International Journal of Pattern Recognition and Artificial Intelligence


# Learning to Assemble Classifiers via Genetic Programming

Niusvel Acosta-Mendoza[1,2,*], Alicia Morales-Reyes[1,*], Hugo Jair Escalante[1], Andrés Gago-Alonso[2]

[1] *Instituto Nacional de Astrofisica, Optica y Electronica (INAOE), Luis Enrique Erro No. 1, Sta. Maria Tonantzintla, Puebla, CP 72840, Mexico*

[2] *Advanced Technologies Application Center (CENATAV), 7a No. 21406 e/ 214 and 216, Siboney, Playa, CP 12200, Havana, Cuba*
*nacosta@cenatav.co.cu, a.morales@inaoep.mx*

This article introduces a novel approach for building heterogeneous ensembles based on genetic programming (GP). Ensemble learning is a paradigm that aims at combining individual classifiers outputs to improve their performance. Commonly, classifiers outputs are combined by a weighted sum or a voting strategy. However, linear fusion functions may not effectively exploit individual models' redundancy and diversity. In this research, a GP based approach to learn fusion functions that combine classifiers outputs is proposed. Heterogeneous ensembles are aimed in this study, these models use individual classifiers which are based on different principles (e.g., decision trees and similarity-based techniques). A detailed empirical assessment is carried out to validate the effectiveness of the proposed approach. Results show that the proposed method is successful at building very effective classification models, outperforming alternative ensemble methodologies. The proposed ensemble technique is also applied to fuse homogeneous models' outputs with results also showing its effectiveness. Therefore, an in deep analysis from different perspectives of the proposed strategy to build ensembles is presented with a strong experimental support.

*Keywords*: Pattern classification; heterogeneous ensembles; genetic programming.

## 1. Introduction

Ensemble learning has been a widely investigated paradigm within computational intelligence and machine learning[6,26]. Ensembles ability has been demonstrated when applied to different machine learning challenges such as pattern classification[6], feature selection[21], and data clustering[22] among others. In pattern classification an ensemble consists on combining several classifiers in order to overcome individual drawbacks, such as low accuracy and high sensitivity to noisy data. Ensembles follow the idea of interaction among several prediction models to take advantage of individual performances and to avoid error propagation. However, combining several classifiers outputs does not guarantee that the best individual classifier is outperformed; although the probability of not selecting the worst individual classifier increases.

A rough ensembles classification divides them in homogeneous and heterogeneous. Homogeneous ensembles combine several instances of the same predictor under different parameters configurations or using different instances/features[11,4]. In contrast, heterogeneous ensembles merge the outputs of individual learners from different nature to build a composed classification model[23,27]. Both kinds of ensembles are normally built following a weighting scheme or a voting strategy which combine classifiers outputs to merge individual decision models. These combination strategies lead to linearly constrained models that possibly are not the best option for ensemble building.

Three main aspects are considered when building ensembles: the selection of training data for individual predictors, the process to obtain ensemble members, and the mechanism to combine individual classifiers[26]. This research focuses on exploring evolutionary computation as the mechanism to combine individual learners. In particular, Genetic Programming (GP) is used to learn a function that combines the predictions of individual classifiers. The underlying hypothesis is that voting or weighted-sum combination strategies may not fully exploit the redundancy and complementarity of individual classifiers. Learning a fusion strategy via GP leads to possibly non-linear combination mechanisms that could better exploit the outputs of multiple predictors. Thus, complex decision spaces can be targeted while exploiting implicit individual model's characteristics such as diversity[3]. The proposed technique focuses on heterogeneous ensembles, as it can automatically deal with different scales of classifiers' outputs. Nevertheless it can also be applied for combining predictions of homogeneous models. Experimental results in forty classification problems show the validity of the proposed method for building heterogeneous and homogeneous ensembles. In fact, ensembles generated with the proposed mechanism outperform traditional fusion techniques.

The proposed approach was first introduced by Escalante et al.[7], where important improvements were achieved by applying GP for ensembles construction on an object recognition data set. This study extends previous research by performing an extensive and comprehensive experimental assessment of the proposed method in a suite of benchmark pattern classification problems. Additionally, an in deep analysis of the solutions generated by the proposed method and its performance under different settings (including homogeneous ensembles and cross-domain ensemble learning) is reported.

This article is organized as follows. In Section 2 related work on ensemble learning with evolutionary algorithms is reported. Next, in Section 3 the approach for learning a fusion mechanism for ensemble generation is presented. In Section 4 experimental results that validate the efficacy of the proposed method are reported and analyzed. Finally, in Section 5 conclusions derived from this work and future work are drawn.

## 2. Related work

Evolutionary and other bio-inspired algorithmic techniques have been applied to classification in general, among them GP has been successfully used at different stages including pre-processing and post-processing tasks that aim at improving prediction model performance[10]. GP is the most recent evolutionary technique with one main characteristic: to allow complex representation structures, such as trees[16]. Although building a prediction model from a data set is the main aim in classification, this research focuses on the post-processing stage: to assemble several classifier models via GP in order to improve individual learners performances. Due to the non-deterministic nature of GP, complex solutions can be evolved while taking advantage of GP's representation flexibility which allows combining operators and functions from non-linear domains.

Both, homogeneous and heterogeneous ensembles, have been built by applying evolutionary and other bio-inspired techniques[5,15,18,19,25]. Finding the best feature-classifier combinations to build an ensemble is approached by Park et al. using a standard genetic algorithm in order to determine an optimal prediction model for lymphoma cancer classification of DNA sequences. The idea was to stochastically search for feature-classifier pairs that provide the best performance to build an ensemble through linear combination[19].

A difficult pharmaceutical problem was tackled by Langdon et al., in which decision trees and neural networks are combined as base learners in order to improve individual performances[15]. Results show that similar performance to a neural network is achieved by combining poor individual learners through GP. This research shows the advantages of implicit GP's flexibility in terms of solutions representation and the combination mechanism of individual predictors.

Other evolutionary techniques have also been applied to build ensembles. Yang et. al. used Particle Swarm Optimization (PSO) algorithm to build ensembles following a weighting scheme[25]. Positive results were achieved while tackling several real problems. It was observed that removing the weakest learner leads to a better overall performance. Three multiple-classifiers systems using PSO were presented by Macas et. al.[18]. Also, linear combination strategies were targeted and results showed accuracy improvement for the proposed approaches when compared to other heuristics. On a wider scope, PSO has been applied to the ensemble model selection problem[9]. Heterogeneous classifiers with optimized parameters are identified and selected for generating an ensemble.

