3

Feature Selection for Ensembles Using the Multi-Objective Optimization Approach

Luiz S. Oliveira[1], Marisa Morita[2], and Robert Sabourin[3]

[1] Pontifical Catholic University of Paraná, Curitiba, PR, Brazil
soares@ppgia.pucpr.br
[2] HSBC Bank Brazil, Curitiba, PR, Brazil
marisa.e.morita@hsbc.com.br
[3] École de Technologie Supérieure, Montreal, Canada
robert.sabourin@etsmtl.ca

Summary. Feature selection for ensembles has shown to be an effective strategy for ensemble creation due to its ability of producing good subsets of features, which make the classifiers of the ensemble disagree on difficult cases. In this paper we present an ensemble feature selection approach based on a hierarchical multi-objective genetic algorithm. The underpinning paradigm is the "overproduce and choose". The algorithm operates in two levels. Firstly, it performs feature selection in order to generate a set of classifiers and then it chooses the best team of classifiers. In order to show its robustness, the method is evaluated in two different contexts: supervised and unsupervised feature selection. In the former, we have considered the problem of handwritten digit recognition and used three different feature sets and multi-layer perceptron neural networks as classifiers. In the latter, we took into account the problem of handwritten month word recognition and used three different feature sets and hidden Markov models as classifiers. Experiments and comparisons with classical methods, such as Bagging and Boosting, demonstrated that the proposed methodology brings compelling improvements when classifiers have to work with very low error rates.

3.1 Introduction

Ensemble of classifiers has been widely used to reduce model uncertainty and improve generalization performance. Developing techniques for generating candidate ensemble members is a very important direction of ensemble of classifiers research. It has been demonstrated that a good ensemble is one where the individual classifiers in the ensemble are both accurate and make their errors on different parts of the input space (there is no gain in combining identical classifiers) [11, 17, 31]. In other words, an ideal ensemble consists of good classifiers (not necessarily excellent) that disagree as much as possible on difficult cases.

L.S. Oliveira et al.: *Feature Selection for Ensembles Using the Multi-Objective Optimization Approach*, Studies in Computational Intelligence (SCI) **16**, 49–74 (2006)
www.springerlink.com

The literature has shown that varying the feature subsets used by each member of the ensemble should help to promote this necessary diversity [12, 31, 24, 35]. Traditional feature selection algorithms aim at finding the best trade-off between features and generalization. On the other hand, ensemble feature selection has the additional goal of finding a set of feature sets that will promote disagreement among the component members of the ensemble. The Random Subspace Method (RMS) proposed by Ho in [12] was one early algorithm that constructs an ensemble by varying the subset of features. More recently some strategies based on genetic algorithms (GAs) have been proposed [31]. All these strategies claim better results than those produced by traditional methods for creating ensembles such as Bagging and Boosting. In spite of the good results brought by GA-based methods, they still can be improved in some aspects, e.g., avoiding classical methods such as the weighted sum to combine multiple objective functions. It is well known that when dealing with this kind of combination, one should deal with problems such as scaling and sensitivity towards the weights.

It has been demonstrated that feature selection through multi-objective genetic algorithm (MOGA) is a very powerful tool for finding a set of good classifiers, since GA is quite effective in rapid global search of large, non-linear and poorly understood spaces [30]. Besides, it can overcome problems such as scaling and sensitivity towards the weights. Kudo and Sklansky [18] have compared several algorithms for feature selection and concluded that GAs are suitable when dealing with large-scale feature selection (number of features is over 50). This is the case of most of the problems in handwriting recognition, which is the test problem in this work.

In this light, we propose an ensemble feature selection approach based on a hierarchical MOGA. The underlying paradigm is the "overproduce and choose" [32, 9]. The algorithm operates in two levels. The former is devoted to the generation of a set of good classifiers by minimizing two criteria: error rate and number of features. The latter combines these classifiers in order to find an ensemble by maximizing the following two criteria: accuracy of the ensemble and a measure of diversity.

Recently, the issue of using diversity to build ensemble of classifiers has been widely discussed. Several works have demonstrated that there is a weak correlation between diversity and ensemble performance [23]. In light of this, some authors have claimed that diversity brings no benefits in building ensemble of classifiers [33], on the other hand, others suggest that the study of diversity in classifier combination might be one of the lines for further exploration [19].

In spite of the weak correlation between diversity and performance, we argue that diversity might be useful to build ensembles of classifiers. We demonstrated through experimentation that using diversity jointly with performance to guide selection can avoid overfitting during the search. In order to show robustness of the proposed methodology, it was evaluated in two different contexts: supervised and unsupervised feature selection. In the former, we

have considered the problem of handwritten digit recognition and used three different feature sets and multi-layer perceptron (MLP) neural networks as classifiers. In such a case, the classification accuracy is supplied by the MLPs in conjunction with the sensitivity analysis. This approach makes it feasible to deal with huge databases in order to better represent the pattern recognition problem during the fitness evaluation. In the latter, we took into account the problem of handwritten month word recognition and used three different feature sets and hidden Markov models (HMM) as classifiers. We demonstrate that it is feasible to find compact clusters and complementary high-level representations (codebooks) in subspaces without using the recognition results of the system. Experiments and comparisons with classical methods, such as Bagging and Boosting, demonstrated that the proposed methodology brings compelling improvements when classifiers have to work with very low error rates.

The remainder of this paper is organized as follows. Section 3.2 presents a brief review about the methods for ensemble creation. Section 3.3 provides a overview of the strategy. Section 3.4 introduces briefly the the multi-objective genetic algorithm we are using in this work. Section 3.5 describes the classifiers and feature sets for both supervised and unsupervised contexts. Section 3.6 introduces how we have implemented both levels of the proposed methodology and Section 3.7 reports the experimental results. Finally, Section 3.8 discusses the reported results and Section 3.9 concludes the paper.

3.2 Related Works

Assuming the architecture of the ensemble as the main criterion, we can distinguish among serial, parallel, and hierarchical schemes, and if the classifiers of the ensemble are selected or not by the ensemble algorithm we can divide them into selection-oriented and combiner-oriented methods [20]. Here we are more interested in the first class, which try to improve the overall accuracy of the ensemble by directly boosting the accuracy and the diversity of the experts of the ensemble. Basically, they can be divided into resampling methods and feature selection methods.

Resampling techniques can be used to generate different hypotheses. For instance, bootstrapping techniques [6] may be used to generate different training sets and a learning algorithm can be applied to the obtained subsets of data in order to produce multiple hypotheses. These techniques are effective especially with unstable learning algorithms, which are algorithms very sensitive to small changes in the training data. In bagging [1] the ensemble is formed by making bootstrap replicates of the training sets, and then multiple generated hypotheses are used to get an aggregated predictor. The aggregation can be performed by averaging the outputs in regression or by majority or weighted voting in classification problems.

While in bagging the samples are drawn with replacement using a uniform probability distribution, in boosting methods [7] the learning algorithm is called at each iteration using a different distribution or weighting over the training examples. This technique places the highest weight on the examples most often misclassified by the previous base learner: in this manner the classifiers of the ensemble focus their attention on the hardest examples. Then the boosting algorithm combines the base rules taking a weighted majority vote of the base rules.

