ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

THESIS PRESENTED TO
ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
Ph.D.

BY
MIRANDA DOS SANTOS, Eulanda

STATIC AND DYNAMIC OVERPRODUCTION AND SELECTION OF CLASSIFIER ENSEMBLES WITH GENETIC ALGORITHMS

MONTREAL, FEBRUARY 27, 2008

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

THESIS PRESENTED TO
ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
Ph.D.

BY
MIRANDA DOS SANTOS, Eulanda

STATIC AND DYNAMIC OVERPRODUCTION AND SELECTION OF CLASSIFIER ENSEMBLES WITH GENETIC ALGORITHMS

MONTREAL, FEBRUARY 27, 2008

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS :

Mr. Robert Sabourin, thesis director
Département de génie de la production automatisée at École de technologie supérieure

Mr. Patrick Maupin, thesis co-director
Recherche et développement pour la défense Canada (Valcartier), groupe de monotoring et analyse de la situation

Mr. Pierre Dumouchel, committee president
Département de génie logiciel et des technologies de l'information at École de technologie supérieure

Mr. Jean Meunier, external examiner
Département d'Informatique et Recherche Opérationnelle at Université de Montréal

Mr. Éric Granger, examiner
Département de génie de la production automatisée at École de technologie supérieure

THIS THESIS WAS PRESENTED AND DEFENDED

BEFORE A BOARD OF EXAMINERS AND PUBLIC

ON FEBRUARY 22, 2008

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGMENTS

I would like to acknowledge the support and help of many people who encouraged me throughout these years of Ph.D. It is not possible to enumerate all of them but I would like to express my gratitude to some people in particular.

First, I would like to thank my supervisor, Dr. Robert Sabourin who has been source of encouragement, guidance and patience. His support and supervision were fundamental for the development of this work.

Thanks also to Patrick Maupin from Defence Research and Development Canada, DRDC-Valcartier. His involvement along the course of this research helped make the thesis more complete.

I would like to thank the members of my examining committee: Dr. Éric Granger, Dr. Jean Meunier and Dr. Pierre Dumouchel, who has made valuable suggestions to the final version of this thesis.

Thanks to the members of LIVIA (*Laboratoire d'imagerie, de vision et d'intelligence artificielle*) who contributed to a friendly and open research environment. Special thanks to Albert Ko, Carlos Cadena, Clement Chion, Dominique Rivard, Eduardo Vellasques, Éric Thibodeau, Guillaume Tremblay, Jonathan Milgram, Luana Batista, Luis Da Costa, Marcelo Kapp, Mathias Adankon, Paulo Cavalin, Paulo Radtke and Vincent Doré.

This research has been funded by the CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), Brazilian Government, and Defence Research and Development Canada under the contract W7701-2-4425. Special thanks to people from CAPES and Defence Research and Development Canada, DRDC-Valcartier, who have had the vision to support this project.

I am indebted to Marie-Adelaide Vaz who has been my family and best friend in Montreal. Words cannot express how grateful I am for everything you have done for me. I also would like to thank Cinthia and Marcelo Kapp, who besides friends, have been very special trip partners. I am also indebted to my family for their love, support, guidance and prayers, not only throughout the years of my PhD, but, throughout my life. To them is all my love and prayers.

Finally, all thanks and praise are due to God for giving me the strength and knowledge to complete this work, "for from him and through him and to him are all things. To him be the glory forever!"

STATIC AND DYNAMIC OVERPRODUCTION AND SELECTION OF CLASSIFIER ENSEMBLES WITH GENETIC ALGORITHMS

MIRANDA DOS SANTOS, Eulanda

ABSTRACT

The overproduce-and-choose strategy is a static classifier ensemble selection approach, which is divided into overproduction and selection phases. This thesis focuses on the selection phase, which is the challenge in overproduce-and-choose strategy. When this phase is implemented as an optimization process, the search criterion and the search algorithm are the two major topics involved. In this thesis, we concentrate in optimization processes conducted using genetic algorithms guided by both single- and multi-objective functions. We first focus on finding the best search criterion. Various search criteria are investigated, such as diversity, the error rate and ensemble size. Error rate and diversity measures are directly compared in the single-objective optimization approach. Diversity measures are combined with the error rate and with ensemble size, in pairs of objective functions, to guide the multi-optimization approach. Experimental results are presented and discussed.

Thereafter, we show that besides focusing on the characteristics of the decision profiles of ensemble members, the control of overfitting at the selection phase of overproduce-and-choose strategy must also be taken into account. We show how overfitting can be detected at the selection phase and present three strategies to control overfitting. These strategies are tailored for the classifier ensemble selection problem and compared. This comparison allows us to show that a global validation strategy should be applied to control overfitting in optimization processes involving a classifier ensembles selection task. Furthermore, this study has helped us establish that this global validation strategy can be used as a tool to measure the relationship between diversity and classification performance when diversity measures are employed as single-objective functions.

Finally, the main contribution of this thesis is a proposed dynamic overproduce-and-choose strategy. While the static overproduce-and-choose selection strategy has traditionally focused on finding the most accurate subset of classifiers during the selection phase, and using it to predict the class of all the test samples, our dynamic overproduce-and-choose strategy allows the selection of the most confident subset of classifiers to label each test sample individually. Our method combines optimization and dynamic selection in a two-level selection phase. The optimization level is intended to generate a population of highly accurate classifier ensembles, while the dynamic selection level applies measures of confidence in order to select the ensemble with the highest degree of confidence in the current decision. Three different confidence measures are presented and compared. Our method outperforms classical static and dynamic selection strategies.

SURPRODUCTION ET SELECTION STATIQUE ET DYNAMIQUE DES ENSEMBLES DE CLASSIFICATEURS AVEC ALGORITHMES GÉNÉTIQUES

MIRANDA DOS SANTOS, Eulanda

RÉSUMÉ

La stratégie de "surproduction et choix" est une approche de sélection statique des ensembles de classificateurs, et elle est divisée en deux étapes: une phase de surproduction et une phase de sélection. Cette thèse porte principalement sur l'étude de la phase de sélection, qui constitue le défi le plus important dans la stratégie de surproduction et choix. La phase de sélection est considérée ici comme un problème d'optimisation mono ou multicritère. Conséquemment, le choix de la fonction objectif et de l'algorithme de recherche font l'objet d'une attention particulière dans cette thèse. Les critères étudiés incluent les mesures de diversité, le taux d'erreur et la cardinalité de l'ensemble. L'optimisation monocritère permet la comparaison objective des mesures de diversité par rapport à la performance globale des ensembles. De plus, les mesures de diversité sont combinées avec le taux d'erreur ou la cardinalité de l'ensemble lors de l'optimisation multicritère. Des résultats expérimentaux sont présentés et discutés.

Ensuite, on montre expérimentalement que le surapprentissage est potentiellement présent lors la phase de sélection du meilleur ensemble de classificateurs. Nous proposons une nouvelle méthode pour détecter la présence de surapprentissage durant le processus d'optimisation (phase de sélection). Trois stratégies sont ensuite analysées pour tenter de contrôler le surapprentissage. L'analyse des résultats révèle qu'une stratégie de validation globale doit être considérée pour contrôler le surapprentissage pendant le processus d'optimisation des ensembles de classificateurs. Cette étude a également permis de vérifier que la stratégie globale de validation peut être utilisée comme outil pour mesurer empiriquement la relation possible entre la diversité et la performance globale des ensembles de classificateurs.

Finalement, la plus importante contribution de cette thèse est la mise en oeuvre d'une nouvelle stratégie pour la sélection dynamique des ensembles de classificateurs. Les approches traditionnelles pour la sélection des ensembles de classificateurs sont essentiellement statiques, c'est-à-dire que le choix du meilleur ensemble est définitif et celui-ci servira pour classer tous les exemples futurs. La stratégie de surproduction et choix dynamique proposée dans cette thèse permet la sélection, pour chaque exemple à classer, du sous-ensemble de classificateurs le plus confiant pour décider de la classe d'appartenance. Notre méthode concilie l'optimisation et la sélection dynamique dans une phase de sélection à deux niveaux. L'objectif du premier niveau est de produire une population d'ensembles de classificateurs candidats qui montrent une grande capacité de généralisa-

tion, alors que le deuxième niveau se charge de sélectionner dynamiquement l'ensemble qui présente le degré de certitude le plus élevé pour décider de la classe d'appartenance de l'objet à classer. La méthode de sélection dynamique proposée domine les approches conventionnelles (approches statiques) sur les problèmes de reconnaissance de formes étudiés dans le cadre de cette thèse.

SURPRODUCTION ET SELECTION STATIQUE ET DYNAMIQUE DES ENSEMBLES DE CLASSIFICATEURS AVEC ALGORITHMES GÉNÉTIQUES

MIRANDA DOS SANTOS, Eulanda

SYNTHÈSE

Le choix du meilleur classificateur est toujours dépendant de la connaissance a priori définie par la base de données utilisée pour l'apprentissage. Généralement la capacité de généraliser sur des nouvelles données n'est pas satisfaisante étant donné que le problème de reconnaissance est mal défini. Afin de palier à ce problème, les ensembles de classificateurs permettent en général une augmentation de la capacité de généraliser sur de nouvelles données.

Les méthodes proposées pour la sélection des ensembles de classificateurs sont réparties en deux catégories : la sélection statique et la sélection dynamique. Dans le premier cas, le sous-ensemble des classificateurs le plus performant, trouvé pendant la phase d'entraînement, est utilisé pour classer tous les échantillons de la base de test. Dans le second cas, le choix est fait dynamiquement durant la phase de test, en tenant compte des propriétés de l'échantillon à classer. La stratégie de "surproduction et choix" est une approche statique pour la sélection de classificateurs. Cette stratégie repose sur l'hypothèse que plusieurs classificateurs candidats sont redondants et n'apportent pas de contribution supplémentaire lors de la fusion des décisions individuelles.

La stratégie de "surproduction et choix" est divisée en deux étapes de traitement : la phase de surproduction et la phase de sélection. La phase de surproduction est responsable de générer un large groupe initial de classificateurs candidats, alors que la phase de sélection cherche à tester les différents sous-ensembles de classificateurs afin de choisir le sous-ensemble le plus performant. La phase de surproduction peut être mise en oeuvre en utilisant n'importe quelle méthode de génération des ensembles de classificateurs, et ce indépendamment du choix des classificateurs de base. Cependant, la phase de sélection est l'aspect fondamental de la stratégie de surproduction et choix. Ceci reste un probléme non résolu dans la littérature.

La phase de sélection est formalisée comme un problème d'optimisation mono ou multicritère. Conséquemment, le choix de la fonction objectif et de l'algorithme de recherche sont les aspects les plus importants à considérer. Il n'y a pas de consensus actuellement dans la littérature concernant le choix de la fonction objectif. En termes d'algorithmes de recherche, plusieurs algorithmes ont été proposées pour la réalisation du processus de sélection. Les algorithmes génétiques sont intéressants parce qu'ils génèrent les N meilleures solutions à la fin du processus d'optimisation. En effet, plusieurs solutions sont dispo-

nibles à la fin du processus ce qui permet éventuellement la conception d'une phase de post-traitement dans les systèmes réels.

L'objectif principal de cette thèse est de proposer une alternative à l'approche classique de type surproduction et choix (approche statique). Cette nouvelle stratégie de surproduction et choix dynamique, permet la sélection du sous-ensemble des classificateurs le plus compétent pour décider la classe d'appartenance de chaque échantillon de test à classer.

Le premier chapitre présente l'état de l'art dans les domaines des ensembles des classificateurs. Premièrement, les méthodes classiques proposées pour la combinaison d'ensemble de classificateurs sont présentées et analysées. Ensuite, une typologie des méthodes publiées pour la sélection dynamique de classificateurs est présentée et la stratégie de surproduction et choix est introduite.

Les critères de recherche pour guider le processus d'optimisation de la phase de sélection sont évalués au chapitre deux. Les algorithmes génétiques monocritère et multicritère sont utilisés pour la mise en oeuvre du processus d'optimisation. Nous avons analysé quatorze fonctions objectives qui sont proposées dans la littérature pour la sélection des ensembles de classificateurs : le taux d'erreur, douze mesures de diversité et la cardinalité. Le taux d'erreur et les mesures de diversité ont été directement comparés en utilisant une approche d'optimisation monocritère. Cette comparaison permet de vérifier la possibilité de remplacer le taux d'erreur par la diversité pour trouver le sous-ensemble des classificateurs le plus performant. De plus, les mesures de diversité ont été utilisées conjointement avec le taux d'erreur pour l'étude des approches d'optimisation multicritère. Ces expériences permettent de vérifier si l'utilisation conjointe de la diversité et du taux d'erreur permet la sélection des ensembles classificateurs plus performants. Ensuite, nous avons montré l'analogie qui existe entre la sélection de caractéristiques et la sélection des ensembles des classificateurs en tenant compte conjointement des mesures de cardinalité des ensembles avec le taux d'erreur (ou une mesure de diversité). Les résultats expérimentaux ont a été obtenus sur un problème de reconnaissance de chiffres manuscrits.

Le chapitre trois constitue une contribution importante de cette thèse. Nous montrons dans quelle mesure le processus de sélection des ensembles de classificateurs souffre du problème de surapprentissage. Etant donné que les algorithmes génétiques monocritère et multicritére sont utilisés dans cette thèse, trois stratégies basées sur un mécanisme d'archivage des meilleures solutions sont présentées et comparées. Ces stratégies sont : la validation partielle, où le mécanisme d'archivage est mis à jour seulement à la fin du processus d'optimisation ; "backwarding", où le mécanisme d'archivage est mis à jour à chaque génération sur la base de la meilleure solution identifiée pour chaque population durant l'évolution ; et la validation globale, qui permet la mise à jour de l'archive avec la meilleure solution identifiée dans la base de données de validation à chaque génération. Finalement, la stratégie de validation globale est présentée comme un outil pour mesurer

le lien entre la diversité d'opinion évaluée entre les membres de l'ensemble et la performance globale. Nous avons montré expérimentalement que plusieurs mesures de diversité ne sont pas reliées avec la performance globale des ensembles, ce qui confirme plusieurs études publiées récemment sur ce sujet.

Finalement, la contribution la plus importante de cette thèse, soit la mise en oeuvre d'une nouvelle stratégie pour la sélection dynamique des ensembles de classificateurs, fait l'objet du chapitre quatre. Les approches traditionnelles pour la sélection des ensembles de classificateurs sont essentiellement statiques, c'est-à-dire que le choix du meilleur ensemble est définitif et celui-ci servira pour classer tous les exemples futurs. La stratégie de surproduction et choix dynamique proposée dans cette thèse permet la sélection, pour chaque exemple à classer, du sous-ensemble de classificateurs le plus confiant pour décider de la classe d'appartenance. Notre méthode concilie l'optimisation et la sélection dynamique dans une phase de sélection à deux niveaux. L'objectif du premier niveau est de produire une population d'ensembles de classificateurs candidats qui montrent une grande capacité de généralisation, alors que le deuxième niveau se charge de sélectionner dynamiquement l'ensemble qui présente le degré de certitude le plus élevé pour décider de la classe d'appartenance de l'objet à classer. La méthode de sélection dynamique proposée domine les approches conventionnelles (approches statiques) sur les problèmes de reconnaissance de formes étudiés dans le cadre de cette thèse.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

ADS	Ambiguity-guided dynamic selection
BAG	Bagging
BKS	Behavior-Knowledge Space
BS	Backward search
BV	Backwarding validation
CSDS	Class strength-based dynamic selection
DCS-LA	Dynamic classifier selection with local accuracy
DCS	Dynamic classifier selection
DOCS	Dynamic overproduce-and-choose strategy
DT	Decision tree
FS	Forward search
FSS	Feature subset selection
GA	Genetic algorithm
GANSEN	GA-based Selective Ensemble
GP	Genetic Programming
GV	Global Validation
HC	Hill-climbing search
Het	Heterogeneous classifiers

IPS	k^{th} Objective Pareto Spread
kNN	K Nearest Neighbors
KNORA	K nearest-oracles
MDS	Margin-based dynamic selection
MLP	Multilayer perceptron
MOGA	Multi-objective genetic algorithm
MOOP	Multi-objective optimization problems
NIST	National Institute of Standards and Technology
NSGA-II	Fast elitist non-dominated sorting genetic algorithm
NSGA	Non-dominated Sorting Genetic Algorithm
OCS	Overproduce-and-choose strategy
OPS	Overall Pareto spread
PBIL	Population-based incremental learning
PSO	Particle Swarm Optimization
PV	Partial Validation
RBF	Radial Basis Function
RSS	Random subspace method
SOCS	Static overproduce-and-choose strategy
SVM	Support Vector Machines
TS	Tabu search

LIST OF SYMBOLS

a_i	Ambiguity of the i-th classifier
$\mathcal{A}$	Auxiliary archive
α_{C_j}	Local candidate ensemble's class accuracy
$\mathcal{C}$	Initial pool of classifiers
$\mathbf{C}^{*'}$	Population of candidate ensembles found using the validation dataset at the optimization level
$\mathbf{C}^{*}$	Population of candidate ensembles found using the optimization dataset at the optimization level
$\mathbf{C}^{q}(g)$	Offspring population
$\mathbf{C}^{r}(g)$	Elitism population
$\mathbf{C}(g)$	Population of ensembles found at each generation g
c_i^*	Winning individual classifier
c_i	Individual classifier
$C_j^{*'}$	The best performing subset of classifiers obtained in the validation dataset
C_j^{*}	The best performing subset of classifiers obtained in the optimization dataset
C_j	Individual candidate ensemble
C_j'	Candidate ensemble with lowest ϵ

$\mathbf{C}_k(g)$	Pareto front found in $\mathcal{O}$ at generation g
$\mathbf{C}_k^*$	Final Pareto fronts found in $\mathcal{O}$
$\mathbf{C}_k^{*'}$	Final Pareto fronts found in $\mathcal{V}$
cov	Coverage function
δ	Double-fault
ϵ	The error rate objective function
F	Number of classifiers in C_j that correctly classify a pattern x
$f(x)$	Unknown function
ϕ	Uncontrolled overfitting
$\mathcal{G}$	Test dataset
g	Each generation step
γ	Ambiguity
$\bar{\gamma}_i$	Local ambiguity calculated for the test sample $\mathbf{x}_{i,g}$
h	Approximation function
H	Hypothesis space
imp	Pareto improvement measure
η	Disagreement
κ	Interrater agreement
l	Number of classifiers in candidate ensemble C_j
λ	Fault majority

max	Maximum rule
$max(g)$	Maximum number of generations
min	Minimum rule
mv	Majority voting combination function
$m(x_i)$	Number of classifiers making error on observation x_i
μ	Measure of margin
N^{ab}	Number of examples classified in X, a, b may assume the value of 1 when the classifier is correct and 0, otherwise.
n	Size of initial pool of candidate classifiers
nb	Naive Bayes
$\mathcal{O}$	Optimization dataset
$\mathcal{P}(\mathcal{C})$	Powerset of $\mathcal{C}$
$p(i)$	Probability that i classifier(s) fail when classifying a sample x
pr	Product rule
$\bar{p}$	Average individual accuracy
Φ	Q-statistic
θ	Difficulty measure
Θ	Strength relative to the closest class
R_j^*	Winning region of competence
R_j	Regions of competence

$r(x)$	Number of classifiers that correctly classify sample x
ρ	Correlation coefficient
sr	Sum rule
σ	Coincident failure diversity
$\mathcal{T}$	Training dataset
τ	Generalized diversity
$\Omega = \{\omega_1, \omega_2 \ldots, \omega_c\}$	Set of class labels
ω_k	Class labels
$\mathcal{V}$	Validation dataset
$v(\omega_k)$	Number of votes for class ω_k
w	Size of the population of candidate ensembles
ξ	Entropy
$\mathbf{x}_{i,g}$	Test sample
$\mathbf{x}_{i,o}$	Optimization sample
$\mathbf{x}_{i,t}$	Training sample
$\mathbf{x}_{i,v}$	Validation sample
y_i	Class label output of the i-th classifier
Y	Proportion of classifiers that do not correctly classify a randomly chosen sample x
ψ	Kohavi-Wolpert
ζ	Ensemble size

INTRODUCTION

The ensemble of classifiers method has become a dominant approach in several different fields of application such as Machine Learning and Pattern Recognition. Such interest is motivated by the theoretical [16] and experimental [31; 96] studies, which show that classifier ensembles may improve traditional single classifiers. Among the various ensemble generation methods available, the most popular are bagging [5], boosting [21] and the random subspace method [32]. Two main approaches for the design of classifier ensembles are clearly defined in the literature: (1) classifier fusion; and (2) classifier selection.

The most common and most general operation is the combination of all classifiers members' decisions. Majority voting, sum, product, maximum, minimum [35], Bayesian rule [86] and Dempster-Shafer [68] are examples of functions used to combine ensemble members' decisions. Classifier fusion relies on the assumption that all ensemble members make independent errors. Thus, pooling the decisions of the ensemble members may lead to increasing the overall performance of the system. However, it is difficult to impose independence among ensemble's component members, especially since the component classifiers are redundant [78], i.e. they provide responses to the same problem [30]. As a consequence, there is no guarantee that a particular ensemble combination method will achieve error independence. When the condition of independence is not verified, it cannot be guaranteed that the combination of classifier members' decision will improve the final classification performance.

Classifier selection is traditionally defined as a strategy which assumes that each ensemble member is an expert in some local regions of the feature space [107]. The most locally accurate classifier is selected to estimate the class of each particular test pattern. Two categories of classifier selection techniques exist: static and dynamic. In the first case, regions of competence are defined during the training phase, while in the second case, they are defined during the classification phase taking into account the characteristics of the sam-

ple to be classified. However, there may be a drawback to both selection strategies: when the local expert does not classify the test pattern correctly, there is no way to avoid the misclassification [80]. Moreover, these approaches, for instance *Dynamic Classifier Selection with Local Accuracy* (DCS-LA) [101], often involve high computing complexity, as a result of estimating regions of competence, and may be critically affected by parameters such as the number of neighbors considered (k value) for regions defined by k nearest neighbors and distance functions.

Another definition of static classifier selection can be found in the Neural Network literature. It is called either the *overproduce-and-choose strategy* [58] or the *test-and-select methodology* [78]. From this different perspective, the overproduction phase involves the generation of an initial large pool of candidate classifiers, while the selection phase is intended to test different subsets in order to select the best performing subset of classifiers, which is then used to classify the whole test set. The assumption behind overproduce-and-choose strategy is that candidate classifiers are redundant as an analogy with the feature subset selection problem. Thus, finding the most relevant subset of classifiers is better than combining all the available classifiers.

Problem Statement

In this thesis, the focus is on the overproduce-and-choose strategy, which is traditionally divided into two phases: (1) overproduction; and (2) selection. The former is devoted to constructing an initial large pool of classifiers. The latter tests different combinations of these classifiers in order to identify the optimal candidate ensemble. Clearly, the overproduction phase may be undertaken using any ensemble generation method and base classifier model. The selection phase, however, is the fundamental issue in overproduce-and-choose strategy, since it focuses on finding the subset of classifiers with optimal accuracy. This remains an open problem in the literature. Although the search for the optimal subset of classifiers can be exhaustive [78], search algorithms might be used when a large

initial pool of candidate classifiers $\mathcal{C}$ is involved due to the exponential complexity of an exhaustive search, since the size of $\mathcal{P}(\mathcal{C})$ is 2^n, n being the number of classifiers in $\mathcal{C}$ and $\mathcal{P}(\mathcal{C})$ the powerset of $\mathcal{C}$ defining the population of all possible candidate ensembles.

When dealing with the selection phase using a non-exhaustive search, two important aspects should be analyzed: (1) the search criterion; and (2) the search algorithm. The first aspect has received a great deal of attention in the recent literature, without much consensus. Ensemble combination performance, ensemble size and diversity measures are the most frequent search criteria employed in the literature. Performance is the most obvious of these, since it allows the main objective of pattern recognition, i.e. finding predictors with a high recognition rate, to be achieved. Ensemble size is interesting due to the possibility of increasing performance while minimizing the number of classifiers in order to accomplish requirements of high performance and low ensemble size [62]. Finally, there is agreement on the important role played by diversity since ensembles can be more accurate than individual classifiers only when classifier members present diversity among themselves. Nonetheless, the relationship between diversity measures and accuracy is unclear [44]. The combination of performance and diversity as search criteria in a multi-objective optimization approach offers a better way to overcome such an apparent dilemma by allowing the simultaneous use of both measures.

In terms of search algorithms, several algorithms have been applied in the literature for the selection phase, ranging from ranking the n best classifiers [58] to *genetic algorithms* (GAs) [70]. GAs are attractive since they allow the fairly easy implementation of ensemble classifier selection tasks as optimization processes [82] using both single- and multi-objective functions. Moreover, population-based GAs are good for classifier selection problems because of the possibility of dealing with a population of solutions rather than only one, which can be important in performing a post-processing phase. However it has been shown that such stochastic search algorithms when used in conjunction to Machine Learning techniques are prone to overfitting in different application problems like

the distribution estimation algorithms [102], the design of evolutionary multi-objective learning system [46], multi-objective pattern classification [3], multi-objective optimization of Support Vector Machines [87] and wrapper-based feature subset selection [47; 22]. Even though different aspects have been addressed in works that investigate overfitting in the context of ensemble of classifiers, for instance regularization terms [63] and methods for tuning classifiers members [59], very few work has been devoted to the control of overfitting at the selection phase.

Besides search criterion and search algorithm, other difficulties are concerned when performing selection of classifier ensembles. Classical overproduce-and-choose strategy is subject to two main problems. First, a fixed subset of classifiers defined using a training/optimization dataset may not be well adapted for the whole test set. This problem is similar to searching for a universal best individual classifier, i.e. due to differences among samples, there is no individual classifier that is perfectly adapted for every test sample. Moreover, as stated by the "No Free Lunch" theorem [10], no algorithm may be assumed to be better than any other algorithm when averaged over all possible classes of problems.

The second problem occurs when Pareto-based algorithms are used at the selection phase. These algorithms are efficient tools for overproduce-and-choose strategy due to their capacity to solve *multi-objective optimization problems* (MOOPs) such as the simultaneous use of diversity and classification performance as the objective functions. They use Pareto dominance to solve MOOPs. Since a Pareto front is a set of nondominated solutions representing different tradeoffs with respect to the multiple objective functions, the task of selecting the best subset of classifiers is more complex. This is a persistent problem in MOOPs applications. Often, only one objective function is taken into account to perform the choice. In [89], for example, the solution with the highest classification performance was picked up to classify the test samples, even though the solutions were optimized regarding both diversity and classification performance measures.

Goals of the Research and Contributions

The first goal of this thesis is to determine the best objective function for finding high-performance classifier ensembles at the selection phase, when this selection is formulated as an optimization problem performed by both single- and multi-objective GAs. Several issues were addressed in order to deal with this problem: (1) the error rate and diversity measures were directly compared using a single-objective optimization approach performed by GA. This direct comparison allowed us to verify the possibility of using diversity instead of performance to find high-performance subset of classifiers. (2) diversity measures were applied in combination with the error rate in pairs of objective functions in a multi-optimization approach performed by *multi-objective* GA (MOGA) in order to investigate whether including both performance and diversity as objective functions leads to selection of high-performance classifier ensembles. Finally, (3) we investigated the possibility of establishing an analogy between feature subset selection and ensemble classifier selection by combining ensemble size with the error rate, as well as with the diversity measures in pairs of objective functions in the multi-optimization approach. Part of this analysis was presented in [72].

The second goal is to show experimentally that an overfitting control strategy must be conducted *during* the optimization process, which is performed at the selection phase. In this study, we used the bagging and random subspace algorithms for ensemble generation at the overproduction phase. The classification error rate and a set of diversity measures were applied as search criteria. Since both GA and MOGA search algorithms were examined, we investigated in this thesis the use of an auxiliary archive to store the best subset of classifiers (or Pareto front in the MOGA case) obtained in a validation process using a validation dataset to control overfitting. Three different strategies to update the auxiliary archive have been compared and adapted in this thesis to the context of single and multi-objective selection of classifier ensembles: (1) *partial validation* where the auxiliary archive is updated only in the last generation of the optimization process; (2) *backward-*

ing [67] which relies on monitoring the optimization process by updating the auxiliary archive with the best solution from each generation and (3) *global validation* [62] updating the archive by storing in it the Pareto front (or the best solution in the GA case) identified on the validation dataset at each generation step.

The global validation strategy is presented as a tool to show the relationship between diversity and performance, specifically when diversity measures are used to guide GA. The assumption is that if a strong relationship between diversity and performance exists, the solution obtained by performing global validation solely guided by diversity should be close or equal to the solution with the highest performance among all solutions evaluated. This offers a new possibility to analyze the relationship between diversity and performance which has received a great deal of attention in the literature. In [75], we present this overfitting analysis.

Finally, the last goal is to propose a dynamic overproduce-and-choose strategy which combines optimization and dynamic selection in a two-level selection phase to allow selection of the most confident subset of classifiers to label each test sample individually. Selection at the optimization level is intended to generate a population of highly accurate candidate classifier ensembles, while at the dynamic selection level measures of confidence are used to reveal the candidate ensemble with highest degree of confidence in the current decision. Three different confidence measures are investigated.

Our objective is to overcome the three drawbacks mentioned above: Rather than selecting only one candidate ensemble found during the optimization level, as is done in *static* overproduce-and-choose strategy, the selection of the best candidate ensemble is based directly on the test patterns. Our assumption is that the generalization performance will increase, since a population of potential high accuracy candidate ensembles are considered to select the most competent solution for each test sample. This first point is particularly important in problems involving Pareto-based algorithms, because our method allows all

equally competent solutions over the Pareto front to be tested; (2) Instead of using only one local expert to classify each test sample, as is done in traditional classifier selection strategies (both static and dynamic), the selection of a subset of classifiers may decrease misclassification; and, finally, (3) Our dynamic selection avoids estimating regions of competence and distance measures in selecting the best candidate ensemble for each test sample, since it relies on calculating confidence measures rather than on performance.

Moreover, we prove both theoretically and experimentally that the selection of the solution with the highest level of confidence among its members permits an increase in the "degree of certainty" of the classification, increasing the generalization performance as a consequence. These interesting results motivated us to investigate three confidence measures in this thesis which measure the extent of consensus of candidate ensembles: (1) *Ambiguity* measures the number of classifiers in disagreement with the majority voting; (2) *Margin*, inspired by the definition of margin, measures the difference between the number of votes assigned to the two classes with the highest number of votes, indicating the candidate ensemble's level of certainty about the majority voting class; and (3) *Strength relative to the closest class* [8] also measures the difference between the number of votes received by the majority voting class and the class with the second highest number of votes; however, this difference is divided by the performance achieved by each candidate ensemble when assigning the majority voting class for samples contained in a validation dataset. This additional information indicates how often each candidate ensemble made the right decision in assigning the selected class.

As marginal contributions, we also point out the best method for the overproduction phase on comparing bagging and the random subspace method. In [73], we first introduced the idea that choosing the candidate ensemble with the largest consensus, measured using ambiguity, to predict the test pattern class leads to selecting the solution with greatest certainty in the current decision. In [74], we present the complete dynamic overproduce-and-choose strategy.

Organization of the Thesis

This thesis is organized as follows. In Chapter 1, we present a brief overview of the literature related to ensemble of classifiers in order to be able to introduce all definitions and research work related to the overproduce-and-choose strategy. Firstly, the combination of classifier ensemble is presented. Then, the traditional definition of dynamic classifier selection is summarized. Finally, the overproduce-and-choose strategy is explained. In this chapter we emphasize that the overproduce-and-choose strategy is based on combining classifier selection and fusion. In addition, it is shown that the overproduce-and-choose strategies reported in the literature are static selection approaches.

In Chapter 2, we investigate the search algorithm and search criteria at the selection phase, when this selection is performed as an optimization process. Single- and multi-objective GAs are used to conduct the optimization process, while fourteen objective functions are used to guide this optimization. Thus, the experiments and the results are presented and analyzed.

The overfitting aspect is addressed in Chapter 3. We demonstrate the circumstances under which the process of classifier ensemble selection results in overfitting. Then, three strategies used to control overfitting are introduced. Finally, experimental results are presented.

In Chapter 4 we present our proposed dynamic overproduce-and-choose strategy. We describe the optimization and the dynamic selection levels performed in the two-level selection phase by population-based GAs and confidence-based measures respectively. Then, the experiments and the results obtained are presented. Finally, our conclusions and suggestions for future work are discussed.

CHAPTER 1

LITERATURE REVIEW

Learning algorithms are used to solve tasks for which the design of software using traditional programming techniques is difficult. Machine failures prediction, filter for electronic mail messages and handwritten digits recognition are examples of these tasks. Several different learning algorithms have been proposed in the literature such as Decision Trees, Neural Networks, k Nearest Neighbors (kNN), Support Vector Machines (SVM), etc. Given sample x and its class label ω_k with an unknown function $\omega_k = f(x)$, all these learning algorithms focus on finding in the hypothesis space H the best approximation function h, which is a classifier, to the function $f(x)$. Hence, the goal of these learning algorithms is the design of a robust well-suited single classifier to the problem concerned.

Classifier ensembles attempt to overcome the complex task of designing a robust, well-suited individual classifier by combining the decisions of relatively simpler classifiers. It has been shown that significant performance improvements can be obtained by creating classifier ensembles and combining their classifier members' outputs instead of using single classifiers. Altinçay [2] and Tremblay et al. [89] showed that ensemble of kNN is superior to single kNN; Zhang [105] and Valentini [93] concluded that ensemble of SVM outperforms single SVM and Ruta and Gabrys [71] demonstrated performance improvements by combining ensemble of Neural Networks, instead of using a single Neural Network. Moreover, the wide applicability of ensemble-of-classifier techniques is important, since most of the learning techniques available in the literature may be used for generating classifier ensembles. Besides, ensembles are effective tools to solve difficulty Pattern Recognition problems such as remote sensing, person recognition, intrusion detection, medical applications and others [54].

According to Dietterich [16] there are three main reasons for the improved performance verified using classifier ensembles: (1) *statistical*; (2) *computational*; and (3) *representational*. Figure 1 illustrates these reasons. The first aspect is related to the problem that arises when a learning algorithm finds different hypotheses h_i, which appear equally accurate during the training phase, but chooses the less competent hypothesis, when tested in unknown data. This problem may be avoided by combining all classifiers. The computational reason refers to the situation when learning algorithms get stuck in local optima, since the combination of different local minima may lead to better solutions. The last reason refers to the situation when the hypothesis space H does not contain good approximations to the function $f(x)$. In this case, classifier ensembles allow to expand the space of functions evaluated, leading to a better approximation of $f(x)$.

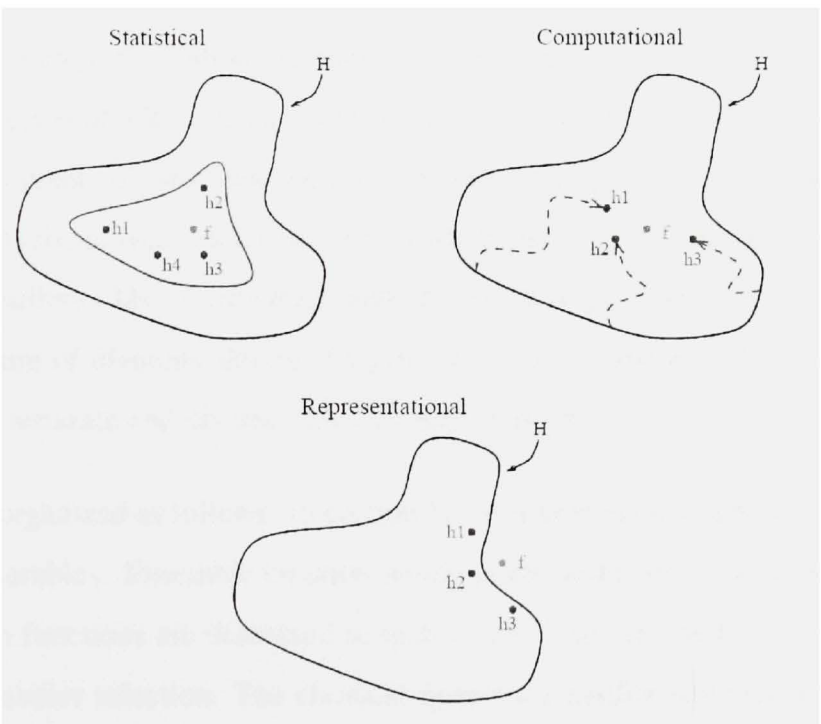


Figure 1 The statistical, computational and representational reasons for combining classifiers [16].

However, the literature has shown that diversity is the key issue for employing classifier ensembles successfully [44]. It is intuitively accepted that ensemble members must be different from each other, exhibiting especially diverse errors [7]. However, highly accurate and reliable classification is required in practical machine learning and pattern recognition applications. Thus, ideally, ensemble classifier members must be accurate and different from each other to ensure performance improvement. Therefore, the key challenge for classifier ensemble research is to understand and measure diversity in order to establish the perfect trade-off between diversity and accuracy [23].

Although the concept of diversity is still considered an ill-defined concept [7], there are several different measures of diversity reported in the literature from different fields of research. Moreover, the most widely used ensemble creation techniques, bagging, boosting and the random subspace method are focused on incorporating the concept of diversity into the construction of effective ensembles. Bagging and the random subspace method implicitly try to create diverse ensemble members by using random samples or random features respectively, to train each classifier, while boosting try to explicitly ensure diversity among classifiers. The overproduce-and-choose strategy is another way to explicitly enforce a measure of diversity during the generation of ensembles. This strategy allows the selection of accurate and diverse classifier members [69].

This chapter is organized as follows. In section 1.1, it is presented a survey of construction of classifier ensembles. Ensemble creation methods are described in section 1.1.1, while the combination functions are discussed in section 1.1.2. In section 1.2 it is presented an overview of classifier selection. The classical dynamic classifier selection is discussed in section 1.2.1; the overproduce-and-choose strategy is presented in section 1.2.2; and the problem of overfitting in overproduce-and-choose strategy is analysed in section 1.2.3. Finally, section 1.3 presents the dicussion.

1.1 Construction of classifier ensembles

The construction of classifier ensembles may be performed by adopting different strategies. One possibility is to manipulate the classifier models involved, such as using different classifier types [70], different classifier architectures [71] and different learning parameters initialization [2]. Another option is varying the data, for instance using different data sources, different pre-processing methods, different sampling methods, distortion, etc. It is important to mention that the generation of an ensemble of classifiers involves the design of the classifiers members and the choice of the fusion function to combine their decisions. These two aspects are analyzed in this section.

1.1.1 Strategies for generating classifier ensembles

Some authors [77; 7] have proposed to divide the ensemble creation methods into different categories. Sharkey [77] have shown that the following four aspects can be manipulated to yield ensembles of Neural Networks: initial conditions, training data, topology of the networks and the training algorithm. More recently, Brown et. al. [7] proposed that ensemble creation methods may be divided into three groups according to the aspects that are manipulated. (1) *Starting point in hypothesis space* involves varying the start points of the classifiers, such as the initial random weights of Neural Networks. (2) *Set of accessible hypothesis* is related to varying the topology of the classifiers or the data, used for training ensemble's component members. Finally, (3) *traversal of hypothesis space* is focused on enlarging the search space in order to evaluate a large amount of hypothesis using genetic algorithms and penalty methods, for example. We present in this section the main ensemble creation methods divided into five groups. This categorization takes into account whether or not one of the following aspects are manipulated: training examples, input features, output targets, ensemble members and injecting randomness.

Manipulating the Training Examples

This first group contains methods, which construct classifier ensembles by varying the training samples in order to generate different datasets for training the ensemble members. The following ensemble construction methods are examples of this kind of approach.

- *Bagging* - It is a bootstrap technique proposed by Breiman [5]. Bagging is an acronym for *B*ootstrap *Agg*regation Learn*ing*, which builds n replicate training datasets by randomly sampling, with replacement, from the original training dataset. Thus, each replicated dataset is used to train one classifier member. The classifiers outputs are then combined via an appropriate fusion function. It is expected that 63,2% of the original training samples will be included in each replicate [5].

- *Boosting* - Several variants of boosting have been proposed. We describe here the Adaboost (short for *Ada*ptive *Boost*ing) algorithm proposed by Freund and Schapire [21], which appears to be the most popular boosting variant [54]. This ensemble creation method is similar to bagging, since it also manipulates the training examples to generate multiple hypotheses. However, boosting is an iterative algorithm, which assigns weights to each example contained in the training dataset and generates classifiers sequentially. At each iteration, the algorithm adjusts the weights of the misclassified training samples by previous classifiers. Thus, the samples considered by previous classifiers as difficult for classification, will have higher chances to be put together to form the training set for future classifiers. The final ensemble composed of all classifiers generated at each iteration is usually combined by majority voting or weighted voting.

- *Ensemble Clustering* - It is a method used in unsupervised classification that is motivated for the success of classifier ensembles in the supervised classification context. The idea is to use a partition generation process to produce different clusters. Afterwards, these partitions are combined in order to produce an improved solution.

Diversity among clusters is also important for the success of this ensemble creation method and several strategies to provide diversity in cluster algorithms have been investigated [26].

It is important to take into account the distinction between *unstable* or *stable* classifiers [42]. The first group is strongly dependent on the training samples, while the second group is less sensitive to changes on the training dataset. The literature has shown that unstable classifiers, such as Decision Trees and Neural Networks, present high variance, which is a component of the bias-variance decomposition of the error framework [19]. Consequently, stable classifiers like kNN and Fischer linear discriminant present low variance. Indeed, one of the advantages of combining individual classifiers to compose one ensemble is to reduce the variance component of the error [59]. Thus, due to the fact that boosting is assumed to reduce both bias and variance [19], this ensemble generation method is efficient using both stable and unstable classifiers. On the other hand, bagging is mostly effective with unstable classifiers, since bagging is assumed to reduce variance [94].

Manipulating the Input Features

These methods construct classifier ensembles manipulating the original set of features available for training. The objective is to provide a partial view of the training dataset to each ensemble member, leading them to be different from each other. In addition, these methods try to reduce the number of features to fight the effects of the so-called *curse of dimensionality* problem [90].

- *Feature Subset Selection* - The objective of ensemble feature selection is to build sets of classifiers using small subsets of features whilst keeping high accurate classifiers, as was done in [51]. An interesting overview about several techniques used to create ensemble feature selection is presented in [90].

- *The Random Subspace Method* - This method introduced by Ho in [32] is considered to be a feature subset selection approach. It works by randomly choosing n different subspaces from the original feature space. Each random subspace is used to train one individual classifier. The n classifiers are usually combined by the majority voting rule. Although the random subspace method is supposed to reduce variance [94], it is assumed be an efficient method for building ensembles using both stable and unstable classifiers.

Manipulating the Output Targets

This group contains methods, which are based on manipulating the labels of the samples contained in the training dataset. In the error-correcting output coding technique used by Dietterich and Bakiri [18], a multi-class problem is transformed into a set of binary problems. At each iteration a new binary division of the training dataset is used to train a new classifier. Another example is the method proposed by Breiman [6], which introduces noise to change some class labels of the training samples.

Manipulating the Ensemble Members (Heterogeneous Ensembles)

These methods work by using different classifier types [70], different classifier architectures [69] or different initializations of the learning parameters [107], whilst maintaining the same training dataset. For instance, Valentini and Dietterich [95] used an ensemble of SVM with kernel RBF (radial basis functions) in which the classifier members were trained using different parameters σ. Ruta and Gabrys [70] employed 15 different learning algorithms, including Quadratic discriminant, Radial Basis Network, k-NN, Decision Tree, and others, in order to compose an ensemble of 15 heterogeneous classifiers.

Injecting randomness

This group contains methods, which inject randomness into the classifier members to produce different classifiers in order to build ensembles. The random initial weights of Neural Networks [78] and the random choice of the feature that decides the split at the internal nodes of Decision Tree [17], are examples of randomness injected into classifier members.

1.1.2 Combination Function

The application of one of the above mentioned ensemble creation methods generates an initial set of classifiers $\mathcal{C}$, where $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$. Figure 2 shows that the ensemble generation method is employed using samples $\mathbf{x}_{i,t}$ contained in the training dataset $\mathcal{T}$. Given such a pool of classifiers, the most common operation is the fusion of all n classifiers. Thus, an effective way of combining the classifier members' outputs must be found. Even though some classification techniques, such as Neural Networks [97] and Polynomial classifiers [20], have been used to combine classifier members, there are many different classifier fusion functions proposed. In this section we present a brief description of some of the most widely used fusion functions.