A multi-objective approach using GP to evolve ensembles for classification of unbalanced data was presented by Bhowan et. al.[2]. The proposed approach is compared to a canonical GP classification system and two other standard classifiers. On an initial stage the competing objectives are each model's accuracy on majority and minority classes. On a second stage, a diversity measure is introduced as a third objective. Results showed an improved performance of the multi-objective approach when dealing with highly unbalanced data. An extension to this work ap-

plied multi-objective GP as a first step to then re-apply GP to solutions associated to the Pareto front[1]. Selective pressure is controlled by limiting the trees depth in order to promote the grouping of cooperative learners. Therefore, accurate, diverse and small ensembles are evolved.

Diversity among ensemble members has been considered as a factor that directly affects model's accuracy but a scientific explanation has not been determined[13]. Kuncheva et. al. studied this relationship through ten diversity measures among binary classifiers[14]. After an exhaustive experimental assessment, results show that both concepts are related depending on specific circumstances. Therefore, a strict link between diversity and ensemble's accuracy does not always exist. Bian et. al. carried out a study on diversity in homogeneous and heterogeneous ensembles[3]. The same diversity measures were assessed on 15 data sets, results led to group similar diversity measures however conclusive remarks were not raised.

Oliveira et. al. used a multi-objective genetic algorithm to investigate the accuracy/diversity dilemma on heterogeneous ensembles[5]. These two concepts were set as objectives and were evaluated together and separately. Results showed an improved performance when considering both metrics. Although three different classifiers (KNN, decision tree, SVM) were used as base predictors, a total of 30 built the ensemble, 10 per type were included. Yet, normalization problem among predictors is an issue not explicitly dealt with.

In this research several classifiers of different nature are fused through a stochastic technique and complex, yet effective, models (possibly non-linear ones) are created. In the proposed approach the diversity challenge is targeted in an implicit way through the evolutionary process and supported by an exhaustive empirical assessment. Schemes guided by weighting or majority voting strategies can be represented by the solutions in the genetic program. Another distinctive feature of the proposed ensemble mechanism is that the normalization problem is automatically approached. The evolutionary mechanism works out a prediction model from the combination of individual learners independently of their scale.

Most existing methods for building ensembles use summing, weighting sums or voting strategies. The proposed technique to combine classifiers outputs could outperform those techniques by building better ensembles that explicitly learned the fusion function of individual predictors. Moreover, once a fusion strategy has been learned, it could be applied in combination with most reviewed works for building ensembles. The proposed approach can be considered an instance of stacked generalization as introduced by Wolpert, where a modeling problem is serially approached in two levels: outputs of individual classifiers are feed to another classifier that determines the labels for objects[24]. However, Wolpert's stacked generalization is a generic framework under which many models fall, in fact, any ensemble can be considered an instance of stacked generalization.

## 3. Evolving ensembles through GP

Genetic programming[16] is an evolutionary technique which algorithmic structure follows the reproductive cycle of other evolutionary algorithms such as Genetic Algorithms (GA): an initial population is created randomly or by a pre-defined criterion, after that individuals are selected, recombined, mutated and then placed back into the solutions pool. GP uses different solutions representation which is normally more complex than other evolutionary techniques. For ensembles building, the advantages of working with a non-deterministic search technique are: possibility to explore difficult search spaces created by the combination of individual prediction models through arithmetic operators, automatic weighting mechanism by including constants affecting individual models, implicit ability to deal with normalization problems, among others. The rest of this section describes in detail the proposed GP approach to build ensembles; a general diagram of the proposed mechanism is shown in Figure 1.

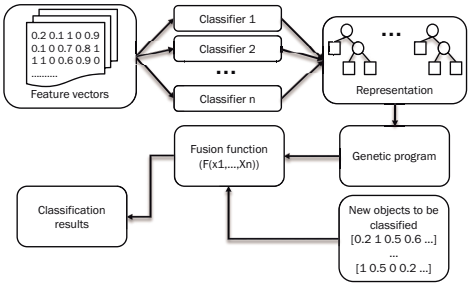


Fig. 1. GP ensemble: General scheme.

### 3.1. *Problem definition*

Consider a data set $\mathcal{D} = (\mathbf{x}_i, y_i)_{\{1,\ldots,N\}}$ with $N$ pairs of instances $(\mathbf{x}_i)$ and labels $(y_i)$ associated to a supervised classification problem. Assume that $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, is a binary classification problem with numeric attributes; and consider $g_k(\mathbf{x}_i) \in [-1, 1]$ as the classifier output $g_k$ for instance $\mathbf{x}_i$. $g_k$ represents the predictor's confidence value for $\mathbf{x}_i$ class. Every $g_k$ term can be modeled as a function $g_k : \mathbb{R}^d \to [-1, 1]$, where the predicted class for $\mathbf{x}_i$, defined by $\hat{y}_i$, is obtained as follows: $\hat{y}_i = sign(g_k(\mathbf{x}_i))$.

A fusion function $f(g_1(\mathbf{x}_i), \ldots, g_L(\mathbf{x}_i))$ is defined for combining $L$ classifiers outputs $g_{\{1,\ldots,L\}}(\mathbf{x}_i)$ for instance $\mathbf{x}_i$:

$$f(g_1(\mathbf{x}_i), \ldots, g_L(\mathbf{x}_i)) = \frac{1}{L} \sum_{k=1}^{L} w_k \cdot g_k(\mathbf{x}_i) \tag{1}$$

where $w_k$ is the $k$ classifier's associated weight. For example, in Adaboost[11] $w_k$ is

iteratively obtained and is related to $g_k$ performance considered as a weak learner. In random forest[4] and other ensembles $w_k$ is a constant equals to one[4]. In majority voting strategies[20], $w_k = 1$ and $g_k(\mathbf{x}_i)$ is replaced by $sign(g_k(\mathbf{x}_i))$.

Analogously, a fusion function for multi-class problems can be defined as follows:

$$f_m(\mathbf{h}_1(\mathbf{x}_i), \ldots, \mathbf{h}_L(\mathbf{x}_i)) = \frac{1}{L} \sum_{k=1}^{L} w_k \cdot \mathbf{h}_k(\mathbf{x}_i) \tag{2}$$

where $\mathbf{h}_k(\mathbf{x}_i)$ is the output of the $k^{th}$ individual multi-class classifier. Assuming a multi-class classification problem with $Q-$classes: $C_1, \ldots, C_Q$, a vector indicating classifier confidence per class, $\mathbf{h}_k(\mathbf{x}_i) = \langle h_k^1(\mathbf{x}_i), \ldots, h_k^Q(\mathbf{x}_i) \rangle$, is provided by each individual learner $k$, see Figure 2. In heterogeneous ensembles, every estimate $h_k^j(\mathbf{x}_i)$, is obtained by predictors of different nature, for example, $h_k^1(\mathbf{x}_i)$ determines class 1 confidence for $x_i$ instance according to a KNN classifier ($k^{th}$ classifier); $h_j^1(\mathbf{x}_i)$ defines class 1 confidence for the same instance according to random forest ( $j^{th}$ classifier), etc. On the other hand, in homogeneous ensembles confidence vectors come from the same classifier but each prediction model has been trained over different data set partitions or has been configured with different parameters.