The second class of methods regards those strategies based on feature selection. The concept behind these approaches consists in reducing the number of input features of the classifiers, a simple method to fight the effects of the classical curse of dimensionality problem. For instance, the random subspace method [12, 35] relies on a pseudorandom procedure to select a small number of dimensions from a given feature space. In each pass, such a selection is made and a subspace is fixed. All samples are projected to this subspace, and a classifier is constructed using the projected training samples. In the classification a sample of an unknown class is projected to the same subspace and classified using the corresponding classifier. In the same vein of the random subspace method lies the input decimation method [37], which reduces the correlation among the errors of the base classifiers, by decoupling the classifiers by training them with different subsets of the input features. It differs from the random subspace as for each class the correlation between each feature and the output of the class is explicitly computed, and the classifier is trained only on the most correlated subset of features.

Recently, several authors have been investigated GA to design ensemble of classifiers. Kuncheva and Jain [21] suggest two simple ways to use genetic algorithm to design an ensemble of classifiers. They present two versions of their algorithm. The former uses just disjoint feature subsets while the latter considers (possibly) overlapping feature subsets. The fitness function employed is the accuracy of the ensemble, however, no measure of diversity is considered. A more elaborate method, also based on GA, was proposed by Optiz [31]. In his work, he stresses the importance of a diversity measure by including it in the fitness calculation. The drawback of this method is that the objective functions are combined through the weighted sum. It is well known that when dealing with this kind of combination, one should deal with problems such as scaling and sensitivity towards the weights. More recently Gunter and Bunke [10] have applied feature selection in conjunction with floating search to create ensembles of classifiers for the field of handwriting recognition. They used handwritten words and HMMs as classifiers to evaluate their algorithm. The feature set was composed of nine discrete features, which makes simpler the feature selection process. A drawback of this method is that one must set a priori the number of classifiers in the ensemble.

3.3 Methodology Overview

In this section we outline the hierarchical approach proposed. As stated before, it is based on an "overproduce and choose" paradigm where the first level generates several classifiers by conducting feature selection and the second one chooses the best ensemble among such classifiers. Figure 3.1 depicts the proposed methodology. Firstly, we carry out feature selection by using a MOGA. It gets as inputs a trained classifier and its respective data set. Since the algorithm aims at minimizing two criteria during the search[1], it will produce at the end a 2-dimensional Pareto-optimal front, which contains a set of classifiers (trade-offs between the criteria being optimized). The final step of this first level consists in training such classifiers.

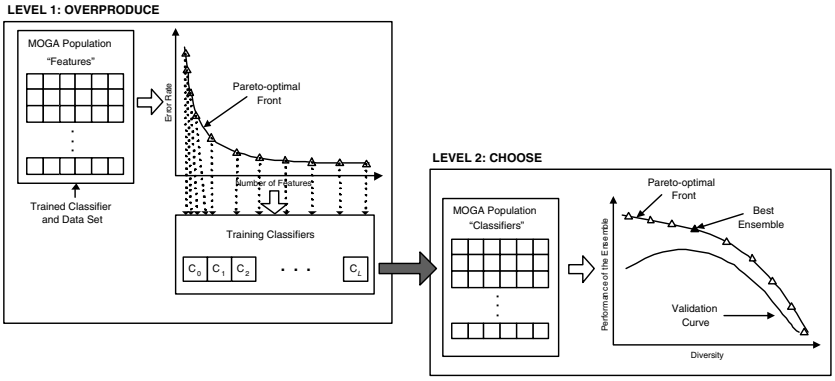


Fig. 3.1. An overview of the proposed methodology.

Once the set of classifiers have been trained, the second level is suggested to pick the members of the team which are most diverse and accurate. Let $A = C_1, C_2, \ldots, C_L$ be a set of L classifiers extracted from the Pareto-optimal and B a chromosome of size L of the population. The relationship between A and B is straightforward, i.e., the gene i of the chromosome B is represented by the classifier C_i from A. Thus, if a chromosome has all bits selected, all classifiers of A will be included in the ensemble. Therefore, the algorithm will produce a 2-dimensional Pareto-optimal front which is composed of several ensembles (trade-offs between accuracy and diversity). In order to choose the best one, we use a validation set, which points out the most diverse and accurate team among all. Later in this paper, we will discuss the issue of using diversity to choose the best ensemble.

[1] Error rate and number of features in the case of supervised feature selection and a clustering index and the number of features in the case of unsupervised feature selection.

In both cases, MOGAs are based on bit representation, one-point crossover, and bit-flip mutation. In our experiments, MOGA used is a modified version of the Non-dominated Sorting Genetic Algorithm (NSGA) [4] with elitism.

3.4 Multi-Objective Genetic Algorithm

Since the concept of multi-objective genetic algorithm (MOGA) will be explored in the remaining of this paper, this section briefly introduces it.

A general multi-objective optimization problem consists of a number of objectives and is associated with a number of inequality and equality constraints. Solutions to a multi-objective optimization problem can be expressed mathematically in terms of nondominated points, i.e., a solution is dominant over another only if it has superior performance in all criteria. A solution is said to be Pareto-optimal if it cannot be dominated by any other solution available in the search space. In our experiments, the algorithm adopted is the Non-dominated Sorting Genetic Algorithm (NSGA) with elitism proposed by Srinivas and Deb in [4, 34].

The idea behind NSGA is that a ranking selection method is applied to emphasize good points and a niche method is used to maintain stable subpopulations of good points. It varies from simple GA only in the way the selection operator works. The crossover and mutation remain as usual. Before the selection is performed, the population is ranked on the basis of an individual's nondomination. The nondominated individuals present in the population are first identified from the current population. Then, all these individuals are assumed to constitute the first nondominated front in the population and assigned a large dummy fitness value. The same fitness value is assigned to give an equal reproductive potential to all these nondominated individuals. In order to maintain the diversity in the population, these classified individuals are made to share their dummy fitness values. Sharing is achieved by performing selection operation using degraded fitness values obtained by dividing the original fitness value of an individual by a quantity proportional to the number of individuals around it. After sharing, these nondominated individuals are ignored temporarily to process the rest of population in the same way to identify individuals for the second nondominated front. These new set of points are then assigned a new dummy fitness value which is kept smaller than the minimum shared dummy fitness of the previous front. This process is continued until the entire population is classified into several fronts.

Thereafter, the population is reproduced according to the dummy fitness values. A stochastic remainder proportionate selection is adopted here. Since individuals in the first front have the maximum fitness value, they get more copies than the rest of the population. The efficiency of NSGA lies in the way multiple objectives are reduced to a dummy fitness function using nondominated sorting procedures. More details about NSGA can be found in [4].

3.5 Classifiers and Feature Sets

As stated before, we have carried out experiments in both supervised and unsupervised contexts. The remaining of this section describes the feature sets and classifiers we have used.

3.5.1 Supervised Context

To evaluate the proposed methodology in the supervised context, we have used three base classifiers trained to recognize handwritten digits of NIST SD19. Such classifiers were trained with three well-known feature sets: Concavities and Contour (CCsc), Distances (DDDsc), and Edge Maps (EMsc) [29].

All classifiers here are MLPs trained with the gradient descent applied to a sum-of-squares error function. The transfer function employed is the familiar sigmoid function. In order to monitor the generalization performance during learning and terminate the algorithm when there is no longer an improvement, we have used the method of cross-validation. Such a method takes into account a validation set, which is not used for learning, to measure the generalization performance of the network. During learning, the performance of the network on the training set will continue to improve, but its performance on the validation set will only improve to a point, where the network starts to overfit the training set, that the learning algorithm is terminated. All networks have one hidden layer where the units of input and output are fully connected with units of the hidden layer, where the number of hidden units were determined empirically (see Table 3.1). The learning rate and the momentum term were set at high values in the beginning to make the weights quickly fit the long ravines in the weight space, then these parameters were reduced several times according to the number of iterations to make the weights fit the sharp curvatures.