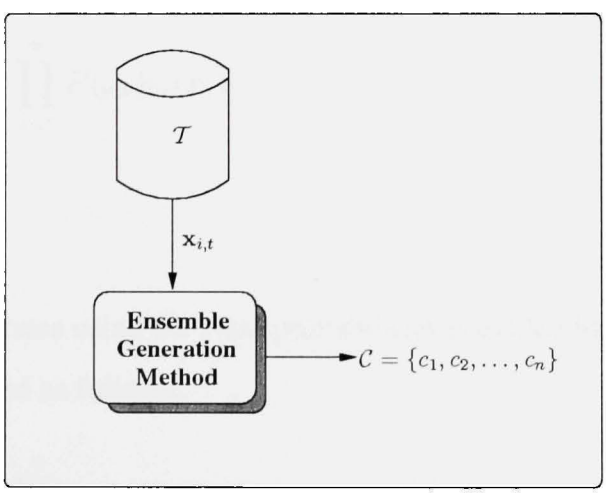


Figure 2 Overview of the creation process of classifier ensembles.

Majority Voting

It is the simplest and most popular method to combine classifiers. The definition presented in Equation 1.1 is also called *Plurality vote* [40]. Considering the set of n classifiers, y_i as the class label output of the i-th classifier, and a classification problem with the following set of class labels $\Omega = \{\omega_1, \omega_2 \ldots, \omega_c\}$, majority voting for sample x is calculated as:

$$mv(x) = \max_{k=1}^{c} \sum_{i=1}^{n} y_{i,k} \tag{1.1}$$

When there is a tie for the number of votes, it may be broken randomly or a rejection strategy must be performed. There are other versions of vote, such as unanimous consensus, weighted voting, etc [40].

Product Rule

It is a simple combination function that is calculated taking into account outputs of classifiers c_i provided as class probabilities $P(\omega_k|y_i(x))$, denoting that the class label of sample x is ω_k if classifier c_i assigns the class label output y_i. The product rule is computed as:

$$pr(x) = \max_{k=1}^{c} \prod_{i=1}^{n} P(\omega_k|y_i(x)) \tag{1.2}$$

Sum Rule

This function also operates using the class probabilities provided by classifier members c_i. The decision is obtained as follows:

$$sr(x) = \max_{k=1}^{c} \sum_{i=1}^{n} P(\omega_k|y_i(x)) \tag{1.3}$$

Maximum Rule

It is possible to approximate the combination functions product (Equation 1.2) and sum (Equation 1.3) by its upper or lower bounds. If the sum is approximated by the maximum of the class probabilities, max rule is obtained as:

$$max(x) = \max_{k=1}^{c} \max_{i=1}^{n} P(\omega_k|y_i(x)) \tag{1.4}$$

Minimum Rule

This function is obtained through approximating the product rule of class probabilities by the minimum:

$$min(x) = \max_{k=1}^{c} \min_{i=1}^{n} P(\omega_k|y_i(x)) \tag{1.5}$$

Naive Bayes

This method, also called Bayesian combination rule [86], assumes that classifier members are mutually independent and operates on the confusion matrix of each classifier member c_i. The objective is to take into account the performance of each classifier c_i, for each class involved in the classification problem, over samples contained in dataset $\mathcal{T}$. Let $\hat{P}(\omega_t|y_i(x) = \omega_k)$ be an estimated of the probability that the true class label of sample x is ω_t if classifier c_i assigns as output the class label ω_k. The probability $\hat{P}(\omega_t|y_i(x) = \omega_k)$ is computed as the ratio between the number of training samples assigned by classifier c_i to class ω_k, whose true class label is ω_t, and the total number of training samples assigned by c_i to class ω_k. Thus, Naive Bayes classifies new samples using these estimated probabilities as follows:

$$nb_k(x) = \prod_{i=1}^{n} P(\omega_t|y_i(x) = \omega_k) \tag{1.6}$$

Naive Bayes decision rule selects the class with the highest probability computed by the estimated probabilities in Equation 1.6.

Dempster-Shafer

This combination method is based on belief functions. Given the set of class labels $\Omega = \{\omega_1, \omega_2 \ldots, \omega_c\}$, the set Ω has subsets which are known as propositions. The set of propositions is denoted as P. Letting A and B denote two arbitrary class sets where $A \in P$ and $B \in P$, a basic probability assignment (bpa) is assigned to each proposition, for instance $bpa(A)$ represents the output of the classifier on A. Thus, the basic probabilities assigned to all subsets of A are added in order to calculate a numeric value in the range $[0, 1]$ that indicates the belief in proposition, as follows:

$$bel(A) = \sum_{B \subseteq A} bpa(A) \tag{1.7}$$

Given the sets A and B with two different basic probabilities, $bpa_1(A)$ for one classifier and $bpa_2(B)$ for the other classifier, C indicates the basic probability of their conjunction $C = A \cap B$, which is proportional to $bpa_1(A)$ x $bpa_2(B)$. Finally, the classifiers are then combined by using the Dempster-Shafer rule as:

$$bpa_1 \oplus bpa_2(C) = \frac{\sum_{A \cap B = C} bpa_1(A) bpa_2(B)}{1 - \sum_{A \cap B = \emptyset} bpa_1(A) bpa_2(B)} \tag{1.8}$$

The fusion functions described in this section are few examples of the variety of classifier combination rules reported in the literature. Other approaches to combine ensemble classifier members include Behavior-Knowledge Space (BKS) [33], Decision Templates [41] and Wernecke's method [98].

Classifier fusion assumes error independence among ensemble's component members. This means that the classifier members are supposed to misclassify different patterns [43].

In this way, the combination of classifier members' decision will improve the final classification performance. However, when the condition of independence is no verified, there is no guarantee that the combination of classifiers will outperform single classifiers [83]. On the one hand, it is difficult to impose independence among ensemble's component members. On the other, the selection of classifiers has focused on finding either the most efficient individual classifier or the best subset of classifiers, rather than combining all available n classifiers. Therefore, classifier selection techniques avoid the assumption of independence. Classifier selection is described in the next section.

1.2 Classifier Selection

It was mentioned in the introduction that classifier selection is traditionally defined as a strategy that assumes each ensemble member as an expert in some regions of competence [55; 107]. The selection is called dynamic or static whether the regions of competence are defined during the test or the training phase respectively. However, several methods reported in the literature as dynamic classifier selection methods define regions of competence during the training phase [107; 81; 79]. These strategies are discussed in section 1.2.1. The overproduce-and-choose strategy, which is the classifier ensemble selection technique studied in this thesis, is classically assumed to be a static classifier selection method [69]. In section 1.2.2 we present an overview of the overproduce-and-choose strategy.

1.2.1 Dynamic Classifier Selection

Classical *dynamic classifier selection* (DCS) methods are divided into three levels, as illustrated in Figure 3. The first level, called *classifier generation* employs one of the ensemble creation methods presented in section 1.1.1, using the samples $\mathbf{x}_{t,g}$ contained in the training dataset $\mathcal{T}$, to obtain classifiers to compose the initial pool of classifiers $\mathcal{C}$. However, rather than combining all available n classifiers using one of the fusion functions mentioned in section 1.1.2, the second level called *region of competence generation*, uses $\mathcal{T}$

or an independent validation dataset $\mathcal{V}$ to produce regions of competence R_j. Finally, *Dynamic Selection* chooses a winning partition R_j^* and the winning classifier c_i^*, over samples contained in R_j^*, to assign the label ω_k to the sample $\mathbf{x}_{i,g}$ from the test dataset $\mathcal{G}$.

Theoretically, level 2 and level 3 are performed during the test phase in DCS methods, i.e. based on samples $\mathbf{x}_{i,g}$. Due to the high computing complexity of estimating regions of competence dynamically [40], several DCS methods preestimate regions of competence during the training phase [107; 81; 79], and perform only the third level during the test phase. Thus, the term DCS will hereafter be used to refer to approaches, which assign label ω_k taking into account the test samples, whatever the phase in which the regions of competence are generated.

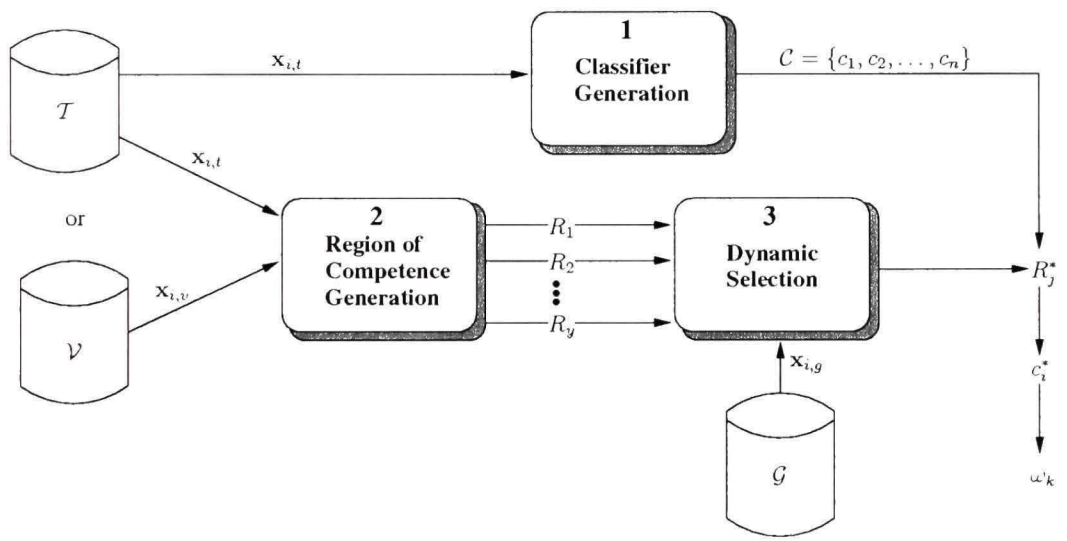


Figure 3 The classical DCS process: DCS is divided into three levels focusing on training individual classifiers, generating regions of competence and selecting the most competent classifier for each region.

The main difference between the various DCS methods is the partition generation strategy employed. K nearest neighbors [101], clustering [38] and various training datasets [81] are examples of techniques used. In DCS-LA (*Dynamic Classifier Selection with Local Accuracy*), proposed by Woods [101], the first level generates a population of five het-

erogeneous classifiers through feature subset selection. The algorithm defines the local region R_j^* as the set of k nearest neighbors from $\mathcal{T}$ surrounding $\mathbf{x}_{i,g}$. Thus, at the third level, the local accuracy of each classifier c_i is estimated, and the most locally accurate classifier c_i^* is then selected to estimate the true class of $\mathbf{x}_{i,g}$. Giacinto and Roli [24] proposed an approach very similar to Woods' method. The difference is that the local region used to estimate the individual performances of each c_i is defined as the nearest neighbors from $\mathcal{V}$ that have a similarity with $\mathbf{x}_{i,g}$ that is higher than a threshold. Such a similarity is measured by comparing the vector of class labels assigned by each c_i to $\mathbf{x}_{i,g}$ and to its neighbors.

In the clustering and selection method proposed by Kuncheva [38], *multilayer perceptrons* (MLPs) with different number of nodes in the hidden layer compose $\mathcal{C}$. Thus, at the second level, the feature space is partitioned into clusters using K means, and cluster centroids are computed. At the third level, the region with a cluster center nearest to $\mathbf{x}_{i,g}$ is picked up as R_j^* and the c_i with the highest classification accuracy is nominated to label $\mathbf{x}_{i,g}$. In the DCS method employed by Sohn and Shin [85], $\mathcal{T}$ is first divided into n clusters, cluster centroids are computed and each cluster is used to train one classifier. The base algorithm was a logistic model. The second level and the third level are conducted as in Kuncheva's method [38]. Liu et al. [45] presented a clustering and selection-based method that first generates three heterogeneous classifiers to compose $\mathcal{C}$. At the second level, $\mathcal{T}$ is divided into two groups for each c_i: (1) correctly classified training samples; and (2) misclassified training samples. These two groups are further partitioned using a clustering algorithm to compose regions of competence, i.e. each c_i has its own. At the dynamic selection level, the cluster closest to $\mathbf{x}_{i,g}$ from each of c_i's regions of competence is pointed out and the most accurate classifier is chosen to assign $\mathbf{x}_{i,g}$'s label.

Singh and Singh [81] described a DCS method for image region labeling in which the first level generates $\mathcal{C}$ by training each c_i (kNN classifiers) with n different training datasets which are obtained through applying n different texture analysis methods. The regions of

competence are defined at the second level as the class centroids of each training dataset. The selection level measures the distance between $\mathbf{x}_{i,g}$ and all the class centroids. Then, the c_i responsible by the closest region is selected to classify $\mathbf{x}_{i,g}$. In [107], a DCS method for data stream mining applications is proposed. For the first level, $\mathcal{T}$ is divided into n chunks which are further used to train the n DT classifiers that compose $\mathcal{C}$. Local regions are generated with statistical information on the attribute values of samples from $\mathcal{V}$. Finally, at the third level, the most accurate classifier in the region sharing the same statistical information on attribute values with $\mathbf{x}_{i,g}$ is selected to label it.

It is important to mention that all these methods pick up only one candidate classifier to make the decision. This may lead to a classifier with a low level of confidence in its decision, or even one with a wrong decision, being chosen. A combination of selection and fusion has been investigated in the literature as a strategy for avoiding this drawback. Kuncheva [39] proposed to use statistical tests to switch between selection and fusion. The classifier c_i^*, selected using clustering and selection [38], is employed to label $\mathbf{x}_{i,g}$ only when it is significantly better than the remaining classifiers. Otherwise, all classifiers in $\mathcal{C}$ are combined through decision templates. Gunes et al. [28] applied a fuzzy clustering algorithm in combination with ambiguity rejection in order make it possible to deal with overlapping regions of competence. They switch between classifying $\mathbf{x}_{i,g}$ using either c_i^* or the combination of the best adapted classifiers whether $\mathbf{x}_{i,g}$ falls into a single cluster or into an ambiguous cluster, respectively. The k-nearest-oracles (KNORA) method proposed by Ko et al. [36] explores the properties of the oracle concept [15] to select the most suitable classifier ensemble for each test sample. KNORA first finds the set of k nearest neighbors from $\mathcal{V}$ surrounding $\mathbf{x}_{i,g}$. Then the algorithm selects each classifier c_i, which correctly classifies this set of neighbors, to compose a classifier ensemble. This selected classifier ensemble is then used for classifying $\mathbf{x}_{i,g}$. In the method used by Tsymbal et al. [92], called Dynamic Voting with Selection (DVS), a weight is determined for each classifier c_i according to its local accuracy measured using the set of k nearest neighbors

from $\mathcal{V}$ surrounding $\mathbf{x}_{i,g}$. Then, the classifiers with the lowest accuracy are removed, while the remaining classifiers are combined through weighted voting to assign the class of $\mathbf{x}_{i,g}$. Finally, Soares et al. [84] have tailored Kuncheva's method [39] to select candidate classifiers based on accuracy and diversity.

Table I summarizes the DCS methods reported in the literature and mentioned in this section. It is interesting to note that heterogeneous classifiers at the first level, clustering at the second level and accuracy as a selection criterion at the third level are most often applied. An alternative to DCS is the overproduce-and-choose strategy, which allows the selection of classifier ensembles instead of only one classifier. Hence, this strategy is also based on combining selection and fusion, as it is explained in the next section.

Table I

Compilation of some of the results reported in the DCS literature highlighting the type of base classifiers, the strategy employed for generating regions of competence, and the phase in which they are generated, the criteria used to perform the selection and whether or not fusion is also used (Het: heterogeneous classifiers).

Reference Number	**Classifier Members**	**Regions of Competence**	**Partition Phase**	**Selection Criteria**	**Selection /fusion**
[107]	DT	Blocks of samples	Training	Accuracy	Selection
[24]	Het	kNN rule	Test	Accuracy	Selection
[101]	Het	kNN rule	Test	Accuracy	Selection
[81]	kNN	Different features	Training	Distance measure	Selection
[38]	MLP	Clustering	Training	Distance & accuracy	Selection
[85]	Logistic	Clustering	Training	Distance & accuracy	Selection
[45]	Het	Clustering	Training	Distance & accuracy	Selection
[39]	MLP/Het	Clustering	Training	Distance & accuracy	Selection or fusion
[28]	Bayesian	Clustering	Training	Distance measure	Selection or fusion
[36]	kNN	kNN rule	Test	Oracle	Selection & fusion
[92]	Het	kNN rule	Test	Accuracy	Selection & fusion
[84]	Het	Clustering	Training	Accuracy & diversity	Selection & fusion

1.2.2 Overproduce-and-Choose Strategy

Given the pool of classifiers $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ generated using any ensemble creation method, classifier fusion combines the n classifiers assuming that they are all important and independent. By contrast, classifier selection, especially DCS, selects one individual classifier c_i^* to assign the label of each sample contained in the test dataset $\mathcal{G}$. A combination of both approaches is the *overproduce-and-choose strategy* (OCS). The objective of OCS is to find the most relevant subset of classifiers, based on the assumption that classifiers in $\mathcal{C}$ are redundant [106]. Once the best subset of classifiers has been selected, the output of its classifier members must be combined.

Methods based on OCS are divided into two phases: (1) *overproduction*; and (2) *selection*. The first phase is related to the first level of DCS, here however, the overproduction phase must construct an initial *large* pool of candidate classifiers $\mathcal{C}$, using the training dataset $\mathcal{T}$. The second phase is devoted to identify the best performing subset of classifiers in $\mathcal{P}(\mathcal{C})$. As mentioned in the introduction, $\mathcal{P}(\mathcal{C})$ is the powerset of $\mathcal{C}$ defining the population of all possible candidate ensembles C_j. The selected ensemble C_j^* is then combined to estimate the class labels of the samples contained in the test dataset $\mathcal{G}$. Figure 4 illustrates the OCS phases.

OCS based in heuristic techniques has been proposed in the literature. Margineantu and Dietterich [48] proposed an OCS to reduce the computational costs of boosting. The pool $\mathcal{C}$ was composed of homogeneous DT generated by boosting. Then, pruning algorithms were applied to select each classifier c_i used to form C_j^*. The authors pointed out a diversity-based pruning algorithm, which used Kappa diversity, as the best pruning method. Partridge and Yates [58] generated the pool $\mathcal{C}$ using two types of Neural Networks, MLP and Radial Basis Function (RBF). The Neural Networks were trained using different number of hidden units and weight initialization. They proposed the use of two heuristics during the selection phase. The first heuristic relies on ranking the candidate

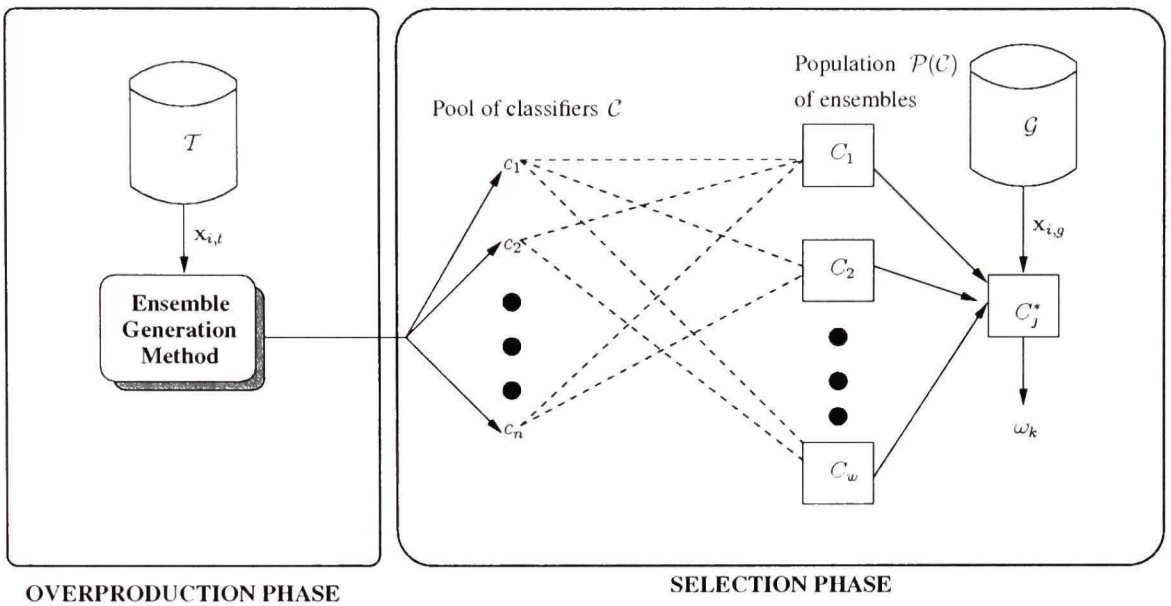


Figure 4 Overview of the OCS process. OCS is divided into the overproduction and the selection phases. The overproduction phase creates a large pool of classifiers, while the selection phase focus on finding the most performing subset of classifiers.

classifiers by its performance and selecting the k best to compose C_j^*, where $k < n$. The second heuristic, picks the classifier with the highest performance, from each type of classifier, to compose C_j^*. In the same light, Canuto et al. [9] generated $\mathcal{C}$ using heterogeneous classifiers such as MLP, RBF, SVM, kNN, and others, and tested different fixed sizes of classifier ensembles. Their selection phase took into account both performance and diversity of each candidate ensemble generated. Aksela and Laaksonen [1] also generated a pool $\mathcal{C}$ of heterogeneous classifiers at the overproduction phase. Neural Networks, SVM, etc, are examples of classifiers used. The authors successfully applied an OCS for selecting classifiers focusing on the diversity of errors. The candidate ensembles were also generated using fixed ensembles' size.

Although these heuristic-based approaches lead to reduce the complexity of the selection phase, there is no guarantee that the optimal solution will be found. An alternative to heuristic-based OCS is to use search algorithms. When performing OCS using search

algorithms, it is important to choose the best search criterion and the best search algorithm for the classifier ensemble selection problem. Sharkey and Sharkey [78] proposed an exhaustive search algorithm to find C_j^* from an initial pool $\mathcal{C}$ composed of MLPs. The classifiers were trained using different number of hidden units and different random initial conditions. Thus, candidate ensembles' performances were used to guide the selection phase. Based on the same idea, Zhou et al. [107] developed equations, which were used to identify the classifiers that should be eliminated from $\mathcal{C}$ in order to keep the combination with optimal accuracy. Their pool $\mathcal{C}$ was composed of different Neural Networks. Nonetheless, non-exhaustive search algorithms might be used when a large $\mathcal{C}$ is available due to the high computing complexity of an exhaustive search, since the size of $\mathcal{P}(\mathcal{C})$ is 2^n.

Several non-exhaustive search algorithms have been applied in the literature for the selection of classifier ensembles. Zhou et al. [107] proposed GANSEN (GA-based Selective Ensemble), which employs GAs to assign random weight to each Neural Network and evolves these weights by assessing the performance of the combination obtained when each Neural Network is included in the ensemble. Then, each Neural Network whose weight is higher than a fixed threshold is used to compose C_j^*. Roli et al. [69] compared forward (FS), backward (BS) and tabu (TS) search algorithms, guided by accuracy and the following three diversity measures: generalized diversity (τ), Q-statistic [44] (Φ) and double-fault (δ), called by the authors compound diversity. Their initial pool $\mathcal{C}$ was composed of heterogeneous classifiers such as MLP, kNN and RBF. They concluded that the search criteria and the search algorithms investigated presented equivalent results. Following the idea of identifying the best search criteria and search algorithm, Ruta and Gabrys [70], used the candidate ensembles' error rate and 12 diversity measures, including the same measures investigated by Roli et al. [69], to guide the following 5 single-objective search algorithms: FS, BS, GA, stochastic hill-climbing (HC) search and population-based incremental learning (PBIL). They concluded that diversity was not a better measure for

finding ensembles that perform well than the candidate ensembles' error rate. Their initial pool of candidate classifiers was composed of 15 heterogeneous classifiers, including MLP, kNN, RBF, Quadratic Bayes, and others.

The combination of the error rate and diversity as search criteria allows the simultaneous use of both measures in OCS. It is not surprising that this idea has already been investigated in the literature. Opitz and Shavlik [53] applied a GA using a single-objective function combining both the error rate and ambiguity diversity measure (as defined in [53]) to search for C_j^* in a population $\mathcal{C}$ of Neural Networks. They showed that this OCS outperformed the combination of all classifiers in $\mathcal{C}$. Zenobi and Cunningham [104] created at the overproduction phase a pool $\mathcal{C}$ of kNN classifiers by applying a feature subset selection approach. At the selection phase, ambiguity (as defined in [104]) and the error rate were used to guide a hill-climbing search method. They showed that their combined approach outperformed the ensembles selected using the error rate as the only objective function. Tremblay et al. [89] used a MOGA (a modified version of Non-dominated Sorting GA - NSGA [11]) guided by pairs of objective functions compose of the error rate with the following four diversity measures: ambiguity [104] (γ), fault majority [70] (λ), entropy (ξ) and Q-statistic (Φ) [44]. They generated a pool $\mathcal{C}$ of kNN classifiers generated by the random subspace method at the overproduction phase. They concluded that MOGA did not find better ensembles than single-objective GA using only the error rate as the objective function. Finally, in the OCS proposed by Oliveira et al. [51], a feature subset selection process was applied to generate a population of Neural Networks at the overproduction phase. The selection phase was also guided by γ and the error rate as the objective functions.

It is important to mention that, these previous works on OCS have one characteristic in common: the solution C_j^* assumed to be the best candidate ensemble, found and analyzed during the selection phase, is used to classify all samples contained in $\mathcal{G}$. Due to this characteristic, we call this method *static* (SOCS). However, as mentioned in the introduction,

there is no guarantee that the C_j^* chosen is indeed the solution most likely to be the correct one for classifying each test sample $\mathbf{x}_{i,g}$ individually. In Chapter 4, we propose a dynamic OCS to avoid this drawback.

Table II presents some of the results reported in the literature dealing with OCS. It is important to see that heterogeneous classifiers at the overproduction phase and GA guided by the error rate at the choice phase, are most often applied. Moreover, the error rate is the search criterion most frequently pointed out as the best evaluation function for selecting C_j^*. These results show that the definition of the best search algorithm and the best search criterion for the selection phase of OCS is still an open issue. Moreover, much less work has been devoted to combining different search criteria in a multi-objective selection phase. These issues are addressed in Chapter 2 of this thesis. In addition, as shown in Table II, most of the search algorithms employed in OCS are stochastic search algorithms, such as GA, HC and PBIL. However, taking into account that the problem of selecting classifier ensembles can be assumed as a learning task, the literature has shown that stochastic search algorithms are prone to overfitting when used in conjunction with Machine Learning techniques. In next section, the problem of overfitting in OCS is described.

Table II

Compilation of some of the results reported in the OCS literature (FSS: Feature Subset Selection, and RSS: Random Subspace.

Reference number	Classifier Members	Ensemble Method	Search criterion	Search algorithm	Best Search criteria
[107]	NN	NN Het	Error rate	GA	-
[58]	NN	NN Het	Error rate	N best	-
[78]	NN	NN Het	Error rate	Exhaustive	-
[70]	Het	Het	Error rate and diversity(12)	GA, FS, BS, HC and PBIL	Error rate
[9]	Het	Het	δ and ξ	N best	No best
[69]	Het	Het	Error rate, τ, ξ and δ	FS, BS and TS	-
[48]	DT	Boosting	Error rate, Kappa	Pruning	Kappa
[1]	Het	Het	Diversity (11)	N best	-
[51]	NN	FSS	A and Error rate	NSGA	-
[89]	kNN	RSS	Error rate, γ, λ, δ and ξ	NSGA	Error rate

1.2.3 Overfitting in Overproduce-and-Choose Strategy

It has been shown in the last section that the research in classifier selection has focused on the characteristics of the decision profiles of ensemble members in order to optimize performance. These characteristics are particularly important in the selection of ensemble members performed in OCS. However, the control of overfitting is a challenge in Machine Learning, and it is difficult to monitor this when creating classifier ensembles.

Overfitting is a key problem in supervised classification tasks. It is the phenomenon detected when a learning algorithm fits the training set so well that noise and the peculiarities of the training data are memorized. As a result of this, the learning algorithm's performance drops when it is tested in an unknown dataset. The amount of data used for the learning process is fundamental in this context. Small datasets are more prone to overfitting than large datasets [37], although, due to the complexity of some learning problems, even large datasets can be affected by overfitting. In an attempt to tackle this issue, several Machine Learning studies have proposed solutions, such as: regularization methods, adding noise to the training set, cross-validation and early stopping [64]. Early stopping is the most common solution for overfitting.

Moreover, overfitting in pattern recognition has attracted considerable attention in optimization applications. Llorà et al. [46] suggested the use of optimization constraints to remove the overfitted solutions over the Pareto front in an evolutionary multi-objective learning system. Wiegand et al. [100] [62] proposed global validation strategies for the evolutionary optimization of Neural Network parameters. Loughrey and Cunningham [47] presented an early-stopping criterion to control overfitting in wrapper-based feature subset selection using stochastic search algorithms such as GA and Simulated Annealing. Finally, Robilliard and Fonlupt [67] proposed "backwarding", a method for preventing overfitting in Genetic Programming.

Overfitting control methods applied to pattern recognition problems may be divided into three categories. The first contains *problem-dependent* methods, which, as the name suggests, cannot be widely reused [46; 91]. The second contains the *early stopping-based* methods [47], which can be directly used in single-objective optimization problems. The method proposed in [47], for example, relies on determining a stopping generation number by averaging the best solution from each generation over a set of 10-fold cross-validation trials. However, defining an early-stopping criterion is more difficult when a Pareto front (i.e. a set of non-dominated solutions) is involved due to the fact that comparing sets of equal importance is a very complex task. Finally, the third contains the *archive-based* methods [62; 100; 67], which have been shown to be efficient tools for tackling overfitting [100; 67] and may be widely reused in all optimization problems [62].

We show in the next Chapter that the selection phase of OCS may be formulated as an optimization problem. Hence, since it has been shown that the optimization process can generate overfitted solutions [47; 65] and considering that the creation of classifier ensembles with a high level of generalization performance is the main objective in learning problems [53], it is important to take the necessary precautions to avoid overfitting in OCS. Even though, very few work has been devoted to the control of overfitting in classifier ensemble selection tasks. Tsymbal et al. [91] suggested that using individual member accuracy (instead of ensemble accuracy) together with diversity in a genetic search can overcome overfitting. Radtke et al. [62] proposed a global validation method for multi-objective evolutionary optimization including ensemble selection. In Chapter 3 we show how overfitting can be detected at the selection phase of OCS and investigate three different archive-based overfitting control method. They are analyzed in the context of population-based evolutionary algorithms with single- and multi-objective functions.

1.3 Discussion

In this chapter we have presented a literature review of classifier ensembles related to our research. It has been observed that classifier fusion and classifier selection are the two approaches available to designing classifier ensembles. In the context of classifier fusion, different methods for generating classifier ensembles and fusion functions have been briefly described. In terms of classifier selection, the two main methods for selecting classifiers have been presented and discussed. These methods are dynamic classifier selection and overproduce-and-choose strategy.

It has been observed that the overproduce-and-choose strategy is the main topic of this thesis. We have seen that, even though several important contributions have been made in this topic, the definition of the best search criterion for finding the best subset of classifiers is still an open question. Moreover, the control of overfitting in the overproduce-and-choose strategy is often neglected. Finally, we have observed that the previous works on overproduce-and-choose strategy define a single classifier ensemble to label all samples contained in the test dataset. This leads to a drawback since the selected ensemble is not assured to be the most likely to be correct for classifying each test sample individually. We have called this strategy as static overproduce-and-choose strategy. In the next chapter we will analyze several search criteria and single- and multi-objective GA as the search algorithms in the context of static overproduce-and-choose strategy.

CHAPTER 2

STATIC OVERPRODUCE-AND-CHOOSE STRATEGY

In the previous chapter we mentioned that the search for the best subset of classifiers in SOCS is frequently conducted using search algorithms, when an initial large pool of classifiers $\mathcal{C}$ is involved. Even though there is no guarantee that a particular non-exhaustive search algorithm will find the optimal subset of classifiers, since the complete powerset $\mathcal{P}(\mathcal{C})$ is not evaluated, search algorithms are employed in order to avoid both problems, the high complexity of generating the powerset $\mathcal{P}(\mathcal{C})$ using an exhaustive enumeration and the low number of candidate ensembles generated using heuristic-based approaches [69]. It is interesting to note that, when dealing with a non-exhaustive search, the selection phase required by SOCS can be easily formulated as an optimization problem in which the search algorithm operates by minimizing/maximizing one objective function or a set of objective functions.

In Figure 5 it is shown how the selection phase may be implemented as an optimization process. In the overproduction phase, $\mathcal{T}$ is used by any ensemble creation method to generate $\mathcal{C}$. Then, the selection phase performs an optimization process conducted by the search algorithm, which calculates the objective function using samples $\mathbf{x}_{\iota,o}$ contained in an optimization dataset $\mathcal{O}$. The objective of the optimization process is to generate and test different combinations of the initial classifiers c_i in order to identify the best performing candidate ensemble.

We may categorize two general strategies for selecting classifier ensembles by performing an optimization process: (1) selection without validation; and (2) selection with validation. The first and simplest procedure relies on selecting the best candidate ensemble on the same dataset as used during the search process. There is no independent validation dataset here, but rather the optimization process is performed using a dataset for a fixed number

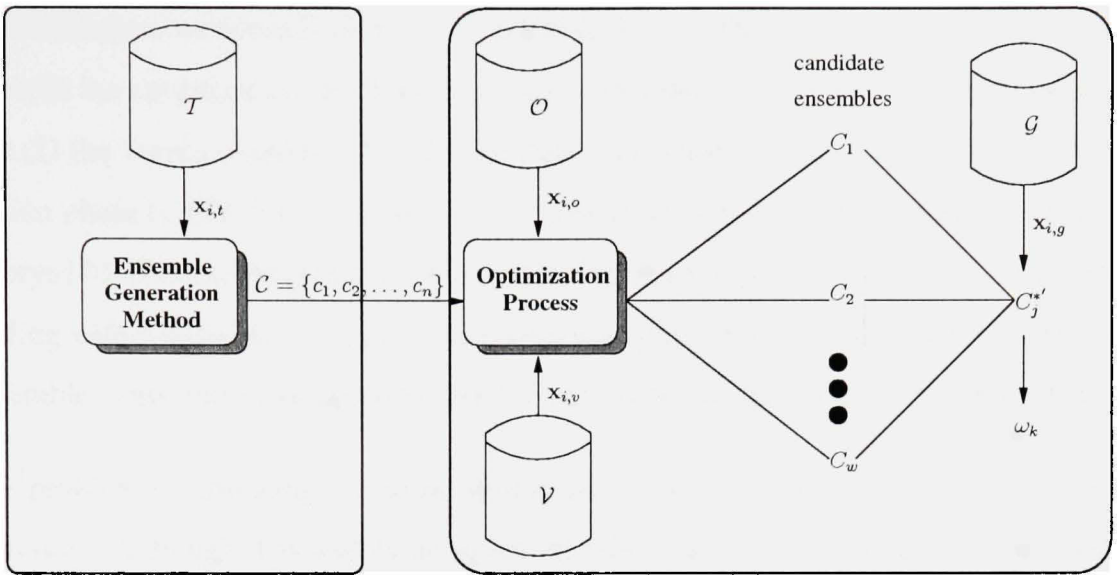


Figure 5 The overproduction and selection phases of SOCS. The selection phase is formulated as an optimization process, which generates different candidate ensembles. This optimization process uses a validation strategy to avoid overfitting. The best candidate ensemble is then selected to classify the test samples.

of generations. A population of solutions is generated and analyzed. Then, the same optimization dataset is used to identify the best performing candidate ensemble C_j^*. This procedure was applied in [90] for ensemble feature selection. However, it is well accepted in the literature that an independent data set must be used to validate selection methods in order to reduce overfitting and increase the generalization ability [78; 65].

Following this idea, in the second selection strategy when the optimization process is finished, the best solution $C_j^{*'}$ obtained on a validation dataset $\mathcal{V}$ is picked up to classify the test samples in $\mathcal{G}$. Tremblay et al. [89] have applied the second procedure in a classifier ensemble selection problem. As is illustrated in Figure 5, the second strategy is employed in this thesis. In the next chapter we describe how the validation procedure is conducted.

In this chapter, our focus is on the two important aspects that must be analyzed when dealing with the optimization process using a non-exhaustive search: (1) the search algorithm; and (2) the search criterion [70]. Evolutionary algorithms appear to fit well with the selection phase of SOCS in the context of optimization processes [82]. Moreover, Ruta and Gabrys [70] observe that population-based evolutionary algorithms allow the possibility of dealing with a population of classifier ensembles rather than one individual best candidate ensemble. This important property enabled us to propose our dynamic SOCS in chapter 5.

The problem of choosing the most appropriate search criterion is the challenge in the literature. Although it is widely accepted that diversity is an important criterion, the relationship between diversity and performance is unclear. As mentioned in the previous chapter, the combination of the classification error rate and diversity as search criteria in a multi-objective optimization approach, offers the possibility of enforcing both search criteria at the selection phase of SOCS. Finally, it can be observed that, since SOCS relies on the idea that component classifiers are redundant, an analogy can be established between feature subset selection and SOCS. Feature subset selection (FSS) approaches work by selecting the most discriminant features in order to reduce the number of features and to increase the recognition rate. Following this analogy, the selection phase of SOCS could focus on discarding redundant classifiers in order to increase performance and reduce complexity. Based on these standpoints, our objective in this chapter is to conduct an experimental study to answer the following questions:

1. Which measure is the best objective function for finding high-performance classifier ensembles?

2. Can we find better performing ensembles by including both performance and diversity as objective functions in a multi-optimization process?

3. Is it possible to establish an analogy between FSS and SOCS, i.e. can we reduce the number of classifiers while at the same time increasing performance?

In sections 2.1 and 2.2 we describe the details of both the overproduction and the selection phases. Then, a description of the parameter settings on experiments is presented in section 2.3.1. Finally, the experiments and the results obtained are presented in section 2.3.

2.1 Overproduction Phase

Random Subspace-based ensembles of kNN are used to perform the overproduction phase of the experiments carried out in this chapter. The Random Subspace (RSS) method is one of the most popular ensemble construction methods apart from Bagging and Boosting. As described in section 1.1.1, each randomly chosen subspace is used to train one individual classifier. In this way, a small number of features are used, reducing the training-time process and the so-called curse of dimensionality. Since the RSS method has the advantage of being capable of dealing with huge feature spaces, kNN appears to be a good candidate as a learner in a RSS-based ensemble. Indeed, Ho [31] maintains that, by using RSS to generate ensembles of kNN, we may achieve high generalization rates and avoid the high dimensionality problem, which is the main problem with kNN classifiers. We use in this chapter an ensemble of 100 kNN classifiers, which was generated using the RSS method. We present in section 2.3.1 details related to the set up of the parameters.

2.2 Selection Phase

We discuss in this section the search criteria and the search algorithm, which are the two main factors analyzed when dealing with the optimization process at the selection phase.

2.2.1 Search Criteria

The previous works on classifier ensemble selection summarized in Table II have one characteristic in common: performance of solutions was the only criterion used to determine whether or not one selection criterion was better than the others. Although Ruta

and Gabrys [70] mentioned the necessity of dealing with performance, complexity and overfitting in selecting classifier ensembles, they did not analyze all three aspects simultaneously. According to the authors, using the ensemble performance as the search criterion meets the requirement for high performance, while using a search algorithm addresses the complexity aspect and using a post-processing approach, such as the selection and fusion approach that they propose, may reduce overfitting. However, as mentioned before, complexity in terms of number of classifiers should also be addressed. Moreover, as it will be detailed in the next chapter, an improvement in generalized performance can be obtained by controlling overfitting during the optimization process without the need for a selection fusion method.

Hence, ensemble error rate, ensemble size and diversity measures are the most frequent search criteria employed in the literature [70]. The first, is the most obvious search criterion. By applying a search on minimizing the error rate (ϵ), we may accomplish the main objective in pattern recognition, which is to find high-performance predictors. In terms of ensemble size, the minimization of the number of classifiers, which is inspired by FSS methods through which it is possible to increase recognition rates while reducing the number of features, appears to be a good objective function. The hope is to increase the recognition rate while minimizing the number of classifiers in order to meet both the performance and complexity requirements. Finally, the important role played by diversity is clearly defined in the literature. Even though, Kuncheva and Whitaker [44] have shown that diversity and accuracy do not have a strong relationship and concluded that accuracy estimation cannot be substituted for diversity. These results were confirmed by Ruta and Gabrys [70] in the context of classifier subset selection. They used diversity measures to guide the selection of classifier ensembles in order to reduce the generalization error. They concluded that diversity is not a better measure for finding ensembles that perform well than ϵ. By contrast, Aksela and Laaksonen [1] successfully applied a method for selecting classifiers focusing on the diversity of errors. Moreover, diversity measures appear to be

an alternative to perform the optimization process without assuming a given combination function [69]. Therefore, there is no consensus about how and where to measure diversity nor which proposed diversity measure is the best one.

Various approaches defining diversity have been proposed. In this section, we describe the diversity measures used in the optimization process conducted in the experiments presented in this chapter. In order to simplify the description of the measures, we use the following notation. Let C_j be the candidate ensemble of classifiers, X the dataset, and l and n their respective cardinalities. N^{ab} denotes the number of examples classified in X, where a, b may assume the value of 1 when the classifier is correct and 0 otherwise. $r(x)$ denotes the number of classifiers that correctly classify sample x. It is worth noting that *dissimilarity* measures must be maximized, while *similarity* measures must be minimized when used as objective functions during the optimization process. The pairwise measures are calculated for each pair of classifiers c_i and c_k, while the non-pairwise measures are calculated on the whole ensemble C_j.

Ambiguity - The classification ambiguity (dissimilarity) measure proposed by Zenobi and Cunningham [104] is defined as:

$$a_i(x) = \begin{cases} 0 & \text{if } y_i = \omega_k \\ 1 & \text{otherwise} \end{cases} \tag{2.1}$$

where a_i is the ambiguity and y_i is the output of the i^{th} classifier on the observation x, and ω_k is the candidate ensemble output. The ambiguity of the ensemble is:

$$\gamma = \frac{1}{n.l} \sum_{i \in C_j} \sum_{x \in X} a_i(x) \tag{2.2}$$

Correlation Coefficient [44] - A pairwise similarity measure calculated as:

$$\rho_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{\sqrt{(N^{11}N^{10}) + (N^{01}N^{00}) + (N^{11}N^{00}) + (N^{10}N^{00})}} \tag{2.3}$$

Difficulty Measure [44] - Given F calculated from $\{\frac{0}{l}, \frac{1}{l}, \ldots, 1\}$, which represents the number of classifiers in C_j that correctly classify a pattern x, this similarity measure may be calculated as:

$$\theta = Var(F) \tag{2.4}$$

Disagreement [44] - A pairwise dissimilarity measure measured as:

$$\eta_{i,k} = \frac{N^{01} + N^{10}}{N^{11} + N^{10} + N^{01} + N^{00}} \tag{2.5}$$

Double-fault [44] - A pairwise similarity measure defined as:

$$\delta_{i,k} = \frac{N^{00}}{N^{11} + N^{10} + N^{01} + N^{00}} \tag{2.6}$$

Entropy [44] - A dissimilarity measure which can be calculated as follows:

$$\xi = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{(l - \lceil l/2 \rceil)} min\{r(x_i), l - r(x_i)\} \tag{2.7}$$

Fault Majority - A pairwise dissimilarity measure proposed by Ruta and Gabrys [70] which selects those classifiers with the highest probability of contributing to the majority

voting error of the classifier combination:

$$\lambda = \sum_{j=\lfloor l/2 \rfloor}^{l} \sum_{i^*=1}^{\lceil l/2 \rceil} x_{i^*j} \tag{2.8}$$

$$x_{i^*j} = \begin{cases} \sum_{k=1}^{n} .[1 - y_{i,k} | m(x_k) = 0]/n & \text{if j=0} \\ l. \sum_{k=1}^{n} .[1 - y_{i,k} | m(x_k) = 0]/n_j & \text{otherwise} \end{cases} \tag{2.9}$$

where $m(x_i)$ is the number of classifiers making error on observation x_i, $m(x_i) = l - \sum_{j=1}^{l} y_{i,j}$, and where $y_{i,j}$ is 1 if correct and 0 otherwise, by classifier j and example i.

