To avoid making the GA too complex, crossover is performed traditionally and after that, if the offspring contains even number of classifiers one randomly selected bit is additionally mutated to bring back the odd number of 1's in the chromosome. To increase exploration ability of the GA, an additional operator of 'pairwise exchange' has been introduced, which simply swaps a random pair of bits within the chromosome preserving the same number of classifiers. In order to preserve the best combinations from generation to generation we applied a specific selection rule, known as elitism, according to which populations of parents and offsprings are put together and then regardless of age a number of best chromosomes equal to the size of population is selected for the next generation. As an attempt to address generalisation problems, a simple diversifying operator has been developed. It forces all chromosomes to be different from each other (unique), by mutating random bits until this requirement is reached. The complete algorithm can be defined as follows:

1. Initialise a random population of n chromosomes.
2. Calculate the fitness for each chromosome.
3. Perform crossover and mutate single bits of offsprings with even number of 1's.
4. Mutate all offsprings at randomly selected points.
5. Apply one or more pairwise exchanges for each offspring.
6. Pool offspring and parents together and select n best, unique chromosomes for the next generation.
7. If convergence then finish, else go to step 2.

Although this particular implementation of GA represents a hill-climbing algorithm, multiple mutation and pairwise exchange together with the diversification operator substantially extend the exploration ability of the algorithm. The convergence condition can be associated with the case when no change in the average fitness is observed for an arbitrarily large number of generations. Preliminary comparative experiments with real classification datasets confirmed the superiority of the presented version of the GA to its standard defini-

tion and highlighted the importance of diversification operator for the classifier selection process.

3.3.2. Stochastic hill-climbing search (SS)

Stochastic hill-climbing search in its standard form is not a population-based algorithm yet shares some similarities with GAs particularly in the encoding of the problem and resembles Tabu search [11,28]. Instead of a population, it uses only a single chromosome, mutated randomly at each step. Due to this fact there can be no crossover and the only genetic change is provided by mutation. This limits strongly the ability of the algorithm to jump into different regions of the search space. Moreover, the changes are accepted only if the new chromosome is fitter than its predecessor. Such direct searching usually reaches convergence much faster than a typical GA, but on the other hand, a global optimum may not be found, as it simply may be unreachable from the initial conditions. Effectively, the stochastic search in its original version quite easily gets trapped in local optima. This problem is usually resolved through multiple searches run from different starting points and the best result is taken as the final solution. However, this operation drastically increases time of searching and eliminates the speed of searching, as is an attractive feature of SS. Another possibility is to introduce multiple consecutive mutations, or perform several differentiating operators before the fitness is examined. Similarly to the solution promoted for the case of GA, mutation together with the pairwise exchange is introduced for SS to increase its exploration abilities. Similarly to GA, rather than a single best solution, the population of the best chromosomes are retained, for which the diversifying operator ensures that no duplicated solutions are stored. The presented version of SS algorithm can be described in the following steps:

1. Create a single random chromosome.
2. Mutate the chromosome at randomly selected one or many points.
3. Apply one or more pairwise bit exchanges.
4. Test the fitness of the new chromosome: if it is fitter than its predecessor, then the changes are accepted, else the changes are rejected.
5. Store the new chromosome if it is among n unique best solutions found so far.
6. If convergence then finish, else go to step 2.

As for the previous algorithm, the convergence condition is satisfied if a pool of n best solutions is not changed for a fixed number of generations.

3.3.3. Population-based incremental learning (PBIL)

Stochastic search represents a fast hill-climbing algorithm. However due to the lack of crossover operator, even after many adjustments, the algorithm partially loses the ability to explore the whole search space. We would like the search algorithm to have the ability of reaching most points of the search space, while keeping convergence at a satisfactory level. An algorithm offering these properties is called population-based incremental learning (PBIL) [25]. It also uses a population of chromosomes, sampled from a special probability vector, which is updated at each step according to the fittest chromosomes. The update process of the probability vector is performed according to a standard supervised learning method. Given the probability vector $\mathbf{p} = [p_1, \ldots, p_M]^{\mathrm{T}}$, and population of chromosomes $P = [\mathbf{v}_1, \ldots, \mathbf{v}_C]$, where $\mathbf{v}_j = [\omega_{j1}, \ldots, \omega_{jM}]^{\mathrm{T}}$, each probability bit is updated as in the following expression:

$$p_i^{\mathrm{new}} = p_i^{\mathrm{old}} + \Delta p_i, \quad \Delta p_i = \eta\left(\frac{\sum_{j=1}^{C} \omega_{ji}}{C} - p_i\right), \tag{20}$$

where $j = 1, \ldots, C$, $i = 1, \ldots, M$ refers to the C fittest chromosomes found and η controls the magnitude of the update. A number of best chromosomes taken to update the probability vector together with the magnitude factor η control a balance between the speed of reaching convergence and the ability to explore the whole search space. According to the standard algorithm, the only information that remains after each step is the probability vector, from which the chromosomes are generated. Note that convergence means that $\Delta p_i \to 0$, which implies $p_i \to \omega_{ji}$ for all the chromosomes in the population. This means that after convergence the probability vector becomes the selection proposition $\boldsymbol{\omega}_j$, which is supposed to be the outcome of searching. As for previous algorithms we adjust the standard PBIL to search for an arbitrary number of unique best chromosomes preserving odd number of unit genes. The uniqueness of the chromosomes is again controlled by a diversification operator introduced for the previous algorithms. The complete PBIL algorithm can be described in the following steps:

1. Create a probability vector of the same length as the required chromosome and initialise it with values of 0.5 at each bit.
2. Create a population of chromosomes according to the probability vector.
3. Evaluate fitness of sample by calculating MVE for the combinations defined by the chromosomes.
4. Update the probability vector using (20).
5. Update the pool of the n best unique solutions.
6. If all elements in probability vector are 0 or 1 then finish, else go to step 2.

PBIL algorithm does not use any genetic operators observed in GA. However it contains a specific mechanism that allows exploiting beneficial information from generation to generation, and therefore preserves the directed search elements of evolutionary algorithms.

Table 1
A list of classifiers used in the experiments

No	Name	Description
1	klclc	Linear classifier using KL expansion of common covariance matrix
2	loglc	Logistic linear classifier
3	fisherc	Fisher's least square linear classifier
4	ldc	Linear discriminant classifier
5	nmc	Nearest mean classifier
6	qdc	Quadratic Bayes normal classifier
7	quadrc	Quadratic discriminant classifier
8	pfsvc	Pseudo-Fisher support vector classifier
9	knnc	K-nearest neighbour classifier
10	parzenc	Parzen density based classifier
11	subsc	Subspace classifier
12	treec	Decision tree classifier
13	lmnc	Levenberg–Marquardt neural network classifier
14	rbnc	Radial basis neural network classifier
15	bpxnc	Feed forward neural network classifier with back-propagation