Among the different strategies of rejection we have tested, the one proposed by Fumera et al [8] provided the better error-reject trade-off for our experiments. Basically, this technique suggests the use of multiple reject thresholds for the different data classes $(T_0, \ldots, T_n)$ to obtain the optimal decision and reject regions. In order to define such thresholds we have developed an iterative algorithm, which takes into account a decreasing function of the threshold variables $R(T_0, \ldots, T_n)$ and a fixed error rate T_{error}. We start from all threshold values equal to 1, i.e., the error rate equal to 0 since all images are rejected. Then, at each step, the algorithm decreases the value of one of the thresholds in order to increase the accuracy until the error rate exceeds T_{error}.

The training (TRDBsc) and validation (VLDB1sc) sets are composed of 195,000 and 28,000 samples from hsf_0123 series respectively while the test set (TSDBsc) is composed of 30,089 samples from the hsf_7. We consider also a second validation set (VLDB2sc), which is composed of 30,000 samples of hsf_7. This data is used to select the best ensemble of classifiers. Figure 3.2 shows the performance on the test set of all classifiers for error rates varying

Table 3.1. Description and performance of the classifiers on TSDB (zero-rejection level).

Feature Set	Number. of Features	Units in the Hidden Layer	Rec. Rate (%)
CC*sc*	132	80	99.13
DDD*sc*	96	60	98.17
EM*sc*	125	70	97.04

from 0.10 to 0.50%, while Table 3.1 reports the performance of all classifiers at zero-rejection level. The curve depicted in Figure 3.2 is much more meaningful when dealing with real applications since they describe the recognition rate in relation to a specific error rate, including implicitly a corresponding reject rate. This rate also allows us to compute the reliability of the system for a given error rate. It can be done by using Equation 3.1.

$$\text{Reliability} = \frac{\text{Rec.Rate}}{\text{Rec.Rate} \quad + \quad \text{Error Rate}} \times 100 \tag{3.1}$$

Figure 3.2 corroborates that recognition of handwritten digits is still an open problem when very low error rates are required. Consider for example our best classifier, which reaches 99.13% at zero-rejection level on the test set. If we allow an error rate of 0.1%, i.e., just one error in 1,000, the recognition rate of such classifier drops from 99.13% to 91.83%. This means that we have to reject 8.07% to get 0.1% of error (Figure 3.2). We will demonstrate that the ensemble of classifiers can significantly improve the performance of the classifiers for low error rates.

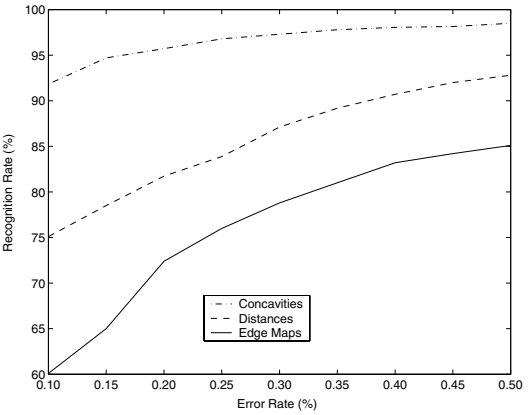


Fig. 3.2. Performance of the classifiers on the test set for error rates varying from 0.10 to 0.50%.

3.5.2 Unsupervised Context

To evaluate the proposed methodology in unsupervised context we have used three HMM-based classifiers trained to recognize handwritten Brazilian month words ("Janeiro", "Fevereiro", "Março", "Abril", "Maio", "Junho", "Julho", "Agosto", "Setembro", "Outubro", "Novembro", "Dezembro"). The training (TRDB*uc*), validation (VLDB1*uc*), and testing (TSDB*uc*) sets are composed of 1,200, 400, and 400 samples, respectively. In order to increase the training and validation sets, we have also considered 8,300 and 1,900 word images, respectively, extracted from the legal amount database. This is possible because we are considering character models. We consider also a second validation set (VLDB2*uc*) of 500 handwritten Brazilian month words [14]. Such data is used to select the best ensemble of classifiers.

Given a discrete HMM-based approach, each word image is transformed as a whole into a sequence of observations by the successive application of preprocessing, segmentation, and feature extraction. Preprocessing consists of correcting the average character slant. The segmentation algorithm uses the upper contour minima and some heuristics to split the date image into a sequence of segments (graphemes), each of which consists of a correctly segmented, an under-segmented, or an over-segmented character. A detailed description of the preprocessing and segmentation stages is given in [28].

The word models are formed by the concatenation of appropriate elementary HMMs, which are built at letter and space levels. The topology of space model consists of two states linked by two transitions that encode a space or no space. Two topologies of letter models were chosen based on the output of our grapheme-based segmentation algorithm which may produce a correct segmentation of a letter, a letter under-segmentation or a letter over-segmentation into two, three, or four graphemes depending on each letter. In order to cope with these configurations of segmentations, we have designed topologies with three different paths leading from the initial state to the final state.

Considering uppercase and lowercase letters, we need 42 models since the legal amount alphabet is reduced to 21 letter classes and we are not considering the unused ones. Thus, regarding the two topologies, we have 84 HMMs which are trained using the Baum-Welch algorithm with the Cross-Validation procedure.

Since no information on recognition is available on the writing style (uppercase, lowercase), the word model consists of two letter HMMs in parallel and four space HMMs linked by four transitions: two uppercase-letters (UU), two lowercase-letters (LL), one uppercase letter followed by one lowercase-letter (UL), and one lowercase letter followed by one uppercase-letter (LU). The probabilities of these transitions are estimated by their frequency of occurrence in the training set. In the same manner, the probabilities of beginning a word by an uppercase-letter (0U) or a lowercase letter (0L) are also estimated in the training set. This architecture handles the problem related to

the mixed handwritten words detecting implicitly the writing style during recognition using the Backtracking of the Viterbi algorithm.

The feature set that feeds the first classifier is a mixture of concavity and contour features (CCuc) [29]. In this case, each grapheme is divided into two equal zones (horizontal) where for each region a concavity and contour feature vector of 17 components is extracted. Therefore, the final feature vector has 34 components. The other two classifiers make use of a feature set based on distances [28]. The former uses the same zoning discussed before (two equal zones), but in this case, for each region a vector of 16 components is extracted. This leads to a final feature vector of 32 components (DDD32$_u c$). For the latter we have tried a different zoning. The grapheme is divided into four zones using the reference baselines, hence, we have a final feature vector composed of 64 components (DDD64$_u c$). Table 3.2 reports the performance of all classifiers on the test set at zero-rejection level. Figure 3.3 shows the performance of all classifiers for error rates varying from 1% to 4%. The strategy for rejection used in this case is the one discussed previously. We have chosen higher error rates in this case due to the size of the database we are dealing with.

Table 3.2. Performance of the classifiers on the test set.

Feature Set	Number of Features	Codebook Size	Rec Rate (%)
CCuc	34	80	86.1
DDD32uc	32	40	73.0
DDD64uc	64	60	64.5

It can be observed from Table 3.3 that the recognition rates with error fixed at 1% are very poor, hence, the number of rejected patterns is very high. We will see in the next sections that the proposed methodology can improve these results considerably.

3.6 Implementation

This section introduces how we have implemented both levels of the proposed methodology. First we discuss the supervised context and then the unsupervised.

