Classifier Ensembles Optimization Guided by Population Oracle

Eulanda M. dos Santos
Federal University of Amazonas
Manaus, Brazil
Email: emsantos@dcc.ufam.edu.br

Robert Sabourin
Ecole de Technologie Superieure
Montreal, Canada
Email: Robert.Sabourin@etsmtl.ca

***Abstract*—Dynamic classifier ensemble selection is focused on selecting the most confident classifier ensemble to predict the class of a particular test pattern. The overproduce-and- choose strategy is a dynamic classifier ensemble selection method which is divided into optimization and dynamic selection phases. The first phase involves the test of different candidate ensembles in order to produce a population composed of the highest performing candidate ensembles. Then, the second phase calculates the domain of expertise of each candidate ensemble to pick up the solution with highest degree of certainty of its decision to classify the unknown test samples. It has been shown that the optimization phase decreases oracle, the upper bound of dynamic selection processes. In this paper we propose a hybrid algorithm to perform the optimization phase of overproduce-and-choose strategy. The proposed algorithm combines stochastic initialization of candidate ensembles of different sizes, with the traditional forward search greedy method. The objective is to apply oracle as search criterion during the optimization phase. We show experimentally that choosing the population of classifier ensembles taking into account the population oracle leads to increase the upper bound of the dynamic selection phase. Moreover, experimental results conducted to compare the proposed method to a multi-objective genetic algorithm (MOGA), demonstrate that our method outperforms MOGA on generating population of candidate ensembles with higher oracle rates.**

***Index Terms*—Dynamic classifier ensemble selection; hybrid search algorithm; multi-objective genetic algorithm; pattern recognition.**

I. Introduction

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. Given sample x and its class label w_k with an unknown function $w_k = f(x)$, learning algorithms focus on finding in the hypothesis space H the best approximation function h, which is a classifier found during a training process. The obtained classifier is then employed to predict the classes of samples which were not used for training. It is expected that the underlying data generating mechanism, or the concept that was learned from the training data does not evolve over time.

However, it has been shown that the learning context (target environment) changes over time in many real-world problems, such as remote sensing [1], commercial applications based on human behavior [2], detecting and filtering out spam e-mail [3], etc. Noise, missing features and illumination failures may lead to changes in non-stationary environment. Another source of changes is related to classes definition, for instance class priors, classes distributions, adding or removing classes, and data distribution. Changes are usually referred to drifts or *concept drift* [4] in Machine Learning.

Pattern Recognition research has recently focused on searching for learning algorithms able to deal with changing environment. Although different methods have been proposed in the literature, there are two general strategies for dealing with changing environment [3]: (1) incremental learning; and (2) classifier ensembles. The first, usually used in Data Mining and commercial applications like credit transactions, is devoted to handle huge databases (*streams data*). In these learning scenarios, the difficulty to store the data increases as the databases become too large. Moreover, in many applications, the data become available in batches over time. Therefore, large databases often lead to concept drift. It is interesting to note that incremental learning algorithms are often proposed to static environment and further extended to cope with concept drift. For instance, Domingos and Hulten proposed The Very Fast Decision Tree (VFDT) for stationary environments [5]. Thus, in [6] they adapted their method to deal with non-stationary environment.

The second strategy has become a dominant approach in Pattern Recognition since it relies on the assumption that combining the decisions of simpler classifiers may improve traditional individual ones. According to Kuncheva [3], the application of classifier ensembles to changing environment problems may be undertaken by means of different strategies, for instance updating the training data, adding and removing classifier members, re-evaluating and/or re-training the classifiers and dynamic classifier selection.

Classifier selection strategies are called dynamic whether the choice of the best classifier or ensemble of classifiers is performed during the classification phase, taking into account the characteristics of the sample to be classified. Classical dynamic classifier selection methods, for instance dynamic classifier selection with local accuracy (DCS-LA) [7], assume that ensemble members are experts in regions of competence. These methods work as follows. One ensemble creation method such as bagging [8] and random subspace [9], is employed using a training dataset to obtain classifiers to

compose an initial pool of classifiers $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$. An independent validation dataset is then used to produce regions of competence (partitions). Finally, a winning partition is selected and a winning classifier over samples contained in the chosen partition, is used to assign the label to the test samples. Indeed, dynamic classifier selection appears to be interesting for changing environment problems because the selection is conducted based on the unknown data.