Table 2
A list of datasets used in the experiments

Name	Size	#Train	#Test	#Feat	#Class
iris	150	150	0	4	3
wine	178	178	0	13	3
biomed	194	194	0	5	2
sonar	208	208	0	60	2
glass	214	214	0	10	6
thyroid	215	215	0	5	3
synthetic	1250	250	1000	2	2
azizah	291	291	0	8	20
liver	345	345	0	6	2
ionosphere	351	351	0	34	2
cancer	569	569	0	30	2
diabetes	768	768	0	8	2
conetorus	800	400	400	2	3
vehicle	946	946	0	18	4
chromo	1143	565	578	8	24
segment	2310	1000	1310	19	7
concentric	2500	1000	1500	2	2
gauss2	5000	1000	4000	2	2
gauss4	5000	1000	4000	4	2
gauss8	5000	1000	4000	8	2
clouds	5000	1000	4000	2	2
phoneme	5404	1000	4404	5	2
texture	5500	1000	4500	40	11
satimage	6435	1000	5435	36	6
cbands	12,000	1000	11,000	30	24
shuttle	58,000	1000	57,000	9	7
letters	20,000	1000	19,000	16	26

3.4. Experimental investigations

A comprehensive series of experiments with 15 classifiers (see Table 1) from PRTOOLS [1] applied to 27 datasets (see Table 2 for details) have been carried out to examine both the quality of the search algorithms and the relevance of the selection criteria. The classifiers have been trained on training sets and tested on separate testing sets producing outputs which have been hardened to the binary form: 0-correct, 1-incorrect. To increase reliability the experiments for each dataset have been repeated 100 times for different splits between training and testing sets. The resulting binary outputs have been organised in a form of binary matrices B that have been further split into validation B_V and testing B_T matrices, such that their sizes are approximately the same, equal to $N/2 \times M$. The selection process is carried out on the validation matrix B_V and the testing matrix B_T is used to evaluate the performance of the combiner applied to selected classifiers. Initially, the heuristic methods SB and NB are applied to all the matrices. These two selection methods have been separated from the rest as they can operate only on the criterion of individual classifier performances. Surprisingly, it turned out that NB returned the individual best classifier for all the datasets and thus was equal to the SB selection. For that reason Table 3 presents the selection results only for the SB method. As is apparent from the results, out of 27 datasets in five cases the best classifier selected for the validation set is different than the best

Table 3
Individual best classifier errors (%) for 27 available datasets

Dataset	E_V	E_T	$E_{V\to T}$	Val best	Test best
iri	2.56	2.29	2.29	4	4
win	1.22	1.76	1.76	6	6
bio	9.35	9.57	9.59	7	6
son	16.46	15.58	15.58	8	8
gla	2.02	1.80	1.80	8	8
thy	3.78	3.70	3.70	15	15
syn	12.86	13.44	13.67	9	10
azi	31.16	30.27	30.27	3	3
liv	32.28	32.23	32.27	15	2
ion	5.73	5.78	5.93	7	6
can	2.89	2.81	2.81	8	8
dia	23.15	23.08	23.08	2	2
cnt	16.59	15.91	15.91	10	10
veh	16.56	16.35	16.35	6	6
chr	52.03	52.89	52.89	6	6
seg	7.74	7.57	7.57	8	8
cnc	1.22	1.05	1.05	15	15
ga2	26.72	26.24	26.24	7	7
ga4	18.15	18.49	18.50	7	6
ga8	9.90	10.32	10.32	6	6
clo	11.38	11.25	11.25	10	10
pho	16.65	16.51	16.51	8	8
tex	0.45	0.50	0.50	4	4
sat	14.95	15.45	15.45	10	10
cba	26.85	26.69	26.69	4	4
shu	1.53	1.54	1.54	8	8
let	26.19	26.21	26.21	6	6

The first three columns correspond to errors obtained for SB method applied to validation matrix, testing matrix and validation matrix but tested on the testing matrix. The following two columns show the index of the best classifier evaluated separately in B_V and B_T matrices.

[1] Pattern Recognition Toolbox (PRTOOLS 3.0) for Matlab 5.0+, implemented by R.P.W. Duin, available free at ftp://ftp.ph.tn.tudelft.nl/pub/bob/prtools.

Table 4
Summary of searching methods, selection criteria and datasets used in experiments

No	Searching methods	Selection criteria
1	ES—Exhaustive search	SB—Single best
2	RS—Random search	ME—Mean classifier error
3	FS—Forward search	MVE—Majority voting error
4	BS—Backward search	MVI—Majority voting improvement
5	SS—Stochastic search	C2—Correlation
6	GS—Genetic search	PM2—Product moment correlation
7	PS—PBIL search	D2—Disagreement measure
8		F2—Double-fault measure
9		Q2—Q statistics measure
10		DI—Difficulty measure
11		EN—Entropy measure
12		IA—Interrater agreement
13		KW—Kohavi–Wolpert variance
14		GD—Generalised diversity
15		CFD—Coincident failure diversity
16		FM—Fault majority

classifier for the testing set. In general the scale of such dissonance depends on many factors including number of classifiers, classes, data size and complexity of the problem.

3.4.1. Evaluation of selection criteria

In the first experiment, greedy and evolutionary searching methods have been applied to binary matrices. For comparison, an exhaustive search (ES) and random search (RS) methods have been included in the experiment. The RS method first randomly picks the odd number of classifiers and then randomly selects required classifiers. For each dataset from Table 2, a combination of 7 selection methods with 16 different selection criteria as summarised in Table 4.

In all experiments, we used the same parameters in the algorithms, for which preliminary experiments showed the best results. Both PBIL and GA used 50 chromosomes in the population. In SS and GA, single bit mutation was applied together with single pairwise exchange operation. The learning rate for PBIL was set to $\eta = 1$. To be able to compare the algorithms in terms of efficiency, in all experiments, the algorithms finished the run after examining a fixed number of chromosomes, which was used instead of specifying convergence conditions.

The criteria of selection were examined in the first instance. For this purpose, the results have been aggregated over the datasets and reduced thus to a matrix of size 16×7 containing average majority voting errors for the best combinations found by specific searching method using specific selection criterion. These results are shown in Table 5 and depicted in Fig. 1.

The results clearly show that searching directly according to the combiner error is optimal and returns well performing combinations. Virtually all the search algorithms work well with this criterion and the combinations of classifiers they return are very similar or identical to the optimal combinations found by exhaustive search. The results related to other criteria confirm our assumptions based on the analysis of the correlation between majority voting error and various diversity measures carried out in [29]. The measures weakly correlated with MVE perform badly as a classifier selection criteria. The measures with better correlation with MVE (ME, F2, FM) guide the selection process much better, although the performance of the

Table 5
Majority voting errors obtained for best combinations of classifiers selected by various searching methods (columns) and selection criteria (rows)