3.6.1 Supervised Context

Supervised Feature Subset Selection

The feature selection algorithm used in here was introduced in [30]. To make this paper self-contained, a brief description is included in this section.

Regarding feature selection algorithms, they can be classified into two categories based on whether or not feature selection is performed independently of the learning algorithm used to construct the classifier. If feature selection is done independently of the learning algorithm, the technique is said to follow a filter approach. Otherwise, it is said to follow a wrapper approach [13]. While the filter approach is generally computationally more efficient than the wrapper approach, its major drawback is that an optimal selection of features may not be independent of the inductive and representational biases of the learning algorithm that is used to construct the classifier. On the other hand, the wrapper approach involves the computational overhead of evaluating candidate feature subsets by executing a given learning algorithm on the database using each feature subset under consideration.

As stated elsewhere, the idea of using feature selection is to promote diversity among the classifiers. To tackle such a task we have to optimize two objective functions: minimization of the number of features and minimization of the error rate of the classifier. Computing the first one is simple, i.e., the number of selected features. The problem lies in computing the second one, i.e., the error rate supplied by the classifier. Regarding a wrapper approach, in each generation, evaluation of a chromosome (a feature subset) requires training the corresponding neural network and computing its accuracy. This evaluation has to be performed for each of the chromosomes in the population. Since such a strategy is not feasible due to the limits imposed by the learning time of the huge training set considered in this work, we have adopted the strategy proposed by Moody and Utans in [26], who use the sensitivity of the network to estimate the relationship between the input features and the network performance.

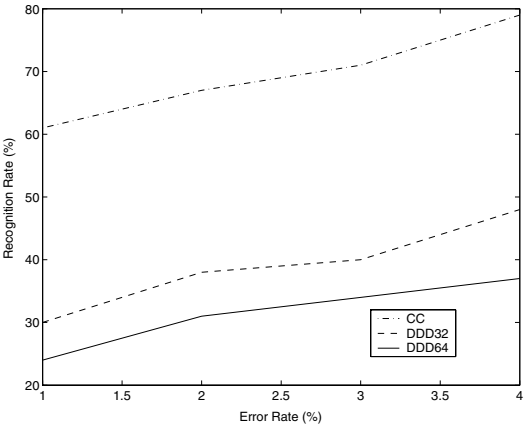


Fig. 3.3. Performance of the classifiers on the test set for error rates varying from 1 to 4%.

The sensitivity of the network model to variable β is defined as:

$$S_\beta = \frac{1}{N}\sum_{j=1}^{N} ASE(\bar{x}_\beta) - ASE(x_\beta) \tag{3.2}$$

with

$$\bar{x}_\beta = \frac{1}{N}\sum_{j=1}^{N} x_{\beta_j} \tag{3.3}$$

where x_{β_j} is the β^{th} input variable of the j^{th} exemplar. S_β measures the effect on the training ASE (average square error) of replacing the β^{th} input x_β by its average $\bar{x}_\beta$. Moody and Utans show that when variables with small sensitivity values with respect to the network outputs are removed, they do not influence the final classification. So, in order to evaluate a given feature subset we replace the unselected features by their averages. In this way, we avoid training the neural network and hence turn the wrapper approach feasible for our problem. We call this strategy modified-wrapper. Such a scheme has been employed also by Yuan et al in [38], and it makes it feasible to deal with huge databases in order to better represent the pattern recognition problem during the fitness evaluation[2]. Moreover it can accommodate multiple criteria such as the number of features and the accuracy of the classifier, and generate the Pareto-optimal front in the first run of the algorithm. Figure 3.4 shows the evolution of the population in the objective plane and its respective Pareto-optimal front.

It can be observed in Figure 3.4b that the Pareto-optimal front is composed of several different classifiers. In order to get a better insight about them, they were classified into 3 different groups: weak, medium, and strong. It can be observed that among all those classifiers there are very good ones. To find out which classifiers of the Pareto-optimal front compose the best ensemble, we carried out a second level of search. Once we did not train the models during the search (the training step is replaced by the sensitivity analysis), the final step of feature selection consists of training the solutions provided by the Pareto-optimal front (3.1).

Choosing the Best Ensemble

As defined in Section 3.3 each gene of the chromosome is represented by a classifier produced in the previous level. Therefore, if a chromosome has all bits selected, all classifiers of will compose the team. In order to find the best ensemble of classifiers, i.e., the most diverse set of classifiers that brings a good generalization, we have used two objective functions during this level

[2] If small databases are considered, then a full-wrapper could replace the proposed modified-wrapper.

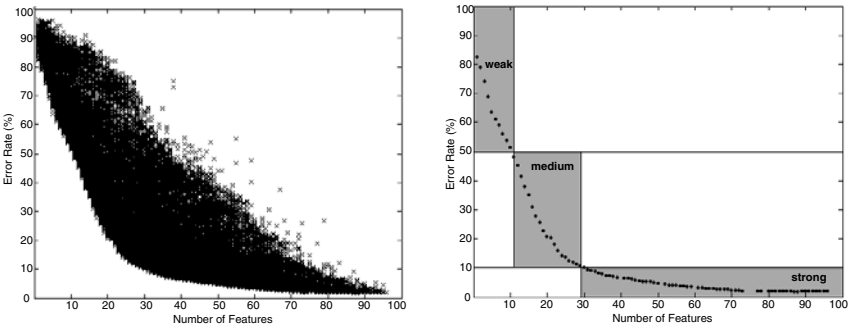


Fig. 3.4. Supervised feature selection using a Pareto-based approach (a) Evolution of the population in the objective plane, (b) Pareto-optimal front and its different classes of classifiers.

of the search, namely, maximization of the recognition rate of the ensemble and maximization of a measure of diversity. We have tried different measures such as overlap, entropy [22], and ambiguity [17]. The results achieved with ambiguity and entropy were very similar. In this work we have used ambiguity as diversity measure. The ambiguity is defined as follows:

$$a_i(x_k) = [V_i(x_k) - \overline{V}(x_k)]^2 \tag{3.4}$$

where a_i is the ambiguity of the i^{th} classifier on the example x_k, randomly drawn from an unknown distribution, while V_i and $\overline{V}$ are the i^{th} classifier and the ensemble predictions, respectively. In other words, it is simply the variance of ensemble around the mean, and it measures the disagreement among the classifiers on input x. Thus the contribution to diversity of an ensemble member i as measured on a set of M samples is:

$$A_i = \frac{1}{M} \sum_{k=1}^{M} a_i(x_k) \tag{3.5}$$

and the ambiguity of the ensemble is

$$\overline{A} = \frac{1}{N} \sum A_i \tag{3.6}$$

where N is the number of classifiers. So, if the classifiers implement the same functions, the ambiguity $\overline{A}$ will be low, otherwise it will be high. In this scenario the error from the ensemble is

$$E = \overline{E} - \overline{A} \tag{3.7}$$

where $\overline{E}$ is the average errors of the single classifiers and $\overline{A}$ is the ambiguity of the ensemble. Equation 3.7 expresses the trade-off between bias and variance in the ensemble, but in a different way than the common bias-variance relation in which the averages are over possible training sets instead of ensemble averages. If the ensemble is strongly biased the ambiguity will be small, because the classifiers implement very similar functions and thus agree in inputs even outside the training set [17].

