Cooperative Evolution of Rules for Classification

Catalin Stoean
University of Craiova
Department of Computer Science
A. I. Cuza, No 13, 200585, Craiova, Romania
catalin.stoean@inf.ucv.ro

Mike Preuss
University of Dortmund
Department of Computer Science
Otto-Hahn 14, 44221 Dortmund, Germany
mike.preuss@cs.uni-dortmund.de

D. Dumitrescu
Babes-Bolyai" University of Cluj-Napoca
Department of Computer Science
M. Kogalniceanu, No 1B, 400084, Cluj, Romania
ddumitr@cs.ubbcluj.ro

Ruxandra Stoean
University of Craiova
Department of Computer Science
A. I. Cuza, No 13, 200585, Craiova, Romania
ruxandra.stoean@inf.ucv.ro

Abstract

A new learning technique based on cooperative coevolution is proposed for tackling classification problems. For each possible outcome of the classification task, a population of if-then rules, all having that certain class as the conclusion part, is evolved. Cooperation between rules appears in the evaluation stage, when complete sets of rules are formed with the purpose of measuring their classification accuracy on the training data. In the end of the evolution process, a complete set of rules is extracted by selecting a rule from each of the final populations. It is then applied to the test data. Some interesting results were obtained from experiments conducted on Fisher's iris benchmark problem.

1. Introduction

Cooperative coevolution for classification (CCC) is inspired from the cooperative coevolution framework proposed in [3], [4]; therein, it was applied for function optimization and each variable of the function to be optimized is evolved by a separate population. When the fitness of an individual from one population is evaluated, $collaborators$ from the other populations that evolve simultaneously have to be chosen in order to form a complete solution that can be measured. The evaluation of this solution is assigned as the fitness of the initial individual, therefore its score directly depends on the chosen collaborators, or, to be more specific, on the way of deciding cooperations.

In our classification approach, the number of populations that coevolve equals the number of outcomes of the classification problem. The data set to be classified is split into two parts, one used for training the evolutionary algorithm (EA) (or for learning the characteristics of each class) and one for testing the performance of the obtained rules. Each population evolves if-then rules with the same outcome; similarly to function optimization using cooperative coevolution, when the quality of an individual (rule) is computed, collaborators from all others populations are selected in order to form a complete rule set to be applied to the training data. In the end of the evolution process, a set of rules is extracted from all final populations and is applied to the test data.

There is a set of parameters of cooperative coevolutionary algorithms that directly affect the values for the fitness evaluations and, through this, indirectly influence the way the algorithm evolves; the parameters refer to the selection of cooperations, the number of collaborators that are picked from each population in order to reach a more objective fitness and to the manner in which fitness is computed when several collaborators are considered from each population. In the experimental section, we will explore some insights of the CCC algorithm while employing several types of parameter configurations with the aim of an optimum setting.

The paper is organized as follows: the next section contains some fundamental aspects of cooperative coevolution, while proposed approach for classification is described in section 3. Experimental results are presented in section 4 and the paper closes with a discussion section.

2. Prerequisites of Cooperative Coevolution

Coevolution is inspired by the way the various species in nature interact; species may be in competition or they may cooperate for a common goal. In competitive coevolution, the evaluation of an individual is determined by a set of competitions between the current individual and other individuals, while in cooperative coevolution, collaborations between two or more individuals are necessary in order to evaluate one individual.

The first step that has to be undertaken when a problem is intended to be solved by cooperative coevolution is to find a proper decomposition of the problem into components. Then, each component is assigned to a population (or species); each population evolves independently, except for the moment when the evaluation process takes place. As each individual in a population represents a component of the problem, collaborators have to be selected from all populations in order to assemble a solution that may be evaluated. Therefore, at each generation, each individual from each population is evaluated by selecting collaborators from the other populations.