Criteria	ES	RS	FS	BS	SS	GS	PS	Average
SB	14.41	18.01	14.41	16.62	17.74	18.61	18.17	16.85
ME	14.41	15.11	14.41	14.91	14.42	14.49	14.41	**14.59**
MVE	13.93	14.24	13.95	13.95	13.97	13.94	13.94	**13.99**
MVI	16.07	16.83	16.24	15.76	16.44	16.22	16.25	16.26
C2	30.11	27.05	23.11	17.25	30.05	29.20	30.11	26.69
PM2	19.95	20.07	14.41	16.80	19.69	20.15	20.03	18.73
D2	21.89	19.09	14.97	15.64	17.02	17.20	16.83	17.52
F2	15.08	15.25	15.15	14.49	15.03	15.05	15.23	**15.04**
Q2	30.51	25.46	19.23	17.83	27.89	27.80	30.23	25.57
DI	32.87	20.43	14.44	16.50	19.97	30.96	32.87	24.01
EN	21.57	21.69	16.05	18.08	21.16	21.61	21.57	20.25
IA	25.52	23.99	23.94	16.75	25.23	25.57	25.52	23.79
KW	25.20	23.42	17.61	17.83	22.38	25.22	25.20	22.41
GD	23.72	21.33	17.95	16.08	18.24	22.08	22.81	20.32
CFD	21.64	19.99	16.42	14.94	16.96	23.29	20.03	19.04
FM	15.07	15.05	15.07	14.75	15.17	14.84	15.07	**15.00**
Average	21.37	19.81	**16.71**	**16.14**	19.46	21.01	21.14	

The results are averaged over 27 datasets. The bottom row and right-most column show the averaged values of MVE for the searching methods and selection criteria, respectively.

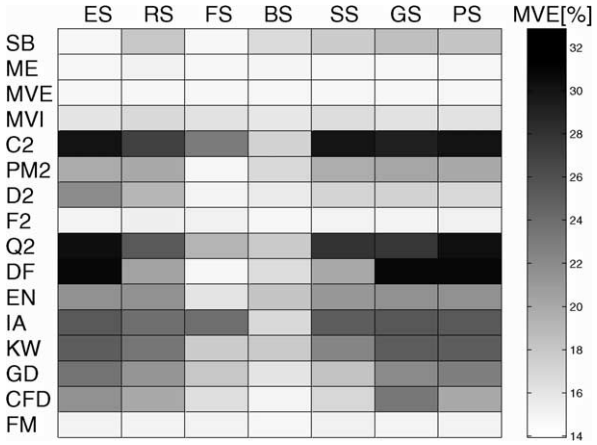

Fig. 1. Visualisation of the majority voting errors presented in Table 5. The lighter the field the lower the majority voting error. Details of classifiers and datasets are provided in Tables 1 and 2.

combinations found as a result of their use is on average around 1% worse than the performance of the combinations found using directly MVE measure. Additional drawbacks of using diversity measures as a selection criteria stem from the fact that most of them are not invariant to the size of combinations, which means they may mislead the search algorithm. An interesting phenomenon has been observed for greedy searches which for most of selection criteria resulted in a singleton or complete ensemble combinations. Surprisingly, the greedy rule turns out to be a property that prevents from misleading effects of using diversity measures. Taking this fact into account, although results indicate that greedy searches perform the best on average, it should be rather said that greedy algorithms are the most resistant to bad selection criteria. The fact worth noting though is that both the selection of individual best classifier or the whole ensemble are quite good and usually reliable solutions. Comparison of the results for the MVE as a selection criterion shows that, apart from random search, all searching methods perform very well with GS and PS almost reaching the level of performance obtained for the optimal exhaustive search.

Similar results from search algorithms with MVE selection criterion do not necessarily prove the excellence of the implemented methods. It may also be the result of a very flat search space where a large amount of different solutions share the same performance. This phenomenon is certainly present in the classifier selection process where swapping only a single classifier changes the solution of the selection process but leaves the performance of the updated combination virtually unchanged.

As the MVE decisively appears to be the best selection criterion, we ignore other measures in subsequent analysis and focus on the properties of classifier selection based on MVE measure.

3.4.2. Searching method evaluation

In terms of the number of returned solutions the presented search algorithms can be divided into individual and population-based methods. The heuristic (SB, NB) and greedy (FS, BS) search algorithms due to their nature are capable of returning only one optimal combination of classifiers. All remaining searching methods either are naturally operating on the population of solutions (GA, PBIL) or can be easily adjusted to do so, just by keeping track of the combinations passed by the algorithm (SS, ES, RS). To start with, Table 6 presents the optimal combinations of classifiers found by exhaustive search for all the datasets. These results provide some initial characteristics of the classifier selection problem. First of all, for around 25% of the datasets, picking the individual best performing classifier is the optimal selection strategy. Secondly the size of the optimal combinations of classifiers may vary, though smaller subsets of classifiers appear to be more successful. Moreover, despite large sizes of the matrices of outputs, differences between validation and testing matrices cause inconsistencies in the selection optimality. As a result of such simulation of the generalisation problem, we see that optimal combinations for validation and testing matrices can be different, causing loss of performance by up to 1% for some datasets.

All the searching methods are further compared in terms of the majority voting error obtained for the best combinations they found. For the SB, FS and BS only the results for the single best combination are shown while for population-based GS, RS, GA and PS the results for both individual best and mean from 50 best combinations are presented. To ensure uniqueness of the combinations for the population-based search algorithms, a diversifying operator has been applied that mutates duplicate binary selection vectors until they are unique in the population. The searching results for the 27 datasets are shown in Table 7 for the validation matrices and the corresponding testing errors are shown in Table 8. The dynamics of the errors for 50 best combinations in population-based methods is further depicted in Fig. 2 for selected datasets. The first thing to note is that PS almost always finds the optimal combination and hence its average performance is virtually equal to the exhaustive search. Another observation is that FS and BS are well performing greedy algorithms, which surprisingly outperformed the genetic algorithm search. In terms of the population of the best solutions, PBIL search reinforces its leading position with a quality very close to the exhaustive search quality. As expected, GS outperformed SS in terms of the average performance of the 50 best combinations found, despite SS returning on average better individual best combinations. The generalisation errors presented in Table 8 and shown in thin grey lines in Fig. 2 show even more surprising results. Greedy algorithms FS and

Table 6
Best combination of classifiers found by the exhaustive search from the ensemble of 15 classifiers

Dat.	E_V	E_T	$E_{V\to T}$	Val. combination	Test combination
iri	1.97	1.60	1.60	1 5 11	1 5 11
win	0.82	1.13	1.33	2 4 5 6 7 14 15	2 3 5 6 14
bio	9.18	9.40	9.53	2 5 6 7 14	1 2 6 7 10
son	16.43	15.44	16.38	5 6 8 9 11 12 15	8 9 15
gla	2.01	1.72	1.79	8	5 8 9
thy	3.37	2.88	3.05	2 7 8 11 15	7 11 15
syn	12.28	13.15	13.34	2 5 9 10 13	9 10 14
azi	30.76	29.53	30.85	1 3 9	3 6 9
liv	29.06	28.56	29.14	1 4 5 7 9 11 12 13 15	2 4 6 8 9 11 12 13 15
ion	4.87	4.65	4.65	6 7 8 12 13	6 7 8 12 13
can	2.72	2.70	2.73	1 8 9 13 15	2 7 8 9 10 13 15
dia	22.95	22.90	23.00	1 2 3 5 10 11 12 13 15	1 2 3 4 5 10 11 12 13 14 15
cnt	15.70	15.71	15.76	5 7 8 10 12	5 8 10 12 13
veh	16.55	16.35	16.35	6	6
chr	52.03	52.88	52.88	6	6
seg	5.41	5.53	5.56	3 6 9 11 12	3 4 6 8 9 11 12
cnc	1.21	1.05	1.05	15	15
ga2	26.71	26.24	26.24	7	7
ga4	18.12	18.46	18.47	6 7 13	6 7 8
ga8	9.89	10.32	10.32	6	6
clo	11.17	11.10	11.28	10 13 15	10 12 13
pho	16.55	16.50	16.68	6 8 12	8
tex	0.30	0.32	0.37	1 2 3 4 6 7 11	2 3 4 6 11
sat	14.50	14.96	15.00	5 6 8 9 10	1 6 8 9 10
cba	25.56	25.67	25.67	4 7 10	4 7 10
shu	1.14	1.14	1.14	8 11 12	8 11 12
let	26.18	26.20	26.20	6	6