Generalized Diversity [44] - A measure also based on the distribution Y defined for the coincident failure diversity. Letting $p(i)$, $p(1)$, $p(2)$ be the probability that i, 1 and 2 classifier(s) respectively, fail when classifying a sample x, we calculate:

$$p(1) = \sum_{i=1}^{l} \frac{i}{l} p_i \tag{2.10}$$

$$p(2) = \sum_{i=1}^{l} \frac{i}{l} \frac{(i-1)}{(l-1)} p_i \tag{2.11}$$

The dissimilarity generalized diversity is calculated as:

$$\tau = 1 - \frac{p(2)}{p(1)} \tag{2.12}$$

Coincident Failure Diversity [44] - The result of a modification to Generalized Diversity (Equation 2.12), which is based on the same distribution proposed for the difficulty

measure. Here, however, $Y = \frac{i}{l}$ denotes the proportion of classifiers that do not correctly classify a randomly chosen sample x. Therefore, $Y = 1 - F$ (F from the difficulty measure). It is also a dissimilarity measure, which is defined as follows:

$$\sigma = \begin{cases} 0 & p_0 = 1.0 \\ \frac{1}{(1-p_0)} \sum_{q=1}^{l} \frac{q}{l} \frac{(q-1)}{(l-1)} p_q & p_0 < 0 \end{cases} \tag{2.13}$$

Interrater Agreement [44] - A similarity measure written as:

$$\kappa = 1 - \frac{\frac{1}{l} \sum_{i=1}^{n} r(x_i)(l - r(x_i))}{n(l-1)\bar{p}(1-\bar{p})} \tag{2.14}$$

where $\bar{p}$ is the average individual accuracy:

$$\bar{p} = \frac{1}{n.l} \sum_{i=1}^{n} r(x_i) \tag{2.15}$$

Kohavi-Wolpert [44] - A dissimilarity measure calculated as:

$$\psi = \frac{1}{n.l^2} \sum_{i=1}^{n} r(x_i)(l - r(x_i)) \tag{2.16}$$

Q-Statics [44] - A pairwise measure calculated as:

$$\Phi_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}} \tag{2.17}$$

Summarizing, fourteen search criteria are used to guide the optimization process of the selection phase presented in this chapter. As shown in Table III, these search criteria comprise twelve diversity measures, plus the ensemble's combined error rate and ensemble size.

Table III

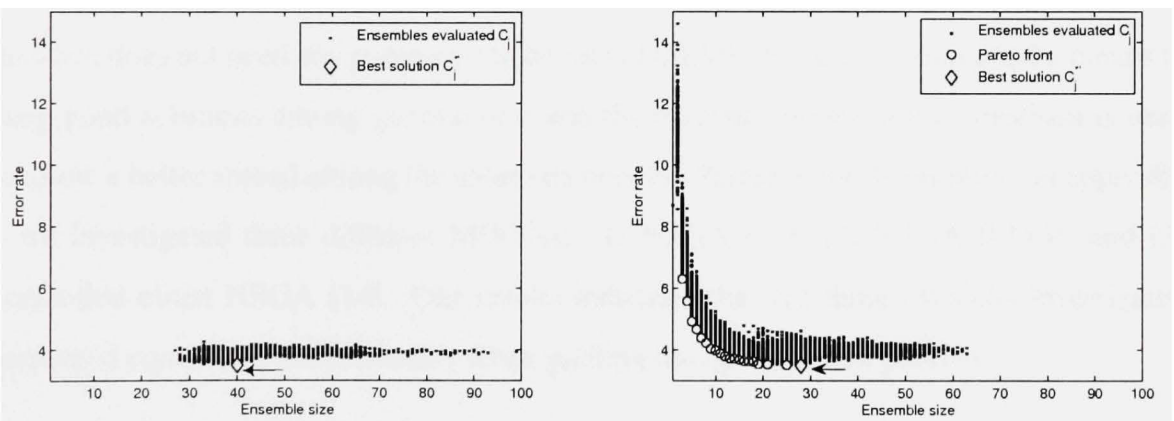
List of search criteria used in the optimization process of the SOCS conducted in this chapter. The type specifies whether the search criterion must be minimized (similarity) or maximized (dissimilarity)

Name	Label	Type
Error rate	ϵ	Similarity
Ensemble size	ζ	Similarity
Ambiguity	γ [104]	Dissimilarity
Coincident failure diversity	σ [44]	Dissimilarity
Correlation coefficient	ρ [44]	Similarity
Difficulty measure	θ [44]	Similarity
Disagreement	η [44]	Dissimilarity
Double-fault	δ [44]	Similarity
Entropy	ξ [44]	Dissimilarity
Fault majority	λ [70]	Dissimilarity
Generalized diversity	τ [44]	Dissimilarity
Interrater agreement	κ [44]	Similarity
Kohavi-Wolpert	ψ [44]	Dissimilarity
Q-statistic	Φ [44]	Similarity

2.2.2 Search Algorithms: Single- and Multi-Objective GAs

Single- and *multi-objective* GA (MOGA) are the two strategies available when dealing with GAs. Traditionally, when the optimization process is conducted as a single-objective problem, GA is guided by an objective function during a fixed maximum number of generations (user defined $max(g)$). The selection of classifier ensembles is applied in the context of GA based on binary vectors. Each individual, called chromosome, is represented by a binary vector with a size n, since the initial pool of classifiers is composed of n members. Initially, a population with a fixed number of chromosomes is randomly created, i.e. a random population of candidate classifier ensembles. Thus, at each generation step g, the algorithm calculates fitness of each candidate ensemble in the population $\mathrm{C}(g)$, which is the population of ensembles found at each generation g. The population is evolved through the operators of crossover and mutation.

Figure 6(a) depicts an example of the evolution of the optimization process for $max(g) = 1,000$ using GA as the search algorithm and the minimization of the error rate ϵ as the objective function. Even though we employed ϵ as objective function, we show in Figure 6(a) plots of ϵ versus number of classifiers ζ to better illustrate the problem. Each point on the plot corresponds to a candidate ensemble C_j taken from $\mathcal{P}(\mathcal{C})$ and evaluated during the optimization process. Indeed, these points represent the complete search space explored for $max(g) = 1,000$. The number of individuals at any $\mathbf{C}(g)$ is 128. It is important to mention that these candidate ensembles are projected onto the validation dataset in Figure 6(a), since our selection phase is conducted with validation. In SOCS, the solution with lowest ϵ is selected as the best solution $C_j^{*'}$, which is further used to classify the test samples, as shown in Figure 5. Diamond represents $C_j^{*'}$ in Figure 6(a).



(a) GA guided by ϵ (b) NSGA-II guided by ϵ and ζ

Figure 6 Optimization using GA with the error rate ϵ as the objective function in Figure 6(a). Optimization using NSGA-II and the pair of objective functions: ϵ and ensemble size ζ in Figure 6(b). The complete search space, the Pareto front (circles) and the best solution $C_j^{*'}$ (diamonds) are projected onto the validation dataset. The best performing solutions are highlighted by arrows.

MOGAs often constitute solutions to optimization processes guided by multi-objective functions. Since the combination of ϵ and diversity measures as search criteria has been investigated in the literature as a strategy to select accurate and diverse candidate ensembles

[89; 104], MOGAs allow the simultaneous use of both measures to guide the optimization process of SOCS. These algorithms use Pareto dominance to reproduce the individuals. A Pareto front is a set of nondominated solutions representing different tradeoffs between the multi-objective functions. In our classifier ensemble selection application, a candidate ensemble solution C_i is said to dominate solution C_j, denoted $C_i \preceq C_j$, if C_i is no worse than C_j on all the objective functions and C_i is better than C_j in at least one objective function. Based on this non-domination criterion, solutions over the Pareto front are considered to be equally important.

Among several Pareto-based evolutionary algorithms proposed in the literature, NSGA-II (elitist non-dominated sorting genetic algorithm) [11] appears to be interesting because it has two important characteristics: a full elite-preservation strategy and a diversity-preserving mechanism using the crowding distance as the distance measure. The crowding distance does not need any parameter to be set [11]. Elitism is used to provide the means to keep good solutions among generations, and the diversity-preserving mechanism is used to allow a better spread among the solutions over the Pareto front. In addition, in appendix 1 we investigated three different MOGAs: (1) NSGA [11]; (2) NSGA-II [13]; and (3) controlled elitist NSGA [14]. Our results indicated that the three MOGAs investigated presented equivalent performances when guiding the optimization process.

NSGA-II [11] works as follows. At each generation step g, a parent population $\mathbf{C}(g)$ of size w evolves and an offspring population $\mathbf{C}^q(g)$, also of size w, is created. These two populations are combined to create a third population $\mathbf{C}^r(g)$ of size $2w$. The population $\mathbf{C}^r(g)$ is sorted according to the nondominance criteria, and different nondominated fronts are obtained. Then, the new population $\mathbf{C}(g+1)$ is filled by the fronts according to the Pareto ranking. In this way, the worst fronts are discarded, since the size of $\mathbf{C}(g+1)$ is w. When the last front allowed to be included in $\mathbf{C}(g+1)$ has more solutions than the $\mathbf{C}(g+1)$ available free space, the crowding distance is measured in order to select the

most isolated solutions in the objective space in order to increase diversity. Algorithm 1 summarizes NSGA-II.

The optimization process performed by NSGA-II for $max(g) = 1,000$ is illustrated in Figure 6(b). NSGA-II was employed, using the pair of objective functions: jointly minimize the error rate ϵ and the ensemble size ζ. Circles on the plot represent the Pareto front. Due to the fact that all solutions over the Pareto front are equally important, the selection of the best candidate ensemble $C_j^{*'}$ is more complex. Several works reported in the literature take into account only one objective function to perform the selection. In [89], [51] and [62], the candidate ensemble with lowest ϵ was chosen as the best solution $C_j^{*'}$, even though the optimization process was guided regarding multi-objective functions. We also select the solution with lowest ϵ as $C_j^{*'}$, to classify the test samples in order to perform SOCS in this chapter. Diamond indicates the solution $C_j^{*'}$ in Figure 6(b). In chapter 5 we propose a dynamics SOCS to select the best candidate ensemble dynamically.

Algorithm 1 NSGA-II

```
Creates initial population C(1) of w chromosomes
while g < max(g) do
    creates C^q(g)
    set C^r(g) = C(g) ∪ C^q(g)
    perform a nondominated sorting to C^r(g) and identify different fronts C_k, k = 1, 2, ..., etc
    while |C(g + 1)| + |C_k| ≤ w do
        set C(g + 1):=C(g + 1) ∪ C_k
        set k:=k + 1
    end while
    perform crowding distance sort to C_k
    set C(g + 1):=C(g + 1) ∪ C_k⌊1 : (w − |C(g + 1)|)⌋
    creates C^q(g + 1) from C(g + 1)
    set g:=g + 1
end while
```

2.3 Experiments

A series of experiments has been carried out to investigate the search criteria and the search algorithm we proposed to deal with at the beginning of this chapter. Our experiments are broken down into three main series. In the first series, 13 different objective functions, including the 12 diversity measures, are applied individually as single-objective functions. In the second series, the diversity measures are used in pairs of objective functions combined with the error rate in a multi-objective approach. Finally, the third series is performed by applying pairs of objective functions combining either the diversity measures or the error rate with ensemble size. We begin our analysis taking into account performance (section 2.3.2), followed by a ensemble size analysis (section 2.3.3). It is important to mention that all the experiments were replicated 30 times and the results were tested on multiple comparisons using the Kruskal-Wallis nonparametric statistical test by testing the equality between mean values. The confidence level was 95% ($\alpha = 0.05$), and the Dunn-Sidak correction was applied to the critical values. First, we present a description of the parameters settings on experiments.

2.3.1 Parameter Settings on Experiments

The details of the parameters used in our experiments are described here. These details are related to the database, the ensemble construction method and the search algorithms.

Database

The experiments were carried out using the NIST Special Database 19 (NIST SD19) which is a popular database used to investigate digit recognition algorithms. It is composed of 10 digit classes extracted from eight handwritten sample form (hsf) series, hsf-{0,1,2,3,4,6,7,8}. It was originally divided into 3 sets: hsf-{0123}, hsf-7 and hsf-4. The last two sets are referred here as data-test1 (60,089 samples) and data-test2 (58,646 samples). Data-test2 is well known to be more difficult to use for classification than data-test1

[25]. On the basis of the results available in the literature, the representation proposed by Oliveira et al. [52] appears to be well defined and well suited to the NIST SD19 database. The features are a combination of the concavity, contour and surface of characters. The final feature vector is composed of 132 components: 78 for concavity, 48 for contour and 6 for surface.

Ensemble Construction Method

As mentioned in section 2.1, RSS is used during the overproduction phase to generate an initial pool of 100 kNN classifiers. Majority voting was used as the combination function. This combination function is shown in Equation 1.1, chapter 1. In [89], rigorous experimental tests were conducted to set up parameters such as: k value and the number of prototypes (to kNN classifiers), the number of subspace dimensions and the number of classifier members to RSS, the size of the optimization and the size of the validation data sets. The best parameters defined in [89] are used in this chapter. Table IV summarizes the parameter sets used.

Table IV

Experiments parameters related to the classifiers, ensemble generation method and database.

Number of nearest neighbors (k)	1
Random subspace (number of features)	32
Training data set (hsf-{0123})	5,000
Optimization data set size (hsf-{0123})	10,000
Validation data set (hsf-{0123})	10,000
Data-test1 (hsf-7)	60,089
Data-test2 (hsf-4)	58,646

Genetic Algorithms

The population-based evolutionary algorithms used in this work are both single- and multi-objective GAs. Since we use an initial pool composed of 100 classifiers, each individual is represented by a binary vector with a size of 100. Experiments were carried out to define the genetic parameters in [4]. Table V shows the parameter settings employed. The same parameters were used for both GAs.

Table V

Genetic Algorithms parameters

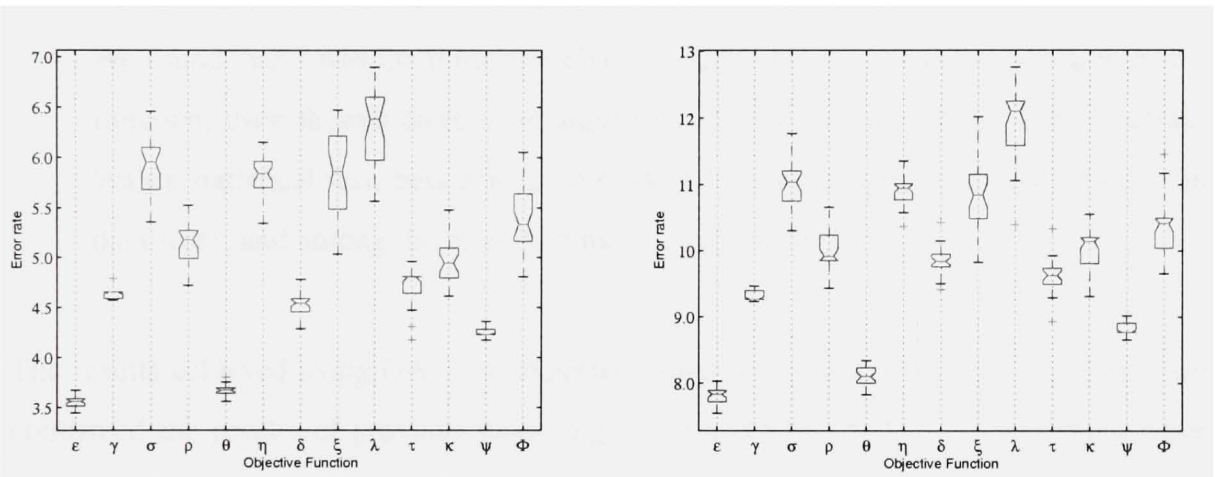
Population size	128
Number of generations	1000
Probability of crossover	0.8
Probability of mutation	0.01
One-point crossover and bit-flip mutation	

2.3.2 Performance Analysis

In order to define the best objective function for our problem, we carried out an experimental investigation focusing on performance (recognition rate). The first question to be answered is: Which measure is the best objective function for finding high-performance classifier ensembles? Among the measures featured in section 2.2.1, the error rate ϵ ($1-$recognition rate) and the diversity measures are the most obvious candidates. The way to compare these measures directly is to apply a single-objective optimization approach. This direct comparison allows us to verify the possibility of using diversity instead ϵ to find high-performance classifier ensembles.

Experiments with GA

GA-based experiments were conducted to compare 13 different objectives functions: ϵ and the 12 diversity measures. As explained previously, each experiment was replicated 30 times in order to arrive at a better comparison of the results. Hence, each of the 13 objective functions employed generated 30 optimized classifier ensembles. Figure 7 shows the comparison results of the 30 replications on data-test1 (Figure 7(a)) and on data-test2 (Figure 7(b)).



(a) data-test1 (b) data-test2

Figure 7 Results of 30 replications using GA and 13 different objective functions. The performances were calculated on the data-test1 (Figure 7(a)) and on the data-test2 (Figure 7(b)).

These experiments show that:

a. Diversity measures are not better than the error rate ϵ as an objective function for generating high-performance classifier ensembles. The Kruskal-Wallis nonparametric statistical test shows that ϵ found ensembles which are significantly different from those found by all the diversity measures.

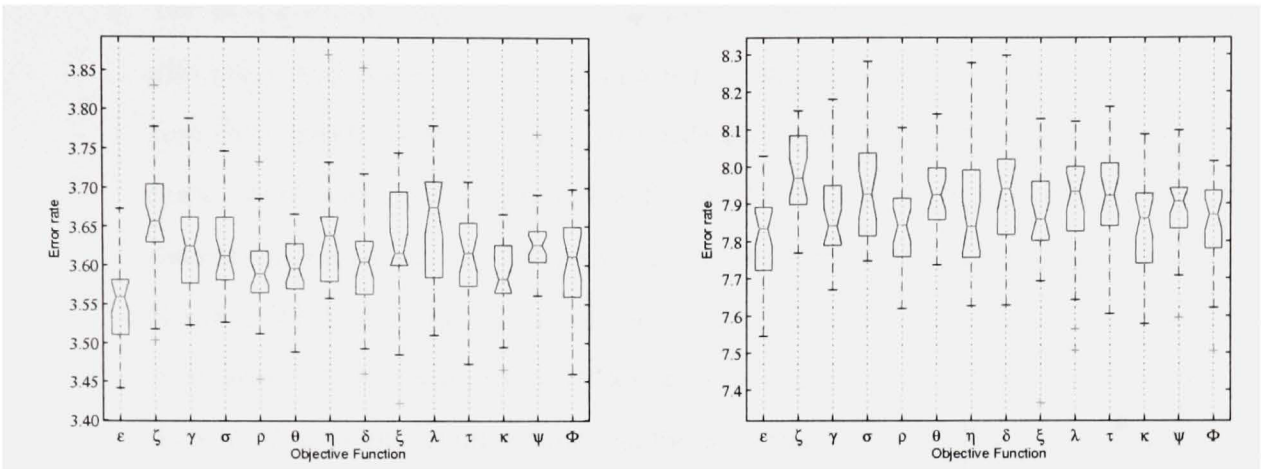
b. The most successful diversity measure was the difficulty measure θ. However, the performance of the classifier ensembles found using this measure was, on average, 0.12% (data-test1) and 0.31% (data-test2) worse than that of the ensembles found using ϵ directly.

c. Fault majority λ was the worst objective function. This is a different result from those presented by Ruta and Gabrys [70]. They observed that measures with better correlation with majority voting error, i.e. fault majority and double-fault δ, are better objective functions for generating high-performance ensembles than the others. We found that δ was the third best diversity measure and λ was the worst objective function, even though there is no significant difference, according to the Kruskal-Wallis statistical test, between λ, coincident failure, disagreement and entropy on data-test1, and among the three first measures on data-test2.

The results achieved using GA were expected, apart from those for λ and δ. In fact, we confirmed the results of previous work, e.g. [70] and [44], that diversity alone cannot substitute for ϵ as an objective function for finding the highest performing classifier ensembles. Since diversity alone is no better than ϵ, can we find better performing ensembles by including both objective functions in the optimization process? We try to answer this question using a multi-objective optimization approach in the next section.

Experiments with NSGA-II

We continue our experimental study using NSGA-II as the search algorithm. The preliminary study with GA suggested that diversity alone is not better than ϵ for generating the best performing classifier ensemble. This observation led us to use both ϵ and diversity jointly to guide the optimization process with NSGA-II, since we have the option of combining different objective functions. The hope is that greater diversity between base classifiers leads to the selection of high-performance classifier ensembles.



(a) data-test1 (b) data-test2

Figure 8 Results of 30 replications using NSGA-II and 13 different pairs of objective functions. The performances were calculated on data-test1 (Figure 8(a)) and data-test2 (Figure 8(b)). The first value corresponds to GA with the error rate ϵ as the objective function while the second value corresponds to NSGA-II guided by ϵ with ensemble size ζ.

Each diversity measure mentioned in section 2.2.1 was combined with ϵ to make up pairs of objective functions to guide the optimization process. Again, the optimization process using each pair of objective functions was replicated 30 times. Figure 8 shows the results of 30 replications on data-test1 (Figure 8(a)) and data-test2 (Figure 8(b)). It is important to mention that the first value corresponds to the results using GA as the search algorithm and ϵ as the objective function. This means that we can compare the single- and multi-objective results. The second value corresponds to the results using NSGA-II guided by ensemble size ζ, with ϵ as the objective functions (discussed in the next section).

Some observations can be made from these results:

a. By including both diversity and ϵ in a multi-objective optimization process, we may find more high-performance classifier ensembles than by using diversity alone; however, the performance of these ensembles is still worse than the performance of the ensembles found using ϵ in the single-objective optimization process.

b. The best diversity measures are difficulty θ, interrater agreement κ, correlation coefficient ρ and double-fault δ on data-test1. The Kruskal-Wallis test shows that the first three measures found classifier ensembles with no significantly different mean ranks from those found using GA with ϵ as the objective function on data-test1. On data-test2, almost all the diversity measure results were similar. According to the Kruskal-Wallis statistical test, except for coincident failure σ and θ, all the diversity measures found classifier ensembles with no significantly different mean ranks on data-test2. It is important to note that the difference between ensembles found using diversity (multi-objective optimization) and ensembles found using only ϵ (single-objective optimization) was dramatically less. The classifier ensembles found using the three best diversity measures were, on average, 0.05% worse than those found using only ϵ on data-test1 and 0.13% worse on data-test2.

c. It is interesting to note that θ found high-performance classifier ensembles on data-test1, but, on data-test2, θ yielded one of the worst performances. Such behavior shows that these two data sets are actually very different.

d. The two measures pointed out by Ruta and Gabrys [70] as the best diversity measures (δ and λ) found better classifier ensembles on multi-objective than on single-objective optimization, as was the case for all the measures. However, especially on data-test1, λ was the worst measure, and, once again, was significantly different from the other measures, as shown by the Kruskal-Wallis test.

As indicated in the beginning of this chapter, besides performance, we have to take into account ensemble size when selecting classifier ensembles. In the following section, this aspect is analyzed.

2.3.3 Ensemble Size Analysis

Our ensemble size analysis relies on the analogy between feature subset selection and SOCS. In this sense, discarding redundant classifiers could improve performance. Hence, the minimization of the ensemble size ζ is the most obvious objective function to use to achieve such a reduction in the number of classifiers. However, ζ cannot be used in a single-objective optimization process, because it might be combined with other measures in order to increase performance while reducing the number of classifiers. In line with this idea, we combine ζ with ϵ and the 12 diversity measures described in section 2.2.1 to make up pairs of objective functions to guide the optimization process using NSGA-II. Figure 9 presents a graph containing the size of the classifier ensembles found in 30 replications generated by each pair of objective functions, while the performances are shown in Figure 10 (10(a), data-test1 and 10(b), data-test2). The first value corresponds to the results obtained using GA as the search algorithm and ϵ as the objective function in order to arrive at a better comparison.

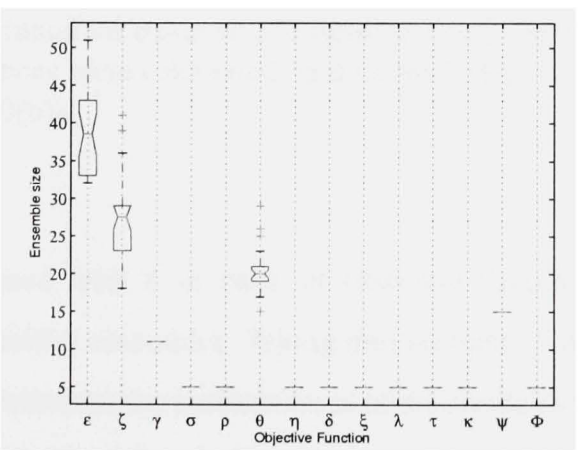
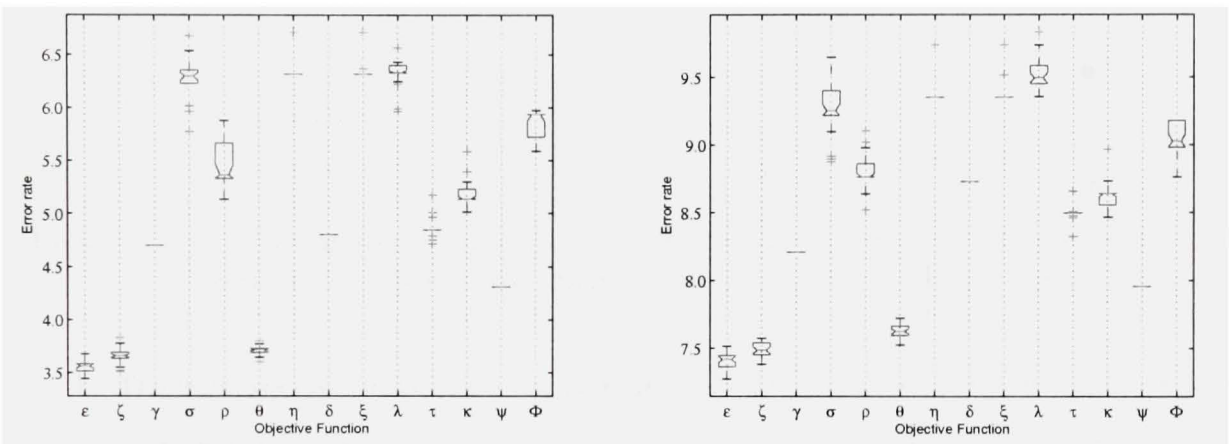


Figure 9 Size of the classifier ensembles found using 13 different measures combined with ensemble size ζ in pairs of objective functions used by NSGA-II.

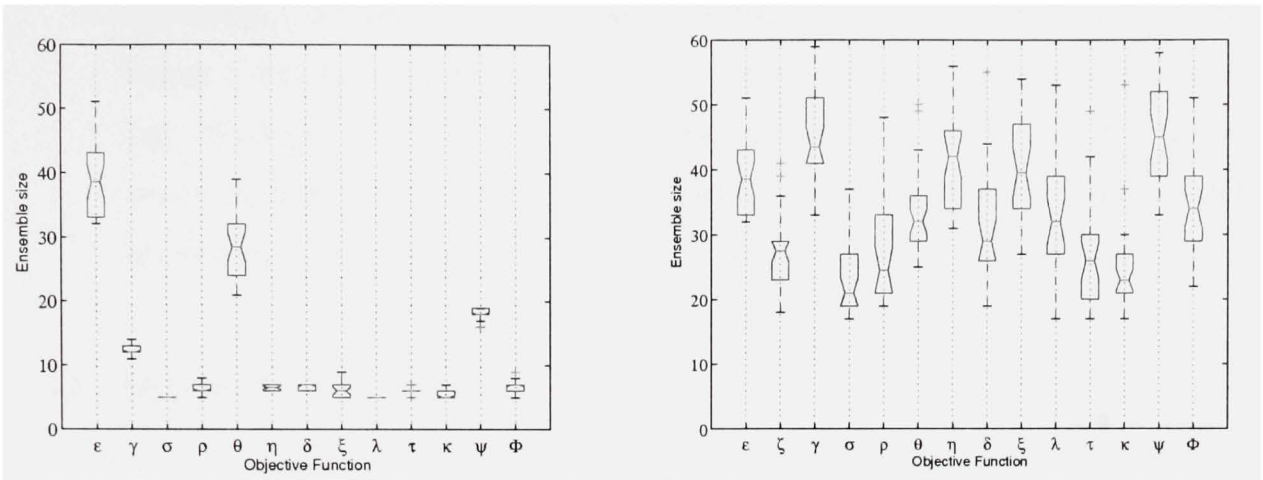
We also show the ensemble size of the classifier ensembles found in the first two series of experiments, i.e., single objective functions and diversity combined with ϵ. This allows us to better analyze the ensemble size issue. Figure 11 shows the ensemble size of the classifier ensembles found with GA (Figure 11(a)) and with NSGA-II combining ϵ with diversity (Figure 11(b)). Based on all these results, we observe that:



(a) data-test1 (b) data-test2

Figure 10 Performance of the classifier ensembles found using NSGA-II with pairs of objective functions made up of ensemble size ζ and the 13 different measures. Performances were calculated on data-test1 (Figure 10(a)) and on data-test2 (Figure 10(b)).

a. Diversity combined with ζ in pairs of objective functions does not find high-performance classifier ensembles. Taking into account all the results obtained in all the series of experiments, the performances of the ensembles found using these pairs of objective functions made up of ζ and diversity showed the worst performances. In contrast, those ensembles were the smallest. It is interesting to note that, using this combination of objective functions, NSGA-II converges to the same solution at each replication, with the exception of the difficulty measure θ. The minimum number of classifiers allowed by the search algorithms was 5. This fixed minimum ensemble size was defined to avoid generating too small classifier ensembles.



(a) GA (b) NSGA-II

Figure 11 Ensemble size of the classifier ensembles found using GA (Figure 11(a)) and NSGA-II (Figure 11(b)). Each optimization process was performed 30 times.

b. The most successful diversity measure in terms of performance was θ. However, the performance of the ensembles obtained using ζ and ϵ are better. According to the Kruskal-Wallis statistical test, the performances are significantly different.

c. The analogy between FSS and SOCS may be established. The performance of our baseline system, i.e. the pool of 100 kNN (96.28% on data-test1), is 0.07% worse than the average result using ζ and ϵ as the objective functions (average of 96.35% on data-test1 (Figure 10), while the averaged ensemble size is 27 classifiers (Figure 40). However, better performing classifier ensembles can be found using GA and ϵ. The Kruskal-Wallis statistical test showed that the performances are significantly different. Moreover, the combination of ζ and ϵ as the objective function did not establish the best trade-off between these two measures. Interrater agreement κ combined with ϵ generated smaller (24 classifiers on average) and better performing classifier ensembles (96.41% and 92.16%, on average, on data-test1 and on data-test2 respectively, Figure 8).

d. Ambiguity γ combined with ϵ, and Kohavi-Wolpert ψ combined with ϵ, found the largest classifier ensembles (45 classifiers on average). What is more interesting is that, although we found the best performing classifier ensemble using GA as the search algorithm and ϵ as the objective function, such single-objective function did not find the largest solutions.

2.4 Discussion

This chapter presented the experimental results of a study using the single- and multi-objective optimization processes to perform the selection phase of SOCS. An ensemble of 100 kNN classifiers generated using the Random Subspace method was used as the initial pool of classifiers. Fourteen different objective functions were applied: 12 diversity measures, error rate and ensemble size. The experiments were divided into three series. In the first, the error rate and the 12 diversity measures were directly compared in a single-optimization approach. In the second, the 12 diversity measures were combined with the error rate to make up pairs of objective functions in a multi-optimization approach. Finally, in the third, the error rate and the 12 diversity measures were combined with ensemble size in pairs of objective functions, again in a multi-optimization approach.

The first series of experiments was conducted in order to answer the first question (1) posed in the beginning of this chapter. Our results confirm the observation made in previous work that diversity alone cannot be better than the error rate at finding the most accurate classifier ensembles. The difficulty measure was the best diversity measure when the diversity measures are compared. In our attempt to answer question (2) when both the error rate and diversity are combined in a multi-objective approach (second series of experiments), we found that the performance of the solutions using diversity is much higher than the performance of the solutions using diversity in a single-objective optimization approach. However, the performance of the classifier ensembles found using diversity measures combined with the error rate to guide the selection were still worse than the per-

formance of the ensembles found using only the error rate, although the difference was reduced.

We are now able to establish an analogy between feature subset selection and SOCS, in response to question (3). By combining ensemble size and the error rate in a pair of objective functions, we increased the initial pool of classifier performances and decreased the number of classifiers to 27, instead of 100, from the initial pool. In contrast, diversity measures achieved the worst performance in the third series of experiments. Moreover, the combined minimization of ensemble size and error rate was not the best pair of objective functions to accomplish the trade-off between complexity and performance. In fact, interrater agreement combined with the error rate established the best trade-off between performance and ensemble size. We conclude, therefore, that it is not necessary to include ensemble size in the optimization process. The reduction in the number of classifiers is a consequence of the selection of classifiers, whatever the objective function used to guide the search. Some objective functions generate smaller classifier ensembles, while others generate bigger ones.

It is also important to observe that our experiments showed that ensemble size and diversity are not conflicting objective functions (see Figure 9). We observe that we cannot decrease the generalization error rate by combining this pair of objective functions. In appendix 3, we present further evidence to show why diversity and ensemble size are not conflicting objective functions. Moreover, we can see in Table XXIV and Table XXV in appendix 2, that the results obtained using diversity measures in a single-objective optimization approach are quite similar to the results obtained using diversity combined with ensemble size in pairs of objective functions in a multi-objective optimization approach. Hence, the minimum number of classifiers is achieved when using most of the diversity measures in single-objective optimization (see Figure 11(a)). Thus, the results related to the first series of our experiments may be different if the minimum number of classifiers is fixed and larger than we have defined in this chapter, i.e. 5 classifiers.

Moreover, in this chapter, we conducted the optimization processes using, besides the traditional optimization dataset, a validation dataset in order to avoid overfitting during the selection phase of SOCS. Despite such apparent overfitting control, we observed that this strategy fails to address the overfitting phenomenon since, although an independent dataset was used to validate the solutions, the overfitting phenomenon may still be present. In the next chapter we show that overfitting can be detected at the selection phase of SOCS and present strategies to control overfitting.

CHAPTER 3

OVERFITTING-CAUTIOUS SELECTION OF CLASSIFIER ENSEMBLES

We show in this chapter that overfitting can be detected during the selection phase of SOCS, this selection phase being formulated as an optimization problem. We attempt to prove experimentally that an overfitting control strategy must be conducted *during* the optimization process. In order to pursue our analysis on population-based evolutionary algorithms, we keep using both single- and multi-objective GA. Taking into account this, we investigate the use of an auxiliary archive $\mathcal{A}$ to store the best performing candidate ensembles (or Pareto fronts in the MOGA case) obtained in a validation process using the validation partition $\mathcal{V}$ to control overfitting. Three different strategies for update $\mathcal{A}$ have been compared and adapted in this chapter to the context of the single- and multi-objective selection of classifier ensembles: (1) *partial validation* where $\mathcal{A}$ is updated only in the last generation of the optimization process; (2) *backwarding* [67] which relies on monitoring the optimization process by updating $\mathcal{A}$ with the best solution from each generation; and (3) *global validation* [62] updating $\mathcal{A}$ by storing in it the Pareto front (or the best solution in the GA case) identified on $\mathcal{V}$ at each generation step.

Besides the ensemble of kNN created at the overproduction phase using the RSS method in the previous chapter, two additional initial pool of classifiers are investigated in this chapter: (1) a pool of DT created using bagging; and (2) a pool of DT created using the RSS method. In addition, considering the results related to the analysis of search criteria obtained previously, we use only four diversity measures (described in section 2.2.1) and ϵ to guide the optimization process presented in this chapter. Diversity measures are applied by NSGA-II in combination with ϵ in pairs of objective functions. Moreover, ϵ, as well as the diversity measures, are employed as single-objective functions by GA. To avoid the problem of reaching a too small ensemble size, we defined a large fixed minimum ensemble size for all diversity measures in our experiments. It is important to mention

that in appendix 2, we carried out an overfitting analysis using both GA and NSGA-II, guided by all the objective functions and combinations of objective functions discussed previously.

In this chapter, the global validation strategy is presented as a tool to show the relationship between diversity and performance, specifically when diversity measures are used to guide GA. The assumption is that, if a strong relationship exists between diversity and performance, the solution obtained by performing global validation solely guided by diversity should be close, or equal, to the solution with the highest performance among all solutions evaluated. This offers a new possibility for analyzing the relationship between diversity and performance, which has received a great deal of attention in the literature [44; 17; 70].

Our objective in this chapter is to answer the following questions:

1. Which is the best strategy employing an archive $\mathcal{A}$ for reducing overfitting at the selection phase of SOCS when this selection is formulated as an optimization problem?

2. Are classifier ensembles generated by bagging and RSS equally affected by overfitting in the selection of classifier ensembles?

The following section demonstrates the circumstances under which the process of classifier ensemble selection results in overfitting. In section 3.2, three strategies used to control overfitting are introduced. The parameters employed for the experiments are described in section 3.3, where the validation strategies are applied using holdout and k-fold cross-validation schemes. Finally, experimental results are presented in section 3.3.

3.1 Overfitting in Selecting Classifier Ensembles

The problem of selecting classifier ensembles, using an optimization dataset $\mathcal{O}$ can be formulated as a learning task, since the search algorithm operates by minimizing/maximizing

the objective function, such as maximizing the classification performance or maximizing the diversity of the members of a given ensemble C_j. Indeed, Radtke et al. [62] showed that multi-objective evolutionary optimization, such as the selection of classifier ensembles, is prone to overfitting. We may define overfitting in the context of ensemble selection inspired by the definition provided by Mitchell [49] in the following way. Let C_j^* and $C_j^{*'}$ be the best performing candidate ensembles found through calculating the error rate ϵ for each element of $\mathcal{P}(\mathcal{C})$ over samples contained in $\mathcal{O}$ and $\mathcal{V}$ respectively. Consider the classification error ϵ of these two candidate ensembles measured using samples from $\mathcal{V}$. We will denote this classification error by $\epsilon(\mathcal{V}, C_j^*)$ and $\epsilon(\mathcal{V}, C_j^{*'})$. In this setting, C_j^* is said to overfit on $\mathcal{O}$ if an alternative candidate ensemble $C_j^{*'} \in \mathcal{P}(\mathcal{C})$ can be found such that $\epsilon(\mathcal{V}, C_j^*) > \epsilon(\mathcal{V}, C_j^{*'})$.

The selection process for classifier ensembles in SOCS is illustrated in Figure 12. An ensemble creation method is employed using $\mathcal{T}$ to generate the initial pool of classifiers $\mathcal{C}$. Thus, the search algorithm calculates fitness on $\mathcal{O}$ by testing different candidate ensembles. The best candidate ensemble $C_j^{*'}$ is identified in $\mathcal{V}$ to prevent overfitting. Finally, the generalization performance of $C_j^{*'}$ is measured using the test dataset ($\mathcal{G}$). Hence, SOCS requires at least these four datasets. It is important to mention that $\mathcal{V}$ is different from the validation dataset typically used to adjust base classifier parameters, such as weights in MLP, the number of neighbors considered (k value) in kNN classifiers, etc. When such parameter adjustment is necessary, a fifth dataset must be used, as was done in [62], to avoid overly optimistic performance assessment. The following sections describe how overfitting can be detected in single- and multi-objective optimization problems performed by GAs.

3.1.1 Overfitting in single-objective GA

In Figure 13, ϵ is employed as the objective function to guide GA using the NIST-digits database as the problem and RSS as the method to generate $\mathcal{C}$ as a pool of 100 kNN clas-

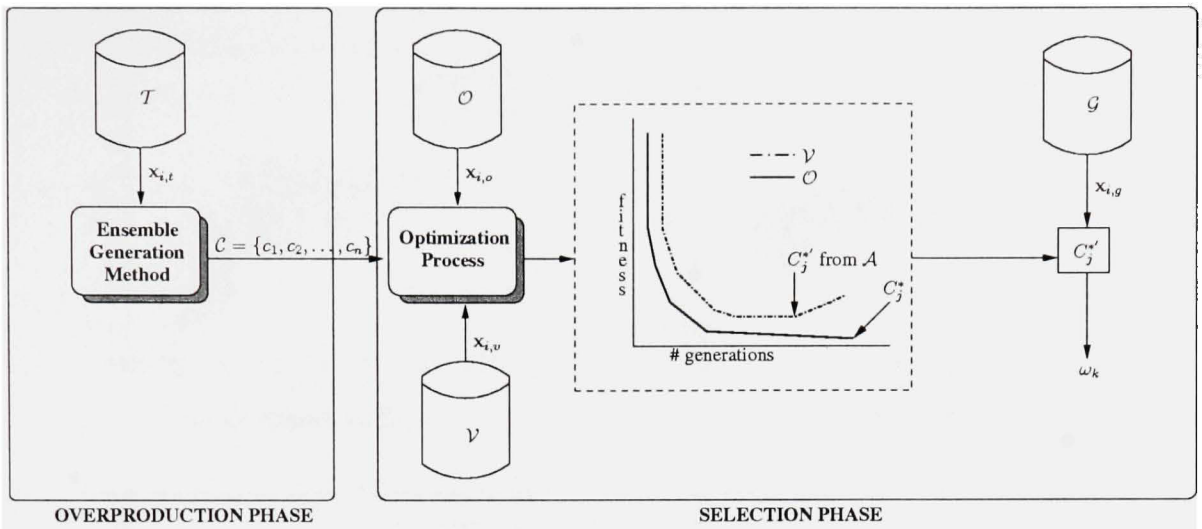
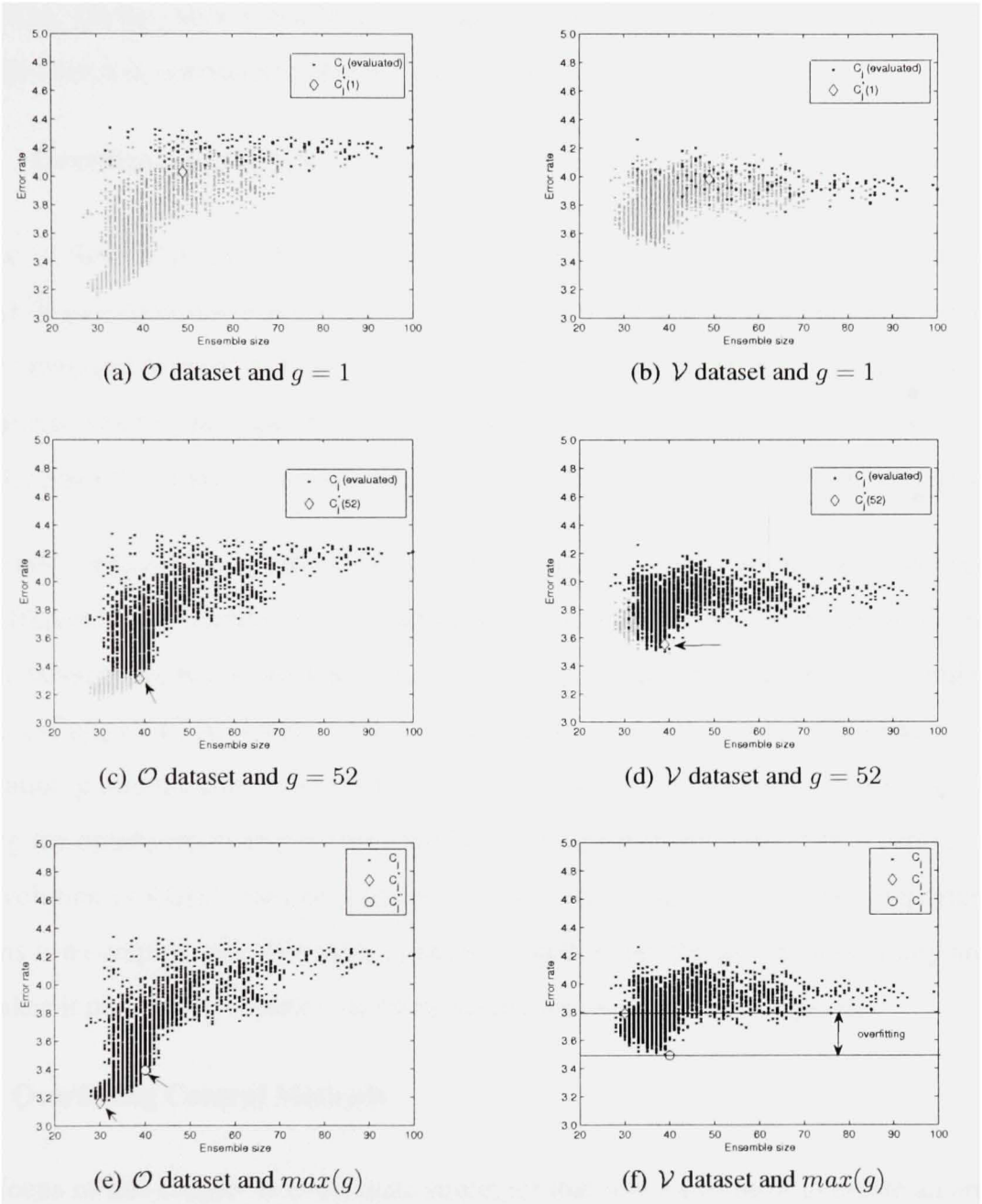


Figure 12 Overview of the process of selection of classifier ensembles and the points of entry of the four datasets used.

sifiers. These same parameters were investigated in last chapter, section 2.3.1. Although GA was only guided by ϵ, we show plots of ϵ versus the ensemble's size to better illustrate the process.