The algorithm that describes this process is outlined below:

Algorithm 1 Cooperative coevolutionary algorithm

```
t = 0;
for each species s do
   randomly initialize population Pop_s(t);
end for
for each species s do
   evaluate Pop_s(t);
end for
while termination condition = false do
   t = t + 1;
   for each species s do
      select population Pop_s(t) from Pop_s(t - 1);
      apply genetic operators to Pop_s(t);
      evaluate Pop_s(t);
   end for
end while
```

The algorithm starts with the initialization of each population. The first evaluation of the individuals in each population is performed by making a random selection of individuals from each of the other populations and evaluating the obtained solutions. After this starting phase, each population is evolved using an EA.

As mentioned before, the choice of collaborators represents the main issue in this process. Consequently, when building a cooperative coevolutionary algorithm, there are three attributes regarding selection that have to be decided [4] on:

1. *Collaborator selection pressure* refers to the way individuals are chosen from each population in order to form complete solutions to the problem, i.e. pick the best individual according to its previous fitness score, pick a random individual or use classic selection schemes in order to select individuals from each of the other populations.

2. *Collaboration pool size* represents the number of collaborators that are selected from each population.

3. *Collaboration credit assignment* decides the way of computing the fitness of the current individual. This attribute appears in case the *collaboration pool size* is higher than one. There are three methods for computing this assignment:

   (a) *Optimistic* - the fitness of the current individual is the value of its best collaboration.

   (b) *Hedge* - the average value of its collaborations is returned as fitness score.

   (c) *Pessimistic* - the value of its worst collaboration is assigned to the current individual.

3. Cooperative Coevolution for Classification

When dealing with a classification problem, the task is to build rules able to classify data. The data set is split into a set used for training the cooperative technique and a set for the testing step. In the process of evolving the rules, only information regarding data from the training set is available. After the evolution process is complete, rules are tested against the test set.

3.1 Representation

Each individual of the EA represents a simple if-then rule for a certain class. The number of genes an individual possesses equals the number of features of a sample of the data set considered for classification.

The decomposition of the classification problem into components is naturally done by assigning to each population the task of evolving rules of a certain class. Therefore the number of populations equals the number of classes the classification problem has.

3.2 The Components of the Evolutionary Algorithm behind Cooperative Coevolution for Classification

The EA used in the evolution of each species is a canonical one. Its components are further on presented.

3.2.1 Initialization of the Populations

The genes of all individuals are randomly initialized following a uniform distribution in the intervals of the corresponding attributes in the data set.

3.2.2 Fitness Evaluation

As stated before, in order to compute the fitness of an individual, collaborators (individuals) from all the other populations have to be selected; thus, complete solution(s) are created and subsequently evaluated.

In the present paper, different values for the collaboration pool size, which is denoted by n, were chosen. Therefore, in order to evaluate an individual from a certain population—that is a rule of a certain outcome—a random collaborator from each of the other populations is selected n times. Every time, the set of rules is applied to the entire training collection. Obtained accuracy represents the evaluation of the current individual. The fitness of an individual c may be given by the best of the n acquired collaboration accuracies (optimistic assignment), by the worst one of them (pessimistic assignment) or by the average of all n accuracies (hedge assignment). Algorithm 2 describes the way evaluation takes place in these cases.

Algorithm 2 Fitness evaluation of an individual c by means of either optimistic, pessimistic or hedge collaboration credit assignment

for i = 1 to n **do**
  $correct_i$ = 0;
  select a random collaborator from each population different from that of c;
  for each sample s in the training set **do**
    find the rule r from the set of all collaborators that is closest to s; found class for s = r's class;
    if found class for s = real class of s **then**
      $correct_i$ = $correct_i$ + 1;
    end if
  end for
end for
if $optimistic$ **then**
  $correct$ = $max_{i=1}^{n}(correct_i)$
else
  if $pessimistic$ **then**
    $correct$ = $min_{i=1}^{n}(correct_i)$
  else
    $correct$ = $avg_{i=1}^{n}(correct_i)$
  end if
end if
$accuracy$ = 100 * $correct$ / number of training samples;

In addition, a novel type of assignment, different from the classical cooperative coevolutionary ones, is considered (Algorithm 3): for a sample s in the training set, multiple sets of rules are formed and applied in order to predict its class. All rules within a set have different outcomes. Scores are computed for the sample s, for each of the possible outcomes in the following manner: when a set of rules is considered for a sample, a certain outcome is established for it. The score of that outcome is increased by unity. Each of the n sets of rules are applied to s. Finally, the class of s is concluded to be the class that obtains the highest score.

