Competitive Coevolution for Classification

Cătălin STOEAN[1], Ruxandra STOEAN[1], Mike PREUSS[2], D. DUMITRESCU[3]

[1] Faculty of Mathematics and Computer Science,
Department of Computer Science,
University of Craiova, Romania
{catalin.stoean, ruxandra.stoean}@inf.ucv.ro
[2] University of Dortmund, Dortmund, Germany
mike.preuss@cs.uni-dortmund.de
[3] Babes-Bolyai University, Cluj-Napoca, Romania
ddumitr@cs.ubbcluj.ro

Abstract. Present paper extends previous work concerned with the development of a classification system based on the coevolution metaphor. Since cooperation has proved to successfully achieve proposed task, we investigate whether competition is also a potential alternative and complement. Validation on one real-world and two benchmark data sets seems to confirm a future potential of this opposite approach.
Keywords: evolutionary computation, competitive coevolution, classification
Math. Subject Classification 2000: 68T20, 68T05, 2D10, 92D15, 92D25

1 Introduction

According to the Darwinian principles, an individual evolves through the interaction with the environment. However, a significant segment of its surroundings is, in fact, represented by other individuals. As a consequence, evolution actually implies coevolution. This interactive process may assume either collaboration towards the achievement of a specific mutual purpose, or, on the contrary, competition for the common resources in the spirit of the survival of the fittest. Accordingly, two kinds of artificial coevolutionary systems exist: cooperative and competitive, respectively. In cooperative coevolution, collaborations between two or more individuals are necessary in order to evaluate one complete potential solution, while in competitive coevolution, the evaluation of an individual is determined by a set of competitions between the current individual and several others.

Classification has been previously targeted by a cooperative system [11], [12], [13]. A potential solution was regarded as a set of IF-THEN conjunctive rules in first order logic and learning was driven by the cooperation between rules towards a complete and accurate rule set. The current work proposes to

embrace the opposed manner of addressing the task by considering the competition between rules and training samples in the direction of extensive and hard testing on each side. The aim is to observe if or how can a competitive classifier be comparable to the current successful cooperative algorithm.

The paper is organized as follows. In the next section the competitive paradigm is outlined and explained. Section three describes the proposed manner of approaching classification from the competitive point of view. Experiments on three data sets, two benchmark and one real-world, together with a comparison to the corresponding results of the cooperative approach are depicted in section four. The ideas are concluded in the last section.

2 Competitive coevolution

Within the competitive model [8], the complementary evolution between species is achieved through an inverse fitness interaction process. This implies that success attained on one side is regarded as failure among the individuals of the other side; the latter species will have to react in order to maintain its chances of survival.

Competitive coevolution represents a predator-prey complex: The strong evolutionary pressure determines the prey to defend itself better while, as a response, the predator develops better attacking strategies. This results in a stepwise adaptation and complexity of involved species. Therefore, the competitive interaction between species represents the force that drives evolution forward.

Accordingly [9], one species corresponds to certain tests a solution must satisfy and the other to the potential solutions for the given task. Competition is achieved through encounters between one individual from the tests population and one from the solution species. The two selected individuals are checked against each other and, if the solution passes the test, then the former is rewarded while the latter is penalized; if it fails, credits are assigned in a reverse manner. Moreover, each individual has a history of its encounters which embodies the penalizations/rewards it has received. The fitness of the individual is computed on this basis, as the sum of its most recent behaviours (successes/failures).

An important remark is that, since tests are a priori defined, it is only the population of potential solutions that evolves; the opposite species contains the same individuals (tests) until the end of the evolutionary process. The only fluctuation that appears within the latter population solely regards the ranking of the individuals according to fitness (their satisfiability hardness). It must be however noted that, in certain cases when tests cannot be exhaustively given, the tests population may also evolve.

Canonical competitive coevolution can be described as in Algorithm 1.

The initial evaluation of the individuals in both populations is based on the results of random encounters between solutions and tests.