At this level of the strategy we want to maximize the generalization of the ensemble, therefore, it will be necessary to use a way of combining the outputs of all classifiers to get a final decision. To do this, we have used the average, which is a simple and effective scheme of combining predictions of the neural networks [16]. Other combination rules such as product, min, and max have been tested but the simple average has produced slightly better results. In order to evaluate the objective functions during the search described above we have used the validation set VLDB1*sc*.

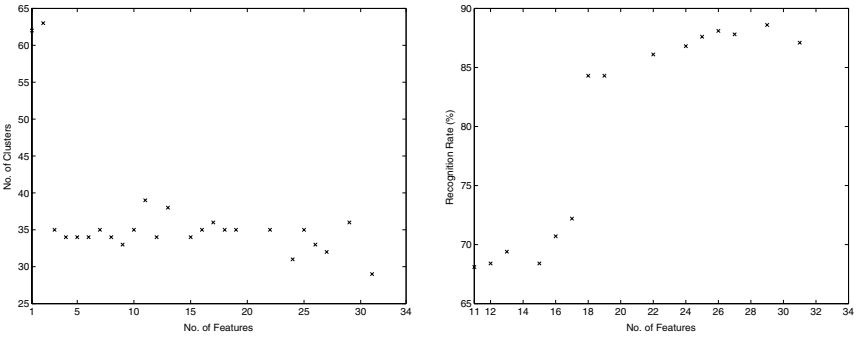
3.6.2 Unsupervised Context

Unsupervised Feature Subset Selection

A lot of work done in the field of handwritten word recognition takes into account discrete HMMs as classifiers, which have to be fed with a sequence of discrete values (symbols). This means that before using a continuous feature vector, we must convert it to discrete values. A common way to do that is through clustering. The problem is that for the most of real-life situations we do not know the best number of clusters, what makes it necessary to explore different numbers of clusters using traditional clustering methods such as the K-means algorithm and its variants. In this light, clustering can become a trial-and-error work. Besides, its result may not be very promising especially when the number of clusters is large and not easy to estimate.

Unsupervised feature selection emerges as a clever solution to this problem. The literature contains several studies on feature selection for supervised learning, but only recently, the feature selection for unsupervised learning has been investigated [5, 15]. The objective in unsupervised feature selection is to search for a subset of features that best uncovers "natural" groupings (clusters) from data according to some criterion. In this way, we can avoid the manual process of clustering and find the most discriminative features in the same time. Hence, we will have at the end a more compact and robust high-level representation (symbols).

In the above context, unsupervised feature selection also presents a multi-criterion optimization function, where the objective is to find compact and well separated hyper-spherical clusters in the feature subspaces. Differently of the supervised feature selection, here the criteria optimized by the algorithm are a validity index and the number of features. [27].



(a) (b)

Fig. 3.5. (a) Relationship between the number of clusters and the number of features and (b) Relationship between the recognition rate and the number of features.

In order to measure the quality of clusters during the clustering process, we have used the Davies-Bouldin (DB)-index [3] over 80,000 feature vectors extracted from the training set of 9,500 words. To make such an index suitable for our problem, it must be normalized by the number of selected features. This is due to the fact that it is based on geometric distance metrics and therefore, it is not directly applicable here because it is biased by the dimensionality of the space, which is variable in feature selection problems.

We have noticed that the value of DB index decreases as the number of features increases. We have correlated this effect with the normalization of DB-index by the number of features. In order to compensate this, we have considered as second objective the minimization of the number of features. In this case, one feature must be set at least. Figure 3.5 depicts the relationship between the number of clusters and number of features and the relationship between the recognition rate on the validation set and the number of features.

Like in the supervised context, here we also divided the classifiers of the Pareto into classes. In this case, we have realized that those classifiers with very few features are not selected to compose the ensemble, and therefore, just the classifiers with more than 10 features were used into the second level of search. In Section 3.7.2 we discuss this issue in more detail. The way of choosing the best ensemble is exactly the same as introduced in Section 3.6.1.

3.7 Experimental Results

All experiments in this work were based on a single-population master-slave MOGA. In this strategy, one master node executes the genetic operators (selection, crossover and mutation), and the evaluation of fitness is distributed among several slave processors. We have used a Beowulf cluster with 17 (one

master and 16 slaves) PCs (1.1Ghz CPU, 512Mb RAM) to execute our experiments.

The following parameter settings were employed in both levels: population size = 128, number of generations = 1000, probability of crossover = 0.8, probability of mutation = $1/L$ (where L is the length of the chromosome), and niche distance (σ_{share}) = [0.25,0.45]. The length of the chromosome in the first level is the number of components in the feature set (see Table 3.1), while in the second level is the number of classifiers picked from the Pareto-optimal front in the previous level.

In order to define the probabilities of crossover and mutation, we have used the one-max problem, which is probably the most frequently-used test function in research on genetic algorithms because of its simplicity [2]. This function measures the fitness of an individual as the number of bits set to one on the chromosome. We have used a standard genetic algorithm with a single-point crossover and the maximum generations of 1000. The fixed crossover and mutation rates are used in a run, and the combination of the crossover rates 0.0, 0.4, 0.6, 0.8 and 1.0 and the mutation rates of $0.1/L$, $1/L$ and $10/L$, where L is the length of the chromosome. The best results were achieved with $P_c = 0.8$ and $P_m = 1/L$. Such results confirmed the values reported by Miki et al in [25]. The parameter σ_{share} was tuned empirically.

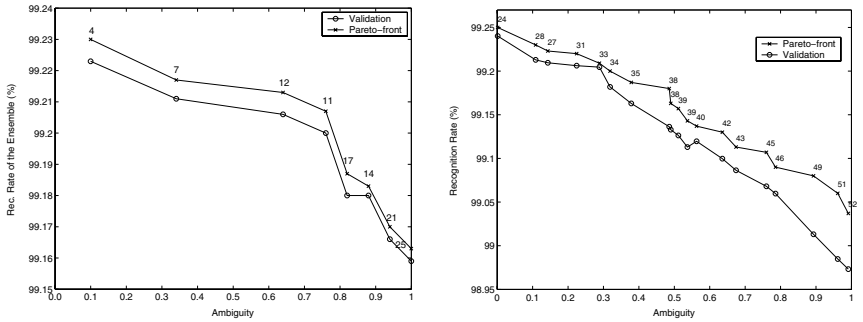
3.7.1 Experiments in the Supervised Context

Once all parameters have been defined, the first step, as described in Section 3.6.1, consists of performing feature selection for a given feature set. As depicted in Figure 3.4, this procedure produces quite a large number of classifiers, which should be trained for use in the second level. After some experiments, we found out that the second level always chooses "strong" classifiers to compose the ensemble. Thus, in order to speed up the training process and the second level of search as well, we decide to train and use in the second level just the "strong" classifiers. This decision was made after we realized that in our experiments the "weak" and "medium" classifiers did not cooperate with the ensemble at all. To train such classifiers, the same databases reported in Section 3.5.1 were used. Table 3.3 summarizes the "strong" classifiers produced by the first level for the three feature sets we have considered.

Table 3.3. Summary of the classifiers produced by the first level.