Algorithm 3 Score based fitness evaluation for an individual c

for each sample s in the training set **do**
  set the score for each possible outcome of s to 0;
end for
for i = 1 to n **do**
  select a random collaborator from each population different from that of c;
  for each sample s in the training set **do**
    find the rule r from the set of all collaborators that is closest to s; increase the score of r's class for s by one unit
  end for
end for
$correct$ = 0;
for each sample s in the training set **do**
  if the real class of s equals the class that had the higher score for s **then**
    s is correctly classified;
    $correct$ = $correct$ + 1;
  end if
end for
$accuracy$ = 100 * $correct$ / number of training samples;

The distance between a sample from the training set x_i = (x_{i1}, x_{i2}, ..., x_{im}) and an individual c = (c_1, c_2, ..., c_m) has to be computed in order to measure which rule is $closest$ to the current sample. The distance does not depend on the outcome and was experimentally chosen as in (1):

$$d(c, x_i) = \sum_{j=1}^{m} \frac{\mid c_j - x_{ij} \mid}{b_j - a_j} \tag{1}$$

where a_j and b_j represent the lower and upper bounds of the j-th attribute. As usually the values for the attributes belong to different intervals, the distance measure has to refer to their bounds.

In both algorithms 2 and 3, the fitness of an individual is computed as the percent of correctly classified samples from the training set (variable $correct$ in the algorithm specifies the number of samples that were successfully labelled).

In Algorithm 3, situations may appear when, for a certain sample, there exist more classes that have the same maximum score. In this case, one class has to be decided and it was considered to choose the first one in the order of outcomes. As herein all combinations of rules count in

the determination of accuracies, we might state that the new choice of assignment is closer to the classical hedge type.

In order to achieve the optimum configurations for the parameters of CCC, experiments were carried out as follows. As the *collaborator selection pressure* attribute is concerned, we used random selection, on the one hand, and, on the other hand, we employed a fitness proportional scheme. All the three types of fitness assignment presented in Algorithm 2 together with the one based on scores are tested. As for *collaboration pool size*, we varied the number of collaborators in order to find the optimum balance between accuracy and runtime.

3.2.3 Selection and Variation Operators

The selection operator discussed in this subsection refers to the EA selection for reproduction, not to the selection of collaborators. We employed proportional selection, but any other selection scheme may be successfully applied.

We used intermediate crossover (with probability p_c) – having two randomly selected parents P and Q, the value of a gene i of the offspring O is obtained according to:

$$O_i = P_i + R * (Q_i - P_i) \tag{2}$$

where R is a uniformly distributed random number over [0,1]. The obtained offspring individual replaces the worst of its two parents.

Mutation with normal perturbation was used for the experiments performed in current paper – a gene i of an individual P is changed with a probability p_m according to:

$$P_i = P_i + R * (b_i - a_i)/ms \tag{3}$$

where R is a random number with normal distribution , b_i and a_i are the upper and lower bounds of the i-th attribute in the data set and ms is the mutation strength parameter. As the intervals for the values of the attributes in the data set have different sizes, we have to refer to the size of the interval for each attribute when we perturb the values of the genes through mutation.

We can not imagine any obstacle for using any other crossover or mutation operators.

3.2.4 Stop Condition

As stop condition of the EA, a fixed number of generations for the evolutionary process was set.

3.3 Rule Interpretation

We have as many populations of individuals (rules) as many classes the classification problem encodes. The test process takes place in a similar manner to the score based evaluation of an individual during the evolution process (changes are emphasized in Algorithm 4) and computes how many samples from the corresponding set are correctly classified.