A general *one-vs-rest* methodology is applied using every classifier to obtain multi-class confidence vectors, see Figure 2. In *one-vs-rest* classification, a binary classifier is trained per class where the $j^{th}-$ classifier uses as positive the training examples from class $j$ and as negative the rest. In this case, $h_k^j(\mathbf{x}_i)$ is the $k^{th}$ binary classifier confidence for instance $\mathbf{x}_i$ on label $C_j$.

The main objective of this research is to determine $f_m^*$, the fusion function that maximizes the classification performance on unseen data. A genetic program is thus applied to search the functions space which is determined by a pre-defined set of arithmetic operators, constants and classifiers outputs.

### 3.2. *Genetic program specifications*

Genetic Programming differs from other evolutionary techniques on its solutions representation. Normally, GP uses trees as data structures, in this research prediction models outputs are represented by leaf nodes. Additionally, constant values, to simulate weighting factors, are also represented by leaf nodes. Non-leaf or internal nodes are a set of arithmetic operators $+, -, \times, \div$ and single arity operators $^2, \sqrt{}, \log_{10}$. These operators were chosen for being commonly used in GP and because they allow representing non-linear models. Although, it is not possible to guarantee that the chosen operators are necessary and sufficient; it is at least guaranteed that traditional ensembles would be built (by considering addition and product). Figure 2 shows a tree example of an individual which encodes a fusion function that dictates how classifiers outputs are combined in an ensemble.

A centralized population has been used considering as the stopping condition a maximum number of generations. Standard mutation and crossover have been

applied[16]. Mutation randomly exchanges a node by a randomly created sub-tree. Crossover randomly exchanges sub-tree structures belonging to selected parents. Roulette-wheel is used as the selection mechanism and the whole population is replaced by the offspring every generation.

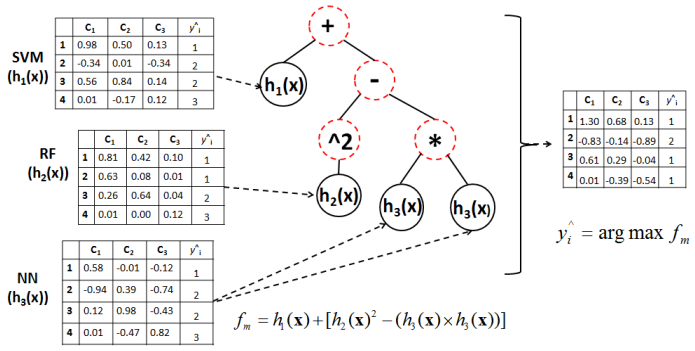


Fig. 2. Individual sample and data flow to combine classifiers outputs considering a 3-class/4-instances problem and 3-classifiers. On the left, a matrix per classifier ($k$), where $i, j$ entry indicates classifier confidence for instance $i$ with correct class $j$ (i.e., $h_k^j(\mathbf{x}_i)$). The last column per matrix shows the actual instances prediction (arg max across rows) using the corresponding matrix. GP's solution sample combines multiple models outputs and returns a fusion function ($f_m^*$) which produces an output matrix.

The whole GP procedure is described next: a set of classifiers outputs $((\mathbf{h}_1(\mathbf{x}_i), \ldots, \mathbf{h}_L(\mathbf{x}_i)))$ are the GP inputs for a training data set $\mathcal{D}$. An instance in $\mathcal{D}$ is classified via 10-fold cross-validation for every learner. These results are the GP inputs which means that for every instance and classifier there is an associated value obtained for that instance belonging to the test partition. In this way, overfitting is avoided as much as possible. The GP evolves and returns the fusion function ($f_m^*$) that achieves the best fitness during the optimization process. Then, $f_m^*$ is evaluated on unseen test data, see Figures 1 and 2.

The fitness value of every solution $f_m$ is calculated by evaluating its corresponding function's performance: 1) the predicted class per instance $\mathbf{x}_i$ is determined as follows: $\hat{y_i} = \arg\max_Q f_m(\mathbf{h}_1(\mathbf{x}_i), \ldots, \mathbf{h}_L(\mathbf{x}_i))$, which is the class index with maximum confidence; 2) $f_m$ predictive performance is assessed through standard measures to determine its fitness. Two performance metrics are assessed as objectives for optimization: accuracy and $f_1$-measure. Accuracy relates to the percentage of instances correctly classified by the evolved ensemble. While $f_1$-measure is the harmonic average between precision ($\frac{TP}{TP+FP}$) and recall ($\frac{TP}{TP+FN}$) per class. The average across classes is reported (also called, macro-average), this way of estimating the $f_1$-measure is known to be particularly useful when tackling unbalanced data sets.

## 4. Experiments and results

In this section results of extensive experimentation are reported in order to show the effectiveness and usefulness of the proposed method. Experiments include an evaluation of the proposed ensemble generation mechanism based on GP in benchmark data; a comparison between the proposed method and baseline ensembles; a performance assessment of the proposed method for generating homogeneous ensembles; and an analysis of the generalization capabilities of the learned fusion functions.

### 4.1. *Experimental settings*

The following classifiers were considered for building ensembles via GP: *random forest, SVM (Support Vector Machine), klogistic, linear-kridge, non-linear kridge, 1NN (Nearest Neighbor), 3NN, naïve Bayes, gkridge, and neural network*. These classifiers were taken from the CLOP toolbox comprising a variety of methodologies that have been widely used to build ensembles[9,7].

The proposed technique has been assessed through 40 data sets from the UCI repository plus the SCEF[a] data set. The latter is associated to an object recognition problem and has been previously evaluated on building heterogeneous ensembles[9].