However, since only one candidate classifier is picked up to make the decision, this may lead to the selection of a wrong classifier. Dynamic ensemble selection has been investigated in the literature as a strategy for avoiding such a drawback, due to the fact that the selection of a classifier ensemble rather than only one classifier may decrease misclassification. The k-Nearest-Oracles [10], Dynamic Voting with Selection [11], and the Dynamic Overproduce-and-Choose Strategy (DOCS) [12] are examples of approaches for dynamic selection of classifier ensembles.

The objective of DOCS is to find the most relevant subsets of classifiers, based on the assumption that classifiers in $\mathcal{C}$ are redundant. The method proposed in [12] is initially divided into two phases: (1) overproduction; and (2) selection. In the first phase, the initial pool of classifiers $\mathcal{C}$ is generated, as defined for classical dynamic classifier selection. The second phase is divided into two levels: (1) optimization; and (2) dynamic selection. The former applies a search algorithm to test different combinations of the initial classifiers in order to generate a population $\mathbf{C}^*$ of ensembles. The latter is devoted to identify dynamically the best ensemble C_j^* over the population $\mathbf{C}^*$, to classify each test sample individually.

A non-exhaustive search algorithms might be used during the optimization level 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. $\mathcal{P}(\mathcal{C})$ being the powerset of $\mathcal{C}$ defining the population of all possible candidate ensembles. Several non-exhaustive search algorithms may be applied. Multi-objective Genetic Algorithms (MOGA) appears to be interesting because it has two important characteristics. First, it allows the possibility of dealing with a population of classifier ensembles (Pareto front) rather than one individual best candidate. This important property enables the dynamic selection level of DOCS being conducted. Second, it allows the fairly easy implementation of ensemble classifier selection tasks as optimization processes using multi-objective functions.

In [12], the optimization level was conducted using MOGA while the selection level was based on the certainty of the candidate ensemble's decision. The results obtained by DOCS were compared to static selection and fusion. In static selection the best performing classifier ensemble is chosen at the optimization phase and used to classify the whole dataset. In fusion, all individual classifiers in $\mathcal{C}$ are combined through majority voting, i.e. there is no selection. Even though the results showed that DOCS outperformed both static selection and fusion, it was also showed that the search algorithm did not help generating a population of candidate ensembles $\mathbf{C}^*$ with high oracle rates. The so-called oracle is an upper bound of dynamic selection strategies. It correctly classifies the test sample if any of candidate ensembles in $\mathbf{C}^*$ predicts the correct label for the sample. According to the authors, the optimization level lead to a major loss of oracle power since it was found a huge difference between oracle obtained using the initial pool of classifiers $\mathcal{C}$ and those obtained using $\mathbf{C}^*$. They observed that by increasing the oracle rate of the population $\mathbf{C}^*$ one may improve the classification rate of DOCS.

In this paper, we propose a hybrid optimization algorithm, which combines stochastic initialization of candidate ensembles of different sizes, with the traditional forward search greedy method, to generate populations $\mathbf{C}$ of classifier ensembles. Then, an auxiliary archive is used to keep stored the population of candidate ensembles which presents the highest oracle rate measured on a validation dataset. The objective is to increase the oracle rates of the population of solutions found at the optimization phase and to verify whether or not the improvement on oracle leads to improvement on classification.

Experiments are conducted using handwritten digits and handwritten uppercase letters databases. Although these databases do not present concept drift, handwritten digits and letters recognition may be prone to drifts in practical applications. The initial pools of classifiers are created using the random subspace method while decision trees and k nearest neighbors (kNN) are the base classifiers. The margin-based dynamic selection strategy also proposed in [12] is used at the dynamic selection level. Moreover, the oracle rates as well as the classification rates are compared to the results obtained using MOGA, NSGA-II (elitist non-dominated sorting GA) [13], at the optimization level.

The experiments show that our method indeed increases oracle rate of the population of candidate ensembles. In addition, our results reveal important information about the relationship between oracle rate and performance, since the classification rates attained at the dynamic selection level were worse than the results obtained using MOGA.