Feature Set	No. of Classifiers	Range of Features	Range of Rec. Rates (%)
CC*sc*	81	24-125	90.5 - 99.1
DDD*sc*	54	30-84	90.6 - 98.1
EM*sc*	78	35-113	90.5 - 97.0

Considering for example the feature set CC_{sc}, the first level of the algorithm provided 81 "strong" classifiers which have the number of features ranging from 24 to 125 and recognition rates ranging from 90.5% to 99.1% on $TSDB_{sc}$. This shows the great diversity of the classifiers produced by the feature selection method. Based on the classifiers reported in Table 3.3 we define four sets of base classifiers as follows: $S_1 = \{CCsc_0, \ldots, CCsc_{80}\}$, $S_2 = \{DDDsc_0, \ldots, DDDsc_{53}\}$, $S_3 = \{EMsc_0, \ldots, EMsc_{77}\}$, and $S_4 = \{S_1 \bigcup S_2 \bigcup S_3\}$. All these sets could be seen as ensembles, but in this work we reserve the word ensemble to characterize the results yielded by the second-level of the algorithm. In order to assess the objective functions of the second-level of the algorithm (generalization of the ensemble and diversity) we have used the validation set (VLDB1sc).



(a) (b)

Fig. 3.6. The Pareto-optimal front produced by the second-level MOGA: (a) S_1 and (b) S_4

Like the first level, the second one also generates a set of possible solutions which are the trade-offs between the generalization of the ensemble and its diversity. Thus the problem now lies in choosing the most accurate ensemble among all. Due to the limited space we have, Figure 3.6 only depicts the variety of ensembles yielded by the second-level of the algorithm for S_1 and S_4. The number over each point stands for the number of classifiers in the ensemble. In order to decide which ensemble to choose we validate the Pareto-optimal front using VLDB2sc, which was not used so far. Since we are aiming at performance, the direct choice will be the ensemble that provides better generalization on $VLDB2_{sc}$. Table 3.4 summarizes the best ensembles produced for the four sets of base classifiers and their performance at zero-rejection level on the test set. For facility, we reproduce in this table the results of the original classifiers.

We can notice from Table 3.4 that the ensembles and base classifiers have very similar performance at zero-rejection level. On the other hand, Figure

Table 3.4. Performance of the ensembles on the test set.

Feature Set	Number of Classifiers	Rec. Rate (%) zero-rejection level	Rec. Rate (%) Original Classifiers
S_1	4	99.22	99.13
S_2	4	98.18	98.17
S_3	7	97.10	97.04
S_4	24	99.25	

3.7 shows that the ensembles respond better for error rates fixed at very low levels than single classifiers. The most expressive result was achieved for the ensemble S_3, which attains a reasonable performance at zero-rejection level but performs very poorly at low error rates. In such a case, the ensemble of classifiers brought an improvement of about 8%. We have noticed that the ensemble reduces the high outputs of some outliers so that the threshold used for rejection can be reduced and consequently the number of samples rejected is reduced. Thus, aiming for a small error rate we have to consider the important role of the ensemble.

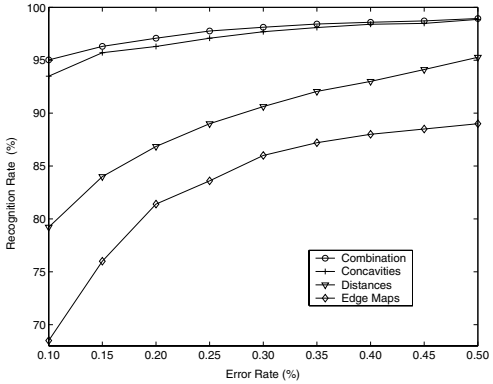


Fig. 3.7. Improvements yielded by the ensembles.

Regarding the ensemble S_4, we can notice that it achieves a performance similar to S_1 at zero-rejection level (see Table 3.4). Besides, it is composed of 24 classifiers, against four of S_1. The fact worths noting though, is the performance of S_4 at low error rates. For the error rate fixed at 1% it reached 95.0% against 93.5% of S_1. S_4 is composed of 14, 6, and 4 classifiers from S_1, S_2, and S_3, respectively. This emphasizes the ability of the algorithm in finding good ensembles when more original classifiers are available.

3.7.2 Experiments in the Unsupervised Context

The experiments in the unsupervised context follow the same vein of the supervised one. As discussed in Section 3.6.2, the main difference lies in the way the feature selection is carried out. In spite of that, we can observe that the number of classifiers produced during unsupervised feature selection is quite large as well. In light of this, we have applied the same strategy of dividing the classifiers into groups (see Figure 3.5). After some experiments, we found out that the second level always chooses "strong" classifiers to compose the ensemble. Thus, in order to speed up the training process and the second level of search as well, we decide to train and use in the second level just "strong" classifiers. To train such classifiers, the same databases reported in Section 3.5.2 were considered. Table 3.5 summarizes the "strong" classifiers (after training) produced by the first level for the three feature sets we have considered.

Table 3.5. Summary of the classifiers produced by the first level.

Feature Set	Number of Classifiers	Range of Features	Range of Codebook	Range of Rec. Rates (%)
CCuc	15	10-32	29-39	68.1 - 88.6
DDD32uc	21	10-31	20-30	71.7 - 78.0
DDD64uc	50	10-64	52-80	60.6 - 78.2

Considering for example the feature set CCuc, the first level of the algorithm provided 15 "strong" classifiers which have the number of features ranging from 10 to 32 and recognition rates ranging from 68.1% to 88.6% on VLDB1uc . This shows the great diversity of the classifiers produced by the feature selection method. Based on the classifiers reported in Table 3.5 we define four sets of base classifiers as follows: $F_1 = \{CCuc_0, \ldots, CCuc_{14}\}$, $F_2 = \{DDD32uc_0, \ldots, DDD32uc_{20}\}$, $F_3 = \{DDD64uc_0, \ldots, DDD64uc_{49}\}$, and $F_4 = \{F_1 \bigcup F_2 \bigcup F_3\}$.

Again, due to the limited space we have, Figure 3.8 only depicts the variety of ensembles yielded by the second-level of the algorithm for F_2 and F_4. The number over each point stands for the number of classifiers in the ensemble. Like in the previous experiments, the second validation set (VLDB2uc) was used to select the best ensemble. After selecting the best ensemble the final step is to assess them on the test set. Table 3.6 summarizes the performance of the ensembles on the test set. For the sake of comparison, we reproduce in Table 3.6 the results presented in Table 3.2.

Figure 3.8b shows the performance of the ensembles generated with all base classifiers available, i.e., Ensemble F_4. Like in the previous experiments (supervised context), the result achieved by the ensemble F_4 shows the ability of the algorithm in finding good ensembles when more base classifiers are

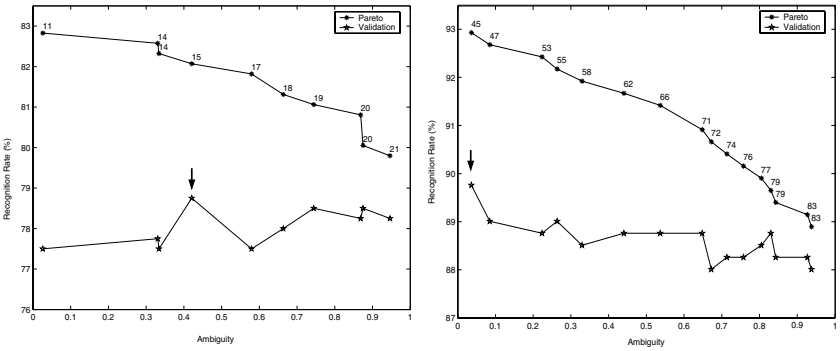


Fig. 3.8. The Pareto-optimal front (and validation curves where the best solutions are highlighted with an arrow) produced by the second-level MOGA: (a) F_2 and (b) F_4.