Algorithm 4 Application of rules to the test set

for each sample s in the test set **do**
  set the score for each possible outcome of s to 0;
end for
for i = 1 to n **do**
  select a random collaborator from each population;
  for each sample s in the **test** set **do**
    find the rule r from the set of all collaborators that is closest to s; increase the score of r's class for s by one;
  end for
end for
$correct$ = 0;
for each sample s in the **test** set **do**
  if the real class of s equals the class that had the higher score for s **then**
    s is correctly classified;
    $correct$ = $correct$ + 1;
  end if
end for
$accuracy$ = 100 * $correct$ / number of **test** samples;

4. Experimental Results

The widely used benchmark problem of Fisher's Iris data set [1] is used for experiments. Each sample has four attributes – width and length of the sepals and petals of the flower. There are three classes, i.e. three types of Iris flowers. There are 150 samples in the data set of which 100 were used for training and the rest for the test phase.

Random training and test sets are selected in each run of the algorithm with the aim of avoiding the chances to pick very well suited (or very unlucky) training and/or test sets and to be able to directly compare obtained accuracy with results reached by other different techniques.

Two ways of choosing collaborators are considered within the experiments: they are either randomly picked or chosen by fitness proportional selection. All the collaboration credit assignment types are used in the experiments. The value for the collaboration pool size parameter is varied from 1 up to 5.

The manually tuned parameters of the EA behind CCC are presented in Table 1. The population size parameter in the first column refers to one population only (all populations have the same cardinal). As there are three populations that coevolve, the overall number of individuals that appear in the algorithm is 450. The last parameter represents the predefined number of generations.

For each choice of the cooperative coevolutionary parameters, we performed 30 runs of the algorithm and com-

Table 1. CCC (EA) parameter values

Population size	p_c	p_m	ms	No. of generations
150	0.4	0.6	150	300

puted the average accuracy on the test set and the standard deviation, as well as the number of fitness evaluation calls and the number of times mutation and crossover took place. Results from Table 2 are approximations of the average obtained after 30 runs with one collaborator, either randomly or fitness proportionally chosen. Only very minor differences were noticed when more collaborators were considered.

Table 2. CCC descriptors independent of the collaboration parameters

Fitness evaluations	Times mutation	Times crossover
189 000	130 000	27 000

Table 3 contains the results obtained when collaborators were selected randomly. When one collaborator is considered, multiple assignments are impossible. In this case, results showed an average of 93.1% in 30 runs with a runtime of about 12 seconds and a standard deviation equal to 4.2%. In the table, n stands for the collaborator pool size parameter. Results are presented for optimistic, pessimistic and hedge credit assignment, as well as for the fitness based on scores. Avg represents the average accuracy on the test set after 30 runs and SD is the standard deviation.

There is a trend to gain accuracy with the increase in the value of the collaboration pool size parameter. Naturally, a higher number for this parameter also brings a higher value for the runtime, reaching around 1 minute for 5 collaborators. The best average accuracy was reached for 5 collaborators when a hedge credit assignment was used.

However, there is not a general rule that more collaborators lead to better results, as a very good result is reached in the case when n = 3 and the assignment based on scores is employed. The suitability of the score based fitness for the CCC algorithm is proven by the observation that the algorithm was more stable when this assignment was chosen, which is demonstrated by the fact that the smallest values for standard deviation were obtained when this type of evaluation was used.

Table 4 presents the average test accuracy obtained after 30 runs when fitness proportional selection was used for choosing the collaborators. In the first generation of the algorithm, for the first evaluation, we selected the collaborators randomly, but in the next generations the collaborators were selected upon their previous fitness values.

Table 3. Average results obtained after 30 runs for random selection of collaborators

n	Collaboration credit assignment							
	Optimistic		Pessimistic		Hedge		Score	
	Avg	SD	Avg	SD	Avg	SD	Avg	SD
2	92.9	3.8	92.5	3.9	92.6	4.8	93.3	2.5
3	93.7	3.7	93.3	3.3	93.9	3.5	94.4	2.4
4	93.4	4	93.8	3.8	93.8	3.8	93.7	2.7
5	93.5	3.9	93.9	3.1	94.7	3.2	93.4	2.6

Overall, the results obtained in this case outperform those when random selection for collaborators was used. The best result was obtained again when the hedge credit assignment was used, but this time for n = 3. The smallest values for the standard deviation were again obtained for the score based fitness.