This paper is organized as follows. Section II presents the definition of oracle in order to clarify how it is used during our optimization process. The proposed method is then described in section III. The parameters employed in the experiments and the results obtained are presented in section IV. Conclusions and suggestions for future work are discussed in section V.

II. Oracle Definitions

Given a pool of classifiers $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ generated using any ensemble construction strategy and base classifier model. The classifier members are trained using a dataset $\mathcal{T}$. Oracle is classically defined as a strategy that correctly classifies each test sample $x_{i,g}$, from a test dataset $\mathcal{G}$, if any of the c_i classifier members of $\mathcal{C}$ predicts the correct label for the sample. In this way, it is assumed that oracle always indicates the correct classifier for assigning the class of each test sample.

However, since there is no guarantee that a particular classifier selection method will achieve such a performance,

oracle has been used as an upper bound of selection strategies. Traditionally, the effectiveness of selection mechanisms reported in the literature is evaluated by comparing their results to oracle performance [14]. Oracle may also be an upper bound for ensemble selection strategies. In this case, it is assumed that oracle correctly classifies the test sample if any of candidate ensembles in $\mathbf{C}^*$ predicts the correct label for the test sample $x_{i,g}$.

It was mentioned in the introduction that the selection mechanism performed in DOCS is divided into optimization and dynamic selection levels. In [12], it is shown through a case study that the optimization phase appears to play an important role on decreasing the oracle rate of DOCS, i.e. its upper bound. The case study was obtained on NIST-letters database. Details about NIST-letters database are presented in section IV. One initial pool of 100 decision tree classifiers was generated using the random subspace method at the overproduction phase. The search algorithm used at the optimization level was NSGA-II guided by the following pair of objective functions: jointly minimize the difficulty diversity measure [15] and the error rate .

Table I summarizes the results achieved in this case study. The error rate obtained by combining the initial pool of 100 classifiers are shown in this table as well as its oracle rate. In fusion, individual classifiers in $\mathcal{C}$ are combined through majority voting (Equation 1), i.e. there is no selection. Table I also shows the error rate achieved by DOCS and its oracle rate. The results from DOCS were computed using only one replication of NSGA-II.

TABLE I
SUMMARY OF THE COMPARISON AMONG ORACLE AND ERROR RATE RESULTS OBTAINED WHEN COMBINING ALL CLASSIFIERS IN THE INITIAL POOL $\mathcal{C}$ AND OVER THE POPULATION $\mathbf{C}^*$ FOUND BY NSGA-II DURING THE OPTIMIZATION LEVEL OF DOCS.

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

In this example it is possible to note that DOCS may not achieve an error rate lower than oracle rate, i.e. 4.81. The huge difference between oracle results of $\mathcal{C}$ and $\mathbf{C}^*$ shows the loss of oracle power at the optimization level. These results could indicate that performances attained by DOCS may be increased as the oracle error rate of the population of candidate ensembles $\mathbf{C}^*$ becomes closer to that obtained using $\mathcal{C}$. Even though, oracle was not used as a selection criterion during the optimization phase in [12].

However, the properties of the oracle concept have been used to guide dynamic ensemble selection in other works. In [10], the k-nearest-oracles selects the set of k nearest neighbors from a validation dataset surrounding the test sample. Then, it picks up each classifier, that correctly classifies this set of neighbors, to compose a classifier ensemble. The selected classifier ensemble is then used to label the test sample. The hybrid optimization algorithm proposed in this paper explores the oracle concept in the context of DOCS. This algorithm is intended to improve the oracle rate of the generated population of candidate ensembles. In this way, it will allow us to verify whether or not the quality of the ensemble population, in terms of oracle rates, leads to DOCS performance improvement. The proposed algorithm is described in next section.

III. THE PROPOSED HYBRID OPTIMIZATION ALGORITHM

Our proposed algorithm applies greedy forward search in the context of DOCS. These two aspects are analyzed in this section before algorithm description.

A. Forward Search

Greedy search algorithms, such as backward and forward searches, have been successfully applied in feature subset selection [16] as well as in static classifier ensemble selection [17]. In static selection the best performing classifier ensemble is chosen at the optimization level and used to classify the whole test dataset.