Columns 2–4 present the MVE values for the best combination found in the validation matrix, testing matrix and validation best tested on the testing matrix, respectively. Columns 4 and 5 show indices of the classifiers forming the best validation and testing combinations. Details of classifiers and datasets are provided in Tables 1 and 2.

BS showed the best average results for individual combinations apparently outperforming the exhaustive search. This may indicate that exhaustive search leads to overfitting the search space. Both validation and testing results for the populations of 50 best combinations shown in Fig. 2 confirm the ranking of searching methods: PS, GS, SS, obtained for the validation matrices, although SS seems to outperform GS for the first 20 best combinations. Despite applied aggregation, the performances of the unstable testing performance among the best combinations, illustrated by thin grey lines in Fig. 2 indicate the level of generalisation problems involved. The thick lines in Fig. 2 showing evolution of the 50 best combinations illustrate the dynamics of the increase in MVE for the validation sets for which the searching was carried out. For all the above experiments the performance reliability measured as a variance of obtained performances over 100 different splits between training and testing sets has been carried out. The reliability results remain consistent with our previous work [30] and due to lack of presentation space have been skipped here. However, on average a consistent reduction of the variance in performance is observed for larger combinations and even further for the combinations of combinations of classifiers.

4. Multistage selection–fusion model (MSF)

The classifier selection models analysed so far involved searching for the optimal combination of classifiers and apply the fusion method on their outputs for the final classification results. Moreover, some potential generalisation losses of performance have been shown for the exhaustive search, and these seem to be equally harmful for other searching methods picking a single combination as a result. Greedy algorithms have been identified as relatively good generalisers, however, they are still capable of returning only individual best combination and are still quite complex: cubically $O(M^3)$ in the current implementation or quadratic if ties are resolved. On the other hand the complexity of population-based searching methods can be flexibly adjusted depending on the size of the population and the number of generations to proceed. Furthermore, the fact that these algorithms return populations of best combinations can potentially be exploited to prevent generalisation problems. A majority voting combiner based on binary classifier outputs has the capacity to incorporate many combinations of classifiers. This is possible as a combination of classifiers is in fact a combination of columns from the binary matrix of outputs and majority voting applied to such a combination returns a column

Table 7
Validation errors (obtained from the validation matrices) of the majority voting combiner obtained for the best combinations and mean from 50 best (if possible) combinations of classifiers found by 8 different search algorithms for 27 datasets

Dat.	ES		SB	RS		FS	BS	SS		GS		PS	
	1b	50b	1b	1b	50b	1b	1b	1b	50b	1b	50b	1b	50b
iri	1.97	2.40	2.56	2.42	2.57	2.13	1.97	1.97	2.46	2.13	2.45	1.97	2.42
win	0.82	0.94	1.22	0.93	1.13	0.91	0.82	0.86	1.01	0.86	0.99	0.82	0.95
bio	9.18	9.32	9.34	9.32	9.78	9.30	9.22	9.18	9.43	9.24	9.39	9.18	9.33
son	16.43	16.73	16.45	16.57	17.34	16.45	16.45	16.43	16.90	16.43	16.79	16.43	16.77
gla	2.01	2.13	2.01	2.07	2.42	2.01	2.07	2.03	2.16	2.03	2.15	2.03	2.13
thy	3.37	3.55	3.77	3.50	3.85	3.40	3.40	3.37	3.66	3.37	3.58	3.37	3.56
syn	12.28	12.52	12.85	12.49	12.83	12.42	12.34	12.34	12.60	12.39	12.59	12.28	12.55
azi	30.76	31.58	31.15	31.37	33.35	30.76	30.76	30.76	31.90	30.76	31.73	30.76	31.62
liv	29.06	29.30	32.27	29.16	29.55	29.06	29.13	29.06	29.41	29.06	29.37	29.06	29.35
ion	4.87	5.26	5.72	5.13	6.27	4.87	4.87	4.87	5.40	4.87	5.39	4.87	5.28
can	2.72	2.79	2.89	2.72	2.92	2.72	2.78	2.72	2.82	2.72	2.82	2.72	2.80
dia	22.95	23.04	23.15	23.09	23.17	22.95	22.95	22.95	23.09	22.95	23.07	22.95	23.05
cnt	15.70	16.01	16.58	16.07	16.76	15.86	15.70	15.70	16.21	15.70	16.07	15.70	16.02
veh	16.55	17.30	16.55	17.36	19.13	16.55	16.55	16.55	17.36	16.62	17.43	16.55	17.34
chr	52.03	54.18	52.03	54.39	57.50	52.03	52.03	52.03	54.82	52.56	54.57	52.03	54.36
seg	5.41	5.74	7.74	5.41	6.34	5.45	5.41	5.41	5.90	5.41	5.81	5.41	5.75
cnc	1.21	1.66	1.21	1.63	1.94	1.21	1.21	1.21	1.68	1.21	1.70	1.21	1.66
ga2	26.71	26.86	26.71	26.82	27.22	26.71	26.71	26.73	26.95	26.71	26.89	26.73	26.88
ga4	18.12	18.37	18.15	18.15	19.15	18.12	18.14	18.12	18.45	18.12	18.41	18.12	18.39
ga8	9.89	10.35	9.89	10.54	11.63	9.89	9.89	9.89	10.64	9.89	10.45	9.89	10.38
clo	11.17	11.45	11.38	11.24	12.14	11.17	11.17	11.17	11.50	11.17	11.48	11.17	11.46
pho	16.55	17.04	16.64	16.98	17.90	16.55	16.72	16.55	17.26	16.55	17.09	16.55	17.04
tex	0.30	0.35	0.45	0.34	0.44	0.32	0.30	0.32	0.37	0.30	0.36	0.30	0.35
sat	14.50	14.79	14.94	14.74	15.47	14.50	14.50	14.50	14.83	14.50	14.88	14.50	14.80
cba	25.56	27.10	26.84	27.06	29.07	25.56	25.56	25.56	27.44	25.56	27.32	25.56	27.11
shu	1.14	1.44	1.52	1.42	1.75	1.14	1.14	1.14	1.46	1.14	1.46	1.14	1.45
let	26.18	28.37	26.18	27.96	32.11	26.18	26.18	26.67	28.62	26.67	28.79	26.18	28.39
Average	13.98	14.47	14.45	14.40	15.32	14.01	14.00	14.00	14.60	14.03	14.56	13.98	14.49

Details of classifiers and datasets are provided in Tables 1 and 2.

of binary outputs, which can be perceived as a higher level classifier. Hence if a fusion method is applied to many such combinations of classifiers, we get many new classifiers, which can subsequently undergo the selection and fusion processes.