Table 3.6. Comparison between ensembles and original classifiers.

Base Classifiers	Number of Classifiers	Rec. Rate (%)	Original Feature Set	Rec. Rate (%)
F_1	10	89.2	CC	86.1
F_2	15	80.2	DDD32	73.0
F_3	36	80.7	DDD64	64.5
F_4	45	90.2		

considered. The ensemble F_4 is composed of 9, 11, and 25 classifiers from F_1, F_2, and F_3, respectively.

In light of this, we decided to introduce a new feature set, which, based on our experience, has a good discrimination power when combined with other features such as concavities. This feature set, which we call "global features", is composed of primitives such as ascenders, descenders, and loops. The combination of these primitives plus a primitive that determines whether a grapheme does not contain ascender, descender, and loop produces a 20-symbol alphabet. For more details, see Ref. [28]. In order to train the classifier with this feature set, we have used the same databases described in Section 3.5.2. The recognition rates at zero-rejection level are 86.1% and 87.2% on validation and testing sets, respectively. This performance compares with the CC*uc* classifier.

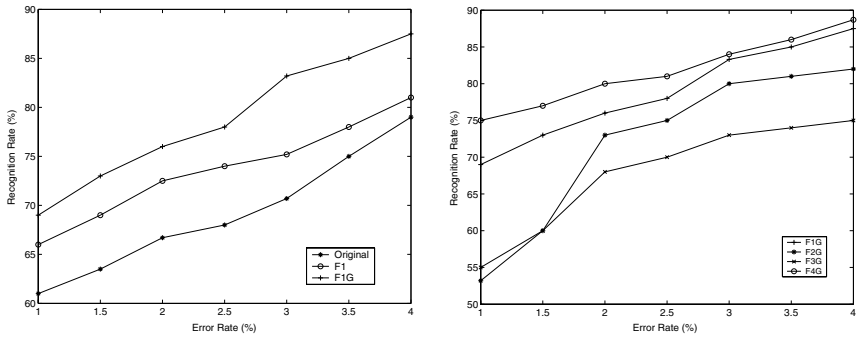
Since we have a new base classifier, our sets of base classifiers must be modified to cope with it. Thus, $F_{1G} = \{F_1 \bigcup G\}$, $F_{2G} = \{F_2 \bigcup G\}$, $F_{3G} = \{F_3 \bigcup G\}$, and $F_{4G} = \{F_1 \bigcup F_2 \bigcup F_3 \bigcup G\}$. In such cases, G stands for the classifier trained with global features. Table 3.7 summarizes the ensembles found using these new sets of base classifiers. It is worthy of remark the reduction of the size of the teams. This shows the ability of the algorithm

Table 3.7. Performance of the ensembles with global features.

Base Classifiers	Number of Classifiers	Rec. Rate (%) Testing
F_{1G}	2	92.2
F_{2G}	2	89.7
F_{3G}	7	85.5
F_{4G}	23	92.0

in finding not just diverse but also uncorrelated classifiers to compose the ensemble [36]. Besides, it corroborates to our claim that the classifier G when combined with other features bring an improvement to the performance.

In Figure 3.9 we compare the error-reject trade-offs for some ensembles reported in Table 3.7. Like the results at zero-rejection level, the improvement observed here also are quite impressive. Table 3.7 shows that F_{1G} and F_{4G} reach similar results on the test set at zero-rejection level, however, F_{1G} contains just two classifiers against 23 of F_{4G}. On the other hand, the latter features a slightly better error-reject trade-off in the long run (Figure 3.9b).



(a) (b)

Fig. 3.9. Improvements yielded by the ensembles: (a) F_1 and (b) Comparison among all ensembles.

Based on the experiments reported so far we can affirm that the unsupervised feature selection is a good strategy to generate diverse classifiers. This is made very clear in the experiments regarding the feature set DDD64. In such a case, the original classifier has a poor performance (about 65% on the test set), but when it is used to generate the set of base classifiers, the second-level MOGA was able to produce a good ensemble by maximizing the performance and the ambiguity measure. Such an ensemble of classifiers brought an improvement of about 15% in the recognition rate at zero-rejection level.

3.8 Discussion

The results obtained here attest that the proposed strategy is able to generate a set of good classifiers in both supervised and unsupervised contexts. To better evaluate our results, we have used two traditional ensemble methods (Bagging and Boosting) in the supervised context. Figure 3.10 reports the results. As we can see, the proposed methodology achieved better results, especially when considering very low error rates.

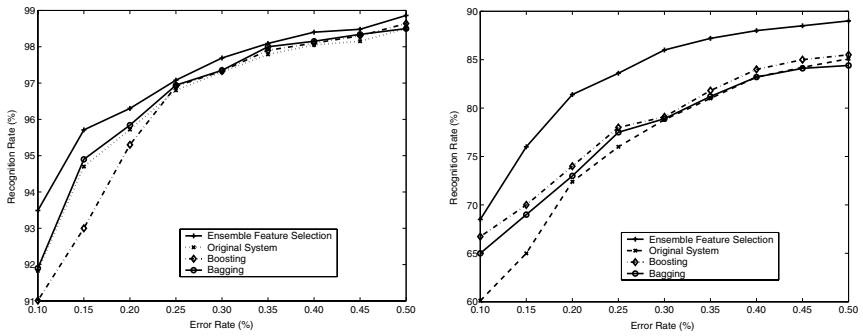


Fig. 3.10. Comparison among feature selection for ensembles, bagging, and boosting for the two feature sets used in the supervised context:(a) CC_{sc} and (b) EM_{sc}

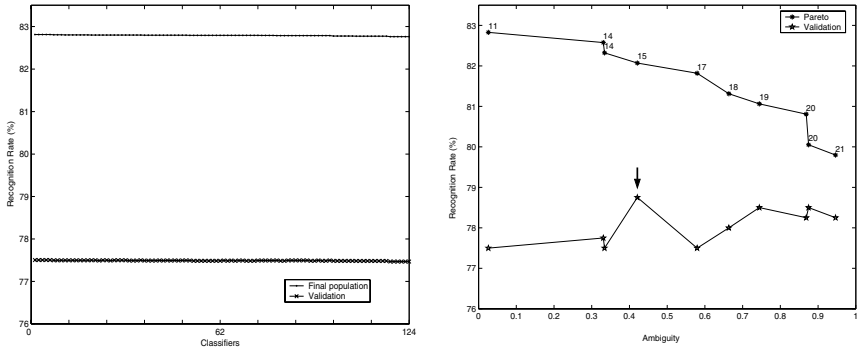
Diversity is an issue that deserves some attention when discussing ensemble of classifiers. As we have mentioned before, some authors advocated that diversity does not help at all. In our experiments, most of the time, the best ensembles of the Pareto-optimal also were the best for the unseen data. This could lead one to agree that diversity is not important when building ensembles, since even using a validation set the selected team is always the most accurate and with less diversity.

However, if we look carefully the results, we will observe that there are cases where the validation curve does not have the same shape of the Pareto-optimal. In such cases diversity is very useful to avoid selecting overfitted solutions.

One can argue that using a single GA and considering the entire final population, perhaps the similar solutions found in the Pareto-optimal produced by the MOGA will be there. To show that it does not happen, we have carried out some experiments with a single GA where the fitness function was the maximization of the ensemble´s accuracy. Since a single-objetive optimization algorithm searches for an optimum solution, it is natural to expect that it will converge towards the fittest solution, hence, the diversity of solutions

presented in the Pareto-optimal is not present in the final population of the single genetic algorithm.