There seems to be a curious tendency that the better results are obtained when an odd number of collaborators is used, i.e. for n = 3 or n = 5.

Table 4. Average results obtained after 30 runs for fitness proportional selection of collaborators

n	Collaboration credit assignment							
	Optimistic		Pessimistic		Hedge		Score	
	Avg	SD	Avg	SD	Avg	SD	Avg	SD
2	94	3.9	93.6	3.7	94	4.2	94.5	1.9
3	94.2	3.2	93.3	2.6	95.4	3	93.7	2.5
4	93.1	3.6	93.3	3.6	93.3	3.1	94.2	2
5	94.2	3.4	94.1	3.5	94.8	2.3	93.9	1.9

For both ways of selecting collaborators, the most appropriate credit assignments were the hedge one, as it yield the best accuracies, and the fitness based on scores, as it provided stability to the algorithm.

4.1 Comparison to Other Approaches

Other results on the same data set provided by different techniques in literature are outlined in Table 5. The four results of the table are reported in [2]; a similar way of selecting the training and test sets was used. The difference to our approach is that 80% of the samples from the Iris data were used for training and the rest for testing and that average accuracies are obtained after 500 runs. Different techniques like nearest neighbor estimators (kNN), nearest-neighbor on the Random Recursive Partitioning dissimilarity matrix (RRP), classification trees and logistic regression were employed in order to reach outlined results.

Table 5. Results obtained for Fisher's Iris data set by other algorithms

Technique	Accuracy	Std. dev.
kNN	95.63	3.3
RRP	93.47	4.2
Classification Trees	94.96	4.1
Linear Regression	96.31	3.4

When directly confronted, results obtained by CCC are comparative to the ones obtained by the other techniques on this data set, which indicates CCC as an significant tool for multi-class classification.

5 Conclusions

A classification technique based on cooperative coevolution is presented in the current paper. CCC is tested against a widely known benchmark problem and obtained results are outlined.

The best choice of attributes pertaining to CCC is discussed and experimentalized. The maximum considered value for the collaboration pool size was 5 as runtime also increases with the rise in the number of collaborators, on the one hand, and, on the other hand, too high values for this parameter eventually drive evolution to some minor decrease in accuracy. A possible explanation for this fact is that exploitation of the solutions space is put above its exploration. This affirmation relies on the fact that the EA selection is too severe and does not leave space for some weaker solutions that may vanish in the near vicinity of some promising regions. All three canonical types of collaboration credit assignment together with a novel score based fitness evaluation are tested. Results demonstrate that best classification accuracy is achieved when a hedge assignment is employed, while best stability is reached when the score based one is used. Finally, fitness proportional selection as the collaborator selection pressure outperforms a random choice of cooperations.

The evolution of more than one rule for one class, i.e. the use of a multimodal algorithm instead of a canonical EA for each population, represents a possibility to considerably improve proposed classifier and constitutes a task for the near future.

Automated tuning of the parameters of the EA might also enhance results.

References

[1] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics 7*, pages 179–188, 1936.

[2] S. Iacus and G. Porro. Missing data imputation, classification, prediction and average treatment effect estimation via random recursive partitioning. *UNIMI - Research Papers in Economics, Business, and Statistics. Statistics and Mathematics*, February 2006. http://services.bepress.com/unimi/statistics/art7/.

[3] M. A. Potter and K. A. D. Jong. A cooperative coevolutionary approach to function optimization. *Proceedings of the Third Conference on Parallel Problem Solving from Nature, Springer*, pages 249–257, 1994.

[4] R. P. Wiegand, W. C. Liles, and K. A. D. Jong. An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. *Proceedings of GECCO 2001*, pages 1235–1245, 2001.