Table 1. Experimental data sets characteristics.

| Data set | Instances | Attributes | Classes | Data set | Instances | Attributes | Classes |
|---|---|---|---|---|---|---|---|
| **SCEF** | 6244 | **737** | 10 | | | | |
| **Australian** | 690 | 14 | 2 | **Phoneme** | 5404 | 5 | 2 |
| **Balance** | 625 | 4 | 3 | **Pima** | 768 | 8 | 2 |
| **Banana** | 5300 | 2 | 2 | **Ring** | **7400** | 20 | 2 |
| **Bands** | 539 | 19 | 2 | **Saheart** | 462 | 9 | 2 |
| **Breast** | 286 | 9 | 2 | **Satimage** | 6435 | 36 | 7 |
| **Bupa** | 345 | 6 | 2 | **Segment** | 2310 | 19 | 7 |
| **Car** | 1728 | 6 | 4 | **Sonar** | 208 | **60** | 2 |
| **Chess** | 3196 | 36 | 2 | **Spambase** | 4597 | 55 | 2 |
| **Contraceptive** | 1473 | 9 | 3 | **Spectfheart** | 267 | 44 | 2 |
| **Crx** | 125 | 15 | 2 | **Splice** | 3190 | **60** | 3 |
| **Flare-Solar** | 1066 | 9 | 2 | **Tae** | 151 | 5 | 3 |
| **German** | 1000 | 20 | 2 | **Texture** | 5500 | 40 | **11** |
| **Haberman** | 306 | 3 | 2 | **Thyroid** | 7200 | 21 | 3 |
| **Heart** | 270 | 13 | 2 | **Tic-tac-toe** | 958 | 9 | 2 |
| **Hepatitis** | 155 | 19 | 2 | **Titanic** | 2201 | 3 | 2 |
| **Housevotes** | 435 | 16 | 2 | **Twonorm** | **7400** | 20 | 2 |
| **Iris** | 150 | 4 | 3 | **Vehicle** | 846 | 18 | 4 |
| **Led7digit** | 500 | 7 | 10 | **Vowel** | 990 | 13 | **11** |
| **Mammographic** | 961 | 5 | 2 | **Wine** | 178 | 13 | 3 |
| **Monks** | 432 | 6 | 2 | **Wisconsin** | 683 | 9 | 2 |

In every experimental sample, data sets are divided in training and testing partitions. Training partitions are used to learn a fusion function and testing partitions are used to assess the built ensemble. In particular, the SCEF data set was partition as follows: $3,615$ testing and $2,629$ training instances. Random partitioning was applied to the rest of databases considering 70% for training and 30% for testing. Table 1 shows data sets characteristics.

Three GP configurations have been defined for the experimental assessment:

[a]http://mklab.iti.gr/project/scef

GPE (Genetic Programming Ensemble) applies the full operators set, GPE-a (Genetic Programming Ensemble by addition) applies only the addition operator simulating the standard approach to learn weights and select ensemble members[19,18,25]; and AVE (Average Voting Ensemble) builds ensembles by a voting strategy, i.e. the fusion function from Equation (2) with $w_k = 1$. Moreover, every configuration is also tested when using only the top-5 models with better performance on training data; aiming to determine the accuracy effect of classifiers assembled via GP.

The processing hardware platform to carry out the experiments was a $64-$bit Intel(R) Core(TM) $i7-3820$ @3.60GHz, 64GB memory. At high level, Matlab $2013a$ and GPLab $v3$ toolbox were used.

### 4.2. *SCEF experimental results*

This section analyzes results obtained by the proposed ensemble generation mechanism for SCEF. This data set is considered as representative because it is among the largest ones in terms of instances and attributes. Preliminary results on the SCEF data set were carried out considering a population size of 50 individual and a stopping condition of 100 generations, 10 experimental samples were executed[7] .

Table 2 shows individual classifiers performance in terms of accuracy and $f_1$-measure. Random forest (RF) significantly outperforms other classifiers. It is expected that the proposed approach improves RF's performance.

Table 2. Results (%) obtained by individual classifiers in terms of accuracy/$f_1$ measure over SCEF data set.

| | RF | SVM | Klogistic | Kridge-l | Kridge-n | 1NN | 3NN | N.Bayes | Gkridge | Neural N |
|---|---|---|---|---|---|---|---|---|---|---|
| Acc. | **90.70** | 55.10 | 70.60 | 13.64 | 74.70 | 69.30 | 69.10 | 26.50 | 20.60 | 55.80 |
| $f_1$ | **79.30** | 49.90 | 62.80 | 2.400 | 63.10 | 60.10 | 57.40 | 21.60 | 3.421 | 37.70 |

In Table 3, performance metrics average and standard deviation after 10 runs obtained by the three GP ensemble variants are presented. The proposed ensemble variants outperform significantly the raw-fusion function (AVE) in terms of both measures with differences between $40-50\%$. GP-ensembles even outperformed AVE when using the top-5 models. This shows the limitations of the raw fusion function for heterogeneous ensembles.

Table 3. Results (%) obtained by different strategies over SCEF data set when optimizing accuracy (top) and $f_1$ (bottom). AVE: raw fusion; GPE-a: GP uses only sums; GPE: proposed GP.

| | **AVE** | **AVE-Top5** | **GPE-a** | **GPE-a-Top5** | **GPE** | **GPE-Top5** |
|---|---|---|---|---|---|---|
| **Acc.** | 31.50 | 81.40 | 90.80(0.001) | 91.10(0.002) | **92.30(0.002)** | 91.20(0.001) |
| $f_1$**.** | 27.2 | 71.90 | 80.40(0.001) | 80.40(0.001) | **85.30(0.003)** | 80.55(0.003) |

All GP ensembles outperformed the best individual classifier. The improvement

for both performance metrics was small for all methods but for GPE. Improvements of more than 1.5% and 6% were obtained by GPE with respect to the best individual classifier, in terms of accuracy and $f_1$, respectively. GPE was able to find very effective fusion functions for heterogeneous classifiers, even when most models performance was low. Moreover, a 6% improvement in $f_1$ is significant when persists across classes, because it focuses on the average performance over classes.

The best results were obtained by the GPE ensemble, i.e., using all operators and classifiers. Using more operators in the GP might allow to obtain better fusion functions. Moreover, the GP has more selection options because it used all classifiers, which explains the improvement over GPE-Top 5.

The best result in Table 3 improved by more than 10% previously reported accuracy for the same data set (81.49%)[9]. Escalante et al. did not optimize the decision threshold thus the ROC curve area (AUC) is also reported[9]. Comparing the best individual AUC (98.44) with the best result reported in[9] (94.05), an improvement of more than 4% is still achieved. These results, to the best of our knowledge, are the best ones so far reported for the SCEF data set.

### 4.3. *Benchmarking results*

This section reports the assessment of the proposed ensemble mechanism considering forty data sets from the UCI repository (see Table 1). The objective of this experimental evaluation is to analyze the behavior of the proposed technique on benchmark data presenting a wide variety in terms of number of: instances, features and classes. Considering previous experimental results[7], a population size of 100 individuals evolving up to 100 generations as standard settings are defined.