We denote $\mathbf{C}(g)$ as the population of candidate classifier ensembles found at each generation g and the best candidate ensemble found by evaluating all individuals from $\mathbf{C}(g)$ on $\mathcal{O}$ by $C_j^*(g)$. Each point on the plot corresponds to a candidate ensemble C_j taken from $\mathcal{P}(\mathcal{C})$ and evaluated during the optimization process. Indeed, these points represent the complete search space explored for $max(g) = 1,000$. The number of individuals at any $\mathbf{C}(g)$ is 128.

Actually, the overfitting phenomenon measured when selecting classifier ensembles presents a behavior very similar to what is seen in a typical Machine Learning process, i.e. the chance that the search algorithm will overfit on samples taken from $\mathcal{O}$ increases with the number of generations. This phenomenon can be observed in Figure 13(e), which shows the evolution of $\epsilon(\mathcal{O}, C_j^{*\prime})$ in the search space. On the one hand, it is clear that the optimization process could be stopped before overfitting starts to occur in this problem



(a) $\mathcal{O}$ dataset and $g = 1$

(b) $\mathcal{V}$ dataset and $g = 1$

(c) $\mathcal{O}$ dataset and $g = 52$

(d) $\mathcal{V}$ dataset and $g = 52$

(e) $\mathcal{O}$ dataset and $max(g)$

(f) $\mathcal{V}$ dataset and $max(g)$

Figure 13 Optimization using GA guided by ϵ. Here, we follow the evolution of $C_j^*(g)$ (diamonds) from $g = 1$ to $max(g)$ (Figures 13(a), 13(c) and 13(e)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figures 13(b), 13(d) and 13(f)). The overfitting is measured as the difference in error between $C_j^{*\prime}$ (circles) and C_j^* (13(f)). There is a 0.30% overfit in this example, where the minimal error is reached slightly after $g = 52$ on $\mathcal{V}$, and overfitting is measured by comparing it to the minimal error reached on $\mathcal{O}$. Solutions not yet evaluated are in grey and the best performing solutions are highlighted by arrows.

using GA. On the other, it is difficult to define a stopping criterion to control overfitting in a multi-objective optimization process, as explained in section 3.1.2.

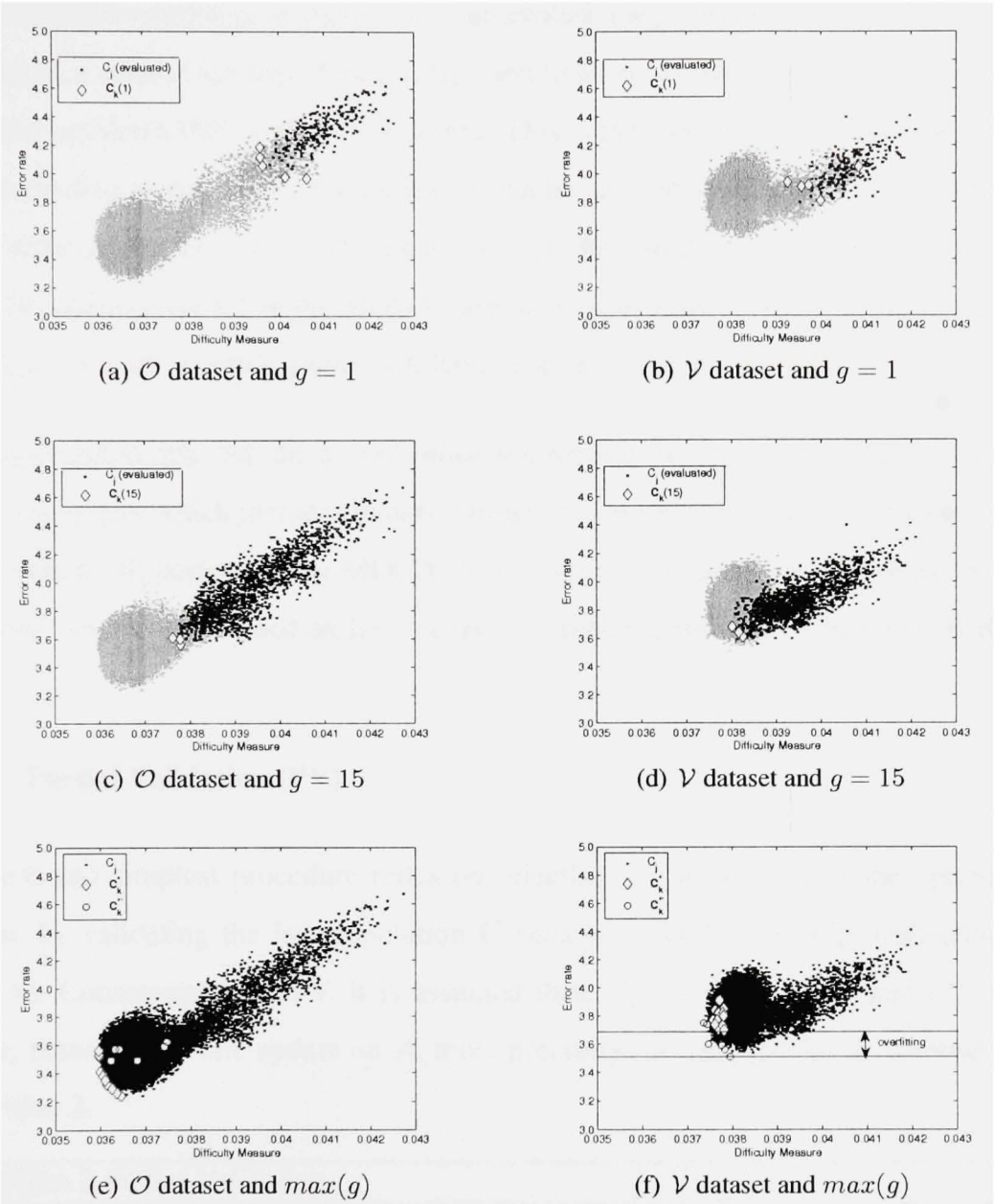
3.1.2 Overfitting in MOGA

Figure 14 (left) illustrates the optimization process performed for $max(g) = 1,000$ using NSGA-II guided by the following pair of objective functions: jointly minimize ϵ and difficulty measure (section 2.2.1) in the same problem investigated with GA in last section. The Pareto front from population $\mathrm{C}(g)$ found in $\mathcal{O}$ is denoted by $\mathrm{C}_k(g)$. Thus, C_k^* (diamonds) and $\mathrm{C}_k^{*'}$ (circles) represent the final Pareto fronts found in $\mathcal{O}$ and $\mathcal{V}$ respectively.

Especially noteworthy in Figure 14(f) is that, besides the fact that the solutions over $\mathrm{C}_k^{*'}$ are different from solutions over C_k^*, which was expected considering that $\mathcal{V}$ and $\mathcal{O}$ are different datasets, the nondominated solutions over $\mathrm{C}_k^{*'}$ are discarded during the optimization process (Figure 14(e)). Hence, the definition of a stopping criterion for multi-objective optimization problems is difficult for the following reasons: (1) solutions over $\mathrm{C}_k(g)$ found during the optimization process may not be nondominated solutions over $\mathcal{V}$ and (2) since the evolution of $\mathrm{C}_k(g)$ must be monitored on $\mathcal{V}$, comparing sets of equally important solutions is a complex task. We show in the next section that the use of an auxiliary archive $\mathcal{A}$ makes it possible to control overfitting taking into account these aspects.

3.2 Overfitting Control Methods

The focus of this chapter is to evaluate strategies that rely on using $\mathcal{V}$ to create an archive $\mathcal{A}$ to control overfitting, since these memory-based strategies may be applied in any optimization problem. The idea is to use the error rate measured on $\mathcal{V}$ as an estimation of the generalization error. In order to accomplish the objectives of this chapter, we have adapted two overfitting control strategies to the context of classifier ensemble selection for both single- and multi-objective optimization problems: (1) *backwarding* originally proposed in [67] to prevent overfitting in GP; and (2) *global validation* proposed in [62] and



(a) $\mathcal{O}$ dataset and $g = 1$

(b) $\mathcal{V}$ dataset and $g = 1$

(c) $\mathcal{O}$ dataset and $g = 15$

(d) $\mathcal{V}$ dataset and $g = 15$

(e) $\mathcal{O}$ dataset and $max(g)$

(f) $\mathcal{V}$ dataset and $max(g)$

Figure 14 Optimization using NSGA-II and the pair of objective functions: difficulty measure and ϵ. We follow the evolution of $\mathbf{C}_k(g)$ (diamonds) from $g = 1$ to $max(g)$ (Figures 14(a), 14(c) and 14(e)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figures 14(b), 14(d) and 14(f)). The overfitting is measured as the difference in error between the most accurate solution in $\mathbf{C}_k^{*'}$ (circles) and in $\mathbf{C}_k^*$ (14(f)). There is a 0.20% overfit in this example, where the minimal error is reached slightly after $g = 15$ on $\mathcal{V}$, and overfitting is measured by comparing it to the minimal error reached on $\mathcal{O}$. Solutions not yet evaluated are in grey.

[67] to control overfitting in multi-objective evolutionary optimization problems. *Partial validation*, a control strategy traditionally used to avoid overfitting in classifier ensemble selection problems [89] is also investigated. This is the strategy used in previous chapter. It is interesting to note that these strategies can be ordered with respect to the cardinality of the solution set used for overfitting control: *partial validation* uses only the last population of solutions (or $\mathbf{C}_k^*$ in the MOGA case) and *backwarding* validates the best solution (or $\mathbf{C}_k^*(g)$) at each g, while *global validation* uses all solutions at each g.

It is important to note that the *global validation* strategy is employed in [62] in a complex two-level system which includes feature extraction, feature selection and classifier ensemble selection, all performed by MOGA. In this thesis, we adapt their strategy to single-objective GA. As mentioned earlier, we use ensembles generated by bagging (BAG) and RSS.

3.2.1 Partial Validation (PV)

The first and simplest procedure relies on selecting $C_j^{*'}$ at the end of the optimization process by validating the last population $\mathbf{C}(max(g))$ (for GA) or $\mathbf{C}_k^*$ (respectively for MOGA). Consequently, in PV, it is assumed that, $C_j^{*'} \in \mathbf{C}(max(g))$ and $C_j^{*'} \in \mathbf{C}_k^*$. Hence, there is only one update on $\mathcal{A}$, more precisely for $max(g)$. PV is summarized in Algorithm 2.

Algorithm 2 Partial Validation

1: Creates initial population $\mathbf{C}(1)$ of w chromosomes
2: $\mathcal{A} = \emptyset$
3: **for** each generation $g \in \{1, \ldots, max(g)\}$ **do**
4: perform all genetic operators and generate new population $\mathbf{C}(g+1)$.
5: **end for**
6: validate all solutions from $\mathbf{C}(max(g))$ (for GA) or $\mathbf{C}_k^*$ (for MOGA)
7: choose as $C_j^{*'}$ the solution with highest recognition rate
8: update $\mathcal{A}$ by storing $C_j^{*'}$
9: **return** $C_j^{*'}$ stored in $\mathcal{A}$

3.2.2 Backwarding (BV)

Overfitting remains uncontrolled even when PV is used, because $\mathbf{C}(max(g))$ (for GA) or $\mathbf{C}_k^*$ for MOGA are composed of overfitted solutions, as explained in section 3.1. An alternative to PV is to validate not only $\mathbf{C}(max(g))$ (or $\mathbf{C}_k^*$), but the $C_j^*(g)$ (or $\mathbf{C}_k(g)$) found at each generation. The BV method [67] proposed for GP problems is based on this idea. The authors advocate that GP is prone to overfitting especially on later generations as in Machine Learning tasks. Motivated by this observation, they proposed BV to control overfitting by monitoring the optimization process on $\mathcal{V}$ to determine the point where GP starts to overfit $\mathcal{O}$. This approach uses $\mathcal{A}$ to store the best solution found before overfitting starts to occur. Even though ϵ is not the only objective function used to guide GA in this chapter, we use ϵ to represent the objective function in Algorithm 3, which shows how BV is employed with GA. However, other objective functions can be used without loss of generality.

Where multi-objective optimization is concerned, Pareto fronts must be stored in $\mathcal{A}$ instead of individual solutions. Taking into account that BV relies on comparing each new solution found on $\mathcal{O}$ to the solution stored on $\mathcal{A}$, this fact leads to the following question: How can Pareto fronts be compared so as to identify whether or not one Pareto front is better than the others. Because of the various tradeoffs over the Pareto front, the definition of a quality measure is much more complex in multi-objective than in single-objective optimization problems. Some quality measures such as the Pareto front spread, the objective functions spread [103], the epsilon indicator and the coverage function [108] have been proposed. In order to determine whether or not the $\mathbf{C}_k(g)$ found at each generation is better than the $\mathbf{C}_k^{*'}$ stored in $\mathcal{A}$, we propose to use the Pareto quality measure called the *coverage* function, introduced by Zitzler et al. [108]. This measure was used in [61] to define a stopping criterion for MOGAs.

Algorithm 3 Backwarding for GA

1: Creates initial population $\mathbf{C}(1)$ of w chromosomes
2: $\mathcal{A} = \emptyset$
3: Find $C_j^*(1)$ and set $C_j^{*'} : C_j^*(1)$
4: Store $C_j^{*'}$ in $\mathcal{A}$
5: **for** each generation $g \in \{1, \ldots, max(g)\}$ **do**
6: perform all genetic operators
7: generate new population $\mathbf{C}(g+1)$ and find $C_j^*(g+1)$ as usual
8: **if** $\epsilon(\mathcal{V}, C_j^*)(g+1) < \epsilon(\mathcal{V}, C_j^{*'})$ **then**
9: set $C_j^{*'} := C_j^*(g+1)$
10: update $\mathcal{A}$ by storing in it the new $C_j^{*'}$
11: **end if**
12: **end for**
13: **return** $C_j^{*'}$ stored on $\mathcal{A}$

The coverage function, measured on $\mathcal{V}$, is based on the weak dominance criteria and indicates the number of candidate ensembles in $\mathbf{C}_k(g)$ that are weakly dominated by at least one solution in $\mathbf{C}_k^{*'}$. Given two solutions $C_i \in \mathbf{C}_k^{*'}$ and $C_j \in \mathbf{C}_k(g)$, we may say that C_i covers C_j if C_i is not worse than C_j in all objective functions. The idea is to verify the Pareto improvement among generations on $\mathcal{A}$. Coverage can be denoted as:

$$cov(\mathbf{C}_k^{*'}, \mathbf{C}_k(g)) \tag{3.1}$$

In this way, the average number of solutions in $\mathbf{C}_k^{*'}$ covering the solutions on $\mathbf{C}_k(g)$ is calculated. Thus, $cov(\mathbf{C}_k^{*'}, \mathbf{C}_k(g)) < 1$ or $cov(\mathbf{C}_k^{*'}, \mathbf{C}_k(g)) = 1$, whether $\mathbf{C}_k^{*'}$ covers the entire Pareto $\mathbf{C}_k(g)$ or not. The Pareto improvement at each generation is then measured as:

$$imp(\mathbf{C}_k(g), \mathbf{C}_k^{*'}) = 1 - cov(\mathbf{C}_k^{*'}, \mathbf{C}_k(g)) \tag{3.2}$$

Improvement reaches its maximum ($imp = 1$) when there is no solution on $\mathbf{C}_k(g)$ covered by solutions on $\mathbf{C}_k^{*'}$ and its lowest possible value ($imp = 0$), when all solutions on

$C_k(g)$ are covered by those on $C_k^{*'}$. Consequently, the update of $\mathcal{A}$ is dependent on the improvement obtained between these two Pareto fronts. When $imp > 0$, $\mathcal{A}$ is updated; otherwise, there is no update. The BV for MOGA is summarized in Algorithm 4.

Algorithm 4 Backwarding for MOGA

```
1: Creates initial population C(1) of w chromosomes
2: A = ∅
3: Find C_k(1) and set C_k^{*'}:=C_k(1)
4: Store C_k^{*'} in A
5: for each generation g, g ∈ {1, ..., max(g)} do
6:    perform all genetic operators;
7:    generate C(g + 1) and find C_k(g + 1) as usual
8:    if imp(C_k(g + 1), C_k^{*'}) > 0 then
9:       set C_k^{*'}:=C_k(g + 1)
10:      update A by storing in it the new C_k^{*'}
11:   end if
12: end for
13: return C_k^{*'} stored on A to pick up C_j^{*'}.
```

3.2.3 Global Validation (GV)

There is a problem with BV because, since, at each generation, GA (or MOGA) creates a population of solutions, $C_j^*(g)$ (or $C_k(g)$ for MOGA) found on $\mathcal{O}$ may not be the best solution (or Pareto front) on $\mathcal{V}$. An approach avoiding such a limitation is global validation GV [62; 100], which relies on using $\mathcal{A}$ to validate the entire population $C(g)$ from each generation.

GV works as follows in the context of multi-objective optimization (Algorithm 5): at each g step, all solutions are validated, and thus the set of non-dominated solutions found on $\mathcal{V}$ is stored in $\mathcal{A}$. When the optimization process is completed, two sets of non-dominated solutions are available: (1) the traditional C_k^* found on $\mathcal{O}$; and (2) $C_k^{*'}$, the set of non-dominated solutions found on $\mathcal{V}$. In Figure 14(f), the solutions composing $C_k^{*'}$ stored in $\mathcal{A}$ are represented by circles. As can be observed in this figure, the solutions stored in $\mathcal{A}$ are different from the solutions over C_k^*.

Algorithm 5 Global Validation for MOGA

1: Creates initial population $\mathbf{C}(1)$ of w chromosomes
2: $\mathcal{A} = \emptyset$
3: **for** each generation $g \in \{1, \ldots, max(g)\}$ **do**
4: perform all genetic operators
5: generate $\mathbf{C}(g+1)$
6: validate all solutions in $\mathbf{C}(g+1)$ over samples contained in $\mathcal{V}$
7: perform a nondominated sorting to $\mathbf{C}(g+1) \cup \mathcal{A}$ to find $\mathbf{C}_k^{*'}$
8: update $\mathcal{A}$ by storing in it $\mathbf{C}_k^{*'}$
9: **end for**
10: **return** $\mathbf{C}_k^{*'}$ stored in $\mathcal{A}$

We propose to change Radtke et al.'s method [62] slightly to adapt it to single-objective optimization problems. Since no Pareto front is involved when using GA, we are interested in the best solution $C_j^{*'}$. In this case, it is sufficient to validate all solutions at each new generation, find the best solution $C_j^{*'}(g)$ and compare it to $C_j^{*'}$, which is stored in $\mathcal{A}$. In this way, we keep the solution $C_j^{*'}$ found on $\mathcal{V}$ stored in $\mathcal{A}$. Letting ϵ represent the objective function, the complete GV algorithm for GA is described in Algorithm 6.

Algorithm 6 Global Validation for GA

1: Creates initial population $\mathbf{C}(1)$ of w chromosomes
2: $\mathcal{A} = \emptyset$
3: Validate all solutions in $\mathbf{C}(1)$ over samples contained in $\mathcal{V}$
4: Find $C_j^{*'}(1)$ from $\mathbf{C}(1)$
5: Set $C_j^{*'}$:=$C_j^{*'}(1)$
6: **for** each generation $g \in \{1, \ldots, max(g)\}$ **do**
7: perform all genetic operations and generate $\mathbf{C}(g+1)$
8: validate all solutions in $\mathbf{C}(g+1)$
9: find $C_j^{*'}(g+1)$
10: **if** $\epsilon(\mathcal{V}, C_j^{*'})(g+1) < \epsilon(\mathcal{V}, C_j^{*'})$ **then**
11: set $C_j^{*'} := C_j^{*'}(g+1)$
12: update $\mathcal{A}$ by storing in it the new $C_j^{*'}$
13: **end if**
14: **end for**
15: **return** $C_j^{*'}$ stored in $\mathcal{A}$

An example of the GV strategy for single-objective GA is illustrated in Figure 13(f). This figure shows overfitting when comparing $\epsilon(\mathcal{V}, C_j^{*'})$ and $\epsilon(\mathcal{V}, C_j^{*})$. The ensemble $C_j^{*'}$ prob-

ably has higher generalization performance than C_j^* obtained in $\mathcal{O}$. In section 3.3, we analyze such an assumption experimentally.

3.3 Experiments

The parameters and the experimental protocol employed are described in sections 3.3.1 and 3.3.2 respectively.

3.3.1 Parameter Settings on Experiments

The databases, ensemble construction methods, search algorithms and objective functions used to conduct the experiments are defined in this section.

Databases

It is important to note that the databases must be large enough to be partitioned into the four above-mentioned datasets: $\mathcal{T}$, $\mathcal{O}$, $\mathcal{V}$ and $\mathcal{G}$, to perform experiments using the holdout validation strategy, as was done in the previous chapter. However, very few large databases containing real classification problems are available in the literature. SOCS is general enough to be conducted using small datasets by applying k-fold cross-validation. Accordingly, we performed experiments using both the holdout and 10-fold cross-validation strategies. Another important aspect taken into account in selecting the databases for our experiments is that relatively high-dimensional feature spaces are necessary for the RSS method.

Two databases were used in the holdout validation experiments: (1) NIST Special Database 19 containing digits (NIST SD19), used in the previous chapter, which we call NIST-digits here; and (2) NIST SD19 containing handwritten uppercase letters, which we call NIST-letters here. We use the representation proposed by Oliveira et al. [52] for both databases. Table VI lists important information about these two databases and the partitions used to compose the four separate datasets. These same partitions were used

in [89] for NIST-digits and in [61] for NIST-letters. We used the two test datasets from NIST-digits, as was done in the previous chapter.

Table VII describes the three databases used in the 10-fold cross-validation experiments: dna, texture and satimage. The **Dna** and **satimage** datasets are provided by Project Statlog on www.niaad.liacc.up.pt/old/statlog; and **texture** is available within the UCI machine Learning Repository.

Table VI

Specifications of the large datasets used in the experiments in section 3.3.3.

Dataset	**# of features**	**Training Set ($\mathcal{T}$)**	**Optimization Set ($\mathcal{O}$)**	**Validation Set ($\mathcal{V}$)**	**Test Set ($\mathcal{G}$)**	**Features RSS**	**Pool $\mathcal{C}$ size**
NIST-digits	132	5,000	10,000	10,000	test1 60,089 test2 58,646	32	100
NIST-letters	132	43,160	3,980	7,960	12,092	32	100

Table VII

Specifications of the small datasets used in the experiments in section 3.3.4.

Dataset	**# of samples**	**# of features**	**Features RSS**	**Pool $\mathcal{C}$ size**
dna	3186	180	45	100
texture	5500	40	20	100
satimage	6435	36	18	100

Ensemble Construction Methods and Base Classifiers

We chose kNN and DT as the base classifiers in our experiments. The C4.5 algorithm [60] (Release 8) was used to construct the trees with pruning. In addition, we used $k = 1$ for kNN classifiers in all databases without fine-tuning this parameter in order to avoid additional experiments. BAG and RSS were applied to generate the initial pools of classifiers in our experiments. Three initial pools of 100 classifiers were created: (1) 100 DT and (2)

100 kNN, which were generated using RSS, and (3) 100 DT, which was generated using BAG (BAG is mostly effective with unstable classifiers, such as DT). The size of the subsets of features used by RSS is shown in Table VI for large datasets and in Table VII for small datasets. The same subspaces are used for both kNN and DT classifiers. Majority voting was used as the combination function.

Objective Functions

In order to reduce redundancy in our experiments, we chose to employ only four diversity measures: (1) difficulty measure (θ), which was pointed out as the best diversity measure when the diversity measures were compared in single-objective optimization problems; (2) double-fault (δ), which was one of the best diversity measure used in combination with the error rate to guide NSGA-II, (3) coincident failure diversity (σ), which was chosen due to the conclusions presented by Kuncheva and Whitaker [44] indicating that this measure is less correlated to the other measures; (4) ambiguity (γ) as defined in [104], which was not investigated in [44]. Kuncheva and Whitaker [44] studied ten diversity measures. They conclude that these diversity measures may be divided into three different groups, taking into account the correlation among measures: the double-fault alone; coincident failure diversity (also alone); and the remaining eight diversity measures.

In terms of ensemble size, we have shown in the previous chapter that the reduction in the number of classifiers is a consequence of the optimization task, whatever the objective function used to guide the search. Hence, there is no need to explicitly include ensemble size in the optimization process.

Genetic Algorithms

The same parameters used in the previous chapter is also employed in our experiments in this chapter.

3.3.2 Experimental Protocol

Our experiments were broken down into three main series. In the first and second series, the holdout and 10-fold cross-validation strategies were implemented to verify the impact of overfitting in large and small databases respectively. The validation control methods PV, BV and GV were also compared in order to determine the best method for controlling overfitting. Diversity measures were employed in pairs of objective functions combined with ϵ by NSGA-II, and only ϵ was used to guide GA in both series of experiments. Finally, in the third series, diversity measures were applied individually as single-objective functions. As a result of this last series of experiments, we show that the relationship between diversity and performance may be measured using the GV strategy.

The selection phase performed for large datasets was also replicated 30 times, as was done in the previous chapter, owing to the use of stochastic search algorithms. To conduct the 10-fold cross-validation experiments, the original whole datasets were divided into 10 folds. Each time, one of the 10 folds was used as $\mathcal{G}$, another fold as $\mathcal{V}$, a third as $\mathcal{O}$ and the other 7 were put together to form $\mathcal{T}$. Thus, $\mathcal{T}$, $\mathcal{O}$ and $\mathcal{V}$ were used at the overproduction phase to generate the initial pools of classifiers, to perform the optimization process and to perform the three validation strategies respectively. This process was repeated 10 times, i.e. the optimization process was repeated for 10 trials. It is also important to mention that the selection phase was replicated 30 times for each trial. Thus, the mean of the error rates over 30 replications for each trial were computed and the error rates reported in all tables of results were obtained as the mean of the error rates across all 10 trials.

Then, in both strategies, the best solution for each run was picked up according to the overfitting control strategy employed. The solutions were tested on multiple comparisons using the Kruskal-Wallis nonparametric statistical test by testing the equality between mean values. The confidence level was 95% ($\alpha = 0.05$), and the Dunn-Sidak correction was applied to the critical values.

3.3.3 Holdout validation results

Table VIII shows the mean error rates obtained using both GA and NSGA-II search algorithms in **NIST-digits data-test1** and **data-test2**, and in **NIST-letters**. The error rates obtained by combining the initial pool of 100 classifiers are also included in this table as well as the results with no overfitting control, denoted NV.

These experiments showed the following:

a. The results from the literature [37], which conclude that complex learning problems are more affected by overfitting, are confirmed. There are more problems presenting overfitting in **NIST-digits data-test2** than in **data-test1**. As we mentioned in section 2.3.1, **NIST-digits data-test2** is more difficult to use for classification.

b. NSGA-II is more prone to overfitting than GA. Even though, in the majority of the experiments using GA, at least one of the validation methods slightly decreased the error rates when compared with NV, the differences are not significant. In contrast, these differences are more likely to be significant in experiments with NSGA-II. Of 36 cases using NSGA-II, an overfitting control decreased the error rates in 30. In 17 of these experiments, the differences were significant.

c. When overfitting was detected, GV outperformed both PV and BV. The Kruskal-Wallis test shows that, among 19 cases where overfitting control decreased the error rates significantly, GV was the best strategy in 18 problems.

d. In terms of ensemble creation methods, our results indicate an order relation between the methods investigated for the **NIST-digits** database. BAG was more prone to overfitting than RSS. In addition, ensembles of DT were less prone to overfitting than ensembles of kNN, both generated using RSS. For the **NIST-letters** database, the results were equivalent.

Table VIII

Mean and standard deviation values of the error rates obtained on 30 replications comparing selection procedures on large datasets using GA and NSGA-II. Values in bold indicate that a validation method decreased the error rates significantly, and underlined values indicate that a validation strategy is significantly better than the others.

NIST-digits - test1 (100kNN = 3.72; 100DT-RSS = 2.92; 100DT-BAG = 5.65)						
		GA	**NSGA-II**			
Method	**Val**	error (ϵ)	ambiguity (γ)	coincident (σ)	difficulty (θ)	double-fault (δ)
kNN-RSS	NV	3.60 (0.06)	3.66 (0.06)	3.67 (0.03)	3.64 (0.05)	3.63 (0.07)
	PV	3.60 (0.06)	3.70 (0.07)	3.68 (0.04)	3.63 (0.05)	3.64 (0.07)
	BV	3.57 (0.07)	3.70 (0.07)	3.65 (0.03)	3.63 (0.07)	3.64 (0.07)
	GV	**3.55** (0.06)	3.63 (0.06)	3.62 (0.06)	3.60 (0.08)	3.60 (0.09)
DT-RSS	NV	2.82 (0.05)	2.96 (0.10)	2.92 (0.07)	2.81 (0.03)	2.81 (0.04)
	PV	2.80 (0.05)	2.97 (0.10)	2.97 (0.11)	2.83 (0.02)	2.82 (0.05)
	BV	2.83 (0.05)	2.97 (0.10)	2.97 (0.11)	2.83 (0.02)	2.82 (0.06)
	GV	2.84 (0.06)	2.94 (0.08)	2.94 (0.09)	2.82 (0.06)	2.84 (0.07)
DT-BAG	NV	5.20 (0.07)	5.73 (0.07)	5.87 (0.12)	5.67 (0.06)	5.73 (0.08)
	PV	5.18 (0.07)	5.71 (0.08)	5.86 (0.10)	5.68 (0.05)	5.73 (0.08)
	BV	5.18 (0.05)	5.71 (0.08)	5.85 (0.10)	5.69 (0.05)	5.72 (0.09)
	GV	5.18 (0.06)	**5.54** (0.06)	**5.56** (0.11)	**5.55** (0.09)	**5.57** (0.07)
NIST-digits - test2 (100kNN = 8.10; 100DT-RSS = 6.67; 100DT-BAG = 10.99)						
kNN-RSS	NV	7.91 (0.14)	7.90 (0.14)	8.09 (0.16)	8.12 (0.09)	8.12 (0.09)
	PV	7.89 (0.12)	7.97 (0.15)	8.11 (0.18)	8.11 (0.11)	8.14 (0.12)
	BV	7.85 (0.17)	7.97 (0.15)	8.10 (0.18)	8.11 (0.10)	8.14 (0.14)
	GV	7.80 (0.13)	7.87 (0.13)	**7.94** (0.15)	**7.93** (0.10)	**7.93** (0.13)
DT-RSS	NV	6.53 (0.08)	6.76 (0.17)	6.77 (0.14)	6.59 (0.11)	6.60 (0.10)
	PV	6.50 (0.09)	6.79 (0.18)	6.88 (0.21)	6.69 (0.06)	6.65 (0.11)
	BV	6.53 (0.09)	6.79 (0.18)	6.88 (0.21)	6.69 (0.06)	6.65 (0.12)
	GV	6.53(0.09)	6.73 (0.13)	6.76 (0.12)	6.59 (0.09)	6.63 (0.13)
DT-BAG	NV	10.16(0.15)	10.99 (0.17)	11.28 (0.28)	11.05 (0.08)	11.13 (0.08)
	PV	10.11 (0.15)	10.96 (0.17)	11.25 (0.23)	11.06 (0.11)	11.12 (0.21)
	BV	10.11(0.13)	10.94 (0.17)	11.24 (0.24)	11.06 (0.10)	11.10 (0.14)
	GV	10.09(0.13)	**10.68** (0.11)	**10.76** (0.22)	**10.76** (0.15)	**10.83** (0.07)
NIST-letter (100kNN = 6.60; 100DT-RSS = 6.06; 100DT-BAG = 7.63)						
kNN-RSS	NV	6.49 (0.11)	6.58 (0.10)	6.73 (0.25)	6.41 (0.09)	6.55 (0.13)
	PV	6.50 (0.09)	6.58 (0.11)	6.71 (0.26)	6.43 (0.11)	6.60 (0.12)
	BV	6.47 (0.12)	6.58 (0.12)	6.71 (0.26)	6.43 (0.11)	6.58 (0.12)
	GV	6.49 (0.09)	6.54 (0.14)	**6.50** (0.13)	6.33 (0.16)	**6.45** (0.12)
DT-RSS	NV	6.03 (0.07)	6.24 (0.11)	6.34 (0.19)	5.93 (0.08)	6.10 (0.12)
	PV	6.03 (0.07)	6.25 (0.11)	6.34 (0.19)	5.92 (0.06)	6.11 (0.12)
	BV	6.05 (0.09)	6.24 (0.11)	6.34 (0.19)	5.92 (0.07)	6.03 (0.13)
	GV	6.01 (0.09)	**6.08** (0.09)	**6.12** (0.14)	5.89 (0.07)	6.02 (0.10)
DT-BAG	NV	7.71 (0.10)	7.68 (0.09)	8.00 (0.14)	7.56 (0.08)	7.78 (0.12)
	PV	7.69 (0.08)	7.64 (0.07)	8.08 (0.18)	7.51 (0.05)	7.78 (0.13)
	BV	7.71 (0.10)	7.64 (0.07)	8.03 (0.16)	7.51 (0.05)	7.76 (0.12)
	GV	7.70 (0.08)	7.64 (0.07)	**7.87** (0.14)	7.54 (0.08)	**7.64** (0.12)

e. Although this thesis does not focus on comparing ensemble creation methods in terms of performance, our results indicate that ensembles generated with BAG performed worse than ensembles generated by RSS. Also, DT ensembles performed better than kNN.

f. The selection of ensembles of classifiers was a better option than combining the initial pools of classifiers. For **NIST-digits**, GA using ϵ as the objective function, and for **NIST-letters**, NSGA-II using θ and ϵ as the pair of objective functions, found solutions better than the baseline composed of 100 classifiers in all three problems.

3.3.4 Cross-validation results

Table IX summarizes the mean values and the standard deviation values achieved on the three small datasets presented in Table VII. We also report the error rates obtained on combining the initial pools of classifiers by majority voting.

Based on these results, it may be observed that:

a. For small databases, NSGA-II was also more prone to overfitting than GA. The overfitting control strategies significantly decreased the error rates of the solutions found by GA only in one case (**dna** with kNN-based ensembles) out of nine, while they significantly reduced the error rates of the solutions found by NSGA-II in 16 of 36 cases.

b. In 15 cases in which overfitting was detected, the difference between the overfitting control methods was significant. GV was also the best strategy for controlling overfitting in this series of experiments.

c. **Texture** is a highly stable database: it has the same class distribution and it can be easily classified, since the error rates can be lower than 1% (see results obtained

Table IX

Mean and standard deviation values of the error rates obtained on 30 replications comparing selection procedures on small datasets using GA and NSGA-II. Values in bold indicate that a validation method decreased the error rates significantly, and underlined values indicate when a validation strategy is significantly better than the others.

Texture (100kNN = 1.11; 100DT-RSS = 2.56; 100DT-BAG = 3.60)						
		GA	**NSGA-II**			
Method	**Val**	error (ϵ)	ambiguity (γ)	coincident (σ)	difficulty (θ)	double-fault (δ)
kNN-RSS	NV	1.09 (0.03)	1.28 (0.03)	1.19 (0.06)	0.87 (0.03)	0.90 (0.05)
	PV	1.09 (0.04)	1.40 (0.06)	1.21 (0.07)	0.84 (0.03)	0.94 (0.06)
	BV	1.08 (0.04)	1.41 (0.06)	1.21 (0.08)	0.88 (0.04)	0.89 (0.06)
	GV	1.11 (0.03)	1.26 (0.06)	1.18 (0.08)	0.95 (0.05)	0.98 (0.09)
DT-RSS	NV	2.54 (0.07)	2.94 (0.11)	3.16 (0.15)	2.40 (0.05)	3.00 (0.14)
	PV	2.52 (0.09)	3.03 (0.14)	3.41 (0.19)	2.41 (0.06)	3.09 (0.11)
	BV	2.55 (0.08)	3.04 (0.15)	3.42 (0.15)	2.44 (0.06)	3.10 (0.10)
	GV	2.51 (0.09)	2.93 (0.15)	3.20 (0.21)	2.41 (0.09)	2.91 (0.18)
DT-BAG	NV	3.58 (0.10)	3.60(0.09)	4.09 (0.15)	3.49 (0.09)	4.02 (0.14)
	PV	3.53 (0.08)	3.63 (0.13)	4.18 (0.20)	3.55 (0.08)	4.14 (0.19)
	BV	3.60 (0.08)	3.62 (0.11)	4.17 (0.18)	3.46 (0.12)	4.14 (0.20)
	GV	3.58 (0.08)	3.63 (0.10)	4.01 (0.15)	3.43 (0.11)	4.04 (0.22)
DNA (100kNN = 6.87; 100DT-RSS = 5.05; 100DT-BAG = 5.02)						
kNN-RSS	NV	8.01 (0.27)	9.61 (0.36)	10.60 (0.51	8.86 (0.22	9.21 (0.35)
	PV	8.02 (0.2)	9.85 (0.35)	10.85 (0.55)	8.93 (0.20)	9.38 (0.39)
	BV	**7.64** (0.24)	9.82 (0.37)	10.76 (0.52)	8.85 (0.21)	9.37 (0.41)
	GV	**7.69** (0.23)	**8.60** (0.37)	**8.75** (0.45)	**8.01** (0.33)	**8.36** (0.30)
DT-RSS	NV	5.10 (0.24)	6.44 (0.25)	7.05 (0.43)	5.15 (0.18)	6.20 (0.27)
	PV	4.99 (0.18)	6.76 (0.31)	7.30 (0.46)	4.96 (0.12)	6.31 (0.30)
	BV	4.98 (0.25)	6.76 (0.30)	7.30 (0.47)	4.93 (0.14)	6.30 (0.30)
	GV	4.97 (0.16)	**5.93** (0.29)	**6.36** (0.42)	**4.77** (0.17)	**5.67** (0.33)
DT-BAG	NV	5.00(0.10)	5.35 (0.14)	5.51 (0.22)	5.13 (0.14)	5.55 (0.23)
	PV	4.99 (0.11)	5.50 (0.12)	5.69 (0.20)	5.08 (0.12)	5.35 (0.14)
	BV	4.98 (0.15)	5.51 (0.11)	5.69 (0.20)	4.99 (0.12)	5.38 (0.15)
	GV	4.99 (0.08)	5.36 (0.20)	5.41 (0.21)	4.93 (0.13)	5.54 (0.22)
Satimage (100kNN = 8.59; 100DT-RSS = 8.64; 100DT-BAG = 9.59)						
kNN-RSS	NV	8.64 (0.09)	8.76 (0.12)	9.12 (0.16)	8.81 (0.23)	9.15 (0.14)
	PV	8.62 (0.07)	8.76 (0.09)	9.25 (0.17)	8.88 (0.25)	9.28 (0.13)
	BV	8.57 (0.10)	8.75 (0.09)	9.23 (0.18)	8.88 (0.26)	9.33 (0.14)
	GV	8.58 (0.11)	**8.69** (0.10)	9.05 (0.17)	8.84 (0.18	9.15 (0.17)
DT-RSS	NV	8.83 (0.11)	8.92 (0.11)	9.44 (0.20)	8.88 (0.10)	9.25 (0.19)
	PV	8.80 (0.09)	9.00 (0.12)	9.63 (0.22)	8.93 (0.12)	9.38 (0.18)
	BV	8.81 (0.09)	9.00 (0.10)	9.61 (0.23)	8.94 (0.11)	9.39 (0.17)
	GV	8.82 (0.12)	**8.77** (0.11)	**9.19** (0.18)	**8.75** (0.13)	**9.03** (0.17)
DT-BAG	NV	9.63 (0.09)	9.81 (0.10)	10.40 (0.19)	9.91 (0.13)	10.14 (0.18)
	PV	9.61 (0.07)	9.81 (0.12)	10.55 (0.18)	9.89 (0.09)	10.34 (0.17)
	BV	9.65 (0.10)	9.82 (0.12)	10.53 (0.17)	9.91 (0.09)	10.35 (0.16)
	GV	9.63 (0.09)	**9.65** (0.14)	10.25 (0.22)	**9.71** (0.13)	10.10 (0.16)

using θ and ϵ to guide NSGA-II). Hence, it is not surprising that our results do not show overfitting in this database.

d. For **dna**, overfitting was detected in the two problems involving ensembles generated using RSS. For **satimage**, DT-based ensembles generated using RSS were more prone to overfitting than the other two methods. Thus, unlike the **NIST-digits** results, BAG was less prone to overfitting than RSS for these two databases.

e. RSS-based ensembles presented lower error rates than BAG-based ensembles in all three databases.

f. The solutions found by NSGA-II guided by θ and ϵ were better than the baseline combination for **texture** and **dna**, except for ensembles of kNN for **dna**. However, for **satimage**, selection did not outperform combination.

3.3.5 Relationship between performance and diversity

When GA was guided by diversity measures as single-objective functions, we observed that a relationship between diversity measures and performance could be measured using the GV strategy, as illustrated in Figure 15. The same optimization problem investigated in section 3.1 is shown here. GA was guided by the minimization of θ as the objective function. It is important to mention that the optimization was only guided by θ. The reason for showing plots of ϵ versus θ in this figure is to demonstrate the relationship between this diversity measure and performance.

Figure 15 shows all C_j evaluated during the optimization process (points) and C_j^* (in terms of θ values) found on $\mathcal{O}$. It is interesting to note that Figure 16 confirms that the overfitting problem is also detected when θ is the objective function used to guide GA, since $\epsilon(\mathcal{V}, C_j^*) > \epsilon(\mathcal{V}, C_j^{*'})$. Thus, in this example, there is an overfitting measured by $\epsilon(\mathcal{V}, C_j^{*'}) - \epsilon(\mathcal{V}, C_j^*) = 0.23$. The use of GV allows us to keep $(C_j^{*'})$ stored in $\mathcal{A}$. However, Figure 16 also shows that $C_j^{*'}$ is not the solution with the smallest ϵ among all candidate

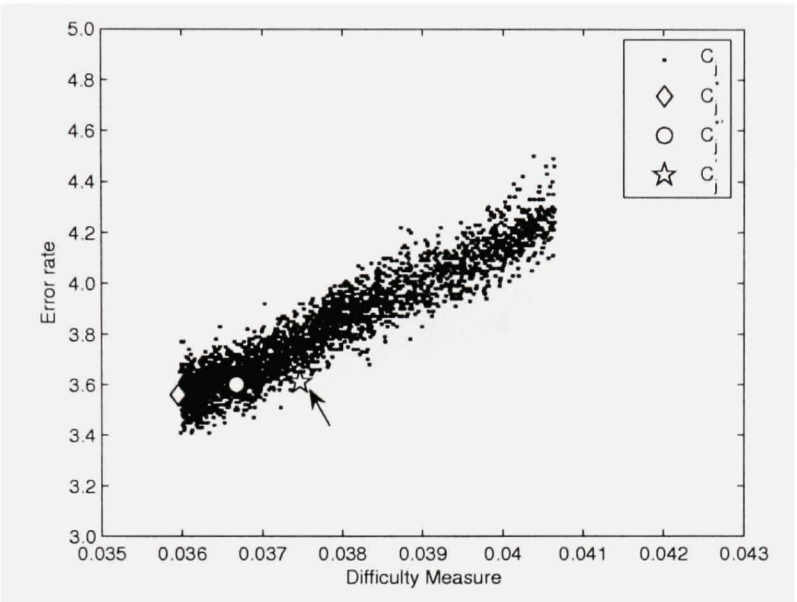


Figure 15 Optimization using GA with the θ as the objective function. all C_j evaluated during the optimization process (points), C_j^* (diamonds), $C_j^{*'}$ (circles) and C_j' (stars) in $\mathcal{O}$. Arrows highlight C_j'.

ensembles evaluated. In fact, the solution denoted C_j' is the candidate ensemble with lowest ϵ.

These observations lead us to propose the use of GV as a tool to measure the relationship between diversity and performance. The assumption is that there is a relationship between these two measures when the generalization performance is increased by keeping the solution $C_j^{*'}$ stored in $\mathcal{A}$ (based on diversity values). In addition, the relationship may range from weak to strong, based on the difference (overfitting) between $\epsilon(\mathcal{V}, C_j^{*'})$ and $\epsilon(\mathcal{V}, C_j')$. For instance, since $\epsilon(\mathcal{V}, C_j') - \epsilon(\mathcal{V}, C_j^{*'}) = 0.05$ in Figure 16, a 0.23% overfitting was controlled using GV and a 0.05% overfitting remains uncontrolled. Because this difference is close to 0, θ is strongly related to performance in this problem.