4.1. Network of outputs

It can be noted that the classifier outputs selected and combined at many layers form a specific network structure. At each layer of such network the aggregation of outputs is followed by the generation of a new layer of outputs returned from the combiner applied to selected subsets of outputs from the previous layer. In our model the combiner (majority voting) realises aggregation or fusion of outputs while the selection method is responsible for generation of new outputs (MVE for the outputs selected at previous layer). Hence, a multistage selection–fusion model realises a network of outputs or more precisely tries to establish the optimal network structure such that if it is applied to classifier outputs, the error from the final layer of the network is minimal. It has to be noted that the condition of optimality is imposed at each layer in the sense that the edges of the network are established as a result of the search algorithm that searches for the optimal combination. The size of the network is completely arbitrary and predefined by the number of layers and the number of best combinations at each layer. In the design of the selection fusion model, the focus is purely on achieving further improvement in performance compared to the traditional single-layer selection. It is anticipated that due to the accumulation of many good yet different combinations of classifiers, such a model would better deal with the generalisation problems. A series of experiments with the five-layer networks are intended to verify these expectations.

4.2. Analysis of generalisation ability

As mentioned above, a major motivation for designing the MSF model is the potential use of the population of best combinations returned by population-based search algorithms in an attempt to improve the system's generalisation performance. For this purpose the experiments intended to evaluate the quality of search algorithms from Section 3.4.2 have been repeated

Table 8
Generalisation errors (evaluated on the testing matrices) of the majority voting combiner obtained for the best combinations and mean from 50 best (if possible) combinations of classifiers found by eight different search algorithms for 27 datasets

Dat.	ES		SB	RS		FS	BS	SS		GS		PS	
	1b	50b	1b	1b	50b	1b	1b	1b	50b	1b	50b	1b	50b
iri	1.60	2.17	2.29	2.02	2.36	1.81	1.60	1.60	2.18	1.81	2.22	1.60	2.18
win	1.33	1.38	1.75	1.42	1.56	1.26	1.33	1.22	1.45	1.33	1.42	1.33	1.40
bio	9.53	9.64	9.59	9.69	9.97	9.81	9.63	9.53	9.72	9.63	9.70	9.53	9.64
son	16.38	16.35	15.58	16.53	16.67	15.58	16.34	16.38	16.42	16.38	16.42	16.38	16.42
gla	1.79	1.81	1.79	1.83	2.05	1.79	1.83	1.79	1.83	1.79	1.81	1.79	1.81
thy	3.05	3.38	3.70	3.37	3.69	3.22	2.88	3.05	3.43	3.05	3.39	3.05	3.39
syn	13.34	13.47	13.66	13.42	13.77	13.20	13.33	13.33	13.56	13.33	13.53	13.34	13.54
azi	30.85	31.09	30.27	31.05	32.81	30.85	30.85	30.85	31.42	30.85	31.23	30.85	31.09
liv	29.14	29.25	32.26	29.12	29.46	29.14	29.09	29.14	29.23	29.14	29.24	29.14	29.28
ion	4.65	5.17	5.93	4.92	6.30	4.65	4.65	4.65	5.32	4.65	5.30	4.65	5.19
can	2.73	2.81	2.81	2.73	2.96	2.73	2.79	2.73	2.85	2.75	2.83	2.75	2.82
dia	23.00	23.10	23.08	23.39	23.19	23.00	23.00	23.00	23.09	23.00	23.12	23.00	23.11
cnt	15.76	16.00	15.91	16.13	16.62	15.97	15.76	15.76	16.16	15.76	16.03	15.76	16.00
veh	16.35	16.92	16.35	16.82	18.68	16.35	16.35	16.35	16.97	16.47	17.02	16.35	16.94
chr	52.88	54.77	52.88	55.37	58.13	52.88	52.88	52.88	55.47	53.37	55.19	52.88	54.95
seg	5.56	5.80	7.56	5.56	6.33	5.53	5.56	5.56	5.88	5.56	5.85	5.56	5.80
cnc	1.05	1.47	1.05	1.41	1.78	1.05	1.05	1.05	1.54	1.05	1.54	1.05	1.46
ga2	26.24	26.41	26.24	26.42	26.80	26.24	26.24	26.25	26.49	26.24	26.41	26.25	26.43
ga4	18.47	18.69	18.49	18.49	19.53	18.47	18.46	18.47	18.79	18.47	18.74	18.47	18.71
ga8	10.32	10.81	10.32	10.99	12.13	10.32	10.32	10.32	11.08	10.32	10.90	10.32	10.84
clo	11.28	11.32	11.25	11.25	11.90	11.28	11.28	11.28	11.39	11.28	11.34	11.28	11.33
pho	16.68	17.08	16.50	16.71	17.91	16.68	16.68	16.68	17.33	16.68	17.12	16.68	17.07
tex	0.37	0.40	0.50	0.40	0.48	0.36	0.37	0.41	0.42	0.37	0.41	0.37	0.40
sat	15.00	15.19	15.44	15.17	15.72	15.00	15.00	15.00	15.24	15.00	15.26	15.00	15.19
cba	25.67	27.12	26.68	26.99	29.18	25.67	25.67	25.67	27.49	25.67	27.32	25.67	27.14
shu	1.14	1.47	1.53	1.36	1.78	1.14	1.14	1.14	1.51	1.14	1.49	1.14	1.47
let	26.20	28.19	26.20	27.71	32.16	26.20	26.20	26.46	28.46	26.46	28.64	26.20	28.21
Average	14.09	14.49	14.43	14.45	15.33	14.08	14.08	14.09	14.62	14.13	14.57	14.09	14.51

Details of classifiers and datasets are provided in Tables 1 and 2.

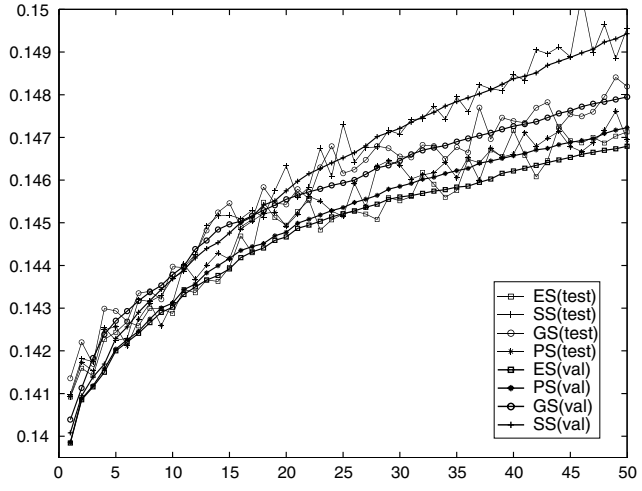

Fig. 2. Comparison of the errors from 50 best combinations of classifiers found by four population-based searching methods: ES, SS, GS, PS.

in an updated form incorporating the subsequent processes of selection and fusion of outputs. Due to the large size of the problem and high space and time complexities, the network has been limited to 5 layers of 15 nodes at each layer. The inputs of such a network represent the outputs from individual classifiers, while the network output can be defined as the MVE output for the best combination selected at the final layer. Such experiments have been run for all 27 datasets and the results obtained in the form of the error from the best combination and the mean from the 15 best combinations at each layer. Due to the huge size of these results an aggregation along the datasets has been applied. Fig. 3 shows these aggregated results for all population-based search algorithms examined on the validation and testing matrices. Results specific to individual datasets are shown in Table 9.