Figures 3 and 4 show performance differences, in terms of accuracy and $f_1$-measure respectively, for every ensemble method with respect to the best individual classifier per data set (i.e., ensemble performance minus best-classifier performance). Values above zero indicate significant improvements over the best individual model. These results are the average over 10 runs per data set. From these figures, it is observed that best results were obtained by the GPE ensemble. Specifically, GPE best performances were achieved for "Wisconsin", "Twonorm", "Housevotes" and "Spambase". Using a raw fusion ensemble (AVE) showed to be the worst approach in all cases; and applying a GP ensemble (GPE-a) based on additions had a similar performance when compared against the best individual classifier in most cases.

Table 4. Average and standard deviation performances for B-Classifier: best classifier; AVE: raw fusion; GPE-a: GP using only sums; GPE: proposed GP.

| | **B-Classifier** | **AVE** | **GPE-a** | **GPE** |
|---|---|---|---|---|
| **Acc.** | $61.01(25.18)_{(1,0,-)}$ | $40.23(27.91)_{(-,-,-)}$ | $62.68(24.45)_{(0,1,-)}$ | $\mathbf{85.75(12.21)}_{(1,1,1)}$ |
| $f_1$**.** | $63.50(26.32)_{(1,0,-)}$ | $31.06(29.00)_{(-,-,-)}$ | $62.91(26.71)_{(0,1,-)}$ | $\mathbf{84.49(14.94)}_{(1,1,1)}$ |

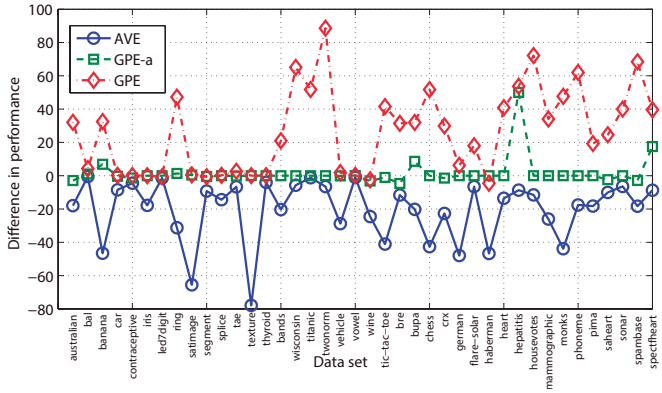


Fig. 3. Accuracy differences between the best individual classifier and the ensembles.

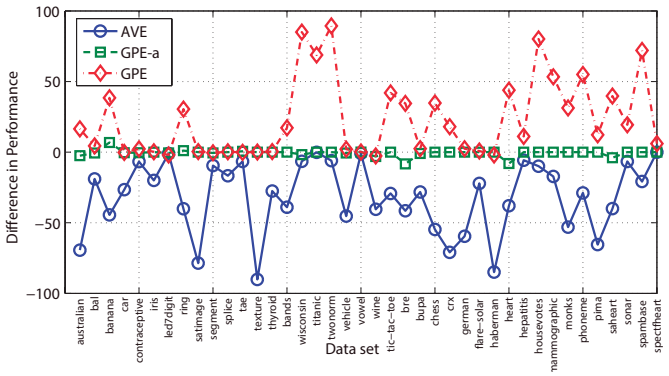


Fig. 4. $f_1$-measure differences the best individual classifier and the ensembles.

In order to validate the experimental assessment reported in this article, a statistical analysis is carried out. Results in Figures 3 and 4 were statistically analyzed using t-test over 10 runs per data set. T-test results show that the proposed technique GPE significantly improves AVE in all data sets. Also, GPE outperforms GPE-a in 31 out of 40 data sets. There is no statistical difference for accuracy between GPE and GPE-a for *iris*, *led7digit*, *texture*, *vowel* and *wine* data sets. On the other hand, $f_1$-measure for GPE and GPE-a is not significantly different for *car*, *flare-solar*, *iris*, *segment*, *tae*, *vowel* and *wine* data sets.

Table 4 presents average and standard deviation results for all experimental data sets obtained by ensemble methods. The best average results are obtained by GPE together with the lowest standard deviation being the most robust approach. Next to each performance metric, statistical tests using t-test are included in parenthesis. For example, accuracy for GPE-a indicates (0,1,-) which means: statistical similarity

to the best individual learner, statistical significant difference as regards AVE and statistical significant deterioration with respect to GPE.

### 4.4. *Comparing to other ensemble methodologies*

This section carries out a comparison among ensembles evolved by the proposed mechanism and alternative techniques. The following ensembles were considered: random forest[4]: a bagging method using decision trees; adaboost[11] a boosting method using SVMs; logitboost[17] a boosting method using regression trees. These ensemble methods have proved to be very effective in many applications and even in academic challenges[11,4,12,9]. Table 5 shows average and standard deviation performances of these methods over all data sets.

Table 5. Average and standard deviation performances for RF: random forest; AVE: raw fusion; GPE-a: GP using only sums; GPE: proposed GP.

| | **RF** | **Adaboost** | **Logitboost** | **AVE** | **GPE-a** | **GPE** |
|---|---|---|---|---|---|---|
| **Acc.** | 40.95(36.17) | 41.69(32.82) | 37.64(28.04) | 40.23(27.91) | 62.68(24.45) | **85.75(12.21)** |
| $f_1$. | 38.64(35.59) | 40.30(31.70) | 32.67(25.95) | 31.06(29.00) | 62.91(26.71) | **84.49(14.94)** |

In addition, a statistical test (t-test) is performed over the results shown in Table 5 to obtain a comparison of GPE, GPE-a and AVE regarding RF, Adaboost and Logitboost. The results of this statistical test show that GPE and GPE-a are better options than RF, Adaboost and Logitboost, while AVE is the worst option. These statistical results and the values in Table 5 show that the proposed method significantly outperforms the three baseline ensembles. These alternative ensemble methods performed poorly, achieving similar results than AVE. In fact, GPE obtained more than twice the performance of the baseline classifiers. One should note that outputs of each alternative ensemble could be considered as another classifier in the proposed fusion mechanism. In fact, random forest outputs are considered in the proposed ensemble mechanism.

### 4.5. *Fusion functions generalization*

In this section the generalization performance of the learned functions is evaluated. For this experiment, a fusion function learned through GP for each data set is selected and assessed in the other 39 data sets. Hence, a total of 40 fusion functions (each learned for a different data set) were used to combine classifiers outputs to test in all data sets. Figure 5 shows the results matrix, where $i, j$ entry indicates the fusion function's classification performance learned for data set $i$ and evaluated in data set $j$; where red zones indicate high performance and blue zones are associated with low performance.