Given $\mathcal{C}$ generated using an ensemble creation method applied over samples contained in the training dataset $\mathcal{T}$, greedy search algorithms calculate an objective function using an optimization dataset $\mathcal{O}$, in order to identify the best subset of classifiers C_j^*. The most common objective function is the minimization of majority voting error. This rule is computed as in Equation 1, given a subset of l classifiers, y_i as the class label output of the i-th classifier, and a classification problem with the following set of class labels $\Omega = \{\omega_1, \omega_2 \ldots, \omega_c\}$.

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

Especially for forward search, the algorithm first selects the most accurate classifier c_i^* in $\mathcal{C}$. Thus, the pair of classifiers, including c_i^*, with lowest majority voting error is identified. Then, at each iteration, a new individual classifier is included into the ensemble. The optimization process is stopped when there is no more improvement on decreasing the majority voting error. In static selection, the classifier ensemble C_j^* found at the end of optimization is then used to classify the whole test dataset $\mathcal{G}$. However, as mentioned in the introduction there is no guarantee that C_j^* is the best solution for each test sample individually.

B. DOCS

In DOCS method, the optimization process generates a population of candidate ensembles $\mathbf{C}^*$, rather than an individual solution. This population is the input to the dynamic selection level, where C_j^* is chosen dynamically for each test sample based on the certainty of the candidate ensembles decision. The assumption is that high consensus among classifier members leads to high level of confidence in the decision. Three different measures of consensus among classifier ensemble members are discussed and compared in [12] so as to identify the best measure for the dynamic selection level. It was observed that all three measures presented results closely similar. Then, in this paper we apply the margin-based measure

at the at the dynamic selection level of DOCS. This measure is described in equation 2.

Let $\mathbf{x}_{i,g}$ be the test samples in the test dataset $\mathcal{G}$ and $v(mv|\mathbf{x}_{i,g})$ the number of votes assigned to the majority voting class, the margin of sample $\mathbf{x}_{i,g}$ for each C_j from the population $\mathbf{C}^*$ may be calculated as:

$$\mu(\mathbf{x}_{i,g}) = \frac{v(mv|\mathbf{x}_{i,g}) - \max_{k \neq mv} v(\omega_k|\mathbf{x}_{i,g})}{|C_j|} \quad (2)$$

where $|C_j|$ is the size of C_j.

C. The Algorithm

The focus of our search algorithm is to achieve the following three objectives: (1) to generate a population of candidate classifier ensembles $\mathbf{C}^*$ for the dynamic level of DOCS; (2) to provide a population of ensembles which presents high oracle rates since the higher the population oracle rate, the higher the possibility of right decision; and, finally, (3) to generate a population of high performance classifier ensembles. The last objective is related to the fact that it has been shown that by searching for minimizing the error rate of the ensembles, we may accomplish the main objective in Pattern Recognition, which is to find high-performance predictors.

As summarized in Algorithm 1 and Figure 1, our method first divides the optimization problem into local forward searches through the creation of a set of slots $S = \{S_1, S_2, \ldots, S_z\}$, where z is the maximum number of slots. The size of each slot must be fixed *a priori* and determines the size of a classifier ensemble. Thus, at each generation g, and during a fixed number of generations $max(g)$, a classifier ensemble is found for each slot using local forward search as follows. For each slot, an initial classifier c_i^* is randomly chosen. Then, the pair of classifiers, including c_i^*, with lowest majority voting error is identified. Further, a third classifier is included into the ensemble. The slot/ensemble size is the stopping point for such a local optimization process.

The solutions found for each slot are used to compose the population of classifier ensembles $\mathbf{C}(g)$ at generation g. Finally, the oracle rate of $\mathbf{C}(g)$ is monitored on a validation dataset $\mathcal{V}$ in order to keep stored in an auxiliary archive $\mathcal{A}$ the population $\mathbf{C}^*$ with highest oracle. Therefore, $\mathcal{A}$ is updated only when $\mathbf{C}(g)$ presents higher oracle rate compared to the population previously stored in $\mathcal{A}$.

It is important to observe that, even though forward search is not a stochastic method, we indeed generate different classifier ensembles at each generation due to the random choice of the initial individual classifier to compose the ensemble for each slot. Besides, the objective is to keep on the auxiliary archive a population of solutions with high oracle rate and to select each ensemble member as the most accurate solution on each slot due to majority voting as search criterion. In this way, the population stored in $\mathcal{A}$ may be used as input to the dynamic selection level of DOCS.