The results clearly show that further selection and fusion beyond the first layer on average bring some improvement to the overall system performance. Particularly valuable is the reduction in the generalisation error which as shown in Fig. 3 (right column) was observed for all population-based searching methods except random search. A further conclusion coming from Fig. 3 is that the improvement of performance is generally observed up to the second or third layer which is followed by a slight increase or plateau in system error. For further understanding of this behaviour Fig. 4

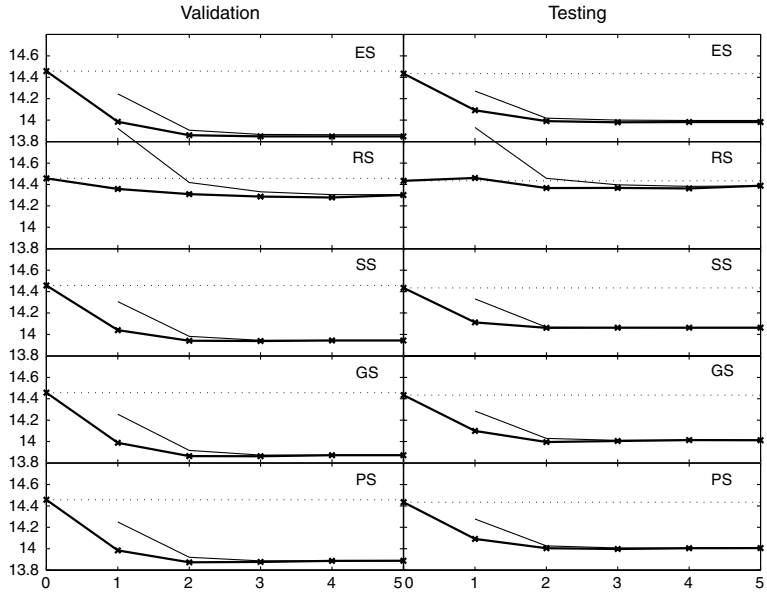

Fig. 3. Evolution of the MVE for the MSF model with a network of 5 layers and 15 nodes at each layer. The thick line shows the MVE values for the best combinations found by different search algorithms at each layer (1–5) of the MSF model. For comparison purposes this line starts from the error of the single best classifier (layer 0), the level of which is also marked by the dotted line. The thin line shows the analogous evolution of the mean MVE from all the combinations selected at each layer. Details of classifiers and datasets are provided in Tables 1 and 2.

depicts the network obtained from the considered MSF applied to the *phoneme* dataset. The complex connections up to the third layer quickly converge to the redundant connections emerging from third layer where the selector is passing the same individual combinations to the next layer.

Although the concept of diversity has been undermined due to insufficient correlation with the combiner performance, the network from Fig. 4 indicates that the best combinations are composed of complementary rather than individually best performing classifiers. For example the best combination at the first layer includes the best performing support vector classifier (No. 8) with an error of 16.66% and two much worse quadratic and decision tree classifiers (No. 6, 12), with errors of 20.67% and 20.81%. These three classifiers combined by majority voting error result in the 16.55% error, improving the performance of the individual best classifier. Such behaviour is typical throughout the datasets confirming the fact that some form of complementarity or diversity among the classifiers plays an important role in classifier fusion.

Among the best combinations at each layer, the minimum error is observed at the second layer and this is consistent with the average results for other datasets. It seems then, that a two-layer network is the optimal design for the MSF model, although it is not clear if and how the number of combinations per layer affects the optimality of the network architecture. The relatively small size of the layer, $M_k = 15$, was set deliberately in the experiments to allow for comprehensive tests with many datasets and searching methods including exhaustive search. A striking similarity with the neural network processing inspires further conclusions. It appears that the MSF model implicitly introduces some form of weighting of the individual classifier results. By stretching the scope of fusion into many layers and allowing for multiple use of individual classifiers' outputs in many combinations, MSF seems to adjust a very inflexible majority voting combiner into a flexible combining system, weighting the influence of individual classifiers.

In terms of reliability of MSF, the experiments revealed moderate but positive tendencies. The reliability was measured as a variance of obtained performances over 100 different splits between training and testing sets. The results showed small but consistent reduction of the variance for higher layers reaching on average around 95% of its original value at the first layer. Boosting the reliability of performance is an expected behaviour consistent with our findings from [30] and resulting from the applied aggregations.

5. Discussion

This work intends to provide some remarks relating to the applicability of diversity analysis to a typical task of the classifier fusion system, which is classifier selection. A number of search algorithms have been presented and adopted with the majority voting combiner considered throughout this paper. The algorithms used a binary vector of classifier incidences, indicating exclusion (0) or inclusion (1) of the classifier in the combi-

Table 9
Generalisation errors (evaluated on the testing matrices) of the majority voting combiner obtained for the best combinations from the five-layer selection–fusion model

Dat.	ES		RS		SS		GS		PS	
	E_{min}	L	E_{min}	L	E_{min}	L	E_{min}	L	E_{min}	L
iri	1.60	1	2.10	1	2.26	2	1.60	1	1.60	1
win	1.20	2	1.26	2	1.26	3	1.17	3	1.17	3
bio	9.34	2	9.42	3	9.34	2	9.38	2	9.38	2
son	15.39	3	16.03	2	15.58	1	16.01	2	15.96	3
gla	1.77	2	1.79	3	1.76	3	1.76	3	1.76	3
thy	3.00	2	3.33	3	3.05	1	2.96	2	2.98	3
syn	13.34	1	13.33	1	13.33	1	13.34	1	13.34	1
azi	30.07	2	30.77	2	30.31	2	30.07	2	30.07	2
liv	29.07	3	28.99	3	29.14	1	28.70	5	28.80	2
ion	4.65	1	4.50	3	4.70	3	4.65	1	4.65	1
can	2.72	3	2.75	2	2.70	2	2.73	1	2.73	1
dia	22.94	2	22.91	4	22.96	2	22.94	2	22.95	2
cnt	15.56	3	15.71	3	15.59	2	15.56	3	15.56	4
veh	16.35	1	16.50	1	16.37	2	16.37	2	16.35	1
chr	52.88	1	54.39	1	53.33	1	52.88	1	52.88	1
seg	5.37	2	5.52	4	5.38	3	5.37	2	5.37	2
cnc	1.05	1	1.34	1	1.05	1	1.05	1	1.05	1
ga2	26.24	1	26.58	4	26.24	1	26.24	1	26.24	1
ga4	18.46	2	18.49	1	18.47	1	18.47	3	18.46	2
ga8	10.32	1	10.89	1	10.32	1	10.32	1	10.32	1
clo	11.11	2	11.27	2	11.20	2	11.14	2	11.13	2
pho	16.24	2	16.55	4	16.31	2	16.24	2	16.24	2
tex	0.36	2	0.37	1	0.37	1	0.36	3	0.36	3
sat	15.00	1	15.05	3	14.91	3	15.00	1	15.00	3
cba	25.67	1	26.89	3	25.67	1	25.67	1	25.65	3
shu	1.12	2	1.42	5	1.14	1	1.12	2	1.13	2
let	26.20	1	26.89	1	26.20	1	26.20	1	26.20	1
Average	13.96		14.26		14.03		13.97		13.97	