From Figure 5 it can be seen that better performance was obtained by functions learned and evaluated in the same data set, see diagonal elements, this is a somewhat

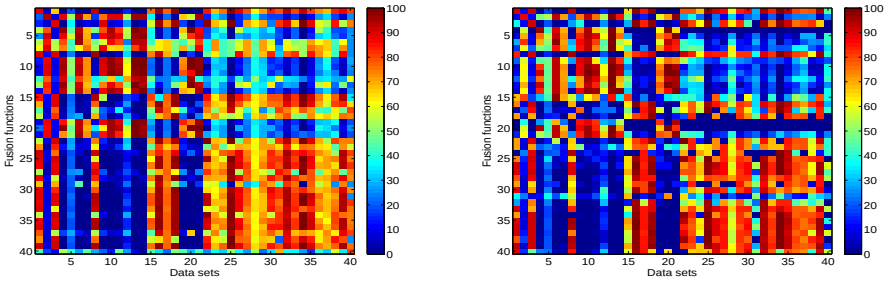


Fig. 5. Functions classification performance (rows) learned for different data sets and evaluated in all data sets (columns). Results correspond to accuracy (left plot) and $f_1$ measure (right plot).

expected result. However, it is interesting that there are clearly distinguishable column-wise and row-wise red/blue zones. Row-wise red (resp. blue) zones indicate fusion functions with high (resp. low) generalization capabilities. Column-wise red (resp. blue) zones indicate easy (resp. difficult) data sets for which most (resp. a few) fusion functions were effective. There are more red-zones than blue ones, thus assessed fusion functions are somewhat generalizable and can be applied to other data sets different to the one they were learned for. However, it is desirable to use a fusion function learned for each specific data set.

Table 6 presents a summary of the main results from this experiment. It can be seen that when using the same data set for learning the function and evaluation (row 2) very competitive performance can be obtained. However, if the best fusion function for each data set is selected the performance would be very close to 100% (row 4). This means that for some data sets, better results were obtained with functions learned from different data sets. The average performance across all data sets is low (row 3), however, the learned functions still have interesting generalization properties. For example, row 5 in Table 6 shows the average number of data sets in which fusion functions outperformed the best individual classifier (row 1). Clearly, each function was helpful in more than 17 and 15 data sets, when optimizing accuracy and $f_1$ measure, respectively.

Table 6. Evaluation summary of performance generalization from fusion functions.

| ID | Measure | **Accuracy** | **f-measure** |
|---|---|---|---|
| 1 | **Perf. Best classifier** | 59.70 | 62.12 |
| 2 | **Perf. when using the ad-hoc weight** | 87.35 | 86.87 |
| 3 | **Avg. performance overall data sets** | 48.81 | 43.51 |
| 4 | **Maximum performance** | 97.08 | 97.47 |
| 5 | **Avg. number of improved data sets** | 17.22 | 15.22 |

### 4.6. *In deep solutions analysis*

In this section different aspects of the evolved fusion functions are analyzed. The aim is to gain an insight into the type of functions that can be learned with the proposed approach.

Figure 6 shows the average frequency (10 runs, 40 UCI data sets) of classifier selection in ensembles generated with the GPE approach. It is clear from this figure that the most used classifier is the most accurate one: random forest (see Table 2). This is somewhat an expected result, however, it is interesting that the second and third more frequently selected classifiers were GKridge and Klogistic, respectively. The latter classifiers are not among the best ones in terms of individual performance, see Table 2. These results confirm findings in ensemble theory that suggest not only accuracy of individual ensembles is important, but also models diversity (i.e., their ability to make uncorrelated errors)[13,14,6,26].

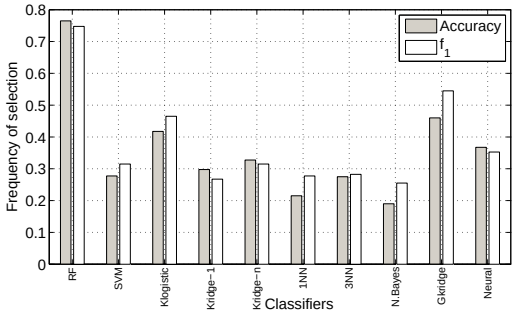


Fig. 6. Average frequency of classifier selection in ensemble functions (40 data sets).

Table 7 shows the average number of terminal and non-terminal nodes (10 runs, 40 UCI data sets) in the fusion functions generated by the proposed method. Recall terminal nodes are associated to classifiers' outputs, while non-terminal nodes correspond to arithmetic operators. From Table 7 it can be seen that there is not a conclusive trend regarding the number of nodes. Nevertheless, it can be noticed that for some data sets very small trees (fusion functions) are obtained, e.g., "Monks $(-(NN, Bayes))$" and "Wine" $((+(\sqrt{(}RF), KL)))$ while for other data sets, large trees are evolved, e.g., "Satimage[b]" . This variety of results reveals GP's adaptive property for generating ad-hoc fusion functions for specific data sets.

[b] $-(KR, -(-(^2(-(0.5, -(KN, -(^2(-(0.8, RF)), \times(\times(0.6, -(0.5, ^2(NN))), \times(KR, KR))))))), RF), +(RF, RF)))$

Table 7. Average number of terminal and non-terminal nodes for fusion functions learned with GPE, each result is the average of 10 independent runs.