In next section we present experimental results employing the proposed hybrid search algorithm at the optimization level of DOCS. We also compare the obtained results to those achieved using NSGA-II.

Algorithm 1 Hybrid Search Algorithm for optimization level of DOCS. (OR=oracle)

```
Set the size of classifier ensembles at each slot S_y
A = ∅
C = ∅
for each generation g ∈ {1, ..., max(g)} do
    for each slot S_y ∈ {S_1, S_2, ..., S_z} do
        Randomly choose an initial classifier c_1^*
        set C_j^*(y) = {c_1^*}
        for i = 2 : i = |S_y| do
            Find c_i^* which included into C_j^*(y) produces the lowest mv error rate
            update C_j^*(y) by setting C_j^*(y) = {C_j^*(y), c_i^*}
        end for
        update the population C(g) by setting C(g) = {C(g), C_j^*(y)}
        if first generation then
            Set C^* := C(g)
            Store C^* in A
        else
            if OR(V, C(g)) > OR(V, C^*) then
                set C^* := C(g)
                update A by storing in it the new C^*
            end if
        end if
    end for
end for
return C^* stored on A
```

IV. Experiments

Experiments have been carried out to verify whether or not the proposed algorithm helps finding a population of classifier ensembles with high oracle rates. We also observe the impact on overall recognition rate of DOCS when increasing oracle rates of the population of candidate ensembles. The obtained results are compared to the results achieved by NSGA-II used as search algorithm guided by the following pair of objective functions: jointly minimize the difficulty diversity measure and the majority voting error rate.

Let C_j be the candidate ensemble of classifiers, l its cardinality, 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, difficulty diversity measure may be calculated as:

$$\theta = Var(F) \quad (3)$$

Two databases were used in our experiments: (1) NIST digits Special Database 19 (NIST SD19), called NIST-digits; and (2) NIST SD19 containing handwritten uppercase letters, called NIST-letters here. There are two test sets from NIST-digits, 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. The original datasets were partitioned into four independent datasets: $\mathcal{T}$, $\mathcal{O}$, $\mathcal{V}$ and $\mathcal{G}$, using the classical holdout validation strategy. This is due to the fact that DOCS requires at least four datasets. We employ