The columns show the minimum errors obtained and the layer indices at which the minimum errors were observed. Details of classifiers and datasets are provided in Tables 1 and 2.

nation, as a representation of the selection solution. Furthermore a diversifying operator was applied to the populations of solutions, which prevented duplication of the same combinations found as a result of the search algorithms. Flexible implementation of the algorithms allowed us to apply many different selection criteria including many diversity measures defined in Section 2.3.

In the extensive experimental work, all the search algorithms have been applied to the classification results from 15 different classifiers applied to 27 typical datasets. This has been repeated in many versions corresponding to different selection criteria. The majority voting has then been applied to the best combinations returned by the algorithms and provided the basis for the assessment of different diversity measures used as selection criteria. As expected, the selection results turned out to be consistent with our findings from [29]. The better the correlation between the measure (selection criterion) and the combiner performance, the higher the performance of the selected combinations. Ultimately, majority voting error used as a selection criterion showed the optimal results, although a selection based on well correlated F2 and FM measures also provided good results.

In an attempt to improve the generalisation ability of the selected combination of classifiers a new MSF model was developed. This method combines the selection and fusion of the classifier outputs applied at many layers. The experiments with the novel method resulted in an improved generalisation performance compared to the individual best and the combination selected by the single-layer selection model. The results of the experiments point to the two-layer network as the optimal architecture of the MSF model, though it is noted that this may vary for different algorithms and the numbers of nodes (best combinations retained at each layer) fixed for each layer. Based on analogies with the neural network design, the advantage of the MSF model is explained as coming from injection of some flexibility into a very inflexible majority voting combiner. This flexibility comes in the form of a weighted influence of individual classifiers and greater freedom in the interaction between the selection and fusion process, although achieved at the price of higher complexity in a model.

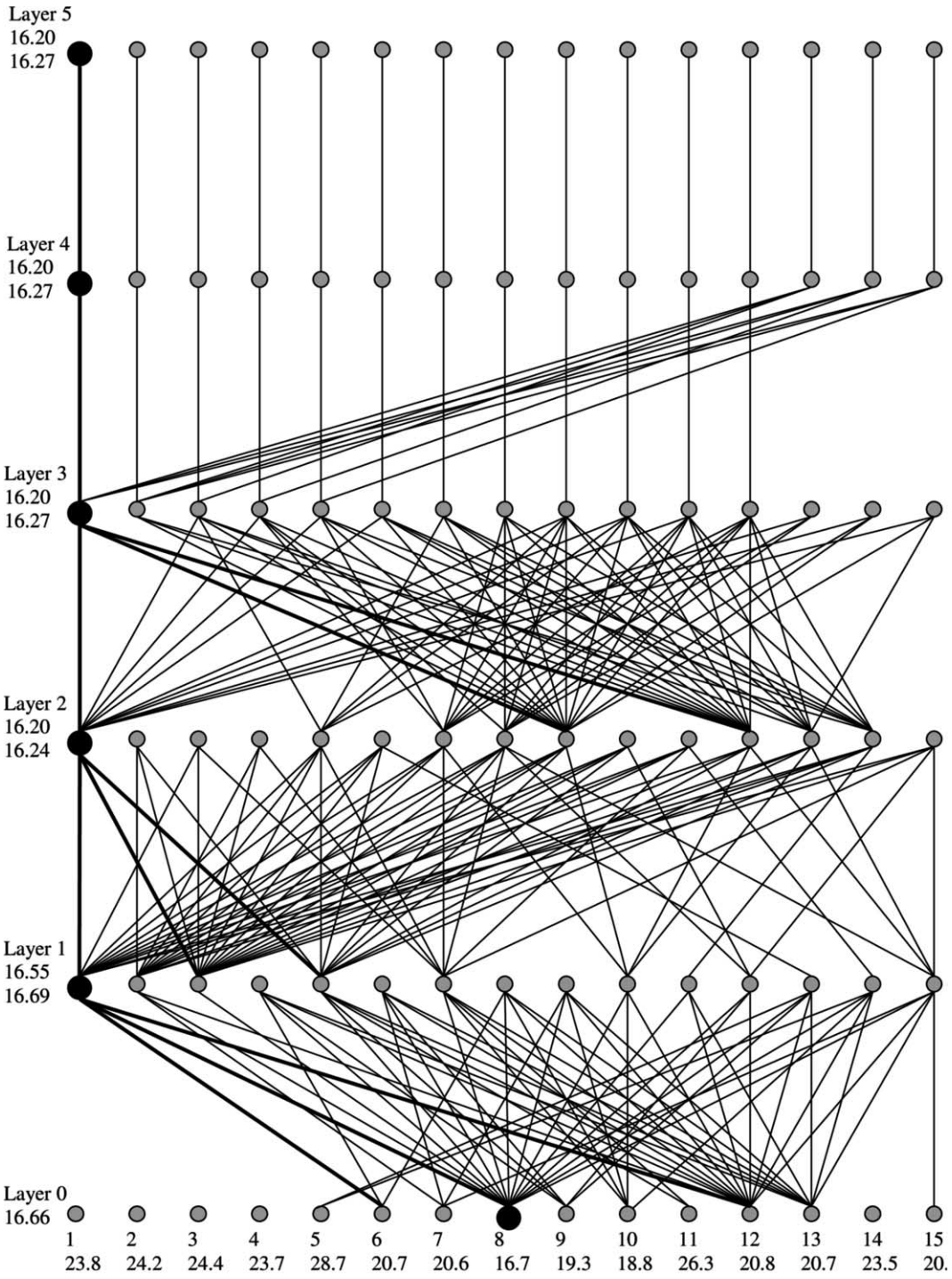

Fig. 4. The network (5×15) resulting from the application of MSF model with $M = 15$ classifiers, majority voting and exhaustive search on the *phoneme* dataset. Layer 0 represents individual classifiers and their individual errors are marked underneath. The best combination at each layer is marked by an enlarged black circle. The validation and testing errors of the best combination at each layer is marked respectively below the layer labels. Details of classifiers and datasets are provided in Tables 1 and 2.

References

[1] S. Baluja, Population-based incremental learning: a method for integrating genetic, search based function optimization and competitive learning, Technical Report No. 163, Carnegie Melon University, Pittsburgh, PA, 1994.

[2] S.-B. Clio, Pattern recognition with neural networks combined by genetic algorithms, Fuzzy Seta and Systems 103 (1999) 339–347.