| | Optimizing accuracy | | Optimizing $f_1$ | |
|---|---|---|---|---|
| | #Non-terminal | #Terminals | #Non-terminal | #Terminals |
| **australian** | 10 | 12 | 16 | 21 |
| **balance** | 16 | 22 | 15 | 19 |
| **banana** | 15 | 18 | 11 | 15 |
| **bands** | 13 | 16 | 15 | 21 |
| **breast** | 13 | 17 | 18 | 25 |
| **bupa** | 8 | 10 | 15 | 20 |
| **car** | 8 | 14 | 11 | 16 |
| **chess** | 11 | 14 | 8 | 12 |
| **contraceptive** | 21 | 28 | 17 | 21 |
| **crx** | 13 | 17 | 12 | 15 |
| **flare-solar** | 13 | 14 | 21 | 31 |
| **german** | 13 | 15 | 13 | 16 |
| **haberman** | 10 | 13 | 12 | 17 |
| **heart** | 8 | 11 | 7 | 9 |
| **hepatitis** | 6 | 8 | 14 | 18 |
| **housevotes** | 10 | 14 | 9 | 12 |
| **iris** | 3 | 4 | 3 | 2 |
| **led7digit** | 12 | 13 | 20 | 28 |
| **mammographic** | 11 | 15 | 17 | 23 |
| **monks** | 2 | 2 | 3 | 2 |
| **phoneme** | 18 | 21 | 14 | 21 |
| **pima** | 8 | 12 | 14 | 21 |
| **ring** | 13 | 16 | 12 | 16 |
| **saheart** | 17 | 24 | 19 | 24 |
| **satimage** | 18 | 20 | 9 | 10 |
| **segment** | 8 | 12 | 17 | 20 |
| **sonar** | 8 | 10 | 10 | 13 |
| **spambase** | 13 | 18 | 10 | 13 |
| **spectfheart** | 10 | 14 | 19 | 24 |
| **splice** | 11 | 12 | 9 | 12 |
| **tae** | 14 | 18 | 17 | 23 |
| **texture** | 4 | 4 | 7 | 7 |
| **thyroid** | 8 | 13 | 8 | 13 |
| **tic-tac-toe** | 10 | 13 | 10 | 12 |
| **titanic** | 8 | 11 | 7 | 9 |
| **twonorm** | 12 | 15 | 10 | 14 |
| **vehicle** | 15 | 16 | 22 | 24 |
| **vowel** | 7 | 10 | 7 | 8 |
| **wine** | 4 | 5 | 2 | 3 |
| **wisconsin** | 7 | 8 | 8 | 11 |

### 4.7. *Homogeneous ensembles*

An experimental assessment to evaluate the suitability of the proposed mechanism to build homogeneous ensembles is also performed. The aim is to make an initial evaluation of the proposed approach when applied to ensemble models from individual classifiers of the same nature. The "Ring" data set is considered for this experiment as it is the largest one. For building homogeneous ensembles the proposed approach is applied as described in Section 3 following experimental constraints of Subsection 4.1, without changes. The only difference is that confidence values $\mathbf{h}_1(\mathbf{x}_i), \ldots, \mathbf{h}_L(\mathbf{x}_i)$ are obtained from the same classification model, but trained on different subsets of the same data set as the source for diversity to build homogeneous ensembles in this study. Specifically, half of training instances and features were randomly selected to train each classification model, where a total of $L = 10$ models were considered for this experiment. Empirical results for homogeneous ensembles are summarized in Table 8. For different ensemble building strategies, average and standard deviation performances in terms of accuracy and $f_1$ measure, are reported after 10 independent runs.

Homogeneous ensembles generated with GPE outperformed the best individual classifier (GKridge: 48.74 accuracy, 65.53 $f_1$) and the other ensemble variants, sim-

Table 8. Average and standard deviation performances for homogeneous ensembles over "Ring" data set; AVE: raw fusion; GPE-a: GP using only sums; GPE: proposed GP.

| | AVE | AVE-Top5 | GPE-a | GPE-a-Top5 | GPE | GPE-Top5 |
|---|---|---|---|---|---|---|
| | | | **KNN** | | | |
| **Acc.** | 10.54(2e-15) | 14.46(4e-15) | 36.92(4.23) | 39.39 (5.16) | 86.52(2.94) | 84.72(2.99) |
| $f_1$. | 18.67(4e-15) | 23.4(4e-15) | 44.21(5.91) | 41.71 (11.75) | 86.43(2.26) | 83.30(2.15) |
| | | | **KRIDGE** | | | |
| **Acc.** | 24.37(0) | 24.64(4e-15) | 46.02(1.03) | 46.02(1.31) | 77.17(0.86) | 77.01(2.05) |
| $f_1$. | 28.89(8e-15) | 29.65(0) | 44.65(2.35) | 45.88(5.31) | 74.75(2.31) | 71.68(1.19) |
| | | | **NAIVE** | | | |
| **Acc.** | 29.59(4e-15) | 28.29(0) | 52.77(1.48) | 47.58(1.07) | 83.39(1.24) | 80.85(1.03) |
| $f_1$. | 21.19(0) | 22.85(4e-15) | 45.11(2.56) | 46.65(7.39) | 76.15(2.71) | 74.73(3.53) |
| | | | **NEURAL** | | | |
| **Acc.** | 20.99(4e-15) | 24.14(4e-15) | 48.27(1.39) | 48.81(1.71) | 79.00(0.17) | 75.66(0.22) |
| $f_1$. | 21.13(0) | 23.02(4e-15) | 48.87(2.11) | 50.86(0.64) | 77.74(0.28) | 76.05(0.31) |
| | | | **RF** | | | |
| **Acc.** | 4.28(0) | 5.50(0) | 51.12 (1.28) | 51.61 (1.35) | 95.30(0.35) | 94.67(0.22) |
| $f_1$. | 2.57(0) | 3.87(0) | 40.26 (14.86) | 37.86 (20.07) | 95.20(0.20) | 93.78(0.62) |
| | | | **KLOGISTIC** | | | |
| **Acc.** | 24.05(4e-15) | 25.09(0) | 47.74(1.11) | 47.68(1.07) | 77.47(1.90) | 77.30(1.84) |
| $f_1$. | 26.57(0) | 28.07(0) | 46.04(3.19) | 46.65(7.39) | 75.68(1.91) | 72.64(0.62) |

ilar behavior was observed by heterogeneous ensembles. Considering results from this case study, it is possible to conclude that the proposed approach is also useful to generate highly effective homogeneous ensembles. However, comparing the best homogeneous ensemble performance for the ring data set to the best heterogeneous ensemble built by GPE (95.95 accuracy, 96.15 $f_1$); the heterogeneous approach significantly improves the others. A partial conclusion indicates that heterogeneous ensembles seem to outperform homogeneous ones; although an extensive study is required to backup these results.

## 5. Conclusions

In this research, a genetic programming approach to learn fusion functions for building heterogeneous ensembles has been presented. The proposed approach consists in combining classifiers outputs through GP. Approaching ensemble construction through an evolutionary technique allows for more complex fusion-functions spaces to arise. GP solutions representation includes arithmetic operators to relate individual learners thus non-linear fusion functions can be evolved in an ensemble. Empirical results on both, benchmark data and a challenging object-recognition data set were reported. The extensive empirical assessment exposed the proposed approach effectiveness. GP based ensembles consistently outperformed the best individual models, a raw-ensemble of heterogeneous classifiers, several configurations to optimize ensemble models and traditional ensembles.