Taking into account that the relationship between diversity and performance has been measured in the literature, using correlation measures [44] and kappa-error diagrams [17] for example, the use of GV offers a different strategy for analyzing such a key aspect in the

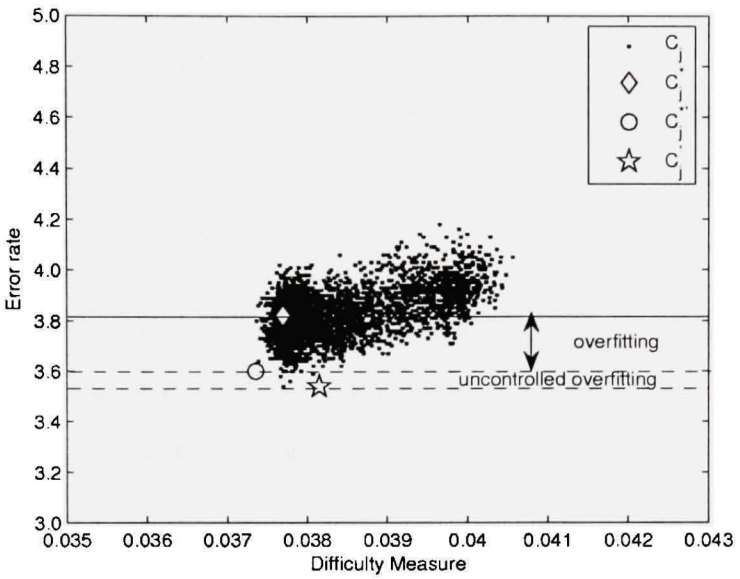


Figure 16 Optimization using GA with the θ as the objective function. All C_j evaluated during the optimization process (points), C_j^* (diamonds), $C_j^{*'}$ (circles) and C_j' (stars) in $\mathcal{V}$. Controlled and uncontrolled overfitting using GV.

ensembles of classifiers literature. However, taking into account the results obtained previously, there may be a drawback to this strategy: diversity measures are not independent of ensemble size, i.e. when diversity measures are employed as single-objective functions, a minimization of the number of classifiers may result. It is expected that ensembles which are too small will perform considerably less well than other approaches [1; 50], such as combining the initial pools of classifiers or selecting the best subset of classifiers using GA only guided by ϵ or MOGA guided by ϵ combined with diversity.

To avoid the problem of reaching a too small ensemble size, we defined a large fixed minimum ensemble size for all diversity measures in this series of experiments. The minimum ensemble size was fixed based on the median size of the ensembles obtained using ϵ as a single-objective function calculated on the three small datasets described in Table VII. Taking into account these results, we set the minimum ensemble size to 49 classifiers.

The four diversity measures investigated in this chapter were used in this series of experiments. The objective of the experiments was to verify which measure is the most closely related to performance. The relationship between the four diversity measures and performance is measured by calculating the uncontrolled overfitting $\phi = \epsilon(\mathcal{V}, C'_j) - \epsilon(\mathcal{V}, C'^*_j)$. The lower ϕ, the stronger the relationship.

Table X summarizes the results obtained. We have also included the results achieved using GA guided only by ϵ, so that the solutions found by all five single-objective functions investigated in this chapter can be compared. These results are the same as those shown in Table IX.

Our results indicate that the differences among the results obtained with diversity measures are small, except for **dna** in RSS-based ensembles. In fact, we confirmed the results of previous work, e.g. [70] and [44], that diversity measures are highly correlated. However, our experiments reveal important information about the relationship between performance and diversity. Taking into account the assumption that the closer to zero the uncontrolled overfitting, the stronger the relationship between the two measures, we can observe the following results:

a. The diversity measure δ was more closely related to performance. The uncontrolled overfitting ϕ was either zero or as close as possible to zero in 5 cases out of a total of 9 cases.

b. The diversity measure γ was less closely related to performance. The results obtained using γ presented the highest ϕ in 6 cases.

c. θ was highly related to performance and appears to be better than ϵ for guiding the selection of the ensembles of classifiers. In 8 cases, θ found solutions better than those found using ϵ as a single-objective function. In addition, the ensembles found

Table X

Mean and standard deviation values of the error rates obtained on measuring the uncontrolled overfitting. The relationship between diversity and performance is stronger as ϕ decreases. The best result for each case is shown in bold.

Method	Strategy	ambiguity (γ)	coincident (σ)	difficulty (θ)	double-fault (δ)
Texture					
	$\epsilon(\mathcal{V}, C_j^{*'})$	1.20 (0.04)	1.12 (0.03)	**1.01** (0.03)	1.01 (0.02)
kNN-RSS	$\epsilon(\mathcal{V}, C_j')$	1.12 (0.03)	1.09 (0.04)	1.09 (0.04)	1.09 (0.05)
GA ϵ = 1.11	ϕ	0.08 (0.03)	0.03 (0.04)	-0.08 (0.04)	-0.08 (0.03)
	$\epsilon(\mathcal{V}, C_j^{*'})$	2.80 (0.04)	2.50 (0.08)	**2.40** (0.06)	2.46 (0.07)
DT-RSS	$\epsilon(\mathcal{V}, C_j')$	2.63 (0.08)	2.50 (0.07)	2.43 (0.09)	2.46 (0.09)
GA ϵ = 2.51	ϕ	0.18 (0.06)	0.00 (0.07)	-0.03 (0.07)	0.00 (0.08)
	$\epsilon(\mathcal{V}, C_j^{*'})$	3.59 (0.09)	3.56 (0.07)	**3.48** (0.06)	3.49 (0.06)
DT-BAG	$\epsilon(\mathcal{V}, C_j')$	3.61 (0.09)	3.61 (0.08)	3.54 (0.06)	3.57 (0.08)
GA ϵ = 3.58	ϕ	-0.02(0.09)	-0.05 (0.08)	-0.06 (0.06)	-0.08 (0.07)
Dna					
	$\epsilon(\mathcal{V}, C_j^{*'})$	10.6 (0.00)	8.33 (0.21)	**7.50** (0.25)	7.89 (0.16)
kNN-RSS	$\epsilon(\mathcal{V}, C_j')$	9.05 (0.00)	8.00 (0.33)	8.44 (0.30)	7.75 (0.29)
GA ϵ = 7.69	ϕ	1.56 (0.00)	0.33 (0.27)	-0.94 (0.30)	0.14 (0.20)
	$\epsilon(\mathcal{V}, C_j^{*'})$	7.40 (0.16)	5.01 (0.18)	**4.50** (0.16)	4.67 (0.10)
DT-RSS	$\epsilon(\mathcal{V}, C_j')$	5.17 (0.18)	5.09 (0.20)	5.24 (0.22)	4.91 (0.19)
GA ϵ = 4.97	ϕ	2.23 (0.17)	-0.07(0.19)	-0.73 (0.19)	-0.24 (0.14)
	$\epsilon(\mathcal{V}, C_j^{*'})$	5.02 (0.11)	5.16 (0.10)	**4.93** (0.09)	4.94 (0.07)
DT-BAG	$\epsilon(\mathcal{V}, C_j')$	5.00 (0.10)	5.08 (0.10)	5.05 (0.12)	5.01 (0.09)
GA ϵ = 4.99	ϕ	0.03 (0.10)	0.08 (0.10)	-0.12 (0.10)	-0.07 (0.08)
Satimage					
	$\epsilon(\mathcal{V}, C_j^{*'})$	8.60 (0.09)	**8.58** (0.11)	8.64 (0.08)	8.62 (0.10)
kNN-RSS	$\epsilon(\mathcal{V}, C_j')$	8.55 (0.07)	8.60 (0.08)	8.66 (0.12)	8.62 (0.09)
GA ϵ = 8.58	ϕ	0.05 (0.08)	-0.02 (0.09)	-0.02 (0.10)	0.00 (0.09)
	$\epsilon(\mathcal{V}, C_j^{*'})$	**8.22** (0.12)	8.77 (0.11)	8.34 (0.03)	8.67 (0.11)
DT-RSS	$\epsilon(\mathcal{V}, C_j')$	8.11 (0.12)	8.72 (0.11)	8.47 (0.14)	8.71 (0.11)
GA ϵ = 8.82	ϕ	0.11 (0.12)	0.05 (0.11)	-0.10 (0.08)	-0.04 (0.10)
	$\epsilon(\mathcal{V}, C_j^{*'})$	9.61 (0.09)	9.64 (0.13)	**9.61** (0.09)	9.71 (0.11)
DT-BAG	$\epsilon(\mathcal{V}, C_j')$	9.66 (0.15)	9.65 (0.13)	9.66 (0.12)	9.70 (0.10)
GA ϵ = 9.63	ϕ	-0.05 (0.12)	-0.01 (0.13)	-0.05 (0.10)	0.01 (0.10)

using θ presented the highest negative ϕ. What this means, in effect, is that guiding GV using θ is better than doing so using ϵ.

d. Our results indicate that diversity measures can be effective objective functions for selecting high-performance ensembles of classifiers. This is a different result from those presented in the previous chapter. The main problem is that diversity measures are critically affected by ensemble size. Since we have fixed quite a large ensemble size, i.e. 49 classifiers, diversity measure performances exceeded those of ϵ.

3.4 Discussion

This chapter presented the experimental results of a study comparing three overfitting control strategies adapted to the selection phase of SOCS, which is performed as an optimization problem using single- and multi-objective GAs. We showed that the task of selecting classifier ensembles may be prone to overfitting, we pointed out the best strategy for controlling overfitting and we presented a global validation strategy GV as a tool to measure the relationship between diversity and performance. The following overfitting control methods were investigated: (1) Partial Validation, in which only the last population of solutions is validated; (2) Backwarding, which relies on validating each best solution at each generation; and (3) GV, in which all solutions at each generation are validated. Three initial pools of 100 classifiers were generated: 100 kNN and 100 DT using RSS, and a third pool of 100 DT using BAG. Five different objective functions were applied: 4 diversity measures and ϵ.

The experiments were divided into three series. The three overfitting control methods were compared in two large databases (first series of experiments) and in three small datasets (second series). The combination error rate ϵ was employed as a single-objective function by GA and the four diversity measures were combined with ϵ to make up pairs of objective functions to guide NSGA-II. Finally, in the third series of experiments, the four diversity measures were directly applied by GA, and the GV strategy was performed to control overfitting. The objective in this third series of experiments was to show that GV can be a tool for identifying the diversity measure more closely related to performance.

The results show that overfitting can be detected at the selection phase of SOCS, especially when NSGA-II is employed as the search algorithm. In response to question (1) posed in the beginning of this chapter, GV may be deemed to be the best strategy for controlling overfitting. In all problems where overfitting was detected, in both large and small databases, GV outperformed the other overfitting control methods. In response to question (2), our results indicate that RSS-based ensembles are more prone to overfitting than BAG-based ones, even though the same behavior was not observed in all databases investigated. In terms of ensembles generated by RSS, our results globally showed that kNN ensembles and DT ensembles are equally affected by overfitting. This aspect is totally problem-dependent.

Finally, our results outlined a relationship between performance and diversity. Double-fault δ and ambiguity γ were the diversity measures more and less closely related to performance respectively. Moreover, we show that the difficulty measure θ can be better than ϵ as a single-objective function for selecting high-performance ensembles of classifiers. It is important, however, to realize that quite a large minimum ensemble size (49) was fixed for all diversity measures. Such a size was defined based on the size of the ensembles found by GA guided by ϵ.

Although we were able to increase the generalization performance of the baseline initial pools of classifiers by employing SOCS in chapter 2 and SOCS with overfitting control in this chapter, the selection of only one candidate ensemble to classify the whole test dataset, does not guarantee that the candidate ensemble most likely to be correct for classifying each test sample individually is selected. In the next chapter of this thesis we propose a dynamic OCS in order to investigate whether or not the gain in performance can be increased still further.

CHAPTER 4

DYNAMIC OVERPRODUCE-AND-CHOOSE STRATEGY

In chapter 2 and chapter 3, we have considered static overproduce-and-choose strategy (SOCS). We have investigated search criteria and search algorithms to guide the optimization process and how this may be affected by the overfitting problem. We also presented methods to control overfitting. The objective was to finding the most accurate subset of classifiers during the selection phase, and using it to predict the class of all the samples in the test dataset. In this chapter, we propose a dynamic overproduce-and-choose strategy, which relies on combining optimization and dynamic selection to compose a two-level selection phase. The optimization process is conducted to generate candidate classifier ensembles and the dynamic selection is performed by calculating a measure of confidence to allow the selection of the most confident subset of classifiers to label each test sample individually.

In our approach, an ensemble creation method, such as bagging or the random subspace method, is applied to obtain the initial pool of candidate classifiers C at the overproduction phase. Thus, the optimization process described in chapter 2 is assumed to be the first level of the selection phase. At this first level, a population-based search algorithm is employed to generate a population $\mathbf{C}^{*'} = \{C_1', C_2', \ldots, C_w'\}$ of highly accurate candidate ensembles C_j'. This population is denoted $\mathbf{C}^{*'}$ to indicate that it was obtained by using the validation process described in chapter 3. Assuming $\mathbf{C}^*$ as the population found using the optimization dataset, $\mathbf{C}^{*'}$ is the alternative population found using the validation dataset to avoid overfitting. At the second level proposed in this chapter, the candidate ensembles in $\mathbf{C}^{*'}$ are considered for dynamic selection in order to identify, for each test sample $\mathbf{x}_{i,g}$, the solution $C_j^{*'}$ most likely to be correct for classifying it.

In the introduction of this thesis we mentioned that our objective by proposing a dynamic overproduce-and-choose strategy is to overcome the following three drawbacks: Rather than selecting only one candidate ensemble found during the optimization level, as is done in SOCS, the selection of $C_j^{*'}$ is based directly on the test patterns. Our assumption is that the generalization performance will increase, since all potential high accuracy candidate ensembles from the population $\mathbf{C}^{*'}$ are considered to select the most competent solution for each test sample. This first point is particularly important in problems involving Pareto-based algorithms, because our method allows all equally competent solutions over the Pareto front to be tested; (2) Instead of using only one local expert to classify each test sample, as is done in traditional classifier selection strategies (both static and dynamic), the selection of a subset of classifiers may decrease misclassification; and, finally, (3) Our dynamic selection avoids estimating regions of competence and distance measures in selecting $C_j^{*'}$ for each test sample, since it relies on calculating confidence measures rather than on performance.

Therefore, in this chapter, we present a new method for dynamic selection of classifier ensembles. The proposed method relies on choosing dynamically from the population $\mathbf{C}^{*'}$ of highly accurate candidate ensembles generated at the optimization level, the candidate ensemble with the largest consensus to predict the test pattern class. We prove both theoretically and experimentally that this selection permits an increase in the "degree of certainty" of the classification [29], increasing the generalization performance as a consequence. We propose in this thesis the use of three different confidence measures, which measure the extent of consensus of candidate ensembles to guide the dynamic selection level of our method. These measures are: (1) *Ambiguity*; (2) *Margin*; and (3) *Strength relative to the closest class*.

The first measure is based on the definition of ambiguity shown in Equation 2.2, which indicates that the solution with least ambiguity among its members presents high confidence level of classification. The second measure, which is inspired by the definition of margin,

measures the difference between the number of votes assigned to the two classes with the highest number of votes, indicating the candidate ensemble's level of certainty about the majority voting class. Finally, the third measure also measures the difference between the number of votes received by the majority voting class and the class with the second highest number of votes; however, this difference is divided by the performance achieved by each candidate ensemble when assigning the majority voting class for samples contained in a validation dataset. This additional information indicates how often each candidate ensemble made the right decision in assigning the selected class. Different candidate ensembles may have different levels of confidence for the same class [8].

Besides these three new confidence measures, DCS-LA is also investigated. DCS-LA is classical dynamic classifier selection method [101]. It has been summarized in chapter 1, more precisely in section 1.2.1. We have tailored DCS-LA to the selection of classifier ensembles to be compared to the three confidence-based dynamic selection methods. Following the same experimental protocol employed in chapter 3, in this chapter, bagging and the random subspace method are used to generate ensemble members, while DT and kNN classifiers are used for the creation of homogeneous ensembles at the overproduction phase, showing that the validity of our approach does not depend on the particulars of the ensemble generation method. Single- and multi-objective GAs are used to perform the optimization of our DOCS, employing the same five objective functions for assessing the effectiveness of candidate ensembles.

This chapter is organized as follows. Our proposed DOCS is introduced in section 4.1. Then, sections 4.2 and 4.3 describe the optimization and dynamic selection performed in the two-level selection phase by population-based GAs and confidence-based measures respectively. Finally, the parameters employed in the experiments and the results obtained are presented in section 4.4. Conclusions are discussed in section 4.5.

4.1 The Proposed Dynamic Overproduce-and-Choose Strategy

The method proposed in this chapter does not follow the classical definition of DCS (see section 1.2.1 in chapter 1), however, and so a partition generation procedure is not needed and the selection level is not based on accuracy. Instead, our approach is based on the definition of OCS. Traditionally, OCS is divided into two phases: (1) overproduction; and (2) selection. The former is devoted to constructing $\mathcal{C}$. The latter tests different combinations of these classifiers in order to identify the optimal solution, $C_j^{*'}$. Figure 17 shows that, in SOCS, $C_j^{*'}$ is picked up from the population of candidate ensembles, $\mathbf{C}^{*'}$, found and analyzed during the selection phase, and is used to classify all samples contained in $\mathcal{G}$. However, as mentioned in the introduction, there is no guarantee that the $C_j^{*'}$ chosen is indeed the solution most likely to be the correct one for classifying each $\mathbf{x}_{i,g}$ individually.

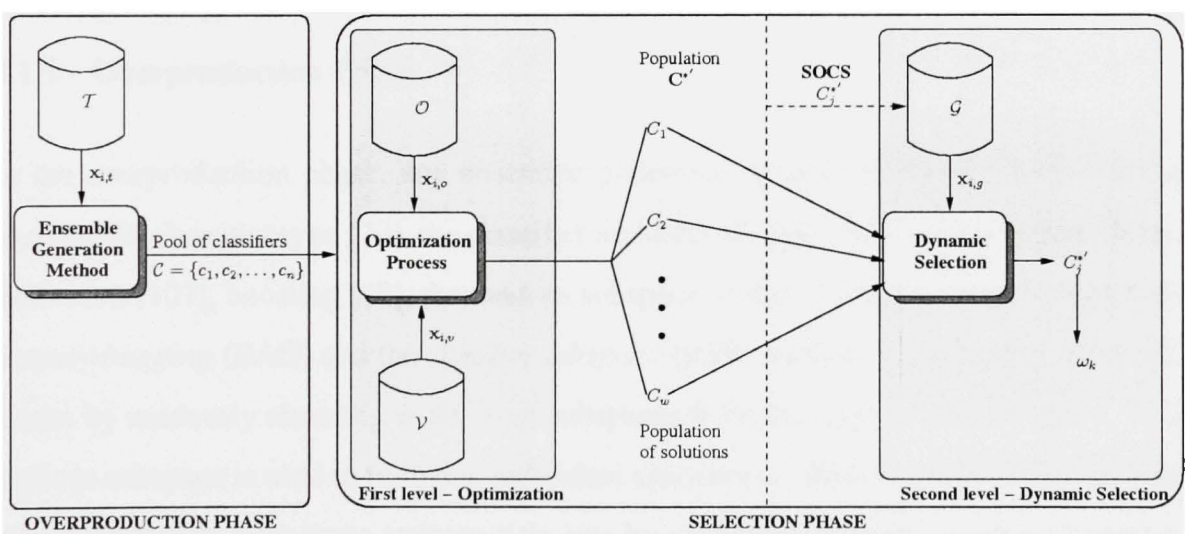


Figure 17 Overview of the proposed DOCS. The method divides the selection phase into optimization level, which yields a population of ensembles, and dynamic selection level, which chooses the most competent ensemble for classifying each test sample. In SOCS, only one ensemble is selected to classify the whole test dataset.

We propose a *dynamic* OCS in this chapter, as summarized in Figure 17 and Algorithm 7. The overproduction phase is performed, as defined in previous chapters, to generate

the initial pool of classifiers $\mathcal{C}$. Thus, the selection phase is divided into two levels: (1) optimization; and (2) dynamic selection. The optimization is performed by GAs guided by both single- and multi-objective functions as was done in all the experiments presented throughout this thesis. According to the global validation strategy defined in chapter 3 to avoid overfitting at this level, the optimization dataset ($\mathcal{O}$) is used by the search algorithm to calculate fitness, and the validation $\mathcal{V}$ is used to keep stored, in the auxiliary archive $\mathcal{A}$, the population of n best solutions for the GA or the Pareto front for NSGA-II found before overfitting starts to occur. As explained below, in the single-objective GA case, here we keep stored in $\mathcal{A}$ the population of n best solutions rather than only one solution, as was done in chapter 3. The population of solutions $\mathbf{C}^{*'}$, is further used at the dynamic selection level, which allows the dynamic choice of $C_j^{*'}$ to classify $\mathbf{x}_{i,g}$, based on the certainty of the candidate ensemble's decision. Finally, $C_j^{*'}$ is combined by majority voting.

4.1.1 Overproduction Phase

In the overproduction phase, any ensemble generation technique may be used, such as varying the classifier type [70], the classifier architecture [78], the learning parameter initialization [107], boosting [48], the random subspace method [72], etc. In this chapter, we employ *bagging* (BAG) and the *random subspace* (RSS) method to generate $\mathcal{C}$. RSS [32] works by randomly choosing n different subspaces from the original feature space. Each random subspace is used to train one individual classifier c_i. BAG is a bootstrap technique [5] which builds n replicate training data sets by randomly sampling, with replacement, from $\mathcal{T}$. Thus, each replicated dataset is used to train one c_i.

In the following sections we describe the two levels of the selection phase of our DOCS. Since we have considered and described the optimization level in previous chapters, this level is briefly presented. We concentrate here in our proposed dynamic selection level.

Algorithm 7 Dynamic Overproduce-and-Choose Strategy (DOCS)

```
Design a pool of classifiers C.
Perform the optimization level using a search algorithm to generate a population of candidate ensembles C*'.
for each test sample x_{i,g} do
  if all candidate ensembles agree on the label then
    classify x_{i,g} assigning it the consensus label.
  else
    perform the dynamic selection level calculating the confidence of solutions in C*';
    if a winner candidate ensemble is identified then
      select the most competent candidate ensemble C_j^{*'} to classify x_{i,g}
    else
      if a majority voting class among all candidate ensembles with equal competence is identified then
        assign the majority voting class to x_{i,g}
      else
        select the second highest competent candidate ensemble
        if a majority voting class among all candidate ensembles with the first and the second highest competence is identified then
          assign the majority voting class to x_{i,g}
        else
          randomly select a candidate ensemble to classify x_{i,g}
        end if
      end if
    end if
  end if
end for
```

4.2 Optimization Level

In order to clarify the proposed DOCS, we will use a case study obtained in one replication using the NIST-letters database (section 3.3.1) throughout this chapter. The initial pool of DT classifiers was generated using RSS. The maximum number of generations is fixed at $max(g) = 1,000$.

Figure 18(a) depicts the optimization level conducted as a single-objective problem. Although GA was guided by the minimization of the error rate ϵ as the objective function,

we show plots of ϵ versus the difficulty measure θ to better illustrate the process. Again, each point on the plot corresponds to a candidate ensemble, i.e. they represent all solutions evaluated for $max(g)$. The population $\mathbf{C}^{*'}$ (circles) is composed of the n best solutions. We fixed $n = 21$ (see section 4.4). In Figure 18(c), only the 21 best solutions are shown.

In SOCS, $C_j^{*'}$ is assumed to be the solution with the lowest ϵ (black circle in Figure 18(c)), without knowing whether the $C_j^{*'}$ chosen is indeed the best solution for correctly classifying each $\mathbf{x}_{i,g}$. Hence, the additional dynamic selection level proposed in this chapter is a post-processing strategy which takes advantage of the possibility of dealing with a set of high-performance solutions rather than only one. In this way, the whole population $\mathbf{C}^{*'}$ is picked up at the dynamic selection level of our method. The parameter n should be defined experimentally.

Figure 18(b) shows all the classifier ensembles evaluated using NSGA-II guided by the following pair of objective functions: jointly minimize θ and ϵ. Here, the Pareto front is assumed to be $\mathbf{C}^{*'}$ (circles in Figures 18(b) and 18(d)). Although the solutions over the Pareto front are equally important, the candidate ensemble with the lowest ϵ (black circle in Figure 18(d)) is usually chosen to be $C_j^{*'}$ in classical SOCS, as was done in [72] and [89].

Considering this case study, Table XI shows the results calculated using samples from $\mathcal{G}$ comparing SOCS and the combination of the initial pool of classifiers $\mathcal{C}$. The selection of a subset of classifiers using SOCS outperformed the combination of $\mathcal{C}$. It is interesting to observe that NSGA-II was slightly superior to GA, and that the sizes of the candidate ensembles in $\mathbf{C}^{*'}$ found using both GAs were smaller than the initial pool size. In addition, NSGA-II found a $C_j^{*'}$ even smaller than the solution found by GA. The assumption of proposing an additional dynamic selection level is that performance may still be increased when selecting $C_j^{*'}$ dynamically for each $\mathbf{x}_{i,g}$.

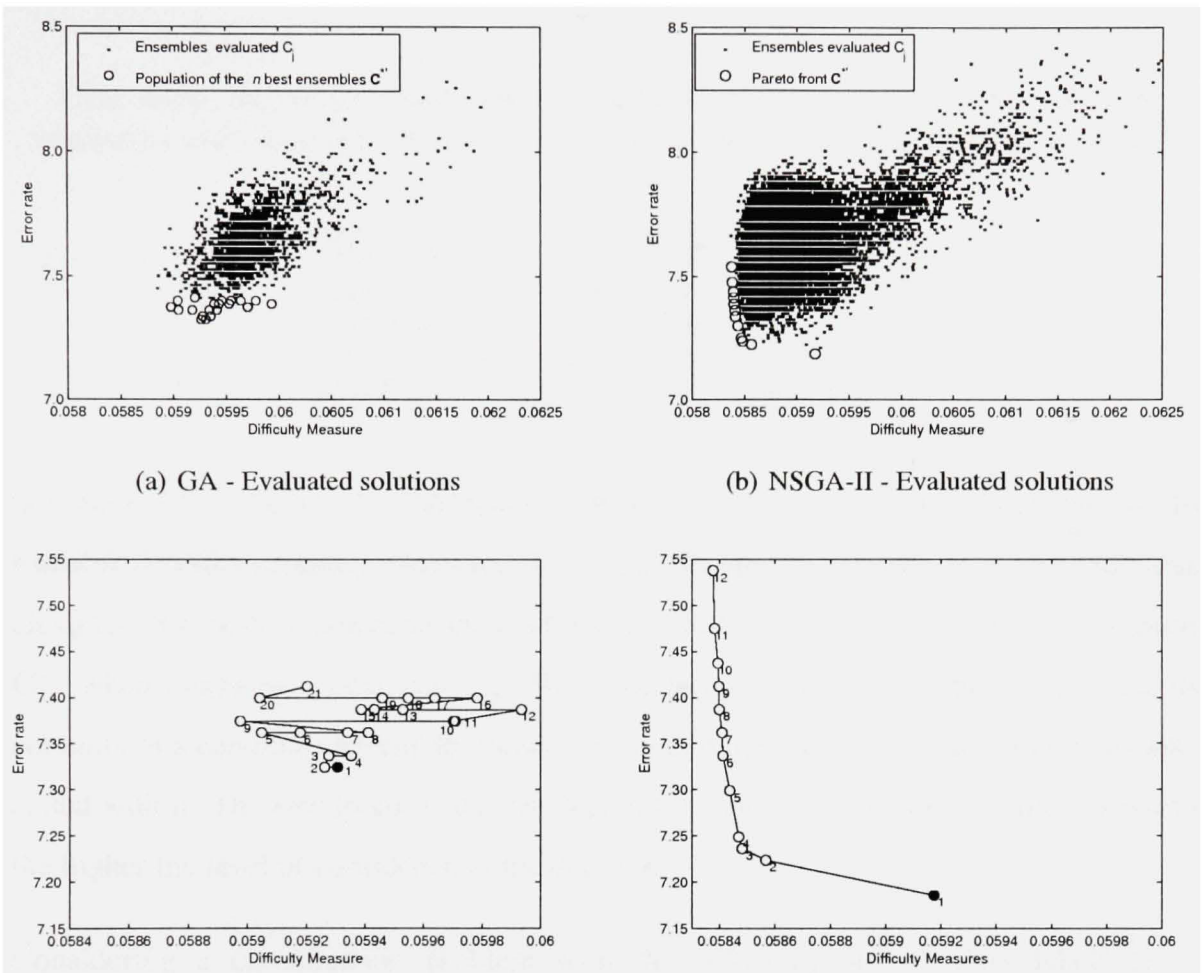


Figure 18 Ensembles generated using single-objective GA guided by ϵ in Figure 18(a) and NSGA-II guided by θ and ϵ in Figure 18(b). The output of the optimization level obtained as the n best solutions by GA in Figure 18(c) and the Pareto front by NSGA-II in Figure 18(d). These results were calculated for samples contained in $\mathcal{V}$. The black circles indicate the solutions with lowest ϵ, which are selected when performing SOCS.

4.3 Dynamic Selection Level

The selection process in classical DCS approaches is based on the certainty of the classifiers decision for each particular $\mathbf{x}_{i,g}$. Consequently, these methods explore the domain of expertise of each classifier to measure the degree of certainty of its decision, as described

Table XI

Case study: the results obtained by GA and NSGA-II when performing SOCS are compared with the result achieved by combining all classifier members of the pool $\mathcal{C}$.

Optimization Level	**Error rate**	$C_j^{*'}$ **size**	**Average size of** $C_j \in \mathbf{C}^{*'}$
Combination	6.06	100	-
NSGA-II (ϵ & θ)	5.86	45	50 (2.50)
GA (ϵ)	5.90	55	55 (3.89)

in section 1.2.1, chapter 1. The dynamic selection level proposed in this chapter is also based on decision certainty. However, instead calculating the confidence of each individual classifier, our method calculates the confidence of each candidate ensemble that composes $\mathbf{C}^{*'}$, when assigning a label for $\mathbf{x}_{i,g}$. We show below that it is possible to calculate the certainty of a candidate ensemble decision by measuring the extent of the consensus associated with it. The standpoint is that the higher the consensus among classifier members, the higher the level of confidence in the decision.

Considering a classification problem with the following set of class labels $\Omega = \{\omega_1, \omega_2 \ldots, \omega_c\}$, the confidence level is related to the *posterior probability* $P(\omega_k|\mathbf{x}_{i,g})$ that $\mathbf{x}_{i,g}$ comes from class ω_k. Hansen et al. [29] have observed that $P(\omega_k|\mathbf{x}_{i,g})$ may be calculated in the context of an ensemble of classifiers by measuring the extent of the consensus of ensembles, as given below:

Given the candidate ensemble $C_j = \{c_1, c_2, \ldots, c_l\}$ and the output of the i-th classifier $y_i(\mathbf{x}_{i,g})$, without loss of generality, we can assume that each classifier produces a class label as output. The number of votes $v(\omega_k|\mathbf{x}_{i,g})$ for class ω_k given $\mathbf{x}_{i,g}$ is obtained as follows:

$$v(\omega_k|\mathbf{x}_{i,g}) = |\{c_i : y_i(\mathbf{x}_{i,g}) = \omega_k\}| \tag{4.1}$$

Assuming majority voting as the combination function, the consensus decision is:

$$mv(\mathbf{x}_{i,g}) = \arg_{k=1}^{c} \max v(\omega_k|\mathbf{x}_{i,g}) \tag{4.2}$$

Thus, the extent of consensus on sample $\mathbf{x}_{i,g}$ is:

$$P(\omega_k|\mathbf{x}_{i,g}) = \frac{v(mv(\mathbf{x}_{i,g})|\mathbf{x}_{i,g})}{|C_j|} \tag{4.3}$$

The extent of consensus measures the number of classifiers in agreement with the majority voting. Consequently, by maximizing the extent of consensus of an ensemble, the degree of certainty that it will make a correct classification is increased. Another important point to mention is that no information on the correctness of the output is needed. These observations allow us to present three confidence measures that calculate the extent of consensus of each candidate ensemble from the population $\mathbf{C}^{*'}$, to be used at the dynamic selection level of our DOCS: (1) ambiguity, (2) margin, and (3) strength relative to the closest class. The first two measures are directly related to the evaluation of the extent of consensus. The third measure also considers the candidate ensembles' performance measured for each class involved in the classification problem. This additional information is calculated over samples contained in $\mathcal{V}$. In addition, DCS-LA is adapted to the context of DOCS to be compared to these three confidence-based strategies.

4.3.1 Ambiguity-Guided Dynamic Selection (ADS)

The classification ambiguity proposed by Zenobi and Cunningham [104] attempts to estimate the diversity of opinions among classifier members. This diversity measure, which is defined in Equation 2.2, chapter 2, appears to be well suited for the dynamic selection level we are proposing, since it does not need knowledge on the correctness of the decision.

It is important to note in Equation 2.2 that, if we calculate the ambiguity γ for a particular test sample $\mathbf{x}_{i,g}$ instead of calculating it for the whole dataset, γ becomes the complement of the extent of consensus in Equation 4.3. Denoting γ calculated for the given $\mathbf{x}_{i,g}$ as $\bar{\gamma}$, we may assume $\bar{\gamma} + P(\omega_k|\mathbf{x}_{i,g}) = 1$. Thus, Equation 2.2 simplifies to:

$$\bar{\gamma} = \frac{1}{|C_j|} \sum_{i \in C_j} a_i(\mathbf{x}_{i,g}) \tag{4.4}$$

Since such a local ambiguity measures the number of classifiers in disagreement with the majority voting, the minimization of $\bar{\gamma}$ leads to the maximization of the extent of consensus. Consequently, the certainty of correct classification is increased. In addition, although $\bar{\gamma}$ does not take into account the label of the given sample, the minimization of $\bar{\gamma}$ also leads to the maximization of the margin in the case of a correct classification. The so-called margin is a measure of confidence of classification. There are two general definitions of margin reported in the literature [27]. The first definition is presented below and the second is presented in section 4.3.2.

The classification margin for sample $\mathbf{x}_{i,g}$ is the following:

$$\mu(\mathbf{x}_{i,g}) = v(\omega_t|\mathbf{x}_{i,g}) - \sum_{k \neq t} v(\omega_k|\mathbf{x}_{i,g}) \tag{4.5}$$

where ω_t is the true class label of $\mathbf{x}_{i,g}$. Hence, the margin measures the difference between the number of votes assigned for the true class label and the number of votes given for any other class. Consequently, the certainty of the classification is increased by trying to maximize the margin. Based on the standpoints presented in this section, our dynamic level guided by $\bar{\gamma}$, denoted ADS, will pick up $C_j'^*$ as the candidate ensemble with lowest $\bar{\gamma}$. The assumption is that the candidate ensemble with the lowest $\bar{\gamma}$ presents the lowest possibility of making a mistake when classifying $\mathbf{x}_{i,g}$.

However, it is important to take into account the difference between γ (Equation 2.2) and $\bar{\gamma}$ (Equation 4.4). The former, called global ambiguity in this chapter, is used to guide the optimization level, and it is calculated for the whole dataset ($\mathcal{O}$ or $\mathcal{V}$). The latter, called local ambiguity, is used in the dynamic selection level calculated for each $\mathbf{x}_{i,g}$ individually. Since the global ambiguity is a dissimilarity measure (see III in chapter 2), it must be maximized at the optimization level.

ADS is summarized in Figure 19. In this example, the output of the optimization level is the Pareto front found by MOGA. $\bar{\gamma}$ is calculated for each solution over the Pareto front. Thus, the solution $C_j^{*'}$ with the lowest $\bar{\gamma}$ is selected and combined to assign the majority voting class mv to the test sample $\mathbf{x}_{i,g}$.

4.3.2 Margin-based Dynamic Selection (MDS)

The second measure proposed for use in guiding dynamic selection in our approach has been inspired by the second definition of the margin. Following this definition, the margin of sample $\mathbf{x}_{i,g}$ is computed as follows:

$$\mu(\mathbf{x}_{i,g}) = v(\omega_t|\mathbf{x}_{i,g}) - \max_{k \neq t} v(\omega_k|\mathbf{x}_{i,g}) \tag{4.6}$$

This equation calculates the difference between the number of votes given to the correct class $v(\omega_t|\mathbf{x}_{i,g})$ and the number of votes given to the incorrect class label with the highest number of votes. In our approach, however, $v(\omega_t|\mathbf{x}_{i,g})$ is unknown, since the dynamic selection is performed for test samples. In order to employ this measure to guide the dynamic selection of our DOCS, we have tailored the margin measure defined in Equation 4.6 to our problem. Considering $v(mv|\mathbf{x}_{i,g})$ as the number of votes assigned to the majority voting class, we propose to replace $v(\omega_t|\mathbf{x}_{i,g})$ by $v(mv|\mathbf{x}_{i,g})$. In this way, the margin

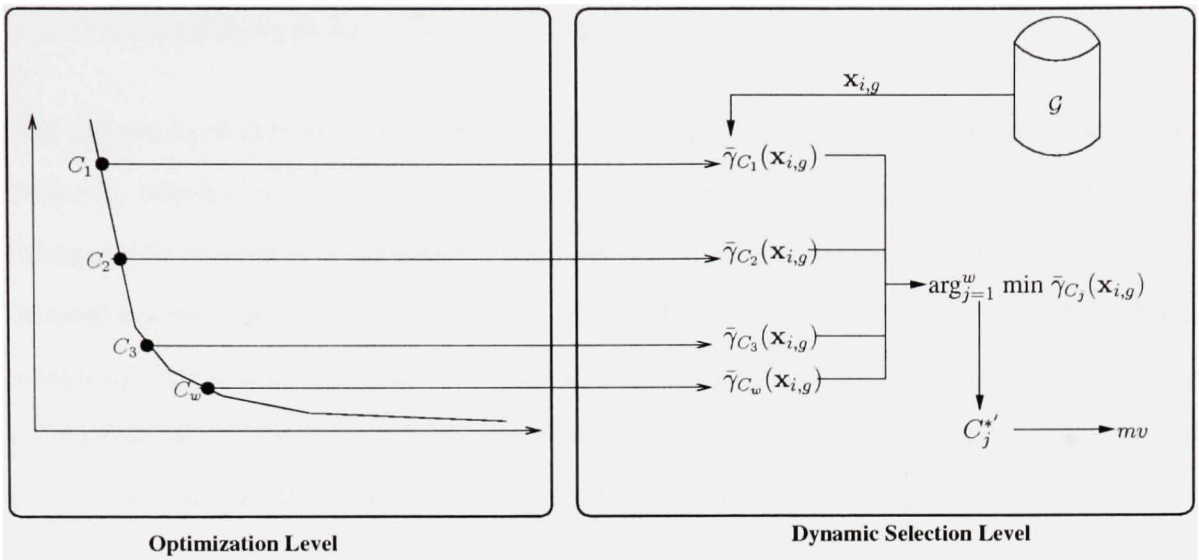


Figure 19 Ambiguity-guided dynamic selection using a Pareto front as input. The classifier ensemble with least ambiguity among its members is selected to classify each test sample.

of sample $\mathbf{x}_{i,g}$ for each C_j from the population $\mathbf{C}^{*'}$ may be calculated as follows:

$$\mu(\mathbf{x}_{i,g}) = \frac{v(mv|\mathbf{x}_{i,g}) - \max_{k \neq mv} v(\omega_k|\mathbf{x}_{i,g})}{|C_j|} \tag{4.7}$$

Hence, our definition of margin measures the difference between the number of votes assigned to the majority voting class and the number of votes assigned to the class with second highest result. Then, the margin value represents the confidence of the classifications, since the higher the margin from Equation 4.7, the higher the confidence of the ensemble consensus decision. Thus, the dynamic selection level guided by the margin, denoted MDS, will choose as $C^{*'}_j$ the candidate ensemble with the highest margin. For instance, when $\mu(\mathbf{x}_{i,g}) = 1$ the majority voting matches well to just one class, indicating the highest level of certainty of correct classification.

4.3.3 Class Strength-based Dynamic Selection (CSDS)

The definition of margin in Equation 4.6 also inspired this third confidence measure. Here, however, besides calculating Equation 4.7, we also consider the candidate ensemble's confidence with respect to the identity of the majority voting class measured in $\mathcal{V}$. This is additional knowledge related to the performance achieved by each candidate ensemble when assigning the chosen class. Our objective is to investigate whether or not performance may help a confidence measure, which does not take into account the correctness of the output, to increase the candidate ensemble's level of certainty of classification.

Strength relative to the closest class is presented in [8] as a method for defining weights in DCS-LA. It is calculated for each individual classifier c_i to verify whether or not the input pattern is closely similar to more than one class. We have adapted this measure to enable us to calculate it for candidate ensembles C_j in the dynamic selection of our DOCS.

Assuming $p_j(mv)$ as the performance of C_j measured over samples contained in $\mathcal{V}$ for the majority voting class, strength relative to the closest class may be calculated for $\mathbf{x}_{i,g}$ as follows:

$$\Theta(\mathbf{x}_{i,g}) = \frac{(v(mv|\mathbf{x}_{i,g}) - \max_{k \neq mv} v(\omega_k|\mathbf{x}_{i,g}))/|C_j|}{p_j(mv)} \tag{4.8}$$

A low value of $\Theta(\mathbf{x}_{i,g})$ means a low level of certainty of correct classification. In contrast, higher $\Theta(\mathbf{x}_{i,g})$ values lead to an increase in the level of confidence of classification. Thus, the dynamic selection level guided by Θ, called CSDS, will choose as $C_j^{*'}$ the candidate ensemble with the highest $\Theta(\mathbf{x}_{i,g})$ to provide a label for $\mathbf{x}_{i,g}$.

4.3.4 Dynamic Ensemble Selection with Local Accuracy (DCS-LA)

As explained in section 1.2.1, DCS-LA dynamically selects the most accurate individual classifier from the population $\mathcal{C}$ to predict the label of the test sample $\mathbf{x}_{i,g}$. Local accuracy

is measured in the region of competence composed as the set of k nearest neighbors from $\mathcal{T}$ surrounding $\mathbf{x}_{i,g}$. Woods et al. [101] compared two strategies to measure local accuracy: (1) overall local accuracy; and (2) local class accuracy. The overall local accuracy is computed as the number of neighbors correctly classified by each classifier c_i. However, in this chapter, we use the second strategy, since Woods et al. have concluded that this is the strategy that achieves the better results.

Given the pool $\mathcal{C}$ and the class assigned by the i-th classifier, ω_y, to the test sample $\mathbf{x}_{i,g}$, we denote N^y as the number of neighbors of $\mathbf{x}_{i,g}$ for which classifier c_i has correctly assigned class ω_y, and $\sum_{i=1}^{k} N^{iy}$ is the total number of neighbors labeled for c_i as class ω_y. According to the definition provided in [101], local class accuracy estimation is computed as follows:

$$\alpha_{c_i}(\mathbf{x}_{i,g}) = \frac{N^y}{\sum_{i=1}^{k} N^{iy}} \tag{4.9}$$

Taking into account that DCS-LA was originally proposed to deal with populations of classifiers, as summarized in Equation 4.9, it cannot be directly employed in problems involving populations of classifier ensembles. Thus, we propose to change the DCS-LA with local class accuracy estimation slightly in order to allow it to be used to guide dynamic selection in our proposed approach. Given a population $\mathbf{C}^{*'}$ of candidate ensembles, we assume ω_y as the class assigned by the candidate ensemble C'_j (composed of l classifiers) to the test pattern $\mathbf{x}_{i,g}$. We define as region of competence the set of k nearest neighbors from $\mathcal{V}$ surrounding $\mathbf{x}_{i,g}$. Clearly, DCS-LA can be critically affected by the choice of the k parameter. The local candidate ensemble's class accuracy is then estimated as follows:

$$\alpha_{C'_j}(\mathbf{x}_{i,g}) = \frac{\sum_{i=1}^{l} \alpha_{c_i}(\mathbf{x}_{i,g})}{|C'_j|} \tag{4.10}$$

To summarize, in this chapter, α_{C_j} for pattern $\mathbf{x}_{i,g}$ is calculated as the sum of the local class accuracy (α_{c_i}) of each classifier composing C_j, divided by the size of C_j. The higher α_{C_j}, the greater the certainty of the decision. Table XII shows a summary of the four different strategies proposed for use at the dynamic selection level of our DOCS.