To illustrate that, we present the results we got using a GA to find ensemble in F_2 (unsupervised context). The parameters used here are the same we have used for the MOGA (Section 3.7). Figure 3.11a plots all the classifiers found in the final population of the genetic algorithm. For the sake of comparison we reproduce Figure 3.8a in Figure 3.11b. As we can see, the population is very homogeneous and it converged, as expected, towards the most accurate ensemble.



(a) (b)

Fig. 3.11. Benefits of using diversity: (a) population (classifiers) of the final generation of the GA and (b) classifiers found by the MOGA.

Some attempts in this direction were made by Optiz [31]. He combined accuracy and diversity through the weighted-sum approach. As stated somewhere, when dealing with this kind of combination, one should deal with problems such as scaling and sensitivity towards the weights. We believe that our strategy offers a clever way to find the ensemble using genetic algorithms.

3.9 Conclusion

We have described a methodology for ensemble creation underpinned on the paradigm "overproduce and choose". It takes two levels of search where the first level overproduces a set of classifiers by performing feature selection while the second one chooses the best team of classifiers.

The feasibility of the strategy was demonstrated through comprehensive experiments carried out in the context of handwriting recognition. The idea of generating classifiers through feature selection was proved to be successful in both supervised and unsupervised contexts. The results attained in both situations and using different feature sets and base classifiers demonstrated the

efficiency of the proposed strategy by finding powerful ensembles, which succeed in improving the recognition rates for classifiers working with a very low error rates. Such results compare favorably to traditional ensemble methods such as Bagging and Boosting.

Finally we have addressed the issue of using diversity to build ensembles. As we have seen, using diversity jointly with the accuracy of the ensemble as selection criterion might be very helpful to avoid choosing overfitted solutions. Our results certainly bring some contribution to the field, but this still is an open problem.

References

[1] L. Breiman. Stacked regressions. *Machine Learning*, 24(1):49–64, 1996.
[2] E. Cantu-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, 2000.
[3] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1(224-227):550–554, 1979.
[4] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley and Sons Ltd, 2^{nd} edition, April 2002.
[5] J. G. Dy and C. E. Brodley. Feature subset selection and order identification for unsupervised learning. In *Proc. 17^{th} International Conference on Machine Learning*, 2000.
[6] B. Efron and Tibshirani R. *An introduction to the Bootstrap*. Chapman and Hall, 1993.
[7] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proc. of 13^{th} International Conference on Machine Learning*, pages 148–156, Bary-Italy, 1996.
[8] G. Fumera, F. Roli, and G. Giacinto. Reject option with multiple thresholds. *Pattern Recognition*, 33(12):2099–2101, 2000.
[9] G. Giacinto and F. Roli. Design of effective neural network ensemble for image classification purposes. *Image Vision and Computing Journal*, 9-10:697–705, 2001.
[10] S. Gunter and H. Bunke. Creation of classifier ensembles for handwritten word recogntion using feature selection algorithms. In *Proc. of 8^{th} IWFHR*, pages 183–188, Niagara-on-the-Lake, Canada, 2002.
[11] S. Hashem. Optimal linear combinations of neural networks. *Neural Networks*, 10(4):599–614, 1997.
[12] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
[13] G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problems. In *Proc. of 11^{th} International Conference on Machine Learning*, pages 121–129, 1994.
[14] J. J. Oliveira Jr., J. M. Carvalho, C. O. A. Freitas, and R. Sabourin. Evaluating NN and HMM classifiers for handwritten word recognition. In *Proceedings of the 15^{th} Brazilian Symposium on Computer Graphics and Image Processing*, pages 210–217. IEEE Computer Society, 2002.

[15] Y. S. Kim, W. N. Street, and F. Menczer. Feature selection in unsupervised learning via evolutionary search. In *Proc. 6^{th} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 365–369, 2000.

[16] J. Kittler, M. Hatef, R. Duin, and J. Matas. On combining classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[17] A. Krogh and J. Vedelsby. Neural networks ensembles, cross validation, and active learning. In G.Tesauro et al, editor, *Advances in Neural Information Processing Systems 7*, pages 231–238. MIT Press, 1995.

[18] M. Kudo and J. Sklansky. Comparision of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33(1):25–41, 2000.

[19] L. Kuncheva. That elusive diversity in classifier ensembles. In *Proc. of ibPRIA, LNCS 2652*, pages 1126–1138, Mallorca, Spain, 2003.

[20] L. Kuncheva, J. C. Bezdek, and R. P. W. Duin. Decision templates for multiple classifier fusion: An experimental comparison. *Pattern Recognition*, 34(2):299–314, 2001.

[21] L. Kuncheva and L. C. Jain. Designing classifier fusion systems by genetic algorithms. *IEEE Trans. on Evolutionary Computation*, 4(4):327–336, 2000.

[22] L. I. Kuncheva and C. J. Whitaker. Ten measures of diversity in classifier ensembles:limits for two classifiers. In *Proc. of IEE Workshop on Intelligent Sensor Processing*, pages 1–10, 2001.

[23] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles. *Machine Learning*, 51:181–207, 2003.

[24] M. Last, H. Bunke, and A. Kandel. A feature-based serial approach to classifier combination. *Pattern Analysis and Applications*, 5:385–398, 2002.

[25] M. Miki, T. Hiroyasu, K. Kaneko, and K. Hatanaka. A parallel genetic algorithm with distributed environment scheme. In *Proc. of International Conference on System, Man, and Cybernetics*, volume 1, pages 695–700, 1999.

[26] J. Moody and J. Utans. Principled architecture selection for neural networks: Application to corporate bond rating prediction. In J. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*. Morgan Kaufmann, 1991.

[27] M. Morita, R. Sabourin, F. Bortolozzi, and C. Y. Suen. Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition. In *Proceedings of the 7^{th} International Conference on Document Analysis and Recognition*, pages 666–670. IEEE Computer Society, 2003.

[28] M. Morita, R. Sabourin, F. Bortolozzi, and Suen C. Y. Segmentation and recognition of handwritten dates: An hmm-mlp hybrid approach. *International Journal on Document Analysis and Recognition*, 6:248–262, 2003.

[29] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen. Automatic recognition of handwritten numerical strings: A recognition and verification strategy. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(11):1438–1454, 2002.

[30] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen. A methodology for feature selection using multi-objective genetic algorithms for handwritten digit string recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 17(6):903–930, 2003.

[31] D. W. Optiz. Feature selection for ensembles. In *Proc. of* 16^{th} *International Conference on Artificial Intelligence*, pages 379–384, 1999.

[32] D. Partridge and W. B. Yates. Engineering multiversion neural-net systems. *Neural Computation*, 8(4):869–893, 1996.
[33] D. Ruta. Multilayer selection-fusion model for pattern classification. In *Proceedings of the IASTED Artificial Intelligence and Application Conference*, Insbruck, Austria, 2004.
[34] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1995.
[35] A. Tsymbal, S. Puuronen, and D. W. Patterson. Ensemble feature selection with the simple Bayesian classification. *Information Fusion*, 4:87–100, 2003.
[36] K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8(3-4):385–404, 1996.
[37] K. Tumer and N. C. Oza. Input decimated ensembles. *Pattern Analysis and Applications*, 6:65–77, 2003.
[38] H. Yuan, S. S. Tseng, W. Gangshan, and Z. Fuyan. A two-phase feature selection method using both filter and wrapper. In *Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 132–136, 1999.