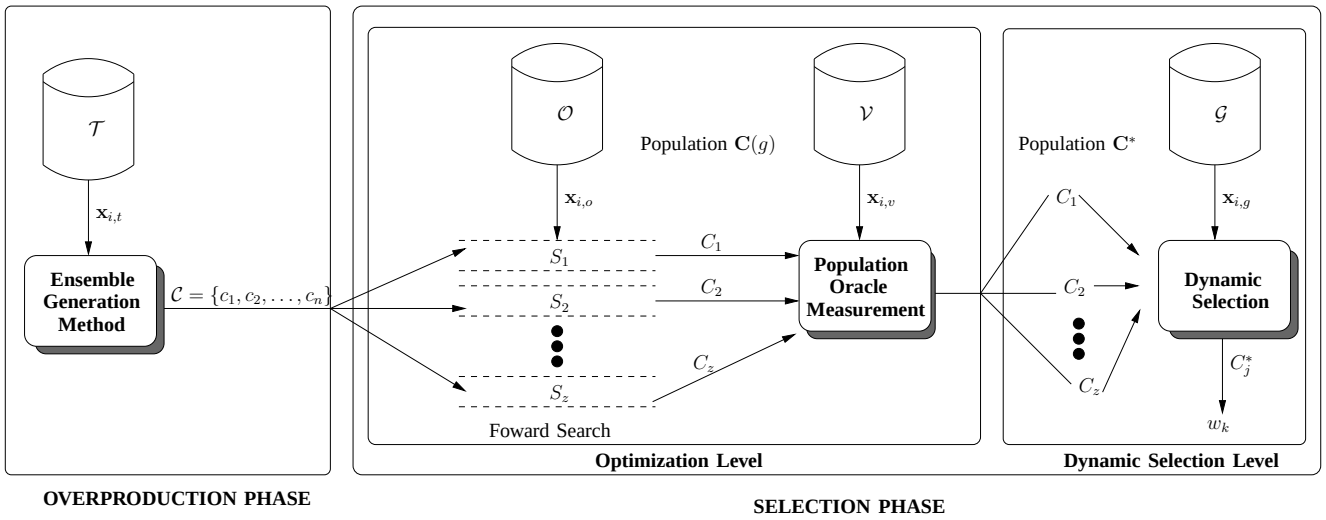

Fig. 1. Overview of the DOCS process highlighting the optimization level guided by our hybrid algorithm. The optimization is divided into local forward searches conducted over a set of slots $S = \{S_1, S_2, \ldots, S_z\}$. Each classifier ensemble C_j found for each slot is used to compose the population of classifier ensembles $\mathbf{C}(g)$ at generation g. Then, the oracle rate of $\mathbf{C}(g)$ is calculated on the validation dataset. The final population $\mathbf{C}^*$ used for the dynamic selection level, is the one with highest oracle.

TABLE II
SPECIFICATIONS OF THE DATASETS USED IN THE EXPERIMENTS.

Dataset	Number of features	Training Set ($\mathcal{T}$)	Optimization Set ($\mathcal{O}$)	Validation Set ($\mathcal{V}$)	Test Set ($\mathcal{G}$)	Number of Features Random Subspace	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

the representation proposed by Oliveira et al. [18], which is composed of 132 features. Table II lists important information about the databases and the partitions used to compose the four separate sets.

We chose kNN and decision trees as base classifiers. The C4.5 algorithm (Release 8) was used to construct the trees with pruning. In addition, we used k=1 for kNN classifiers. The random subspace method was used to generate the initial pools of 100 homogeneous classifiers. The size of the subsets of features used by random subspace is shown in Table II.

In our experiments, the proposed algorithm is employed to generate populations of candidate classifier ensembles in order to be the input to the dynamic selection level of DOCS. The majority voting error rate was applied individually as single-objective function during the local forward search, for each slot. Moreover, we set the number of slots $z = 13$ since it is closely related to the number of solutions over the Pareto front found by NSGA-II in [12]. When $z = 1$ the slot size was set as 9. Then, for $z = 13$ the slot size was set as 57, i.e. each slot size was increased with 4 classifiers. Finally, the maximum number of generations was 100, due to the size of the initial pool of classifiers. Table III shows the parameter settings employed in our experiments.

The optimization level applied using NSGA-II is based on binary vectors. We defined the same genetic parameters employed in [12] (Table IV).

TABLE III
PROPOSED SEARCH ALGORITHM PARAMETERS

Classifier ensemble population size	13
Number of generations	100
Sizes of slots	9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57.

TABLE IV
NSGA-II 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	

Table V shows the results comparing the proposed algorithm to NSGA-II in terms of oracle rates of the population of kNN ensembles . We also report the recognition rate achieved by DOCS comparing the two search algorithms. Table VI summarizes the same comparison for ensemble of decision trees. These results indicate that our hybrid search algorithm reduces the loss of oracle power at the optimization level when compared to NSGA-II. Therefore, the upper bound of DOCS is increased. However, such an improvement on upper bound did not bring about an improvement in the performance attained by DOCS. Our results show that the recognition rate

achieved by DOCS decreased using the population of classifier ensembles found by our search algorithm compared to the populations generated by NSGA-II. What this means, in effect, is that performing the optimization level of DOCS taking into account the population oracle, can be effective to increase the upper bound of the selection level. Even though, this does not guarantee that the candidate ensemble most likely to be correct for classifying each test sample individually will be selected at the dynamic selection level.

TABLE V
COMPARISON OF ORACLE AND ERROR RATE VALUES CALCULATED USING ENSEMBLE OF KNN CLASSIFIERS

	Ensemble of kNN			
Database	**Oracle**		**Error**	
	Proposed Algorithm	NSGA-II	Proposed Algorithm	NSGA-II
NIST-digits (data-test1)	2.00	2.91	3.72	3.58
NIST-digits (data-test2)	4.55	6.59	8.14	7.97
NIST-letters	3.32	5.18	6.68	6.27

TABLE VI
COMPARISON OF ORACLE AND ERROR RATE VALUES CALCULATED USING ENSEMBLE OF DECISION TREES

	Ensemble of Decision Trees			
Database	**Oracle**		**Error**	
	Proposed Algorithm	NSGA-II	Proposed Algorithm	NSGA-II
NIST-digits (data-test1)	1.25	2.17	3.18	2.77
NIST-digits (data-test2)	3.12	5.21	7.09	6.50
NIST-letters	2.89	5.04	6.19	5.84