Table XII

Summary of the four strategies employed at the dynamic selection level. The arrows specify whether or not the certainty of the decision is greater if the strategy is lower (↓) or greater (↑).

Name	**Label**	↑ / ↓
Ambiguity-Guided Dynamic Selection	ADS	(↓)
Margin-based Dynamic Selection	MDS	(↑)
Class Strength-based Dynamic Selection	CSDS	(↑)
Dynamic Ensemble Selection with Local Accuracy	DCS-LA	(↑)

Using the case study mentioned in section 4.2, we compare, in Table XIII, the results obtained in $\mathcal{G}$ using the combination of the initial pool $\mathcal{C}$, SOCS and our DOCS employing the four dynamic strategies presented in this section. These preliminary results show that, except for CSDS, our dynamic method guided by confidence measures outperformed SOCS performed by NSGA-II. In terms of the single-objective GA, static and dynamic selection methods presented similar results, except for CSDS and DCS-LA.

Table XIII

Case study: comparison among the results achieved by combining all classifiers in the initial pool $\mathcal{C}$ and by performing classifier ensemble selection employing both SOCS and DOCS.

Optimization	**Combination**	**SOCS**	**ADS**	**MDS**	**CSDS**	**DCS-LA**
NSGA-II (ϵ & θ)	6.06	5.86	5.74	**5.71**	6.01	5.74
GA (ϵ)	6.06	5.90	5.90	5.88	6.13	5.98

Figures 20 and 21 show histograms of the frequency of selection of each candidate ensemble performed using each dynamic selection strategy presented in this section. The histograms in Figure 20 were obtained for the population $\mathbf{C}^{*'}$ composed of n best candidate ensembles generated by GA shown in Figure 18(c), while the histograms in Figure 21 were obtained considering the Pareto front determined by II shown in Figure 18(d). Especially noteworthy is the fact that the test set $\mathcal{G}$ used for the case study is composed of 12,092 samples (Table VI), even though the dynamic selection level was conducted over only 325 and 436 of test samples for MOGA and GA respectively. All candidate ensembles in $\mathbf{C}^{*'}$ agreed on the label for the remaining test samples.

It is also important to observe in Figure 21 that ADS, MDS and DCS-LA more often selected as $C_j^{*'}$ the same candidate ensemble selected statically (C_1 in Figure 18(d)). In contrast, the opposite behavior is shown in Figure 20. These results indicate why DOCS did not outperform SOCS for the GA's population of candidate ensembles (see Table XIII). In addition, two of the confidence-based dynamic strategies, namely ADS and MDS, more frequently selected the same candidate ensembles as selected by DCS-LA, which is an accuracy-oriented strategy. These results support our assumption that selecting the candidate ensemble with the largest consensus improves the performance of the system. Moreover, considering the results obtained by CSDS, we can conclude that measures of confidence that do not take into account the correctness of the output provide enough information about a candidate ensemble's level of certainty of classification. The additional information calculated by CSDS through measuring the performance of each candidate ensemble over samples in $\mathcal{V}$ did not help in finding better performing ensembles. In next section, we present experimental results to verify whether or not these preliminary results are general, considering other databases and ensemble generation methods.

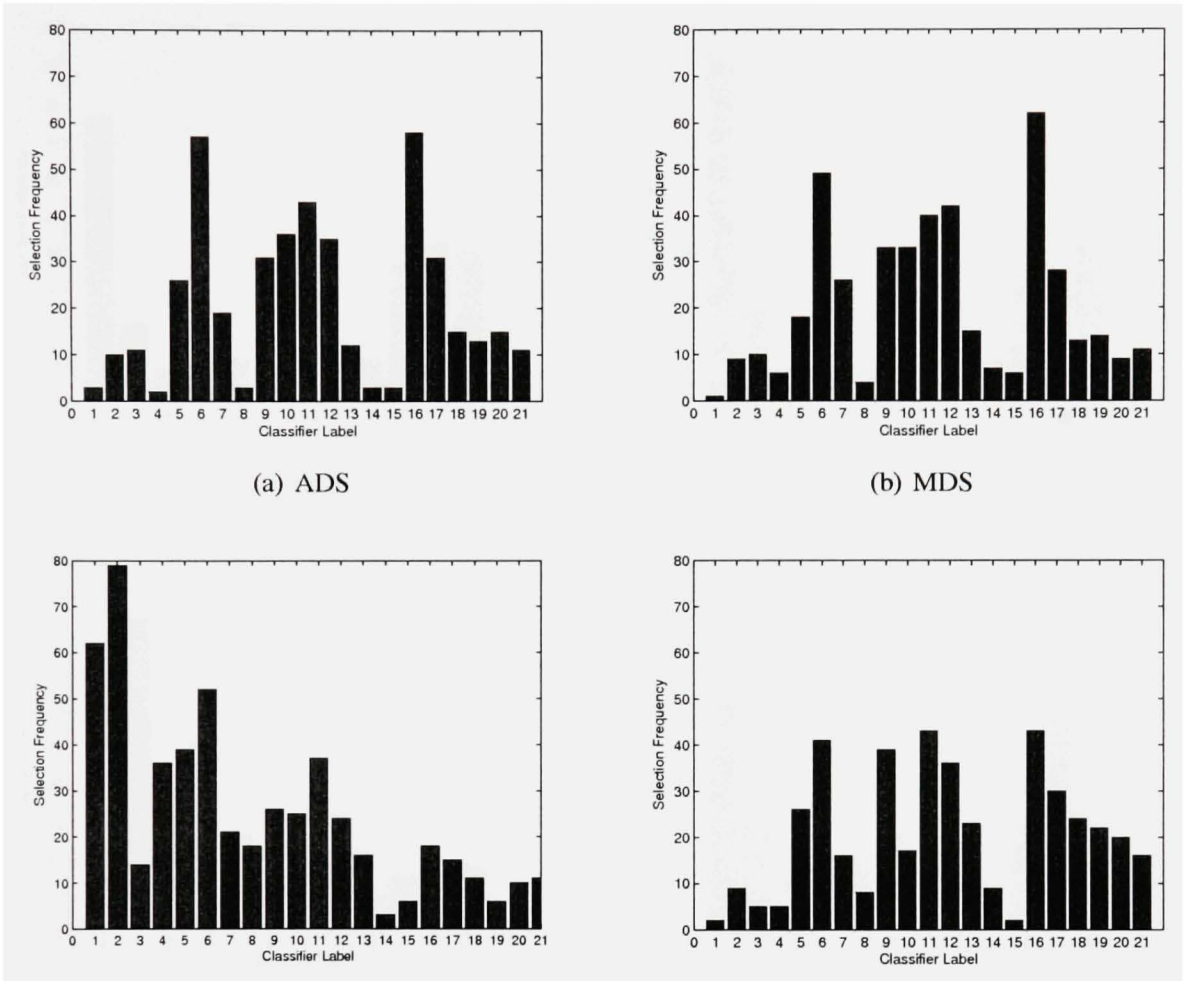


Figure 20 Case study: histogram of the frequency of selection of candidate ensembles performed by each dynamic selection strategy. The population $\mathbf{C}^{*'}$ is the n best solution generated by GA (see Figure 18(c)).

4.4 Experiments

A series of experiments has been carried out to determine the best strategy for the dynamic selection level proposed in our approach and to show whether or not DOCS is better than SOCS. As a consequence, we also point out the best method for the overproduction phase on comparing RSS and BAG, and the best search algorithm and search criteria for the optimization level. We used seven datasets divided into two groups: (1) two large; and (2)

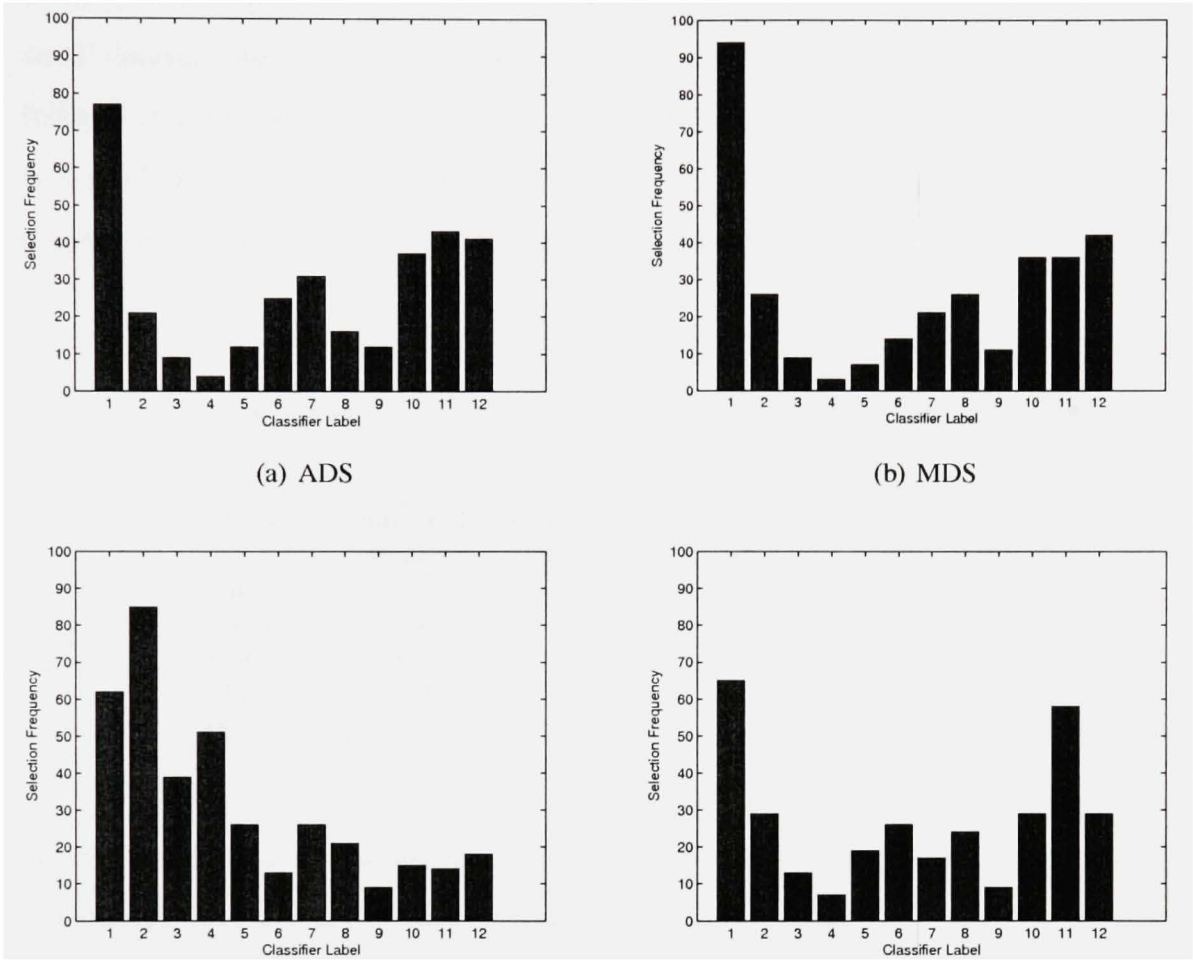


Figure 21 Case study: histogram of the frequency of selection of candidate ensembles performed by each dynamic selection method. The population $C^{*'}$ is the Pareto front generated by NSGA-II (see Figure 18(d)).

five small, Tables VI and XIV respectively. The datasets from group 1 are the same large datasets used in chapter 3. Therefore, they are large enough to be partitioned into the four independent datasets, illustrated in Figure 17: $\mathcal{T}$, $\mathcal{O}$, $\mathcal{V}$ and $\mathcal{G}$ using the classical holdout validation strategy for the evaluation of performance. By contrast, 10-fold cross-validation is applied for the evaluation of performance using small datasets due to the small number of samples available for evaluation.

Table XIV describes the five small datasets: the **Dna**, **satimage** and **texture** are the same small datasets used in chapter 3. We used two additional small datasets in this chapter: **feltwell** is a multisensor remote-sensing dataset [76] and **ship** is a dataset composed of forward-looking infra-red (FLIR) ship images [56]. The two new datasets are traditionally employed in works dealing with dynamic classifier selection.

Table XIV

Specifications of the small datasets used in the experiments.

Dataset	**Number of samples**	**Number of classes**	**Number of features**	**Features RSS**	**Pool C size**
Dna	3186	3	180	45	100
Feltwell	10944	5	15	8	100
Satimage	6435	6	36	18	100
Ship	2545	8	11	6	100
Texture	5500	11	40	20	100

In this chapter we follow the same experimental protocol employed in chapter 3. To summarize, for the overproduction phase, three different initial pools of 100 homogeneous classifiers were created: 100 kNN and 100 DT generated by RSS, and 100 DT generated by BAG. The selection phase was divided into two levels: for the optimization level, each diversity measure mentioned in section 3.3.1 was employed in combination with ϵ to guide NSGA-II, while only ϵ was used to guide single-objective GA. For the dynamic selection level, all four dynamic strategies defined in section 4.3 were tested so that they could be compared. We set $k = 10$ for experiments with DCS-LA, as employed in [84] and [79]. The results obtained are given in subsequent sections.

4.4.1 Comparison of Dynamic Selection Strategies

A summary of the experimental results comparing the four dynamic selection strategies defined in section 4.3 is given in Tables XV, XVI and XVII. The best result for each dataset is shown in bold. These results indicate that ADS and MDS were the best strate-

Table XV

Mean and standard deviation values obtained on 30 replications of the selection phase of our method. The overproduction phase was performed using an initial pool of kNN classifiers generated by RSS. The best result for each dataset is shown in bold.

Dataset	**Method**	**NSGA-II**				GA
		ambiguity (γ)	**coincident** (σ)	**difficulty** (θ)	**double-fault** (δ)	**error** (ϵ)
DNA	DCS-LA	9.33 (0.27)	9.84 (0.28)	7.48 (0.24)	9.36 (0.30)	7.47 (0.15)
	ADS	10.31 (0.28)	9.72 (0.47)	7.51 (0.22)	9.30 (0.23)	7.24 (0.21)
	MDS	10.34 (0.30)	9.57 (0.41)	7.47 (0.21)	9.16 (0.28)	**7.24** (0.15)
	CSDS	10.56 (0.30)	9.89 (0.19)	7.48 (0.19)	8.36 (0.30)	7.25 (0.17)
Felwell	DCS-LA	11.17 (0.21)	10.46 (0.25)	**9.70** (0.14)	10.59 (0.31)	9.87 (0.13)
	ADS	10.51 (0.11)	10.34 (0.25)	9.82 (0.16)	10.62 (0.28)	9.95 (0.17)
	MDS	10.51 (0.11)	10.34 (0.26)	9.82 (0.14)	10.59 (0.29)	9.95 (0.16)
	CSDS	10.57 (0.10)	10.40 (0.25)	9.84 (0.14)	10.66 (0.32)	9.97 (0.17)
Nist-digits Test1	DCS-LA	4.12 (0.07)	4.59 (0.17)	3.72 (0.05)	4.43 (0.26)	3.60 (0.06)
	ADS	3.80 (0.04)	3.95 (0.05)	3.58 (0.03)	7.49 (2.29)	3.53 (0.05)
	MDS	3.80 (0.04)	3.94 (0.05)	3.58 (0.02)	7.46 (2.32)	**3.53** (0.05)
	CSDS	5.21 (0.09)	7.25 (0.26)	3.80 (0.04)	6.67 (0.22)	3.80 (0.06)
Nist-digits Test2	DCS-LA	8.57 (0.14)	9.34 (0.28)	8.18 (0.11)	9.48 (0.43)	7.91 (0.10)
	ADS	8.11 (0.06)	8.55 (0.08)	7.97 (0.05)	8.40 (0.48)	**7.78** (0.10)
	MDS	8.12 (0.07)	8.53 (0.09)	7.97 (0.05)	8.36 (0.55)	7.78 (0.10)
	CSDS	9.95 (0.11)	9.89 (0.33)	8.21 (0.08)	9.42 (0.45)	8.09 (0.10)
Nist-letters	DCS-LA	7.63 (0.43)	8.22 (0.69)	6.48 (0.15)	7.20 (0.33)	6.55 (0.16)
	ADS	7.23 (0.17)	8.22 (0.12)	6.48 (0.07)	6.95 (0.12)	6.43 (0.06)
	MDS	7.16 (0.16)	7.07 (0.11)	**6.27** (0.07)	6.88 (0.12)	6.41 (0.08)
	CSDS	10.15 (0.45)	14.53 (0.63)	6.47 (0.14)	9.35 (0.41)	6.70 (0.09)
Satimage	DCS-LA	8.75 (0.08)	9.60 (0.15)	8.73 (0.13)	9.40 (0.13)	8.63 (0.07)
	ADS	8.67 (0.08)	9.05 (0.17)	8.64 (0.13)	9.16 (0.16)	**8.61** (0.09)
	MDS	8.68 (0.09)	9.04 (0.18)	8.65 (0.14)	9.16 (0.18)	8.61 (0.09)
	CSDS	8.96 (0.13)	10.13 (0.24)	8.80 (0.16)	9.78 (0.19)	8.67 (0.11)
Ship	DCS-LA	14.24 (0.32)	10.31 (0.29)	9.34 (0.24)	9.24 (0.23)	10.40 (0.20)
	ADS	13.25 (0.24)	9.60 (0.33)	9.25 (0.20)	**9.13** (0.21)	9.81 (0.16)
	MDS	13.37 (0.26)	9.66 (0.31)	9.24 (0.20)	9.15 (0.22)	9.83 (0.13)
	CSDS	14.06 (0.25)	10.17 (0.34)	9.39 (0.21)	9.31 (0.29)	10.23 (0.18)
Texture	DCS-LA	1.51 (0.06)	1.64 (0.08)	0.97 (0.05)	1.02 (0.07)	1.18 (0.03)
	ADS	1.37 (0.05)	1.23 (0.09)	**0.94** (0.05)	0.98 (0.07)	1.11 (0.01)
	MDS	1.37 (0.05)	1.22 (0.09)	0.94 (0.05)	0.98 (0.07)	1.11 (0.02)
	CSDS	1.37 (0.07)	1.28 (0.10)	0.94 (0.06)	0.98 (0.09)	1.11 (0.03)

gies for performing the dynamic selection level of our approach, considering all three ensemble creation methods investigated, i.e. (1) ensemble of kNN; (2) ensemble of DT generated through RSS (Tables XV and XVI); and (3) ensembles of DT generated by BAG

Table XVI

Mean and standard deviation values obtained on 30 replications of the selection phase of our method. The overproduction phase was performed using an initial pool of DT classifiers generated by RSS. The best result for each dataset is shown in bold.

Dataset	**Method**	**NSGA-II**				GA
		ambiguity (γ)	**coincident** (σ)	**difficulty** (θ)	**double-fault** (δ)	**error** (ϵ)
DNA	DCS-LA	7.05 (0.21)	7.54 (0.28)	4.63 (0.15)	7.23 (0.23)	5.14 (0.14)
	ADS	7.94 (0.23)	6.50 (0.29)	**4.59** (0.17)	6.27 (0.23)	4.95 (0.18)
	MDS	7.92 (0.23)	6.54 (0.30)	4.63 (0.17)	6.27 (0.19)	4.92 (0.19)
	CSDS	8.01 (0.24)	6.59 (0.28)	4.62 (0.17)	6.34 (0.22)	4.93 (0.19)
Felwell	DCS-LA	12.68 (0.19)	13.27 (0.47)	11.65 (0.32)	12.79 (0.51)	11.59 (0.14)
	ADS	11.60 (0.15)	12.71 (0.40)	11.65 (0.33)	12.51 (0.51)	**11.50** (0.17)
	MDS	11.59 (0.14)	12.74 (0.42)	11.65 (0.32)	12.49 (0.48)	11.53 (0.16)
	CSDS	11.66 (0.15)	12.80 (0.43)	11.65 (0.32)	12.51 (0.50)	11.51 (0.16)
Nist-digits Test1	DCS-LA	3.82 (0.12)	5.37 (0.29)	2.91 (0.04)	3.68 (0.20)	2.89 (0.04)
	ADS	3.35 (0.04)	4.11 (0.20)	**2.77** (0.03)	3.59 (0.23)	**2.77** (0.09)
	MDS	3.35 (0.04)	3.83 (0.13)	2.77 (0.02)	3.40 (0.12)	2.77 (0.04)
	CSDS	5.63 (0.09)	5.53 (0.12)	2.98 (0.06)	4.47 (0.78)	3.12 (0.06)
Nist-digits Test2	DCS-LA	8.18 (0.18)	10.06 (0.50)	6.79 (0.09)	8.07 (0.37)	6.66 (0.08)
	ADS	7.38 (0.07)	9.16 (0.32)	**6.45** (0.05)	8.11 (0.40)	**6.45** (0.05)
	MDS	7.34 (0.06)	8.61 (0.20)	6.50 (0.05)	7.80 (0.20)	6.45 (0.05)
	CSDS	9.05 (0.14)	8.77 (0.54)	6.50 (0.09)	8.69 (0.50)	6.98 (0.10)
Nist-letters	DCS-LA	7.57 (0.30)	6.12 (0.50)	6.03 (0.14)	7.20 (0.39)	6.17 (0.14)
	ADS	7.13 (0.09)	9.31 (0.29)	**5.84** (0.06)	7.12 (0.14)	5.96 (0.06)
	MDS	7.12 (0.09)	7.69 (0.19)	5.84 (0.06)	6.93 (0.10)	5.95 (0.06)
	CSDS	10.65 (0.11)	15.92 (0.14)	5.98 (0.08)	10.95 (0.46)	6.29 (0.09)
Satimage	DCS-LA	9.30 (0.09)	10.66 (0.19)	8.97 (0.09)	10.12 (0.15)	8.96 (0.09)
	ADS	8.73 (0.10)	9.22 (0.19)	8.64 (0.10)	9.17 (0.15)	8.78 (0.12)
	MDS	8.73 (0.10)	9.22 (0.19)	**8.63** (0.09)	9.16 (0.15)	8.77 (0.13)
	CSDS	9.13 (0.13)	11.23 (0.24)	8.94 (0.12)	10.64 (0.26)	8.86 (0.12)
Ship	DCS-LA	10.65 (0.30)	9.16 (0.30)	7.53 (0.18)	7.93 (0.29)	7.24 (0.14)
	ADS	8.86 (0.25)	8.26 (0.34)	7.17 (0.15)	7.72 (0.28)	6.98 (0.14)
	MDS	8.94 (0.26)	8.26 (0.32)	7.18 (0.15)	7.75 (0.29)	**6.95** (0.13)
	CSDS	9.97 (0.24)	9.15 (0.41)	7.18 (0.17)	8.03 (0.27)	7.35 (0.12)
Texture	DCS-LA	3.42 (0.17)	3.85 (0.14)	2.47 (0.07)	3.51 (0.16)	2.84 (0.09)
	ADS	2.98 (0.08)	3.02 (0.14)	**2.35** (0.06)	2.97 (0.14)	2.44) (0.05
	MDS	3.00 (0.08)	3.04 (0.14)	2.35 (0.05)	2.65 (0.16)	2.44 (0.05)
	CSDS	3.01 (0.08)	3.06 (0.15)	2.35 (0.05)	2.96 (0.16)	2.44 (0.06)

(Table XVII). These two dynamic strategies presented equivalent performances, which confirms the preliminary results in our case-study problem (see Table XIII). CSDS was the worst dynamic selection strategy for 1 and 3, while DCS-LA was most likely to be the

Table XVII

Mean and standard deviation values obtained on 30 replications of the selection phase of our method. The overproduction phase was performed using an initial pool of DT classifiers generated by Bagging. The best result for each dataset is shown in bold.

Dataset	Method	NSGA-II				GA
		ambiguity (γ)	**coincident** (σ)	**difficulty** (θ)	**double-fault** (δ)	**error** (ϵ)
DNA	DCS-LA	5.15 (0.15)	5.20 (0.18)	**4.75** (0.15)	5.30 (0.15)	4.88 (0.09)
	ADS	5.20 (0.16)	5.20 (0.16)	4.86 (0.11)	5.32 (0.21)	4.95 (0.06)
	MDS	5.17 (0.17)	5.20 (0.18)	4.86 (0.10)	5.31 (0.23)	4.95 (0.06)
	CSDS	5.19 (0.15)	5.22 (0.17)	4.86 (0.11)	5.36 (0.25)	4.97 (0.07)
Felwell	DCS-LA	12.50 (0.36)	12.47 (0.28)	12.00 (0.22)	12.40 (0.32)	**11.69** (0.20)
	ADS	12.70 (0.21)	12.28 (0.34)	12.04 (0.18)	12.24 (0.31)	11.95 (0.14)
	MDS	12.70 (0.17)	12.28 (0.35)	12.03 (0.18)	12.24 (0.29)	11.94 (0.14)
	CSDS	12.75 (0.19)	12.38 (0.33)	12.04 (0.19)	12.34 (0.32)	11.98 (0.15)
Nist-digits Test1	DCS-LA	5.74 (0.09)	6.72 (0.36)	5.67 (0.09)	6.04 (0.21)	5.23 (0.06)
	ADS	5.53 (0.05)	5.79 (0.11)	5.51 (0.05)	5.47 (0.10)	**5.14** (0.06)
	MDS	5.52 (0.05)	5.69 (0.10)	5.51 (0.05)	5.44 (0.09)	5.14 (0.05)
	CSDS	5.87 (0.12)	6.86 (0.56)	5.64 (0.09)	6.72 (0.41)	5.44 (0.06)
Nist-digits Test2	DCS-LA	11.00 (0.16)	11.60 (0.55)	10.98 (0.15)	10.64 (0.37)	10.21 (0.12)
	ADS	10.71 (0.09)	10.81 (0.25)	10.71 (0.09)	10.59 (0.18)	**10.06** (0.11)
	MDS	10.70 (0.09)	11.02 (0.24)	10.70 (0.10)	10.54 (0.12)	10.06 (0.10)
	CSDS	11.18 (0.20)	12.76 (0.91)	10.88 (0.12)	11.68 (0.61)	10.41 (0.11)
Nist-letters	DCS-LA	7.81 (0.17)	9.13 (0.62)	7.64 (0.12)	8.45 (0.44)	7.79 (0.12)
	ADS	7.61 (0.06)	8.36 (0.15)	**7.50** (0.06)	8.06 (0.17)	7.64 (0.07)
	MDS	7.63 (0.06)	8.30 (0.10)	7.50 (0.05)	8.02 (0.14)	7.64 (0.07)
	CSDS	7.85 (0.07)	10.16 (0.94)	7.60 (0.08)	9.84 (0.54)	7.78 (0.01)
Satimage	DCS-LA	9.70 (0.38)	11.35 (0.57)	9.81 (0.11)	10.99 (0.22)	9.74 (0.09)
	ADS	9.34 (0.43)	10.36 (0.60)	9.62 (0.09)	10.25 (0.13)	9.61 (0.11)
	MDS	**9.21** (0.37)	10.44 (0.58)	9.63 (0.07)	10.22 (0.11)	9.61 (0.12)
	CSDS	9.78 (0.17)	12.08 (0.22)	9.78 (0.11)	11.82 (0.28)	9.68 (0.12)
Ship	DCS-LA	7.79 (0.19)	8.32 (0.29)	7.81 (0.29)	8.58 (0.29)	**7.72** (0.13)
	ADS	8.14 (0.20)	8.70 (0.29)	8.36 (0.19)	8.83 (0.27)	8.07 (0.09
	MDS	8.16 (0.20)	8.71 (0.29)	8.38 (0.17)	8.84 (0.28)	8.05 (0.09)
	CSDS	8.46 (0.23)	9.04 (0.32)	8.53 (0.21)	9.12 (0.27)	8.15 (0.16)
Texture	DCS-LA	3.69 (0.09)	4.40 (0.43)	3.44 (0.09)	4.17 (0.13)	3.66 (0.07)
	ADS	3.57 (0.09)	4.02 (0.56)	**3.41** (0.008)	4.03 (0.15)	3.60 (0.06)
	MDS	3.56 (0.09)	3.93 (0.63)	3.42 (0.08)	4.04 (0.15)	3.60 (0.06)
	CSDS	3.63 (0.10)	4.01 (0.18)	3.43 (0.11)	3.43 (0.11)	3.58 (0.08)

worst dynamic selection strategy for ensembles of DT generated by RSS. In terms of the optimization level, single-objective GA and NSGA-II presented equivalent performances for ensembles of kNN generated by RSS and ensembles of DT generated by BAG, while

NSGA-II found the best results for populations of ensembles of DT generated by RSS. θ was clearly the best diversity measure for composing, with ϵ, a pair of objective functions to guide NSGA-II, while γ and σ were the worst diversity measures.

It is important to note that DCS-LA was better than the other dynamic selection strategies in 3 of the 8 datasets in problems involving DT generated by BAG. The reason for this behavior is that BAG provides the complete representation of the problem to each classifier member, whereas RSS provides only a partial representation. Thus, DCS-LA calculates the local accuracy of each classifier more accurately when BAG is used as the ensemble creation method. Also important is the fact that CSDS was less effective in problems involving a large number of classes, such as **NIST-letters**, because it takes into account the performance of the candidate ensembles for each class involved in the classification problem, in addition to the extent of consensus of the ensembles. Moreover, for the same reason, CSDS is much more critically affected by the quality of the population of candidate ensembles found at the optimization level. For instance, since γ and σ were the worst objective functions when guiding the optimization level, CSDS was much worse than the other three strategies when used to perform the dynamic selection on populations of candidate ensembles found by these two diversity measures.

4.4.2 Comparison between DOCS and Several Methods

In this section, we summarize the best results obtained by our DOCS for each dataset, in order to show that the proposed approach outperforms other related methods. Table XVIII reports the results attained by the following methods:

- Fusion of the initial pool of classifiers C by majority voting;
- Selection of the best individual classifier from the initial pool C;
- Individual kNNs and DTs trained using all available features;

- DOCS using the best dynamic selection strategy;
- Oracle for each initial pool of candidate classifiers.

Oracle is a lower bound of selection strategies, because it correctly classifies the test sample if any of the classifier members predicts the correct label for the sample. Values are shown in bold for the best result obtained for each dataset considering each ensemble generation method, and are shown underlined for the best overall result obtained for each dataset, whatever the ensemble creation method used. From this table, some observations can be made:

Table XVIII

Error rates attained by several methods. NSGA-II (X), GA (Y), I (DCS-LA), II (ADS), III (MDS), IV (CSDS). Values in bold and underlined indicate the best result in each dataset for each overproduction method and the best overall result for each dataset respectively.

Method	Dna	Feltwell	NIST-digits		NIST-letters	Satimage	Ship	Texture
			Test1	Test2				
Bagging DT	X-I θ	Y-I ϵ	Y-II ϵ	Y-II ϵ	X-II θ	X-III γ	Y-I ϵ	X-II θ
Initial Pool	5.02	12.80	5.65	10.99	7.63	9.59	8.09	3.60
Best Classifier	5.87	**10.59**	9.70	16.62	14.31	12.77	9.08	6.64
Oracle (C)	0.38	2.57	0.24	0.63	0.29	0.11	035	0.02
Best DOCS	**4.75**	11.69	**5.14**	**10.06**	**7.50**	**9.21**	**7.72**	**3.41**
RSS DT	X-II θ	Y-II ϵ	Y-II θ	X-II θ	X-II θ	X-III θ	Y-III ϵ	X-II θ
Initial Pool	5.05	11.86	2.92	6.67	6.06	8.64	**6.80**	2.56
Best Classifier	11.33	11.86	11.07	19.18	17.13	11.83	10.45	6.07
DT all features	6.85	16.81	10.3	18.2	13.5	14.17	10.92	7.56
Oracle (C)	0.03	0.60	0.01	0.04	0.04	0.22	0.24	0.02
Best DOCS	**4.59**	**11.50**	**2.77**	**6.45**	**5.84**	**8.63**	6.95	**2.35**
RSS kNN	Y-II ϵ	X-I θ	Y-II ϵ	Y-II ϵ	X-III θ	Y-II ϵ	X-II δ	X-II θ
Initial Pool	**6.87**	10.44	3.72	8.10	6.60	**8.59**	9.94	1.11
Best Classifier	23.10	**9.46**	7.52	13.99	14.47	8.95	10.26	**0.62**
kNN all features	26.30	12.35	6.66	9.76	7.82	9.84	11.24	1.13
Oracle (C)	0.03	0.67	0.05	0.17	0.18	0.36	0.28	0.04
Best DOCS	7.24	9.70	**3.53**	**7.78**	**6.27**	8.61	**9.13**	0.94

- Analyzing the results obtained for ensembles of DT generated using BAG, our DOCS outperformed the other methods, except for **feltwell**. The same scenario was observed for ensembles of DT generated using RSS, but the exception was the **ship** dataset. For ensembles of kNN generated using RSS, however, the proposed method outperformed the other methods in only 4 of the 8 cases. For the remaining 4 datasets, our DOCS was the second best method. The combination of the initial pool $\mathcal{C}$ was the best method for **dna** and **satimage** while the individual best classifier from $\mathcal{C}$ was the best method for **feltwell** and **texture**.

- Confidence-based dynamic strategies were better than DCS-LA for performing the dynamic selection of the selection phase. Even though ADS and MDS presented similar behavior, we show in Table XVIII values obtained using ADS to perform the comparison in next section.

- The search algorithms presented equivalent performances. The best results found using DOCS were obtained in populations of candidate ensembles optimized by NSGA-II in 13 cases and by single-objective GA in 11 cases out of a total of 24 cases investigated.

- θ was the best diversity measure for combining with ϵ to guide NSGA-II.

- RSS was better than BAG for use during the overproduction phase. Results obtained using ensembles generated by RSS were both better than those obtained using ensembles generated by BAG.

- DT was the best classifier model to be used as a base classifier during the overproduction phase. Five of the best overall results were obtained with ensembles of DT, whereas in the remaining three datasets they were obtained with ensembles of kNN. Both ensembles were generated by RSS.

Table XIX presents some of the results reported in the literature dealing with the selection of classifiers, except for references [32] and [66], on the databases used in this chapter. In this way, it is possible to gain an overview of the results obtained in the literature, even though the method of partition of the data used in some of these works was not the same as that used in this chapter.

Table XIX

The error rates obtained, the data partition and the selection method employed in works which used the databases investigated in this chapter (FSS: feature subset selection).

Database	Reference Number	Ensemble Creation	Classifiers Members	Partition Strategy	Selection Type	Error (%)
Dna	[32]	RSS	DT	Holdout	Fusion	9.19
Feltwell	[15]	Het	Het	Cross-validation	DCS-LA	13.62
NIST-digits Test1	[89]	RSS	kNN	Holdout	SOCS	3.65
NIST-letters	[61]	FSS	MLP	Holdout	SOCS	4.02
Satimage	[15]	Het	Het	Cross-validation	DCS-LA	10.82
Ship	[66]	Het	Het	Holdout	Fusion	5.68
Texture	[101]	DT	Het	Holdout	DCS-LA	0.75

To verify whether or not our DOCS is better that the classical SOCS, in the next section, we concentrate our analysis on the methods that attained the best results in this section, i.e. DT ensembles generated by RSS, ADS as dynamic strategy; MOGA guided by θ in combination with ϵ; and GA guided by ϵ.

4.4.3 Comparison of DOCS and SOCS Results

The static selection results were obtained by picking up the candidate ensemble presenting the lowest ϵ value of all the solutions composing the n best solution (for single-objective GA) or the Pareto front (for NSGA-II). These ensembles are represented in Figures 18(c) and 18(d) by black circles. It is important to mention that the results were tested on multiple comparisons using the Kruskal-Wallis nonparametric statistical and the confidence level was 95% ($\alpha = 0.05$), as was done in previous chapters.

Table XX summarizes the mean and the standard deviation of the error rates obtained on all eight databases comparing the static and dynamic OCS. These results indicate, without exception, that our DOCS was better than the traditional SOCS. In two specific situations (**feltwell** and **NIST-letters**), the differences between the two methods were not significant. It is important to note that the results achieved using NSGA-II guided by both θ and ϵ outperformed the results found using GA guided by ϵ the single-objective function. This is a different result from those presented in [72] and [89]. It appears that performing dynamic selection in the selection phase resulted in exploiting all possible potentials of the population of candidate ensembles over the Pareto front, leading to more benefits for NSGA-II with the new level. However, only with the **texture** database was GA better than NSGA-II in SOCS, whereas in DOCS the opposite is true.

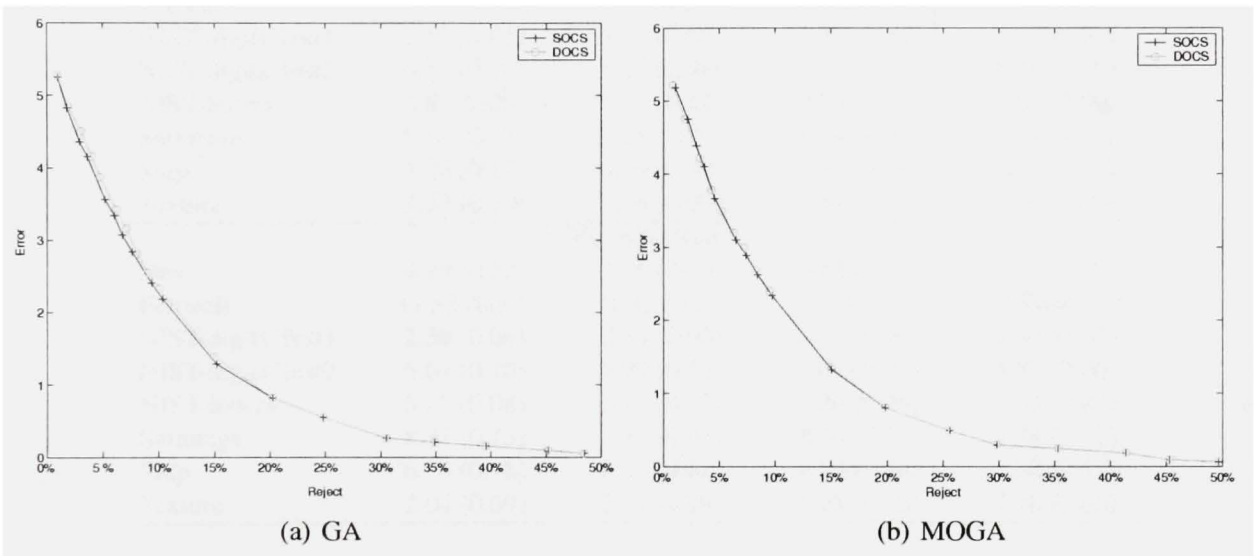
Table XX

Mean and standard deviation values of the error rates obtained on 30 replications comparing DOCS and SOCS with no rejection. Values in bold indicate the lowest error rate and underlined when a method is significantly better than the others.

Dataset	**SOCS**		**DOCS**	
	NSGA-II (θ & ϵ)	GA ϵ	NSGA-II (θ & ϵ)	GA ϵ
Dna	4.77 (0.17)	4.97 (0.16)	**4.59** (0.17)	4.95 (0.19)
Feltwell	12.02 (0.38)	11.86 (0.06)	11.65 (0.33)	**11.50** (0.17)
NIST-digits Test1	2.82 (0.06)	2.84 (0.06)	**2.77** (0.03)	**2.77** (0.03)
NIST-digits Test2	6.59 (0.09)	6.53 (0.09)	**6.45** (0.05)	**6.45** (0.05)
NISt-letters	5.89 (0.07)	6.02 (0.09)	**5.84** (0.06)	5.96 (0.06)
Satimage	8.76 (0.13)	8.82 (0.12)	**8.64** (0.10)	8.78 (0.12)
Ship	7.35 (0.17)	7.14 (0.23)	7.17 (0.15)	**6.98** (0.14)
Texture	2.41 (0.09)	2.51 (0.09)	**2.35** (0.06)	2.44 (0.05)

All these results were obtained with a zero reject rate. However, as advocated by Hansen et al. [29], the reject mechanism for an ensemble of classifiers is based on the extent of consensus among its members. This means that the decision to reject a pattern is related to the confidence level of the ensemble. They assume that it is better to reject the pattern if the ensemble presents a low confidence level to take a decision. Such a confidence level

is clearly related to $\bar{\gamma}$ (Equation 4.4), which is used to guide ADS. Thus, since $\bar{\gamma}$ is the criterion used to perform the reject mechanism, we might assume that the difference between SOCS and DOCS will increase as the rejection rate increases. We analyze such an assumption in Figure 22 taking into account the case study problem investigated throughout this chapter. This preliminary result does not confirm this assumption. Both the static and the dynamic OCS presented similar error-reject curves for ensembles generated by GA 22(a) and NSGA-II 22(b).



(a) GA (b) MOGA

Figure 22 Case Study: Error-reject curves for GA (22(a)) and NSGA-II (22(b)).

In order to show more general results about the error-reject tradeoff, we fixed four different reject rates: 0.5% and 1.0% (Table XXI), 5.0% and 10.0% (Table XXII), to investigate all the problems dealt with in this chapter. These results confirm that we should not assume that DOCS is more effective than SOCS when increasing the rejection rate.

4.5 Discussion

We propose a dynamic overproduce-and-choose strategy which is composed of the traditional overproduction and selection phases. The novelty is to divide the selection phase into two levels: optimization and dynamic selection, conducting the second level using

Table XXI

Mean and standard deviation values of the error rates obtained on 30 replications comparing DOCS and SOCS with rejection. Values in bold indicate the lowest error rate attained and underlined when a method is significantly better than the others.

Dataset	**SOCS**		**DOCS**	
	NSGA-II (θ & ϵ)	GA ϵ	NSGA-II (θ & ϵ)	GA ϵ
		0.5%-rejection		
Dna	4.50 (0.18)	4.72 (0.18)	**4.38** (0.19)	4.78 (0.18)
Feltwell	11.79 (0.38)	11.62 (0.24)	11.42 (0.33)	**11.29** (0.17)
NIST-digits Test1	2.55 (0.06)	2.55 (0.06)	**2.52** (0.04)	2.53 (0.03)
NIST-digits Test2	6.32 (0.10)	6.26 (0.08)	6.25 (0.05)	**6.20** (0.06)
NIST-letters	5.60 (0.09)	5.71 (0.11)	**5.53** (0.07)	5.67 (0.06)
Satimage	8.52 (0.13)	8.58 (0.13)	**8.44** (0.09)	8.54 (0.12)
Ship	7.06 (0.17)	6.89 (0.24)	6.88 (0.18)	**6.87** (0.16)
Texture	2.22 (0.10)	2.26 (0.09)	**2.17** (0.06)	2.25 (0.06)
		1.0%-rejection		
Dna	4.29 (0.17)	4.54 (0.20)	**4.22** (0.18)	4.63 (0.19)
Feltwell	11.53 (0.37)	11.38 (0.23)	11.14 (0.33)	**11.08** (0.17)
NIST-digits Test1	**2.30** (0.06)	2.34 (0.06)	2.31 (0.04)	2.34 (0.03)
NIST-digits Test2	6.04 (0.10)	5.99 (0.11)	6.09 (0.07)	**5.97** (0.06)
NIST-letters	5.27 (0.08)	5.41 (0.11)	**5.24** (0.06)	5.41 (0.07)
Satimage	8.31 (0.13)	8.38 (0.14)	**8.24** (0.11)	8.34 (0.12)
Ship	6.74 (0.18)	6.59 (0.21)	6.57 (0.18)	**6.52** (0.17)
Texture	2.04 (0.09)	2.07 (0.09)	**2.01** (0.06)	2.06 (0.05)

confidence measures based on the extent of consensus of ensembles. The two classical ensemble creation methods, the random subspace method and bagging, were used for the overproduction phase, while single- and multi-objective GAs were used at the optimization level. The ensemble error rates and diversity measures guided the optimization. Finally, three confidence measures were applied at the dynamic selection level: (1) ambiguity; (2) margin; and (3) strength relative to the closest class. In addition, DCS-LA was compared to the confidence-based strategies.

Experiments conducted using eight datasets demonstrated that the proposed approach outperforms both static selection and the fusion of the initial pool of classifiers. Ambiguity and margin were the best measures to use at the dynamic selection level, presenting equiva-

Table XXII

Mean and standard deviation values of the error rates obtained on 30 replications comparing DOCS and SOCS with rejection. Values in bold indicate the lowest error rate attained and underlined when a method is significantly better than the others.