[3] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, Now York, 1991.
[4] H. Drucker, C. Cortes, L.D. Jackel, Y. LeCun, V. Vapnik, Boosting and other ensemble methods, Neural Computation 6 (1994) 1289–1301.
[5] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, John Wiley and Sons, New York, 2001.
[6] J.L. Pleiss, Statistical Methods for Rates and Proportions, John Wiley and Sons, 1981.
[7] G. Giacinto, P. Roli, Methods for dynamic classifier selection, in: Proceedings of the 10th International Conference on Image—Analysis and Processing, Venice, Italy, 1999, pp. 659–664.
[8] G. Giacinto, F. Roli, A theoretical framework for dynamic classifier selection, in: Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain, Lecture Notes in Computer Science, vol. II, 2000, pp. 8–11.
[9] G. Giacinto, P. Roli, Design of effective neural network ensembles for image classification purposes, Image Vision and Computing Journal 19 (9–10) (2001) 669–707.
[10] G. Giacinto, F. Roli, Dynamic classifier selection based on multiple classifier behaviour, Pattern Recognition 34 (2001) 1879–1881.
[11] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, Boston, 1997.
[12] M. Hamburg, Statistical Analysis for Decision Making, Harcourt Bruce and World, New York, 1970.
[13] H. Handels, T. Ross, J. Kreusch, H.H. Wolff, S.J. Poppl, Feature selection for optimized skin tumor recognition using genetic algorithms, Artificial Intelligence in Medicine 16 (1999) 283–297.
[14] S. Hansen, P. Salamon, Neural network ensembles, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (10) (1990) 993–1001.
[15] J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Michigan, 1975.
[16] J. Kim, K. Seo, K. Chung, A systematic approach to classifier selection on combining multiple classifiers for handwritten digit recognition, in: Proceedings of the 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997, pp. 459–462.
[17] R. Kohavi, D.H. Wolpert, Bias plus variance decomposition for zero-one loss function, in: Machine Learning: Proceedings of the 13th International Conference, Morgan Kaufmann, 1996, pp. 275–283.
[18] L.I. Kuneheva, Cluster-and-selection method for classifier combination, in: Proceedings of the 4th International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, Brighton, UK, 2000, pp. 185–188.
[19] L.I. Kuneheva, L.C. Jain, Designing classifier fusion systems by genetic algorithms, IEEE Transactions on Evolutionary Computation 4 (4) (2000) 327–336.
[20] L.I. Kuneheva, C.J. Whitaker, Feature subsets for classifier combination: an enumerative experiment, in: Proceedings of the 2nd International Workshop on Multiple Classifier Systems, Cambridge, UK, Lecture Notes in Computer Science, LNCS 2096, Springer-Verlag, 2001, pp. 228–237.
[21] L.I. Kuneheva and C.J. Whitaker, Ten measures of diversity in classifier ensembles: limits for two classifiers, in: Proceedings of the IEE Workshop on Intelligent Sensor Processing, Birmingham, UK, 2001, pp. 10/1–10/6.
[22] L.I. Kundieva, C.J. Whitaker, Measures of diversity in classifier ensembles, Machine Learning 51 (2003) 181–207.
[23] L.I. Kuncheva, C.J. Whitaker, C.A. Shipp, R.P.W. Duin, Limits on the majority vote accuracy in classifier fusion, Pattern Analysis and Applications 6 (2003) 22–31.
[24] R. Liu, B. Yuan, Multiple classifiers combination by clustering arid selection, Information Fusion 2 (2001) 163–168.
[25] D. Partridge, W. Krzanowski, Software diversity: practical statistics for its measurement arid exploitation, Information and Software Technology 39 (10) (1997) 707–717.
[26] D. Partridge, W.B. Yates, Engineering multiversion neural-net systems, Neural Computation 8 (1996) 869–893.
[27] G. Rogova, Combining the results of several neural network classifiers, Neural Networks 7 (5) (1994) 777–781.
[28] F. Roli, G. Giadnto, Design of multiple classifier systems, in: Hybrid Methods in Pattern Recognition, World Scientific Publishing, 2002, pp. 199–226.
[29] D. Ruta, B. Gabrys, Analysis of the correlation between majority voting error and the diversity measures in multiple classifier systems, in: Proceedings of the 4th International Symposium, on Soft Computing, Paper No. 1824-025, Paisley, UK, 2001, ISBN: 3-906454-27-4.
[30] D. Ruta, B. Gabrys, Application of the evolutionary algorithms for classifier selection in multiple classifier systems with majority voting, in: Proceedings of the 2nd International Workshop on Multiple Classifier Systems, Cambridge, UK, Lecture Notes in Computer Science, LNCS 2096, Springer-Verlag, 2001, pp. 399–408.
[31] D. Ruta, B. Gabrys, New measure of classifier dependency in multiple classifier systems, in: F. Roll, J. Kittler (Eds.), Proceedings of the 3rd International Workshop on Multiple Classifier Systems, Cagliari, Italy, Lecture Notes in Computer Science, LNCS 2364, Springer-Verlag, 2002, pp. 127–136.
[32] D. Ruta, B. Gabrys, Set analysis of coincident errors and its applications for combining classifiers, in: Pattern Recognition and String Matching, Combinatorial Optimisation, vol. 13, Kluwer Academic Publishers, 2002, ISBN: 1-4020-0953-4.
[33] D. Ruta, B. Gabrys, A theoretical analysis of the, limits of majority voting errors for multiple classifier systems, Pattern Analysis and Applications 5 (4) (2002) 333–350.
[34] R.E. Schapire, Y. Freud, P. Bartlett, W.S. Lee, Boosting the margin: a new explanation for the effectiveness of voting methods, The Annals of Statistics 26 (5) (1998) 1651–1686.
[35] A.J.C. Sharkey, N.E. Sharkey, Combining diverse neural nets, The Knowledge Engineering Review 12 (3) (1997) 231–247.
[36] A.J.C. Sharkey, N.E. Sharkey, U. Gerecke, G.O. Ghandroth, The 'test and select' approach to ensemble combination, in: J. Kittler, F. Roli (Eds.), Proceedings of the 1st International Workshop on Multiple Classifier Systems, Cagliari, Italy, Lecture Notes in Computer Science, LNCS 1857, Springer-Verlag, 2000, pp. 30–44.
[37] C.A. Shipp, L.I. Kuncheva, Relationship between combination methods and measures of diversity in combining classifiers, Information Fusion 3 (2) (2002) 135–148.
[38] D.B. Skalak, The sources of increased accuracy for two proposed boosting algorithms, in: Proceedings of the AAAI'96, Integrating Multiple Learned Models Workshop, Portland, OR, 1996.
[39] K. Woods, W.P. Kegelmeyer, K. Bowyer, Combination of multiple classifiers using local accuracy estimates, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (4) (1997) 405–410.
[40] G. Zeuobi, P. Cunningham, Using diversity in preparing ensembles of classifiers based on different feature subsets to minimise generalisation error, in: Proceedings of the 12th European Conference on Machine Learning, 2001, pp. 576–587.