An analysis of evolved solutions, in terms of individual classifiers frequency to form an ensemble, not only confirmed the expected high membership of the best individual classifier but also showed at second and third ranking positions two weak learners. This confirms, to a certain level, diversity importance among ensemble members that make uncorrelated errors. Another empirical assessment applied an ensemble solution from a specific data set to the rest of benchmark data sets. Tendencies showed high performances which imply generalization properties of evolved fusion functions. Also, homogeneous ensembles were built by the proposed

approach demonstrating flexibility.

Several research directions were identified: including the suitability of the proposed approach to learn fusion functions for other tasks, including multi-modal information retrieval [8] and ensemble feature selection [21].

**Acknowledgments**

This work was partly supported by the National Council of Science and Technology of Mexico (CONACyT) through scholarship grant 287045.

**References**

1. U. Bhowan, M. Johnston, Mengjie Zhang, and Xin Yao. Reusing genetic programming for ensemble selection in classification of unbalanced data. *IEEE Trans. Evol. Comput.*, In press, DOI 10.1109/TEVC.2013.2293393.
2. U. Bhowan, M. Johnston, Mengjie Zhang, and Xin Yao. Evolving diverse ensembles using genetic programming for classification with unbalanced data. *IEEE Trans. Evol. Comput.*, 17(3):368–386, 2013.
3. S. Bian and W. Wang. On diversity and accuracy of homogeneous and heterogeneous ensembles. *International Journal of Hybrid Intelligent Systems*, 4:103–128, 2007.
4. L. Breiman. Random forest. *Mach. Learn.*, 24(2):123–140, 2001.
5. D.F. de Oliveira, A.M.P. Canuto, and M. C P De Souto. Use of multi-objective genetic algorithms to investigate the diversity/accuracy dilemma in heterogeneous ensembles. In *Proc. of IJCNN*, pages 2339–2346, 2010.
6. T. Dietterich. Ensemble methods in machine learning. volume 1857 of *LNCS*, pages 1–15. Springer, 2000.
7. H. J. Escalante, N. Acosta-Mendoza, A. Morales-Reyes, and A. Gago-Alonso. Genetic programming of heterogeneous ensembles for classification. volume 8258 of *LNCS*, pages 9–16. Springer, 2013.
8. H. J. Escalante, C. Hernandez, E. Sucar, and M. Montes. Late fusion of heterogeneous methods for multimedia image retrieval. In *Proc. of the ACM Multimedia Information Retrieval Conference*, 2008.
9. H. J. Escalante, M. Montes, and L. E. Sucar. Ensemble particle swarm model selection. In *Proc. of IJCNN*, pages 1–10, 2010.
10. P. Espejo, S. Ventura, and F. Herrera. A survey on the application of genetic programming to classification. *IEEE T. Syst. Man. Cyb. C*, 40(2):121–144, 2010.
11. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55:119–139, 1997.
12. I. Guyon, A. Saffari, G. Dror, and G. Cawley. Analysis of the ijcnn 2007 agnostic learning vs. prior knowledge challenge. *Neural Networks*, 21(2-3):544–550, 2008.
13. L. I. Kuncheva. That elusive diversity in classifier ensembles. volume 2652 of *LNCS*, pages 1126–1138. Springer, 2003.
14. L.I. Kuncheva and C.J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Mach. Learn.*, 51(2):181–207, 2003.
15. W. B. Langdon, S. J. Barret, and B.F. Buxton. Combining decision trees and neural networks for drug discovery. volume 2278 of *LNCS*, pages 60–70. Springer, 2002.
16. W. B. Langdon and R. Poli. *Foundations of Genetic Programming*. Springer, 2001.
17. R.W. Lutz. Logitboost with trees applied to the wcci 2006 performance prediction challenge datasets. In *Proc. of IJCNN*, pages 1657–1660. IEEE, 2006.
18. M. Macas, D. R. B. Gabrys, and L. Lhotska. Particle swarm optimization of multiple classifier systems. volume 4507 of *LNCS*, pages 333–340. Springer, 2007.

19. C. Park and S. Cho. Evolutionary computation for optimal ensemble classifier in lymphoma cancer classification. volume 2871 of *LNAI*, pages 521–530. Springer, 2003.
20. D. Ruta and B. Gabrys. Classifier selection for majority voting. *Information Fusion*, 6:63–81, 20051.
21. Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques. In *Proc. of ECML/PKDD*, volume 5112 of *LNAI*, pages 313–325. Springer, 2008.
22. S. Vega-Pons and J. Ruiz-Schulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition Artificial Intelligence*, 25(337), 2011.
23. J. D. Wichard, C. Merkwirth, and M. Ogorzalek. Building ensembles with heterogeneous models. In *7th Course on the Intl. School on Neural Nets*, 2002.
24. David H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.
25. L. Yang and Z. Qin. Combining classifiers with particle swarms. volume 3611 of *LNCS*, pages 756–763. Springer, 2005.
26. Cha Zhang and Ma Yunqian. *Ensemble Machine Learning, Methods and Applications*. Springer, 2012.
27. L. Zhang, L. Zhang, W. Teng, and Y. Chen. Based on information fusion technique with data mining in the application of finance early-warning. *Procedia Computer Science*, 17:695–703, 2013.

**Niusvel Acosta-Mendoza** obtained a BEng on Computational Science from University of Informatics Sciences, Cuba, in 2007. In July 2013 he received the MSc degree in Computer Science at INAOE, Mexico. Currently, he is a research fellow in the Data Mining Department at CENATAV, Cuba and a PhD student at INAOE. His research interests include: knowledge discovery and data mining in graph-based content, machine learning and evolutionary computation.

**Hugo Jair Escalante** obtained his degree of PhD in computer science from INAOE in Mexico, where he is now associate researcher. He is member of the Mexican System of Researchers (SNI) since 2011, and co-director of ChaLearn, the Challenges in Machine Learning organization (2011-2014). His main research interests are on machine learning and computational intelligence with applications in text mining and high-level computer vision.

**Alicia Morales-Reyes** was admitted to the PhD degree in the College of Science and Engineering at the University of Edinburgh-UK, in 2011. She received the MSc degree in Computer Science (INAOE) in 2006 and a BEng on Electrical and Electronics Engineering (UNAM) in 2002, Mexico. She is an associate researcher at INAOE collaborating within the Reconfigurable and High Performance Computing research group.

**Andrés Gago-Alonso** obtained a BEng on Computer Science from Havana University, Havana, Cuba, in 2004. He holds the MSc degree in Mathematics from the same university in 2007. He completed his PhD degree in Computational Sciences at INAOE in January 2010. Currently, he is an associate researcher in the Data Mining Department at CENATAV, Cuba. His research interests include: knowledge discovery, data mining in graph-based content and evolutionary computation