Dataset	**SOCS**		**DOCS**	
	NSGA-II (θ & ϵ)	GA ϵ	NSGA-II (θ & ϵ)	GA ϵ
		5.0**%-rejection**		
Dna	**3.13** (0.15)	3.37 (0.14)	3.18 (0.17)	3.54 (0.16)
Feltwell	9.70 (0.35)	9.46 (0.22)	9.48 (0.36)	**9.26** (0.20)
NIST-digits Test1	**1.21** (0.05)	1.22 (0.04)	1.24 (0.05)	1.29 (0.02)
NIST-digits test2	4.38 (0.14)	**4.34** (0.10)	4.41 (0.10)	4.44 (0.08)
NIST-letters	**3.64** (0.11)	3.79 (0.10)	**3.64** (0.09)	3.83 (0.06)
Satimage	6.72 (0.13)	6.71 (0.12)	**6.68** (0.12)	6.71 (0.11)
Ship	5.05 (0.21)	**4.99** (0.17)	5.05 (0.16)	5.16 (0.17)
Texture	1.01 (0.05)	1.03 (0.05)	**0.98** (0.07)	1.04 (0.04)
		10.0**%-rejection**		
Dna	2.06 (0.18)	2.14 (0.18)	**2.05** (0.19)	2.36 (0.13)
Feltwell	7.45 (0.40)	7.50 (0.20)	7.51 (0.29)	**7.39** (0.22)
NIST-digits Test1	**0.54** (0.03)	0.56 (0.04)	0.54 (0.03)	0.58 (0.03)
NIST-digits Test2	**2.83** (0.10)	**2.83** (0.10)	2.89 (0.09)	2.93 (0.09)
NIST-letters	**2.27** (0.07)	2.30 (0.08)	2.31 (0.08)	2.42 (0.05)
Satimage	5.05 (0.10)	5.02 (0.09)	5.02 (0.10)	**4.98** (0.09)
Ship	**3.59** (0.13)	3.62 (0.11)	3.64 (0.14)	3.79 (0.12)
Texture	0.39 (0.05)	0.37 (0.03)	0.37 (0.05)	**0.36** (0.04)

lent performances. It was shown that NSGA-II guided by the difficulty measure combined with the error rate found better ensembles than single-objective GA guided only by the error rate. In addition, although not as clearly as one might have hoped, our results indicate that our method is especially valuable for tasks using NSGA-II, since differences between the results found using NSGA-II and GA are greater in the dynamic than in the static overproduce-and-choose strategy. Ideally, we should exploit all the potential of the population of candidate ensembles over a Pareto front using the proposed dynamic selection level.

However, the quality of the population of candidate ensembles found at the optimization level critically affects the performance of our approach. We can confirm this observation

taking into account the concept of oracle. As mentioned before, the so-called oracle is a lower bound of selection strategies. Table XXIII shows results attained by the fusion of the initial pool of classifiers $\mathcal{C}$ and its oracle for the same case study investigated throughout this chapter. Only NSGA-II guided by θ combined with ϵ is considered in this example.

Table XXIII

Case study: comparing oracle results.

Fusion of initial pool $\mathcal{C}$	**Oracle of initial pool $\mathcal{C}$**	**SOCS**	**DOCS**	**Oracle of population of ensembles $\mathbf{C}^{*'}$**
6.06	0.04	5.86	5.71	4.81

It is important to mention that, instead of $\mathcal{C}$, the Pareto front (population of ensembles $\mathbf{C}^{*'}$) is the input to our dynamic selection level. Hence, we should consider the oracle of the population $\mathbf{C}^{*'}$ as the lower bound for our DOCS. Thus, it is assumed that oracle correctly classifies the test sample if any of candidate ensembles in $\mathbf{C}^{*'}$ predicts the correct label for the sample. In Table XXIII, it is shown that our DOCS may not achieve an error rate lower than 4.81. The large difference between the results of the $\mathcal{C}$ and $\mathbf{C}^{*'}$ oracles leads us to conclude that there is a major loss of oracle power at the optimization level. Populations of candidate ensembles with oracle error rates closer to those obtained using $\mathcal{C}$ will bring about an effective improvement in the performances attained by our method.

CONCLUSION

The focus of this thesis was the selection of classifier ensembles. Our efforts were concentrated towards the overproduction-and-choose strategy. We have seen that this strategy is classically composed of the overproduction and selection phases. While the overproduction phase may be performed using any ensemble construction strategy and base classifier model, the selection phase is the main challenge in the overproduction-and-choose strategy, since it focuses on finding the best performing subset of classifiers. We have called this approach static overproduction-and-choose strategy due to the fact that the candidate ensemble selected as the best performing subset of classifiers is used to classify the whole test dataset.

In our first investigation we analyzed the search criterion, which is one of the two main aspects that should be analyzed when the selection phase performs an optimization process conducted by non-exhaustive search algorithms. Single- and multi-objective optimization processes were conducted using GAs. NSGA-II was the multi-objective GA used (in appendix 1 we show experimental results which support our choice). Fourteen different objective functions were applied: 12 diversity measures, the error rate and ensemble size, while an ensemble of 100 kNN classifiers generated using the random subspace method was used as the initial pool of classifiers created at the overproduction phase. The results of our experiments have shown that diversity alone cannot be better than the error rate at finding the most accurate classifier ensembles. However, when both the error rate and diversity are combined in a multi-objective approach, we observed that the performance of the solutions found using diversity is much higher than the performance of the solutions found using diversity in a single-objective optimization approach. Finally, we showed that it can be established an analogy between feature subset selection and static overproduction-and-choose strategy by combining ensemble size and the error rate in a pair of objective functions. However, we observed that the reduction in the number of classifiers is a consequence of the selection process, then, it is not necessary to include

explicitly ensemble size in the optimization process. Moreover, our experiments showed that ensemble size and diversity are not conflicting objective functions, and should not be used combined to guide the optimization process in a multi-objective approach. This observation, which is also confirmed in appendix 3, leads to the conclusion that the results obtained using single-objective GA in our experiments may be different if the minimum number of classifiers is fixed and larger than we have defined in chapter 2.

We then proceeded to investigate the problem of overfitting detected at the optimization process of the selection phase. We presented three overfitting control methods: (1) partial validation; (2) backwarding; and (3) global validation. Three initial pools of 100 classifiers were generated: 100 kNN and 100 Decision Trees using the random subspace method, and a third pool of 100 Decision Trees using bagging. Five different objective functions were applied: four diversity measures and the error rate were used to guide both single- and multi-objective GAs. We confirmed that overfitting can be detected at the selection phase, especially when NSGA-II is employed as the search algorithm. Global validation outperformed the other overfitting control methods, while random subspace-based ensembles were more prone to overfitting than bagging-based ones. We also proposed to use global validation as a tool for identifying the diversity measure most closely related to performance. Our results indicated that double-fault was the diversity measure more closely related to performance. In addition, we showed that the difficulty measure can be better than the error rate as a single-objective function for selecting high-performance ensembles of classifiers. However, taking into account our previous experimental results, which show that the minimum number of classifiers is achieved when using most of the diversity measures in single-objective optimization, quite a large minimum ensemble size was fixed in our experiments. Therefore, these results are still dependent of the minimum ensemble size defined.

Lastly, we have proposed a dynamic overproduce-and-choose strategy which combines optimization and dynamic selection in a two-level selection phase to allow the selection of

the most confident subset of classifiers to label each test sample individually. In this way, instead of choosing only one solution to classify the whole test dataset, all potential high accuracy candidate ensembles from the population generated by the search algorithms, are considered to select the most competent solution for each test sample. Moreover, we proposed to conduct our dynamic selection level using confidence measures based on the extent of consensus of candidate ensembles. These measures lead to the selection of the solution with the highest level of confidence among its members, allowing an increase in the "degree of certainty" of the classification. The same three initial pools of classifiers, generated by the random subspace method and bagging, were used as output of the overproduction phase. Single- and multi-objective GAs guided by the same five objective functions, i.e. four diversity measures and the error rate, were used at the optimization level. Finally, three different confidence measures were proposed to be used at the dynamic selection level.

Our experiments, which were carried out on eight real classification problems including handwritten digits and letters, texture, multisensor remote-sensing images and forward-looking infra-red ship images, demonstrated that our method outperforms static overproduce-and-choose strategy, a classical dynamic classifier selection method and the combination of the decisions of the classifiers in the initial pools. In despite of these performance improvements, we observed that the quality of the population of candidate ensembles found at the optimization level critically affects the performance of our approach. Still more interesting is the observed major loss of oracle power at the optimization level. Taking into account that oracle is the lower bound of selection strategies, the optimization level of our method increases the lower bound, consequently avoiding the dynamic selection level to attain more effective improvement in the performances.

Future Works

Even though this thesis successfully investigated the static overproduce-and-choose strategy and proposed a dynamic overproduce-and-choose strategy, some issues were not addressed owing to time. The following future directions are possible and worthy of investigation:

- Determine strategies to improve the quality of the population of ensembles. We believe that the optimization level can be improved in order to decrease the lower bound oracle for our dynamic overproduce-and-choose strategy.
- Investigate different strategies for the generation of the population of classifier ensembles, which is the input of the dynamic selection level of our method.
- Develop new confidence measures focusing on increasing the confidence of the decisions.
- Investigate the impact of our dynamic overproduce-and-choose strategy in problems involving missing features.

APPENDIX 1

Comparison of Multi-Objective Genetic Algorithms

Genetic Algorithms are the population-based search algorithms investigated in this thesis. In chapter 2 we mentioned that single- and multi-objective GAs are two possible options when dealing with GAs. Three MOGAs were investigated in order to allow us to define the best option to be used in all the experiments conducted in this thesis. The following three MOGAs were investigated: (1) Non-dominated sorting GA (NSGA) [11]; (2) Fast elitist non-dominated sorting GA (NSGA-II) [13]; and (3) controlled elitist NSGA [14].

The general GA parameters presented in Table V, chapter 2, are used for all three MOGAs. However, NSGA and controlled elitist NSGA need some parameters to be set. In the first case, a niche distance parameter must be defined. This parameter, which indicates the maximum distance allowed between any two solutions to participate into a niche, is used to control the number of solutions allowed to be concentrated over small regions (niches) over the Pareto front. In the case of controlled elitist NSGA, the portion of the population that is allowed to keep the best non-dominated solutions must be set, which controls the extent of elitism.

The same initial pool of candidate classifiers generated at the overproduction phase in chapter 2 is used here. This is a pool of 100 kNN classifiers (k=1) created using the random subspace method. Moreover, the same NIST SD19 database is investigated. However, only data-test1 is concerned here. In Table IV, all these parameters are summarized. The three main search criteria, i.e. combination performance, ensemble size and diversity measures are employed in the experiments shown in this appendix. However, only ambiguity (as defined in Equation 2.1) is used as the diversity measure. Thus, the following two pairs of objective functions are used to guide the three MOGAs: the maximization of ambiguity combined with the minimization of the error rate; and the minimization of ensemble size combined with the minimization of the error rate.

The level of elitism defined is equal to 50% for controlled elitist NSGA. In terms of NSGA's parameters, we set the best niche distance values defined by Tremblay in [88].

In his work an extensive experimental study was conducted in order to determine the best parameters considering the same two pairs of objective functions investigated in this appendix. Taking into account his results, we set the niche distance values as defined bellow. The level of elitism for controlled elitist NSGA is also mentioned.

- NSGA guided by ambiguity and the error rate: niche distance=0.025;
- NSGA guided by ensemble size and the error rate: niche distance=0.05;
- Controlled elitist NSGA elitism level: 50%.

1.1 Performance Evaluation

Figure 23 shows the comparison results of 30 replications on data-test1 obtained using the first pair of objective functions. These results show that controlled elitism NSGA and NSGA2 presented equivalent performances, while NSGA was slightly worse.

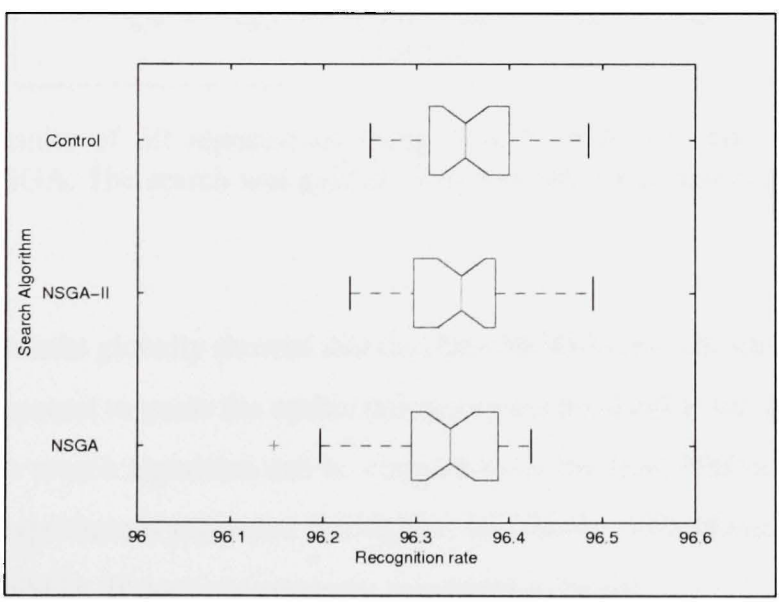


Figure 23 Results of 30 replications using NSGA, NSGA-II and controlled elitist NSGA. The search was guided using ambiguity and the error rate.

The results obtained by combining ensemble size and the error rate in a pair of objective functions to guide the three MOGAs are shown in Figure 24. Using this second pair of objective functions we observe that controlled elitism NSGA and NSGA presented equivalent performances. Thus, unlike the previous pair of objective results, NSGA2 was the worst search algorithm. However, it is important to note that the variance of the recognition rates achieved by each algorithm is small (from 96.1% to 96,6%).

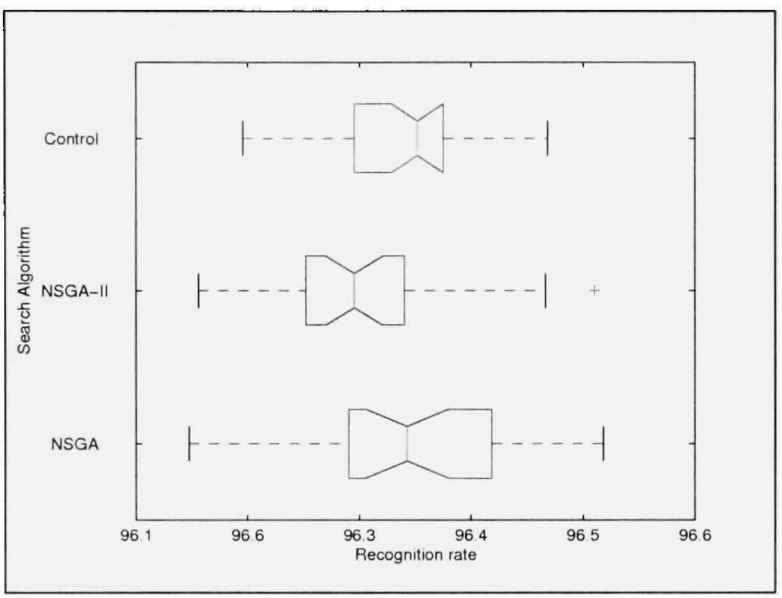


Figure 24 Results of 30 replications using NSGA, NSGA-II and controlled elitist NSGA. The search was guided using ensemble size and the error rate.

Therefore, our results globally showed that the three MOGAs investigated in this appendix are equally competent to guide the optimization process involved at the selection phase of SOCS. Since no search algorithm can be claimed to be the best, NSGA-II is the MOGA applied in the experiments presented throughout this thesis. This choice is specially due to the fact that NSGA-II does not need any parameter to be set.

APPENDIX 2

Overfitting Analysis for NIST-digits

In this appendix we carried out a comparative study using only two overfitting control strategies described in chapter 3: (1) partial validation PV; and (2) global validation GV. The objective is to verify whether or not overfitting is detected in optimization process using both GA and NSGA-II, guided by all the objective functions and combinations of objective functions discussed in chapter 2. Hence, we show here the overfitting analysis for the three series of experiments carried out in chapter 2: single-objective functions, diversity measures combined with the error rate ϵ and diversity measures combined with ensemble size ζ. The minimum number of classifiers allowed by the search algorithms was 5 and the pool of 100 kNN classifiers generated with the random subspace method was used in this experiments.

Table XXIV summarizes the average results on data-test1, and Table XXV summarizes the results on data-test2. The results are shown in bold when GV decreased the generalization error rate significantly, according to the Kruskal-Wallis statistical test. The results are shown underlined when GV increased the generalization error rate. The results with no overfitting control NV are also included in these tables.

Table XXIV

Comparing overfitting control methods on data-test1. Values are shown in bold when GV decreased the error rates significantly, and are shown underlined when it increased the error rates.

	GA - Single objective functions												
Method	ϵ	γ	σ	ρ	θ	η	δ	ξ	λ	τ	κ	ψ	Φ
NV	3.60	4.70	6.36	5.34	3.76	6.32	4.80	6.11	6.00	4.76	5.17	4.28	5.73
PV	3.60	4.70	6.35	5.61	3.76	6.32	4.80	6.12	6.00	4.84	5.16	4.28	5.73
GV	**3.55**	**4.63**	**5.92**	**5.15**	**3.67**	**5.81**	**4.53**	5.86	6.32	4.72	**4.96**	**4.25**	**5.43**
	NSGA-II - Pairs of objective function with the error rate ϵ												
NV	-	3.66	3.67	3.60	3.64	3.66	3.63	3.66	3.69	3.65	3.62	3.67	3.65
PV	-	3.70	3.68	3.59	3.63	3.69	3.64	3.67	3.70	3.63	3.60	3.67	3.63
GV	-	3.63	3.62	3.59	**3.60**	**3.64**	3.60	3.63	3.65	3.61	3.59	**3.63**	3.60
	NSGA-II - Pairs of objective function with ensemble size ζ												
NV	3.66	4.70	6.28	5.50	3.76	6.39	4.80	6.35	6.34	4.85	5.18	4.31	5.84
PV	3.67	4.70	6.28	5.50	3.83	6.39	4.80	6.35	6.34	4.85	5.18	4.31	5.84
GV	3.65	5.20	6.27	5.72	**3.71**	6.47	4.85	6.45	6.63	4.97	5.33	4.49	6.08

Table XXV

Comparing overfitting control methods on data-test2. Values are shown in bold when GV decreased the error rates significantly, and are shown underlined when it increased the error rates.

	GA - Single objective functions												
Method	ϵ	γ	σ	ρ	θ	η	δ	ξ	λ	τ	κ	ψ	Φ
NV	7.89	9.42	11.79	10.42	8.43	11.70	10.45	11.30	11.39	10.29	10.24	8.89	10.97
PV	7.91	9.42	11.79	10.64	8.44	11.70	10.45	11.33	11.39	9.98	10.21	8.89	10.97
GV	7.80	**9.31**	**11.02**	**10.01**	**8.11**	**10.92**	**9.85**	**10.82**	11.92	**9.64**	**10.01**	**8.81**	**10.37**
	NSGA-II - Pairs of objective function with the error rate ϵ												
NV	-	7.90	8.09	7.91	8.12	7.99	8.12	7.96	8.02	8.08	7.96	7.96	7.98
PV	-	7.97	8.11	7.90	8.11	8.01	8.14	7.98	8.01	8.00	7.92	7.97	7.94
GV	-	7.87	**7.94**	7.84	**7.93**	**7.88**	**7.93**	**7.87**	**7.90**	7.92	7.84	7.89	**7.84**
	NSGA-II - Pairs of objective function with ensemble size ζ												
NV	8.08	9.42	11.61	10.63	8.44	11.83	10.45	11.77	12.11	10.00	10.22	8.91	11.08
PV	8.08	9.42	11.61	10.63	8.49	11.83	10.45	11.77	12.11	10.00	10.22	8.91	11.08
GV	7.98	10.27	11.58	10.76	**8.25**	11.80	**10.42**	11.79	12.54	10.06	10.37	9.20	11.21

These experiments show that:

a. The overfitting phenomenon was detected and controlled in all the multi-objective optimization results when diversity measures were combined with ϵ. The GV procedure allowed us to find classifier ensembles with a higher power of generalization, whatever pair of objective functions was used to guide the NSGA-II search.

b. Except for fault majority λ, the GV procedure helped to find classifier ensembles with higher recognition rates in all the single-objective optimization problems studied in this paper. Thus, overfitting was detected even when diversity was used as the objective function.

c. The overfitting phenomenon was not detected on multi-objective optimization when diversity measures were combined with ζ. Thus, other than when ζ is combined with ϵ, the difficulty measure θ or coincident failure σ, GV leads to a decrease in the generalization performances. We can explain this apparent discrepancy by the fact that the combination of ζ and diversity is not related to performance.

The relationship between diversity and error rate is illustrated in Figures 25 and 26. For each diversity measure employed in the single-objective optimization experiments described in section 2.2.1, we have measured the uncontrolled overfitting $\phi = \epsilon(\mathcal{V}, C_j') - \epsilon(\mathcal{V}, C_j^{*\prime})$. These figures show ϕ based on 30 replications on data-test1 (Figure 25) and on data-test2 (Figure 26) for each diversity measure. According to these results, θ was the diversity measure that was more closely related to the error rate, while λ was the diversity measure that was less closely related to the error rate, since a stronger relationship between diversity and error rate leads to a lower ϕ.

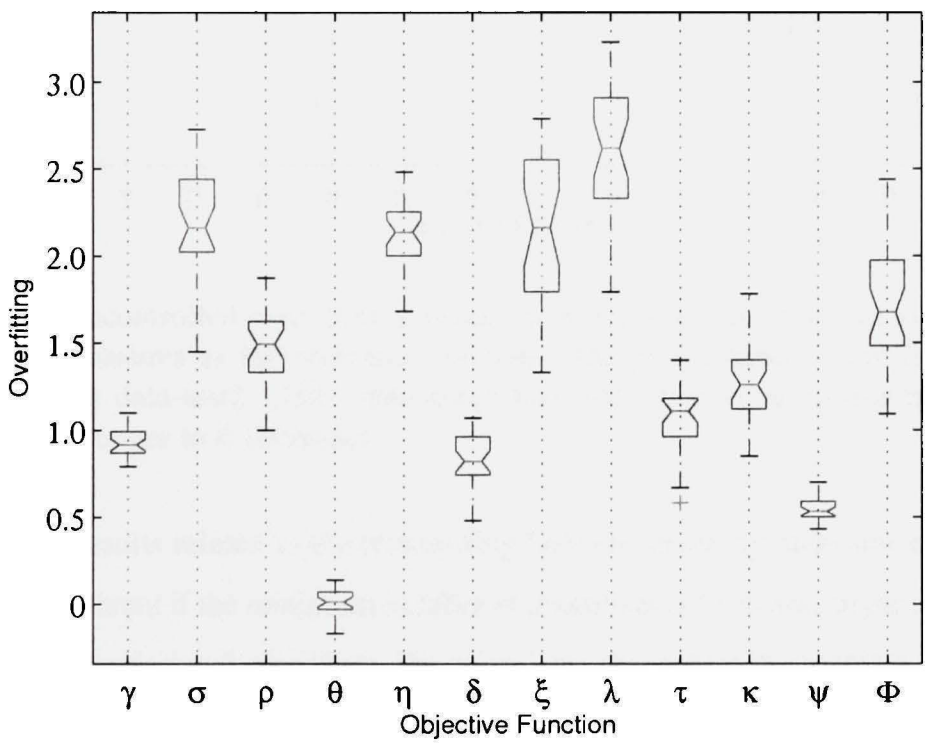


Figure 25 Uncontrolled overfitting ϕ based on 30 replications using GA with diversity measures as the objective function. The performances were calculated on the data-test1. The relationship between diversity and error rate becomes stronger as ϕ decreases.

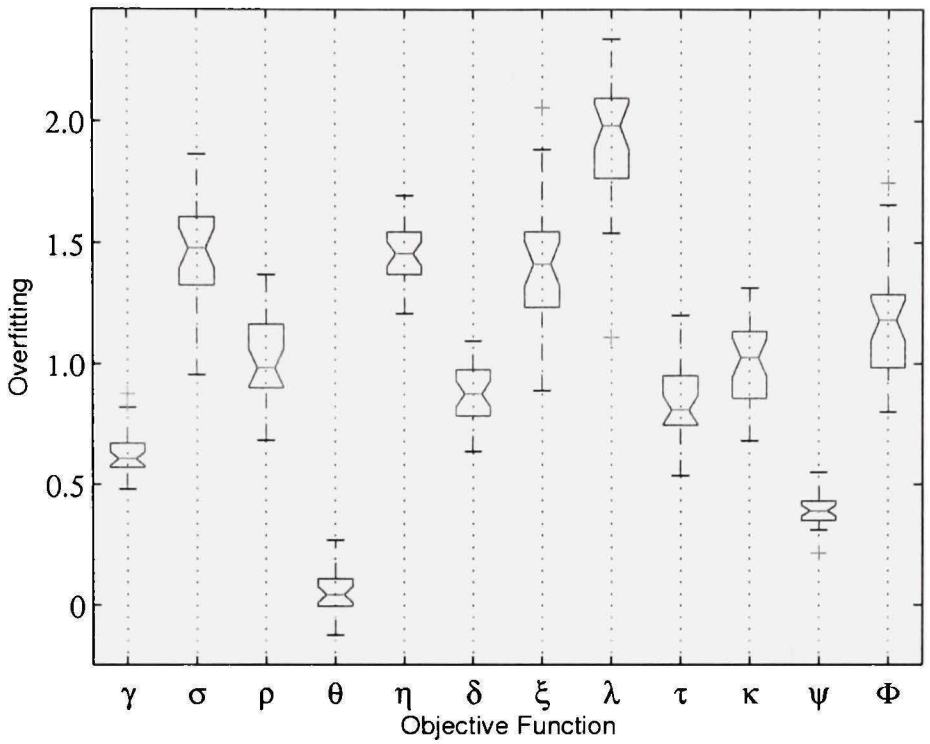


Figure 26 Uncontrolled overfitting ϕ based on 30 replications using GA with diversity measures as the objective function. The performances were calculated on the data-test2. The relationship between diversity and error rate becomes stronger as ϕ decreases.

However, the results related to the relationship between diversity measures and the error rate may be different if the minimum number of classifiers is fixed and larger than we have defined in this work, i.e. 5 classifiers. We investigate this aspect in chapter 3.

APPENDIX 3

Pareto Analysis

3.1 Introduction

Based on the results obtained in appendix 2 we observed that the performance of the solutions found in both single- and multi-objective optimization processes was increased by applying the global validation GV method to reduce overfitting. However, when ensemble size ζ and diversity were combined to make up pairs of objective functions to guide the optimization and the validation processes, the results were very different. GV increased the generalization error rates. We explained this behavior in appendix 2 by the following assumption: the combination of ζ and diversity is not related to the performance. First, in despite of the fact that good results were obtained using GV guided by diversity in the single-objective optimization approach, our experiments outlined that, although reduced, overfitting was not totally controlled. Second, even if we can establish an analogy between feature subset selection and SOCS, we cannot increase performance by selecting classifier ensembles with the minimum ζ, specially because we need to deal with a large initial pool of classifiers. Therefore, we cannot decrease the generalization error rate by combining this pair of objective functions.

In this appendix, we present further evidence on Pareto front analysis to show why diversity and ζ are not strongly related to the performance

3.2 Pareto Analysis

Objective functions in multi-objective optimization problems are often conflicting. Since different tradeoffs are established over the Pareto front, when one solution is better according to one objective, it is often worse according to the remaining objective functions. Indeed, Deb [12] points out that, for instance in a two conflicting objective problem, if a ranking of nondominated solutions is carried out in an ascending order according to one objective function, a ranking in a descending order is obtained according to the other objective function.

These observations are illustrated in Figure 27. NSGA-II was employed as the search algorithm. In Figure 27(a) the search was guided by the minimization of the error rate ϵ and the difficulty measure θ. The Pareto front solutions are shown in an ascending order of ϵ, consequently, in a descending order of the θ. Figure 27(b) shows the Pareto front found using the following pair of objective functions: jointly minimize ϵ and ζ. Once again, the solutions were ordered according to ϵ, and in a descending order of ζ. In Figure 27(c) the pair of objective functions employed was the minimization of ζ and θ. The Pareto solutions are shown in an ascending order of θ (descending order of ζ). However, the same behavior cannot be detected in Figure 27(d) where it is shown an example of the evolution of the optimization process after 1,000 generations which was generated using the following pair of objective functions: jointly minimize ζ and the interrater agreement κ. It can be seen that only one solution was found over the Pareto front.

The first two figures show that either θ or ζ combined with ϵ are conflicting objectives. In Figure 27(c) it is shown that θ and ζ are also conflicting objectives. In contrast, κ and ζ are not conflicting objectives. In order to show whether or not the pairs of objective functions applied in chapter 2 are conflicting objective functions we apply here two quality metrics based on calculating the overall Pareto spread and k^{th} objective Pareto spread introduced by Wu and Azarm [103]. Given a multi-objective problem with m objective functions $f_1, f_2, \ldots, f_m$, we show how to calculate the two measures of spread taking into account, without loss of generality, all objective functions to be minimized and equally important.

Given s_w be the possible worst solution and s_b be the possible best solution, the objective space should be scaled by the following equation:

$$\bar{f}_j(x_k) = \frac{f_j(x_k) - f_j^b}{f_j^w - f_j^b} \tag{3.1}$$

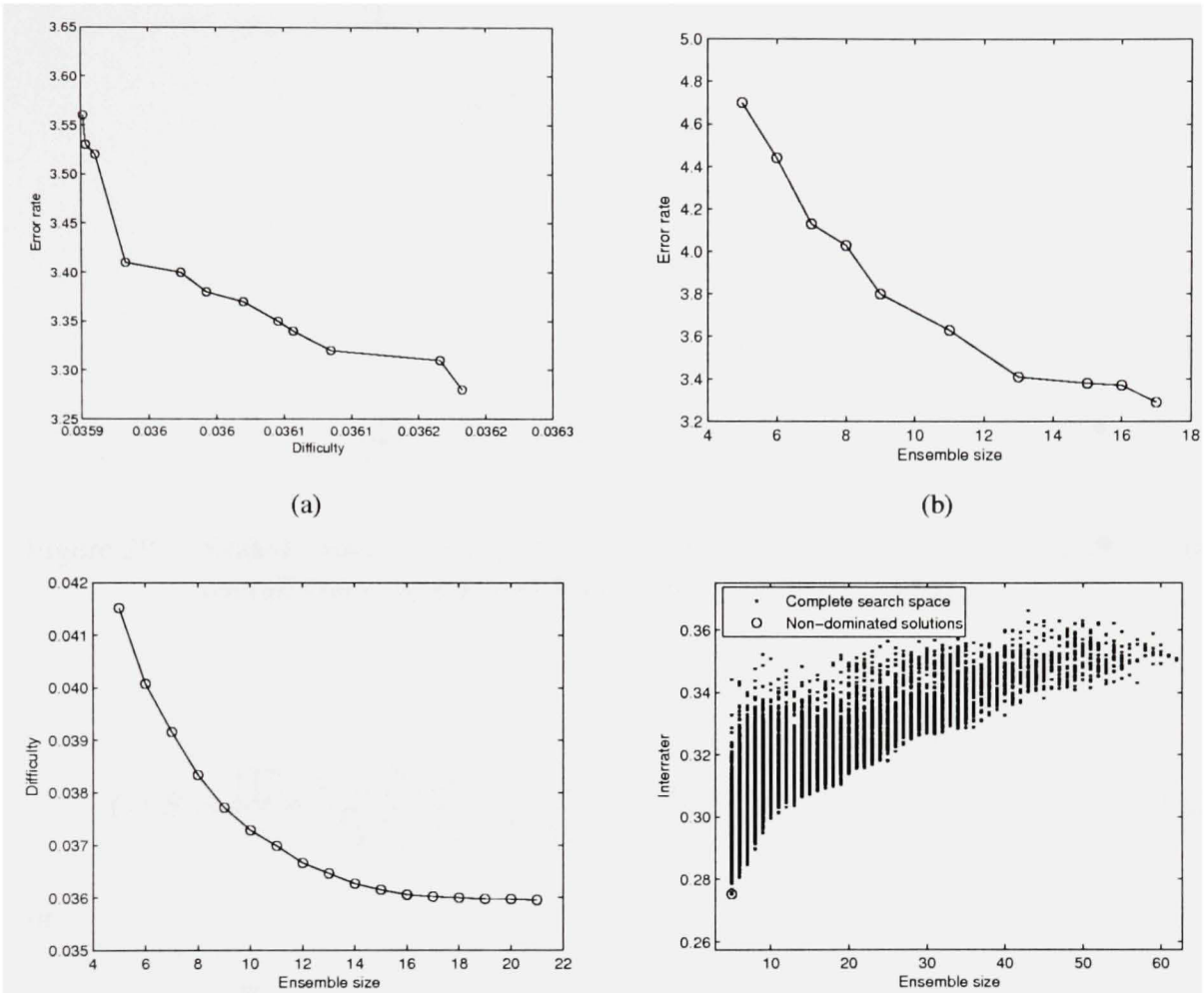


Figure 27 Pareto front after 1000 generations found using NSGA-II and the pairs of objective functions: jointly minimize the error rate and the difficulty measure (a), jointly minimize the error rate and ensemble size (b), jointly minimize ensemble size and the difficulty measure (c) and jointly minimize ensemble size and the interrater agreement (d).

In a two-objective problem, the scaled objective space is a hyper-rectangle, as shown in Figure 28 defined by the scaled worst $(1, 1)$ and best $(0, 0)$ solutions. The overall Pareto spread is defined as:

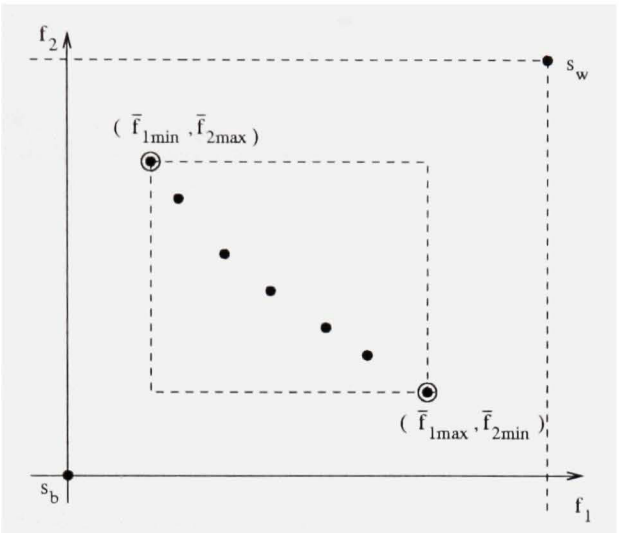


Figure 28 Scaled objective space of a two-objective problem used to calculate the overall Pareto spread and the k^{th} objective Pareto spread.

$$OPS = \frac{\prod_{i=1}^{m} |max_{k=1}^{\bar{np}}(p_k)_i - min_{k=1}^{\bar{np}}(p_k)_i|}{\prod_{i=1}^{m} |(p_w)_i - (p_b)_i|} \tag{3.2}$$

or

$$OPS = \prod_{i=1}^{m} |max_{k=1}^{\bar{np}}[\bar{f}_i(x_k)] - min_{k=1}^{\bar{np}}[\bar{f}_i(x_k)]| \tag{3.3}$$

where $\bar{np}$ is the total number of Pareto solutions. When we are dealing with no conflicting objective functions OPS is equal to 0. Hence, a wider spread leads to more diversity over the Pareto front which is a desired Pareto property. However, OPS does not measure the individual objective spread. For example, if instead of having 7 solutions over the Pareto front as shown in Figure 28, we had only two solutions, for instance $(\bar{f}_{1min}, \bar{f}_{2max})$ and $(\bar{f}_{1max}, \bar{f}_{2min})$, the same OPS value would be obtained but the diversity over a Pareto with only two solutions is much smaller than the diversity over a Pareto with 7 solutions. In order to overcome this problem, the k^{th} Objective Pareto Spread (IPS), also proposed by Wu and Azarm[103], is applied. The IPS_k is calculated as:

$$IPS_k = \frac{|max_{i=1}^{\bar{np}}((p_i)_k) - min_{i=1}^{\bar{np}}((p_i)_k)|}{|(p_w)_k - (p_b)_k|} \tag{3.4}$$

or

$$IPS_k \quad |max_{i=1}^{\bar{np}}(\bar{f}_k(x_i)) - min_{i=1}^{\bar{np}}(\bar{f}_k(x_i))| \tag{3.5}$$

In the two-objective problem shown in Figure 28, the k^{th} objective Pareto Spread and the overall Pareto spread are respectively:

$$IPS_{f_1} = |\bar{f}_{1max} - \bar{f}_{1min}| \tag{3.6}$$

$$IPS_{f_2} = |\bar{f}_{2max} - \bar{f}_{2min}| \tag{3.7}$$

$$OPS = IPS_{f_1} IPS_{f_2} \tag{3.8}$$

The k^{th} objective Pareto Spread is important to measure the diversity over the Pareto front and to show whether or not spread is wider for one objective function than for the others. We try to identify if there is more variation in one objective function or if the objective functions are equally diverse. Table XXVI presents the k^{th} objective Pareto Spread and the overall Pareto spread calculated for each Pareto front shown in Figure 27.

The difficulty measure θ and the error rate ϵ are conflicting objective functions but there is much more variation among ϵ values than among θ values. Ensemble size ζ and ϵ are conflicting objective functions and there is more variation in ζ values than in ϵ values. In addition, θ and ζ are conflicting objective functions but once again, there is less variation among θ values. It is important note that there is more variation among θ values when this diversity measure is combined with ζ than when it is combined with ϵ. Finally, the interrater agreement κ and ζ are not conflicting objective functions.

Table XXVI

Worst and best points, minima and maxima points, the k^{th} objective Pareto spread values and overall Pareto spread values for each Pareto front.

Objective Functions	**Original values**				**Scaled values**					
	s_w	s_b	min	max	s_w	s_b	min	max	IPS	OPS
Difficulty θ (f_1)	1	0	0.0359	0.0362	1	0	0.0359	0.0362	0.0003	
Error rate ϵ (f_2)	100	0	3.2800	3.5600	1	0	0.0328	0.0356	0.0028	0.1×10^{-5}
Ensemble size ζ (f_1)	100	0	5	17	1	0	0.0500	0.1700	0.1200	
Error rate ϵ (f_2)	100	0	3.2900	4.7000	1	0	0.0329	0.0470	0.0141	0.17×10^{-2}
Ensemble size ζ (f_1)	100	0	21	5	1	0	0.0500	0.2100	0.1600	
Difficulty θ (f_2)	1	0	0.0415	0.0356	1	0	0.0356	0.0415	0.0059	0.9×10^{-3}
Ensemble size ζ (f_1)	100	0	5	5	1	0	0.0500	0.0500	0.0000	
Interrater κ (f_2)	1	0	0.2752	0.2752	1	0	0.2752	0.2752	0.0000	0.0000

We have calculated both quality measures in order to show which measures are conflicting or not. The same pairs of objective functions investigated in appendix 2 were used here: diversity measures combined with ϵ; diversity measures combined with ζ and ϵ combined with ζ. Table XXVII shows the average results from 30 replications for each pair of objective functions. This table shows the values of the k^{th} objective Pareto spread and the overall Pareto spread, as well as the difference between the k^{th} objective Pareto spreads. This difference indicates the variation among objective functions values.

These results show that all diversity measures are conflicting objective functions when combined with ϵ. Except Kohavi-Wolpert ψ, θ and ambiguity γ, there is more variation among diversity values than among ϵ values. Ensemble size ζ and ϵ are also conflicting

objective functions and there is more variation among ζ values. However, there are only three diversity measures which are conflicting objective functions when combined with ensemble size, the same ψ, θ and γ.

The literature has shown that the pairwise diversity measures and ζ are not conflicting objective functions. Aksela and Laaksonen [1] observe that pairwise diversity measures always try to find the two most diverse classifiers which leads to the minimization of the number of classifiers during the optimization process. Our results confirm this observation since all pairwise measures (see Chapter 2, section 2.2.1 employed in our experiments are not conflicting objective functions when combined with ζ. However, our results show that some of the nonpairwise measures lead also to the minimization of ζ. Three of the nonpairwise measures employed: γ, θ and ψ, do not minimize ζ.

Table XXVII

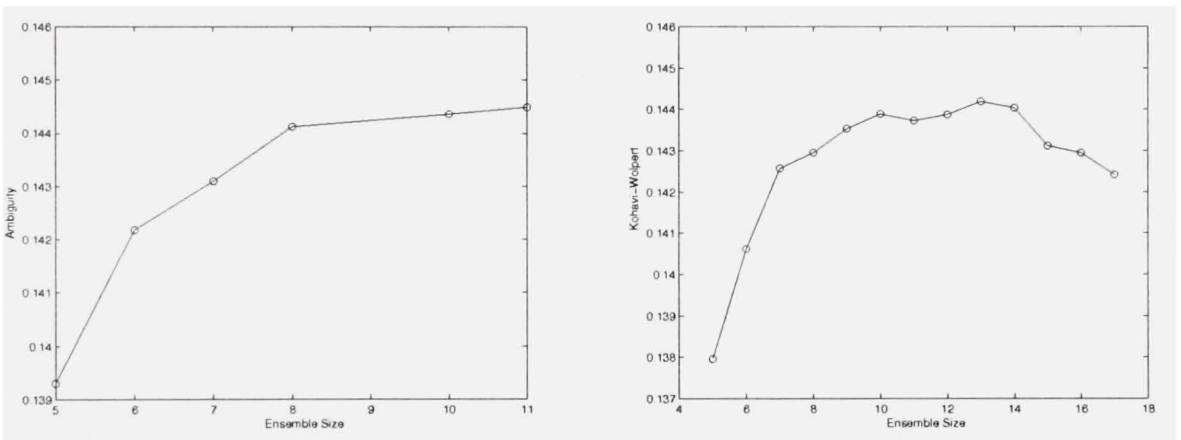
The average k^{th} objective Pareto spread values and the average overall Pareto spread values for each pair of objective function.

	Error rate				**Ensemble size**			
Diversity	IPS_ϵ	IPS_{div}	$IPS_{div} - IPS_\epsilon$	OPS	IPS_ζ	IPS_{div}	$IPS_\zeta - IPS_{div}$	OPS
γ	0.0187	0.0254	0.0067	0.5×10^{-3}	0.0503	0.0030	-0.0474	0,0001
σ	0.0134	0.0368	0.0234	0.5×10^{-3}	0.0000	0.0000	0.0000	0.0000
ρ	0.0213	0.0442	0.0229	0.9×10^{-3}	0.0000	0.0000	0.0000	0.0000
θ	0.0025	0.0006	-0.0019	0.2×10^{-5}	0.1733	0.0058	-0.1675	0.0010
η	0.0330	0.0428	0.0098	0.14×10^{-2}	0.0000	0.0000	0.0000	0.0000
δ	0.0159	0.0068	-0.0091	0.1×10^{-3}	0.0000	0.0000	0.0000	0.0000
ξ	0.0249	0.0659	0.0410	0.17×10^{-2}	0.0000	0.0000	0.0000	0.0000
λ	0.0192	0.0460	0.0268	0.9×10^{-3}	0.0000	0.0000	0.0000	0.0000
τ	0.0150	0.0357	0.0207	0.5×10^{-3}	0.0000	0.0000	0.0000	0.0000
κ	0.0159	0.0447	0.0288	0.7×10^{-3}	0.0000	0.0000	0.0000	0.0000
ψ	0.0119	0.0113	-0.0006	0.1×10^{-3}	0.1000	0.0079	-0.0921	0.0008
Φ	0.0265	0.0927	0.0662	0.25×10^{-2}	0.0000	0.0000	0.0000	0.0000
	Ensemble size ζ and Error rate ϵ							
	IPS_ϵ		IPS_ζ		$IPS_\zeta - IPS_\epsilon$		OPS	
	0.0119		0.2163		0.2045		0.0025	

According to Whitaker and Kuncheva [99] nonpairwise measures are calculated by using either entropy or correlation between individual outputs and the ensemble's output or distribution of "difficulty" of the data samples. Among the three diversity measures, which

are conflicting objective functions when combined with ζ, two measures are based on the variance over the dataset (θ and ψ) and γ is based on the variance among ensemble's members. Thus, our observation is that, besides the pairwise diversity measures pointed out by Aksela and Laaksonen [1], all non-pairwise diversity measures based on entropy or correlation between individual outputs are also not able to find the best ensemble's size, i.e. they minimize ζ during the optimization process.