We can point out the following reasons for these results:

- The confidence measure used at the dynamic selection level may not be the best strategy for choosing the best classifier ensemble for each test sample.
- Oracle is not independent of population size. It is expected that oracle rate from the initial pool of classifiers $\mathcal{C}$, which has 100 members, will be higher than the oracle rate of population of ensembles $\mathbf{C}^*$, which has 13 classifier ensembles on average.
- Oracle is too optimistic. According to Didaci et al. [14], oracle can correctly classify even patterns which would be wrongly classified according to the Bayesian decision theory. The authors advocate that Bayesian classifier is the true upper bound for dynamic selection approaches.

V. CONCLUSION

In this paper we have presented a hybrid search algorithm combining local forward search and stochastic initialization of individuals to select a population of classifier ensembles guided by oracle rates. Oracle is an upper bound of selection strategies. The objective was to reduce the loss of oracle rate at the optimization level of Dynamic Overproduce-and-Choose Strategy. The experiments demonstrated that our method outperforms NSGA-II on generating population of classifier ensembles with higher upper bound of selection. Even though, the improvement on population oracle does not help a confidence measure, which does not take into account the correctness of the output, to improve the overall classification rate of Dynamic Overproduce-and-Choose Strategy. For future work we plan to apply different selection approaches in order to achieve a classification rate closer to the oracle rate. We also plan to investigate different selection upper bound.

ACKNOWLEDGMENT

The authors would like to thank CNPq and FAPESP, Brazil, for supporting INCT-SEC, under the contracts 573963/2008-8 and 08/57870-9.

REFERENCES

[1] F. Bovolo, G. Camps-Valls, and L. Bruzzone, "A support vector domain method for change detection in multitemporal images," *Pattern Recognition Letters*, vol. 7, no. 5, pp. 777–781, 2009.
[2] D.J.Hand, "Classifier technology and the illusion of progress," *Statistical Science*, vol. 21, pp. 1–14, 2006.
[3] L. Kuncheva, "Classifier ensembles for changing environments," in *Proceedings of International Workshop on Multiple Classifier System*, Cagliari, Italy, 2004, pp. 1–15.
[4] L. Kuncheva and I. Zliobaite, "On the window size for classification in changing environments," *Intelligente Data Analysis*, vol. 13, pp. 314–323, 2009.
[5] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 71–80.
[6] G. Hulten, L. Spencer, and P. Domingos, "Mining time changing data streams," in *Proceedings of International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 97–106.
[7] K. Woods, "Combination of multiple classifiers using local accuracy estimates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 405–410, 1997.
[8] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
[9] T. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.
[10] A. Ko, R. Sabourin, and A. B. Jr., "From dynamic classifier selection to dynamic ensemble selection," *Pattern Recognition*, vol. 41, no. 5, pp. 1718–1731, 2008.
[11] A. Tsymbal, M. Pechenizkiy, P. Cunningham, and S. Puuronen, "Dynamic integration of classifiers for handling concept drift," *Information Fusion*, vol. 9, no. 1, pp. 56–68, 2008.
[12] E. D. Santos, R. Sabourin, and P. Maupin, "A dynamic overproduce-and-choose strategy for the selection of classifier ensembles," *Pattern Recognition*, vol. 41, pp. 2993–3009, 2008.
[13] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, LTD, 2001.
[14] L. Didaci, G. Giacinto, F. Roli, and G. Marcialis, "A study on the performances of the dynamic classifier selection based on local accuracy estimation," *Pattern Recognition*, vol. 28, pp. 2188–2191, 2005.
[15] L. Kuncheva and C. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.
[16] M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," *Pattern Recognition*, vol. 33, pp. 25–41, 2000.
[17] D. Ruta and B. Gabrys, "Classifier selection for majority voting," *Information Fusion*, vol. 6, no. 1, pp. 163–168, 2005.
[18] L. Oliveira, R. Sabourin, F. Bortolozzi, and C. Suen, "Automatic recognition of handwritten numerical strings: A recognition and verification strategy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1438–1454, 2002.