Moreover, there is no guarantee of finding the best ensemble's size using one of the three diversity measures based on variance, specially ambiguity, which generates the less diverse Pareto front (Table XXVII and Figure 29(a)) and the smallest classifier ensembles (Figure 29(a)). However, these three measures have such a property that makes them able to generate larger classifier ensembles, which is useful in OCS methods since we deal with a large initial pool of classifiers. The two diversity measures based on the variance over the dataset found more diverse Pareto front and larger classifier ensembles than ambiguity (based on the variance of the classifiers' output). The difficulty measure θ was better than Kohavi-Wolpert ψ in these two aspects, as it can seen in Figure 29(b) for ψ and in Figure 27(c) for θ, as well as in Table XXVII.



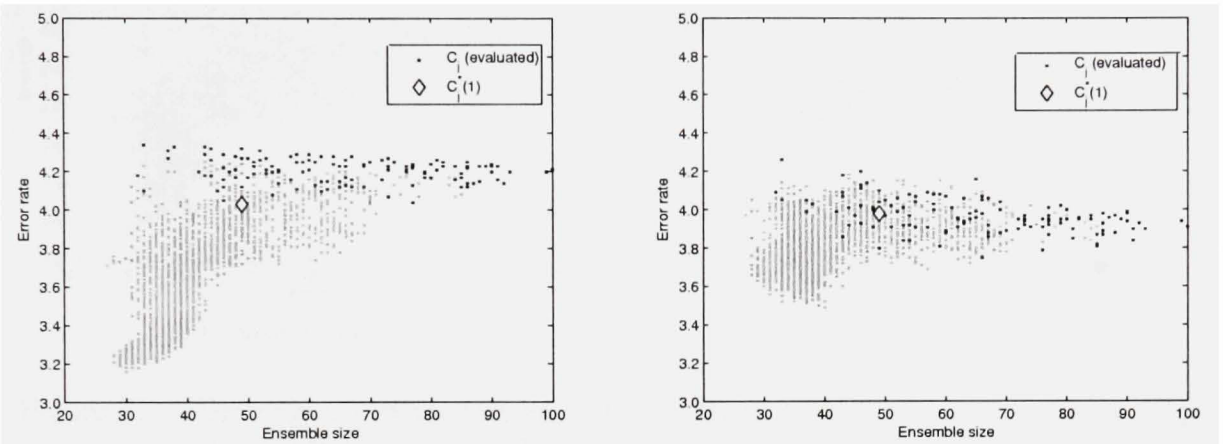
(a) (b)

Figure 29 Pareto front after 1000 generations found using NSGA-II and the pairs of objective functions: minimize ensemble size and maximize ambiguity (a) and minimize ensemble size and maximize Kohavi-Wolpert (b).

APPENDIX 4

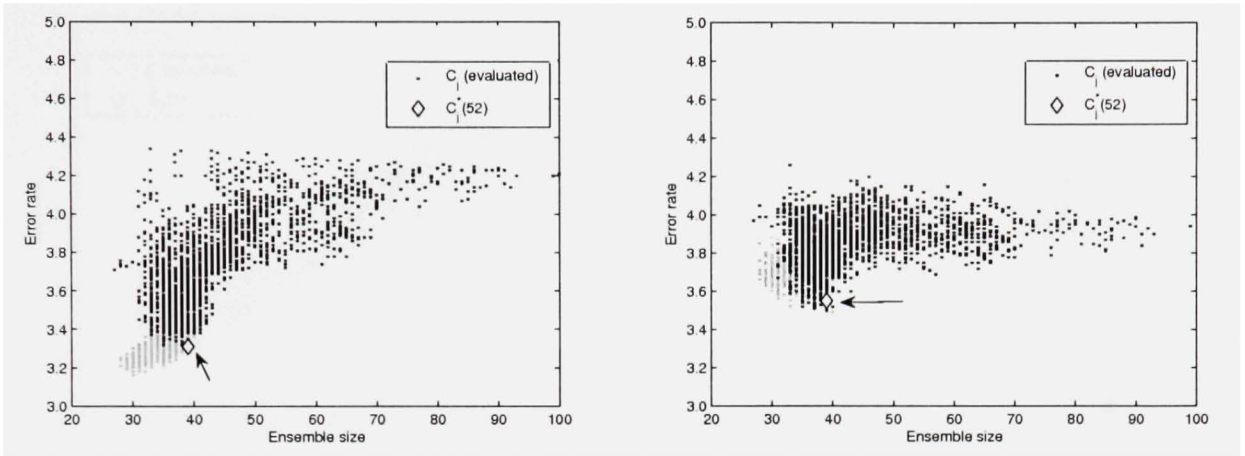
Illustration of overfitting in single- and multi-objective GA

In this appendix we show the Figures 13 and 14, which illustrated the overfitting problem in single- and multi-objective GA in chapter 3, in larger versions.



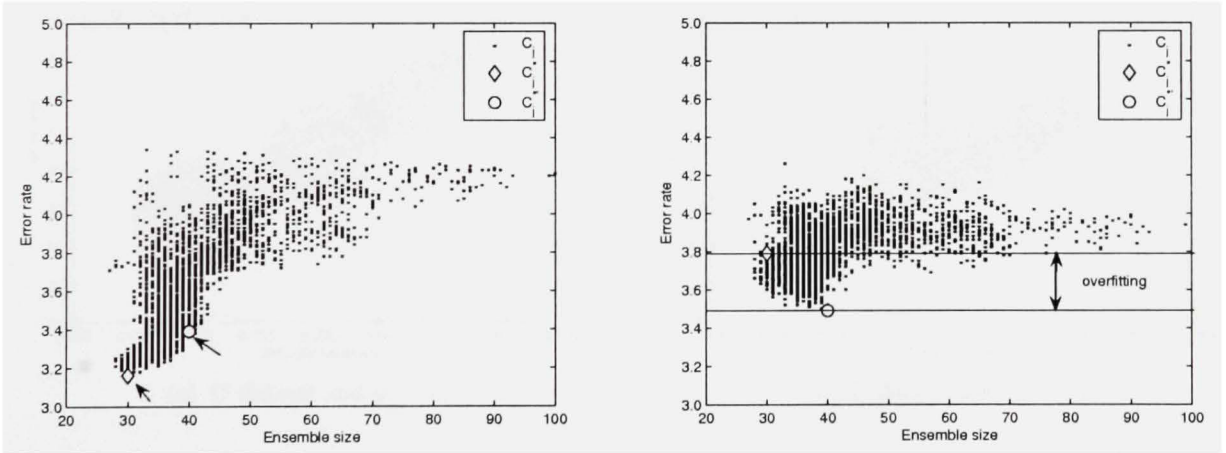
(a) $\mathcal{O}$ dataset and $g = 1$ (b) $\mathcal{V}$ dataset and $g = 1$

Figure 30 Optimization using GA guided by ϵ. Here, we follow the evolution of $C_j^*(g)$ (diamonds) for $g = 1$ (Figure 30(a)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figure 30(b)). Solutions not yet evaluated are in grey and the best performing solutions are highlighted by arrows.



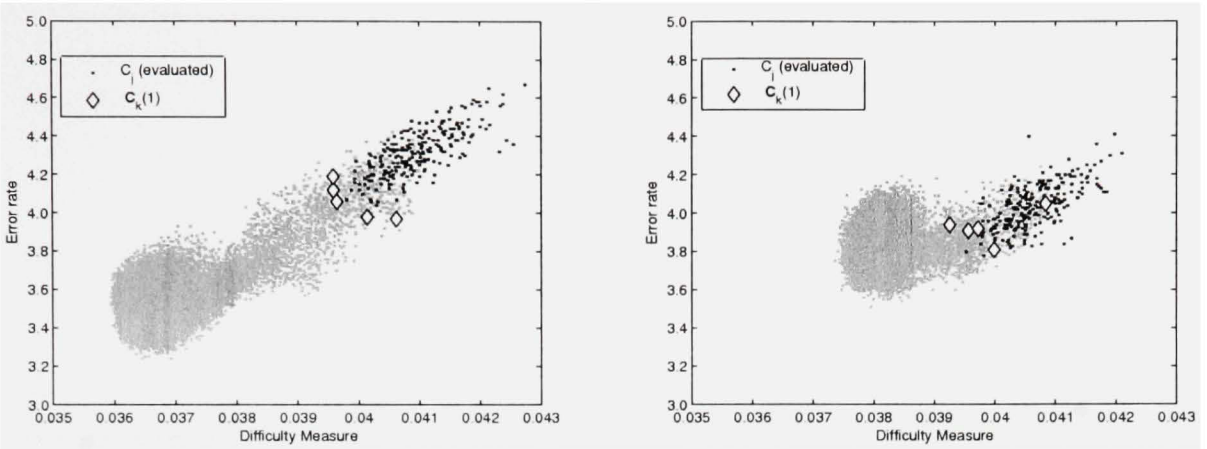
(a) $\mathcal{O}$ dataset and $g = 52$ (b) $\mathcal{V}$ dataset and $g = 52$

Figure 31 Optimization using GA guided by ϵ. Here, we follow the evolution of $C_j^*(g)$ (diamonds) for $g = 52$ (Figure 31(a)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figure 31(b)). The minimal error is reached slightly after $g = 52$ on $\mathcal{V}$, and overfitting is measured by comparing it to the minimal error reached on $\mathcal{O}$. Solutions not yet evaluated are in grey and the best performing solutions are highlighted by arrows.



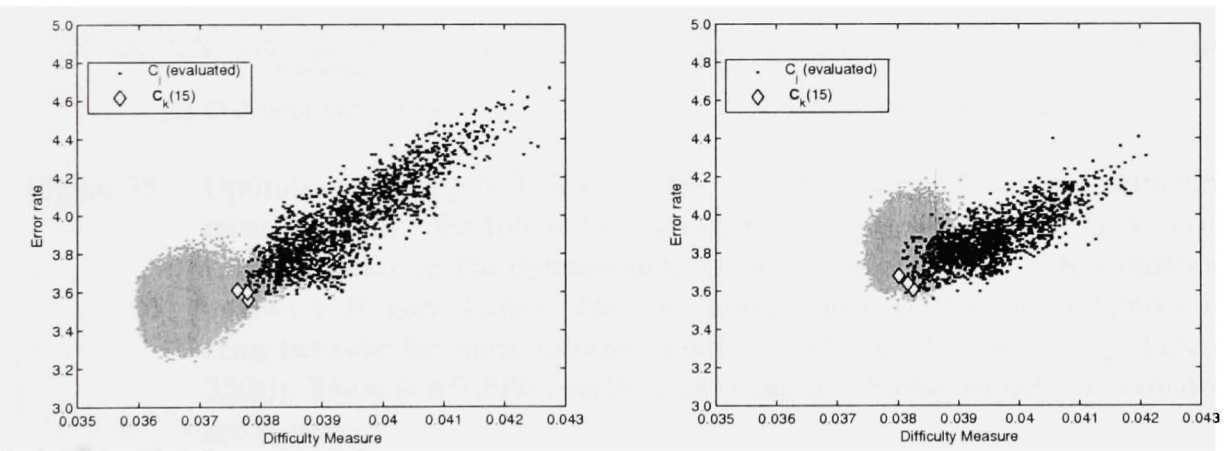
(a) $\mathcal{O}$ dataset and $max(g)$ (b) $\mathcal{V}$ dataset and $max(g)$

Figure 32 Optimization using GA guided by ϵ. Here, we follow the evolution of $C_j^*(g)$ (diamonds) for $max(g)$ (Figure 32(a)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figure 32(b)). The overfitting is measured as the difference in error between $C_j'^*$ (circles) and C_j^* (Figure 32(b)). There is a 0.30% overfit in this example. Solutions not yet evaluated are in grey and the best performing solutions are highlighted by arrows.



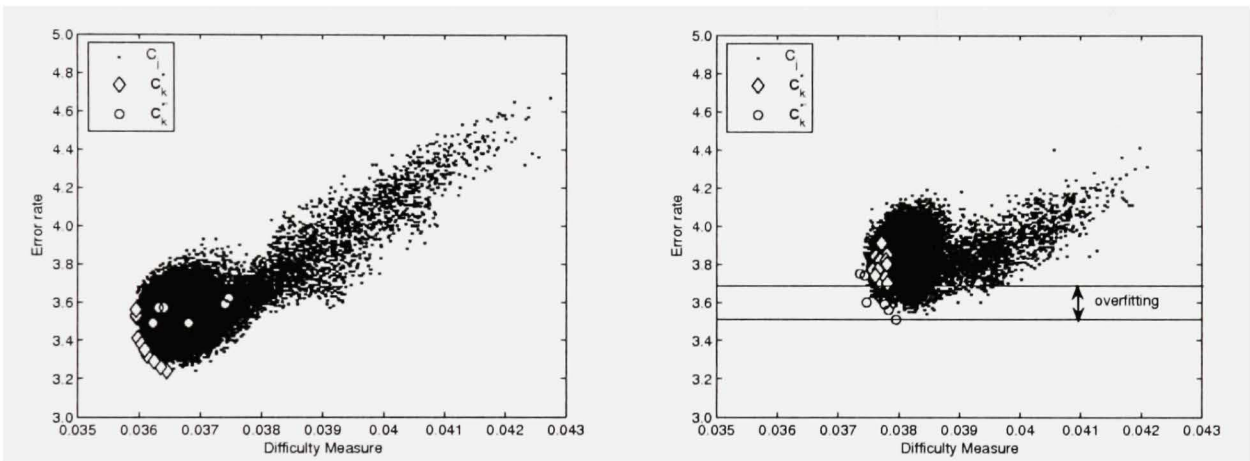
(a) $\mathcal{O}$ dataset and $g = 1$ (b) $\mathcal{V}$ dataset and $g = 1$

Figure 33 Optimization using NSGA-II and the pair of objective functions: difficulty measure and ϵ. We follow the evolution of $\mathbf{C}_k(g)$ (diamonds) for $g = 1$ (Figure 33(a)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figure 33(b)). Solutions not yet evaluated are in grey.



(a) $\mathcal{O}$ dataset and $g = 15$ (b) $\mathcal{V}$ dataset and $g = 15$

Figure 34 Optimization using NSGA-II and the pair of objective functions: difficulty measure and ϵ. We follow the evolution of $\mathbf{C}_k(g)$ (diamonds) for $g = 15$ (Figure 34(a)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figure 34(b)). The minimal error is reached slightly after $g = 15$ on $\mathcal{V}$, and overfitting is measured by comparing it to the minimal error reached on $\mathcal{O}$. Solutions not yet evaluated are in grey.



(a) $\mathcal{O}$ dataset and $max(g)$ (b) $\mathcal{V}$ dataset and $max(g)$

Figure 35 Optimization using NSGA-II and the pair of objective functions: difficulty measure and ϵ. We follow the evolution of $\mathbf{C}_k(g)$ (diamonds) for $max(g)$ (Figure 35(a)) on the optimization dataset $\mathcal{O}$, as well as on the validation dataset $\mathcal{V}$ (Figure 35(b)). The overfitting is measured as the difference in error between the most accurate solution in $\mathbf{C}_k^{*'}$ (circles) and in $\mathbf{C}_k^*$ (Figure 35(b)). There is a 0.20% overfit in this example. Solutions not yet evaluated are in grey.

APPENDIX 5

Comparison between Particle Swarm Optimization and Genetic Algorithm taking into account overfitting

The objective of this appendix is to present the results of a short comparative study between Particle Swarm Optimization (PSO) [34] and GA in order to show which algorithm is more prone to overfitting. These experiments were conducted using only the error rate ϵ as the objective function. The comparative study is focused on detecting and controlling overfitting in both GA and PSO search algorithms at the optimization process, by applying the global validation strategy GV.

5.1 Particle Swarm Optimization

PSO simulates the behaviors of bird flocking or fish schooling. Each individual of the population is called particles. All particles have fitness values which are evaluated during the optimization process of the selection phase of SOCS. Algorithm 8 summarizes the variant of the PSO algorithm used in this appendix. This variant is called *global neighborhood* [57] in the literature.

Algorithm 8 PSO

```
1: for each particle do
2:    Initialize particle
3: end for
4: while maximum iterations or stop criteria are not attained do
5:    for each particle do
6:       Compute fitness value
7:       if fitness value is better than the best fitness (pBest) in history then
8:          Set current value as the new pBest
9:       end if
10:   end for
11:   Choose the particle with the best fitness as the gBest
12:   for each particle do
13:      Compute its velocity
14:      Update its position
15:   end for
16: end while
```

The solution $gBest$ denotes the best particle in the whole population. In PSO the population is called swarm. Letting $\mathbf{C}(g)$ denote the swarm in iteration g and ϵ represent the objective function, the complete GV algorithm for PSO is described in Algorithm 9.

Algorithm 9 Global Validation for PSO

1: Creates initial swarm
2: $\mathcal{A} = \emptyset$
3: Validate all solutions in $\mathbf{C}(1)$ over samples contained in $\mathcal{V}$
4: Find $C_j'^*(1)$ from $\mathbf{C}(1)$
5: Set $C_j'^*$:=$C_j'^*(1)$
6: **for** each iteration $g \in \{1, \ldots, max(g)\}$ **do**
7: run PSO and generate $\mathbf{C}(g+1)$
8: validate all solutions in $\mathbf{C}(g+1)$
9: find $C_j'^*(g+1)$
10: **if** $\epsilon(\mathcal{V}, C_j'^*)(g+1) < \epsilon(\mathcal{V}, C_j'^*)$ **then**
11: set $C_j'^* := C_j'^*(g+1)$
12: update $\mathcal{A}$ by storing in it the new $C_j'^*$
13: **end if**
14: **end for**
15: **return** $C_j'^*$ stored in $\mathcal{A}$

5.2 Experiments

In order to develop our experiments we use the two large datasets described in chapter 3, Table VI: NIST-digits and NIST-letters. The same initial pool of 100 kNN classifiers generated by applying the Random Subspace method is investigated here.

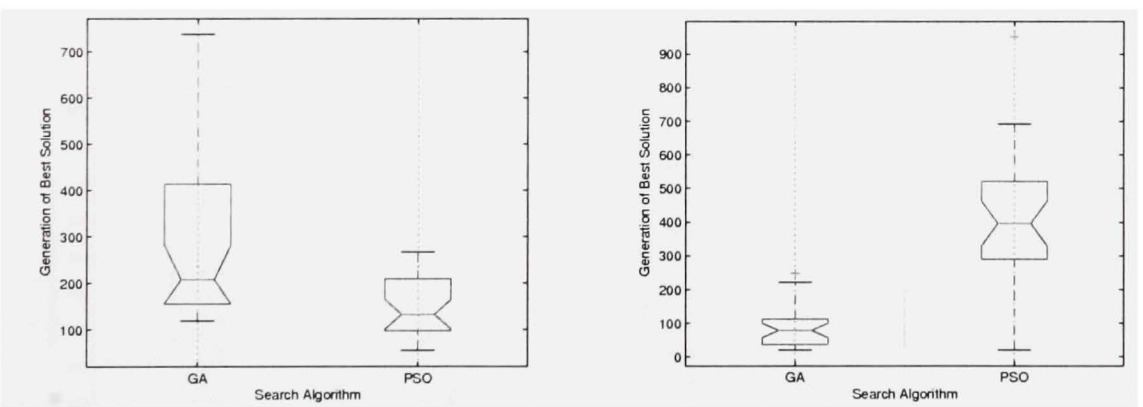
Table XXVIII shows the average results from 30 replications obtained using NIST-digits dataset. Values are shown in bold when GV decreased the generalization error significantly, according to the Kruskal-Wallis statistical test. The results with no overfitting control NV are also included in this table. These initial results show that although both search algorithms are prone to overfitting, GA appears to be more affected. It is important to note that this behavior confirms Reunamen's work [65]. He pointed out that the degree of overfitting increases as the intensity of search increases. Figure 36(a) shows that GA

needs more generations to find the best solution during the optimization process than PSO. On the other hand, the best solution $C_j^{*'}$ stored in $\mathcal{A}$ is found much earlier using GA than PSO (Figure 36(b)). This result can be explained by the fact that, since during the optimization process PSO keeps searching by solutions even worse than $gBest$, it is possible to generate solutions that are better in generalization after finding $gBest$.

Table XXVIII

Comparing GA and PSO in terms of overfitting control on NIST-digits dataset. Values are shown in bold when GV decreased the error rates significantly. The results were calculated using data-test1 and data-test2.

Validation Method	**GA**		**PSO**	
	data-test1	data-test2	data-test1	data-test2
NV	3.60	7.91	3.62	8.02
GV	**3.55**	7.80	3.61	7.88



(a) (b)

Figure 36 NIST-digits: the convergence points of GA and PSO. The generation when the best solution was found on optimization (36(a)) and on validation (36(b)).

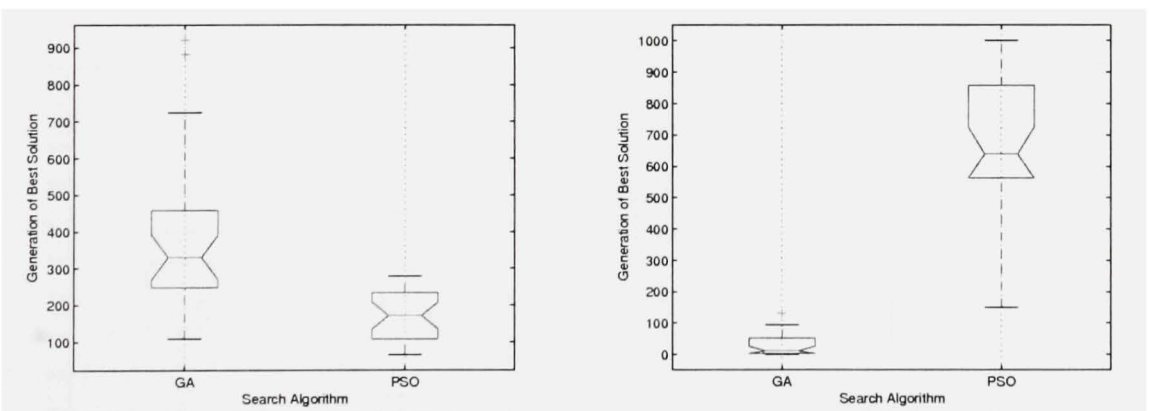
However, the experiments with NIST-letters did not confirm our first observations. The average results from 30 replications obtained using NIST-letters dataset are reported in Table XXIX. These results show that GV only slightly decreased the generalization error

rate for NIST-letters. In despite of this, the results obtained using NIST-letters dataset allow us to confirm that GA performs a more intense search process. Figure 37(a) shows that GA needs more generations to find the best solution during the optimization process than PSO also on NIST-letters dataset. Again, GA finds $C_j^{*'}$ much earlier than PSO (Figure 37(b)). Moreover, the performance of the solutions found by GA were slightly better using both NIST-digits (Figure 38) and NIST-letters (Figure 39(a)) datasets. However, the classifier ensembles found employing GA were larger than those found using PSO, as illustrated in Figure 40 for NIST-digits and in Figure 39(b) for NIST-letters.

Table XXIX

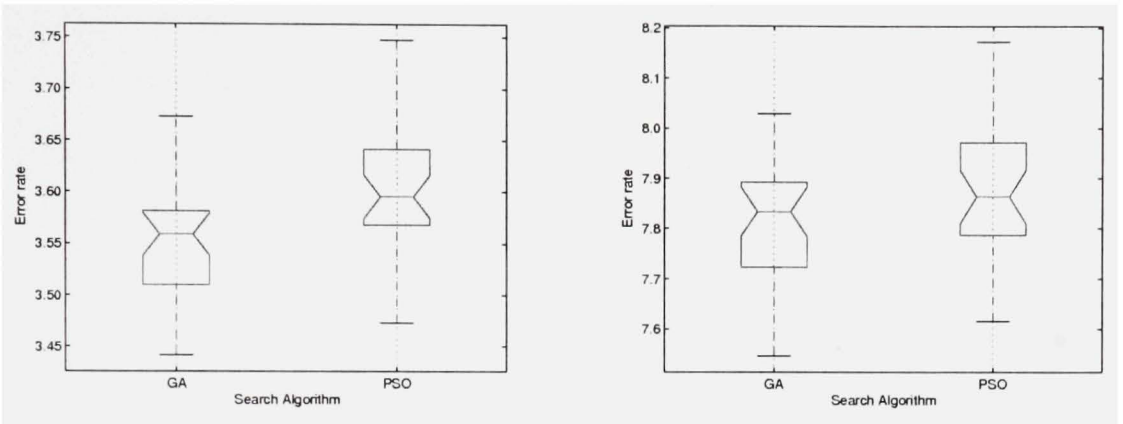
Comparing GA and PSO in terms of overfitting control on NIST-letter dataset.

Data set	**No validation**		**Global validation**	
	GA	PSO	GA	PSO
NIST-letters	6.49	6.54	6.44	6.49



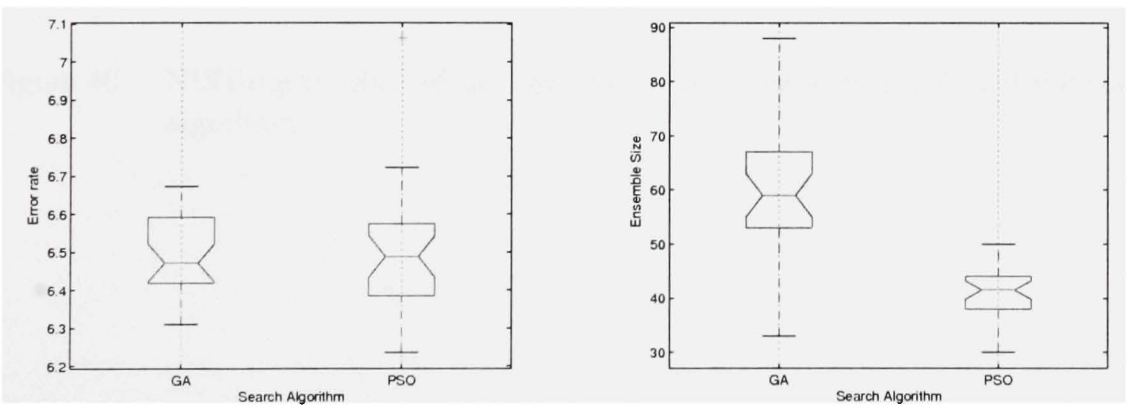
(a) (b)

Figure 37 NIST-letters: the convergence points of GA and PSO. The generation when the best solution was found on optimization (37(a)) and on validation (37(b)).



(a) (b)

Figure 38 NIST-digits: error rates of the solutions found on 30 replications using GA and PSO. The performances were calculated on the data-test1 (38(a)) and on the data-test2 (38(b)).



(a) (b)

Figure 39 NIST-letters: error rates of the solutions found on 30 replications using GA and PSO (39(a)). Size of the ensembles found using both GA and PSO search algorithms (39(b)).

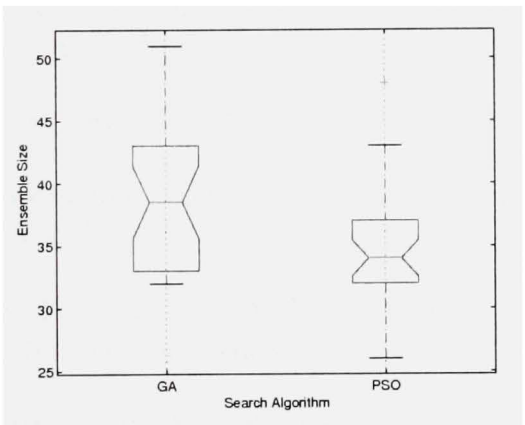


Figure 40 NIST-digits: Size of the ensembles found using both GA and PSO search algorithms.

BIBLIOGRAPHY

[1] M. Aksela and J. Laaksonen. Using diversity of errors for selecting members of a committee classifier. *Pattern Recognition*, 39(4):608–623, 2006.

[2] H. Altinçay. Ensembling evidential k-nearest neighbor classifiers through multi-modal pertubation. *Applied Soft Computing*, 7(3):1072–1083, 2007.

[3] S. Bandyopadhyay, S.K. Pal, and B. Aruna. Multiobjective gas, quantitative indices, and pattern classification. *IEEE Transactions on System, Man, and Cybernetics - PartB*, 34(5):2088–2099, 2004.

[4] N. Benahmed. Optimisation de réseaux de neurones pour la reconnaissance de chiffres manuscrits isolés: Sélection et pondération des primitives par algorithmes génétiques. Master's thesis, École de technologie supérieure, 2002.

[5] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[6] L. Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3):229–242, 2000.

[7] G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[8] A. Canuto, G. Howells, and M. Fairhurst. The use of confidence measures to enhance combination strategies in multi-network neuro-fuzzy systems. *Connection Science*, 12(3/4):315–331, 2000.

[9] A. Canuto, M.C.C.Abreu, L.M. Oliveira, J.C. Xavier Jr., and A. M. Santos. Investigating the influence of the choice of the ensemble members in accuracy and diversity of selection-based and fusion-based methods for ensembles. *Pattern Recognition Letters*, 28:472–486, 2007.

[10] D. Corne and J. Knowles. No free lunch and free leftovers theorems for multiobjective optimization problems. In *Proceedings of Evolutionary Multi-Criterion Optimization*, pages 327–341, Faro, Portugal, 2003.

[11] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, LTD, 2001.

[12] K. Deb. Unveiling innovative design principles by means of multiple conflicting objectives. *Engineering Optimization*, 35(5):445–470, 2003.

[13] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 849–858, Paris, France, 2000.

[14] K. Deb and T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. In *Proceedings of First Evolutionary Multi-Criterion Optimization*, pages 67–81, 2001.

[15] L. Didaci, G. Giacinto, F. Roli, and G.L. Marcialis. A study on the performances of the dynamic classifier selection based on local accuracy estimation. *Pattern Recognition*, 28:2188–2191, 2005.

[16] T. G. Dietterich. Ensemble methods in machine learning. In *Proceedings of I International Workshop on Multiple Classifier System*, pages 1–15, Cagliari, Italy, 2000.

[17] T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–158, 2000.

[18] T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correting output codes. *Journal of Artificial Intteligence Research*, 2:263–286, 1995.

[19] P. Domingos. A unified bias-variance decomposition and its applications. In *Proc. 17th International Conf. on Machine Learning*, pages 231–238. Morgan Kaufmann, San Francisco, CA, 2000.

[20] J. Franke and M. Oberlander. Writing style detection by statistical combination of classifiers in form reader applications. In *Proceedings of II International Conference on Document Analysis and Recognition*, pages 581–584, 1993.

[21] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proceedings of XIII International Conference on Machine Learning*, pages 148–156, 1996.

[22] H. Frohlich and O. Champelle. Feature selection for support vector machines by means of genetic algorithms. In *Proceedings of IEEE International Conference on Tools with AI*, pages 142–148, 2003.

[23] B. Gabrys and D. Ruta. Genetic algorithms in classifier fusion. *Applied Soft Computing*, 6(4):337–347, 2006.

[24] G. Giacinto and F. Roli. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition*, 33:1879–1881, 2001.

[25] P.J. Gother. *NIST Special Database 19 - Handprited forms and characters database*. National Institute of Standard and Technology - NIST : database CD documentation, 1995.

[26] D. Greene, A. Tsymbal, N. Bolshakova, and P. Cunningham. Ensemble clustering in medical diagnostics. In *Proceedings of the IEEE Symposium on Computer-Based Medical Systems*, pages 570–575, Bethesda, USA, 2004.

[27] A.J. Grove and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. In *Proceedings of 15th National Conference on Artificial Intelligence*, pages 692–699, 1998.

[28] V. Gunes, M. Menard, and P. Loonis. A multiple classifier system using ambiguity rejection for clustering-classification cooperation. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 8(6):747–762, 2000.

[29] L.K. Hansen, C. Liisberg, and P. Salomon. The error-reject tradeoff. *Open Systems & Information Dynamics*, 4(2):159–184, 1997.

[30] T. K. Ho. Multiple classifier combination: Lessons and next steps. In A. Kandel and H. Bunke, editors, *Hybrid Methods in Pattern Recognition*, pages 171–198. World Scientific, 2002.

[31] T.K. Ho. Nearest neighbors in random subspaces. In *Proceedings of the Second International Workshop on Statistical Techniques in Pattern Recognition*, pages 640–648, Sydney, Australia, 1998.

[32] T.K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[33] Y.S. Huang and C. Suen. Combination of multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):90–94, 1995.

[34] J. Kennedy and R.C. Eberhart. Particle swarm intelligence. In *Proceedings of International Conference on Neural Network*, pages 1942–1948, 1995.

[35] J. Kittler, M. Hatef, RPW. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–238, 1998.

[36] A. Ko, R. Sabourin, and A. Britto Jr. A new dynamic ensemble selection method for numeral recognition. In *Proceedings of the 7th International Workshop on Multiple Classifier Systems*, pages 302–311, Prague-Czech Republic, 2007.

[37] R. Kohavi and D. Sommerfield. Feature subset selection using the wrapper model: Overfitting and dynamic search space topology. In *Proceedings of The First International Conference on Knowledge Discovery and Data Mining*, pages 192–197, 1995.

[38] L.I. Kuncheva. Cluster-and-selection model for classifier combination. In *Proceedings of International Conference on Knowledge Based Intelligent Engineering Systems and Allied Technologies*, pages 185–188, 2000.

[39] L.I. Kuncheva. Switching between selection and fusion in combining classifiers: An experiment. *IEEE Transactions on Systems, Man, and Cybernetics*, 32(2):146–156, 2002.

[40] L.I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Jhon Wiley & Sons, LTD, USA, 2004.

[41] L.I. Kuncheva, J. Bezdek, and R. Duin. Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recognition*, 34(2):299–314, 2001.

[42] L.I. Kuncheva, M.Skurichina, and R.P.W. Duin. An experimental study on diversity for bagging and boosting with linear classifiers. *Information Fusion*, 3(4):245–258, 2002.

[43] L.I. Kuncheva and J. Rodriguez. Classifier ensembles with a random linear oracle. *IEEE Transactions on Knowledge and Data Engineering*, 19(4):500–508, 2007.

[44] L.I. Kuncheva and C.J. Whitaker. Measures of diversity in classifier ensembles an their relationship with the ensemble accuracy. *Machine Learning*, 51:181–207, 2002.

[45] R. Liu and B. Yuan. Multiple classifiers combination by clustering and selection. *Information Fusion*, 2(4):163–168, 2001.

[46] X. Llorà, D.E. Goldberg, I. Traus, and E. Bernadó i Mansilla. Accuracy, parsimony, and generality in evolutionary learning systems via multiobjective selection. In *International Workshop in Learning Classifier Systems*, pages 118–142, 2002.

[47] J. Loughrey and P. Cunningham. Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets. In *Proceedings of International Conference on Innovative Techniques and Applications of AI*, pages 33–43, 2004.

[48] D.D. Margineantu and T.G. Dietterich. Pruning adaptive boosting. In *Proceedings of the International Conference on Machine Learning*, pages 358–366, 1997.

[49] T. M. Mitchell. *Machine Learning*. McGraw-Hill, USA, 1997.

[50] N. Narasimhamurthy. Evaluation of diversity measures for binary classifier ensembles. In *Proceedings of MCS*, pages 267–277, Seaside, USA, 2005.

[51] L.S. Oliveira, M. Morita, R. Sabourin, and F. Bortolozi. Multi-objective genetic algorithms to create ensemble of classifiers. In *Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization*, pages 592–606, 2005.

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

[53] D. W. Opitz and J. W. Shavlik. Actively searching for an effective neural-network ensemble. *Connection Science*, 8(3/4):337–353, 1996.

[54] N.C. Oza and K. Tumer. Classifier ensembles: select real-world applications. *Information Fusion*, 9(1):4–20, 2008.

[55] D. Parikh and R. Polikar. An ensemble-based incremental learning approach to data fusion. *IEEE Transactions on Systems, Man, and Cybernetics-PartB*, 37(2):500–508, 2007.

[56] Y. Park and J. Sklansky. Automated design of linear tree classifiers. *Pattern Recognition*, 23(12):1393–1412, 1990.

[57] K.E. Parsopoulos and M.N. Vrahatis. Recent approaches to global optimization problems through particle swarm optimization. *Natural Computing*, 1(2):235–306, 2002.

[58] D. Partridge and W.B. Yates. Engineering multiversion neural-net systems. *Neural Computation*, 8(4):869–893, 1996.

[59] M. Prior and T. Windeatt. Over-fitting in ensembles of neural networks classifiers within ecoc frameworks. In *Proceedings of International Workshop on Multiple Classifier System*, pages 286–295, Seaside, USA, 2005.

[60] J.R. Quinlan. *Programs for Machine Learning*. Morgan Kaufmann, 1993.

[61] P.V.W. Radtke. *Classification Systems Optimization with Multi-Objective Evolutionary Algorithms*. PhD thesis, Ecole de Technologie Superieure - Universite du Quebec, 2006.

[62] P.V.W. Radtke, R. Sabourin, and T. Wong. Impact of solution over-fit on evolutionary multi-objective optimization for classification systems. In *Proceedings of IJCNN*, Vancouver, Canada - accepted, 2006.

[63] G. Ratsch, T. Onoda, and K-R. Muller. An improvement of adaboost to avoid overfitting. In *Proceedings of The Fifth International Conference on Neural Information Processing*, pages 506–509, Kitakyushu, Japan, 1998.

[64] R. Reed. Pruning algorithms - a survey. *IEEE Transactions on Neural Networks*, 4(5):740–747, 1993.

[65] J. Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382, 2003.

[66] F. Rheaume, A-L. Jousselme, D. Grenier, E. Bosse, and P. ValinThomas. New initial basic probability assignments for multiple classifiers. In *Proceedings of XI Signal Processing, Sensor Fusion, and Target Recognition*, pages 319–328, 2002.

[67] D. Robilliard and C. Fonlupt. Backwarding: An overfitting control for genetic programming in a remote sensing application. In *Proceedings of the 5th European Conference on Artificial Evolution*, pages 245–254, 2000.

[68] G. Rogova. Combining the results of several neural network classifiers. *Neural Networks*, 7(5):777–781, 1994.

[69] F. Roli, G. Giacinto, and G. Vernazza. Methods for designing multiple classifier systems. In *Proceedings of the International Workshop on Multiple Classifier System*, pages 78–87, 2001.

[70] D. Ruta and B. Gabrys. Classifier selection for majority voting. *Information Fusion*, 6(1):163–168, 2005.

[71] D. Ruta and B. Gabrys. Neural network ensembles for time series prediction. In *Proceedings of the IEEE International Joint Conference on Neural Network*, pages 1204–1209, Orlando, USA, 2007.

[72] E.M. Dos Santos, R. Sabourin, and P. Maupin. Single and multi-objective genetic algorithms for the selection of ensemble of classifiers. In *Proceedings of International Joint Conference on Neural Networks*, pages 5377–5384, 2006.

[73] E.M. Dos Santos, R. Sabourin, and P. Maupin. Ambiguity-guided dynamic selection of ensemble of classifiers. In *Proceedings of International Conference on Information Fusion*, Quebec, CA, 2007.

[74] E.M. Dos Santos, R. Sabourin, and P. Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Submitted to Pattern Recognition*, 2007.

[75] E.M. Dos Santos, R. Sabourin, and P. Maupin. Overfitting cautious selection of classifier ensembles with genetic algorithm. *Submitted to Information Fusion*, 2007.

[76] S.B. Serpico, L. Bruzzone, and F. Roli. An experimental comparison of neural and statistical non-parametric algorithms for supervised classification of remote-sensing images. *Pattern Recognition Letters*, 17(13):1331–1341, 1996.

[77] A.J.C. Sharkey. Multi-net systems. In *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*, pages 1–27. Spring-Verlag, 1999.

[78] A.J.C. Sharkey and N.E Sharkey. The "test and select" approach to ensemble combination. In J. Kittler and F. Roli, editors, *Multiple Classifier System*, pages 30–44. Spring-Verlag, 2000.

[79] H.W. Shin and S.Y. Sohn. Combining both ensemble and dynamic classifier selection schemes for prediction of mobile internet subscribers. *Expert Systems with Applications*, 25:63–68, 2003.

[80] H.W. Shin and S.Y. Sohn. Selected tree classifier combination based on both accuracy and error diversity. *Pattern Recognition*, 38(2):191–197, 2005.

[81] S. Singh and M. Singh. A dynamic classifier selection and combination approach to image region labelling. *Signal Processing: Image Communication*, 20:219–231, 2005.

[82] K. Sirlantzis, M.C. Fairhurst, and R.M. Guest. An evolutionary algorithm for classifier and combination rule selection in multiple classifier system. In *Proceedings of ICPR*, pages 771–774, 2002.

[83] P. C. Smits. Multiple classifier systems for supervised remote sensing image classification based on dynamic classifier selection. *IEEE Transactions on Geoscience and Remote Sensing*, 40(4):801–813, 2002.

[84] R.G.F. Soares, A. Santana, A.M.P. Canuto, and M.C.P. de Souto. Using accuracy and diversity to select classifiers to build ensembles. In *Proceedings of International Joint Conference on Neural Networks*, pages 2289–2295, 2006.

[85] S.Y. Sohn and H.W. Shin. Experimental study for the comparison of classifier combination methods. *Pattern Recognition*, 40(1):33–40, 2007.

[86] C.Y. Suen and Louisa Lam. Multiple classifier combination methodologies for different output levels. In *First International Workshop on Multiple Classifier Systems*, pages 52–66, London, UK, 2000.

[87] T. Suttorp and C. Igel. Multi-objective optimization of support vector machines. *Studies in Computational Intelligence - Multi-objective Machine Learning*, 16:199–220, 2006.

[88] G. Tremblay. Optimisation d'ensembles de classifieurs non paramétriques avec aprentissage par représentation partielle de l'information. Master's thesis, École de technologie supérieure, 2004.

[89] G. Tremblay, R. Sabourin, and P. Maupin. Optimizing nearest neighbour in random subspaces using a multi-objective genetic algorithm. In *Proceedings of International Conference Pattern Recognition*, pages 208–211, Cambridge, UK, 2004.

[90] A. Tsymbal, M. Pechenizkiy, and P. Cunningham. Diversity in search strategies for ensemble feature selection. *Information Fusion*, 6(1):83–98, 2005.

[91] A. Tsymbal, M. Pechenizkiy, and P. Cunningham. Sequential genetic search for ensemble feature selection. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 877–882, 2005.

[92] A. Tsymbal, M. Pechenizkiy, P. Cunningham, and S. Puuronen. Dynamic integration of classifiers for handling concept drift. *Information Fusion*, 9(1):56–68, 2008.

[93] G. Valentini. *Ensemble methods based on bias-variance analysis.* PhD thesis, Genova University, 2003.

[94] G. Valentini and T. Dietterich. Low bias bagged support vector machines. In *Proc. International Conf. on Machine Learning*, pages 776–783, 2003.

[95] G. Valentini and T. G. Dietterich. Bias-variance analysis and ensembles of svm. In F. Roli and J. Kittler, editors, *Multiple Classifier System*, pages 222–231. Spring-Verlag, 2002.

[96] X. Wang and H. Wang. Classification by evolutionary ensembles. *Pattern Recognition*, 39(4):595–607, 2006.

[97] C. Wenzel, S. Baumann, and T. Jager. Advances in document classification by voting of competitive approaches. In J.J. Hull and S. Lebowitz, editors, *Document Analysis Systems II*, pages 385–405. World Scientific, 1998.

[98] K-D. Wernecke. A coupling procedure for the discrimination of mixed data. *Biometrics*, 48(2):497–506, 1992.

[99] C.J. Whitaker and L.I. Kuncheva. Examining the relationship between majority vote accuracy and diversity in bagging and boosting. Technical report, School of Informatics, Unversity of Wales, 2003.

[100] S. Wiegand, C. Igel, and U. Handmann. Evolutionary multi-objective optimization of neural networks for face detection. *International Journal of Computational Intelligence and Applications*, 4(3):237–253, 2004.

[101] K. Woods. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):405–410, 1997.

[102] H. Wu and J.L. Shapiro. Does overfitting affect performance in estimation of distribution algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 433–434, Seattle, USA, 2006.

[103] J. Wu and S. Azarm. Metrics for quality assessment of a multiobjective design optimization solution set. *Transactions ASME, Journal of Mechanical Design*, 123:18–25, 2001.

[104] G. Zenobi and P. Cunningham. Using diversity in preparing ensembles of classifiers based on differente feature subsets to minimize generalization error. In *Proceedings of XII European Conference on Machine Learning*, pages 576–587, Freiburg, Germany, 2001.

[105] Z. Zhang. Mining relational data from text: From strictly supervised to weakly supervised learning. *Information Systems, In Press*, 2008.

[106] Z-H. Zhou, J. Wu, and W. Tang. Ensembling neural networks: Many could be better than all. *Artificial Intelligence*, 137(1-2):239–263, 2002.

[107] X. Zhu, X. Wu, and Y. Yang. Dynamic selection for effective mining from noisy data streams. In *Proceedings of Fourth IEEE International Conference on Data Mining*, pages 305–312, 2004.

[108] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.