Algorithm 1 A canonical competitive coevolutionary algorithm

```
t ← 0;
randomly initialize solutions population Pop_Sol(t);
create history and evaluate individuals in Pop_Sol(t);
create history and evaluate individuals in Pop_Test(t);
while termination condition = false do
  t ← t + 1;
  for i = 1, 2, ..., number of encounters do
    select solution from Pop_Sol(t − 1);
    select test from Pop_Test(t − 1);
    obtain result from encounter between solution and test;
    update history and evaluation of solution according to result;
    update history and evaluation of test according to result;
  end for
  select two solutions from Pop_Sol(t − 1);
  apply variation operators to obtain one offspring;
  evaluate offspring;
  Pop_Sol(t) ← Pop_Sol(t − 1)
  insert offspring into Pop_Sol(t);
end while
```

When such an encounter takes place, only the current individual is rewarded / penalized without the inverse score attribution for its competitor happening as well.

An evolution cycle is then entered. A predefined number of encounters between solutions and tests takes place. Those opposite individuals that meet are decided following a ranking selection. As a result, the fittest solutions and tests are more frequently involved in such "tournaments": The best performing solutions must prove their superiority more often, while, concomitantly, the algorithm focuses upon the most difficult tests. As soon as the reward/penalization is established for the two selected competitors, their corresponding history is updated: The score of the most recent encounter replaces that of the oldest one and evaluation is revised.

After the considered encounters are finished, a single offspring is created. Two parents are selected according to the same selection scheme and recombination and mutation on the resulting solution are subsequently applied. A personal history of the offspring is created through a number of encounters equal to the defined history length. The tests are again selected according to a ranking scheme. Following such an encounter, only the history of the offspring is modified; unlike a standard encounter between a solution and a test, no simultaneous penalization/reward of the involved test is conducted. This stems from the simple reason that a mediocre offspring might lead to an unreliable change in the behaviour of the considered test. After the offspring is evaluated, it will replace the weakest individual in the solutions population.

During the entire evolutionary process, the tests population suffers no variation.

The two species thus evolve together, through the inverse fitness interaction mechanism: As soon as the potential solutions satisfy certain tests, the latter receive a weaker evaluation score which leads to omission from further selection. As a result, other more difficult tests are subsequently more often selected for tournaments, while the solutions must evolve to adapt to the new requirements that must be fulfilled.

The parameters that are associated with competitive coevolution are the history length of an individual (the number of meetings that provide a measure of its performance) and the number of encounters between solutions and tests within an evolutionary cycle.

The importance of the personal history is manifold [8]. For one, it offers a continuous evaluation of an individual. Then, its partial nature leads to a major decrease in the computational expense of testing a potential solution against all the given tests, while it offers dynamics and keep of pace between the two species.

The competitive paradigm has been applied to a wide range of problems, i.e. path planning [6], constraint satisfaction [5] and classification. As classification is concerned, known techniques involve the evolution of neural networks [4], decision trees [10], cellular automata rules [2], [7] and the use of genetic programming for the problem of intertwined spirals [3]. Again, it has to be stated that to the best of our knowledge, the competitive coevolution between simple IF-THEN rules and the training set has not been achieved yet.

3 Competitive coevolution for classification

Classification can assume different characterizations, however herein it is regarded from a general point of view. Given $\{(x_i, y_i)\}_{i=1,2,...,m}$, a training set where every $x_i \in R^n$ represents a data sample (values that correspond to a sequence of attributes or indicators) and each $y_i \in \{1, 2, ..., p\}$ represents a class (outcome, decision attribute), a classification task consists in learning the optimal mapping that minimizes the discrepancy between the given classes of data sample and the actual classes produced by the learning machine. Subsequently, the learnt patterns are confronted with each of the test data samples, without an a priori knowledge of their real classes. The predicted outcome is then compared with the given class: If the two are identical for a certain sample, then the sample is considered to be correctly classified. The percentage of correctly labelled test data is reported as the classification accuracy of the constructed learning machine.

Within proposed competitive approach, the population of tests is represented by the samples in the training data, while the other population, that of solutions, will contain only the rules that are to be evolved.

3.1 Training stage. The evolutionary algorithm behind

The task is to build p rules, one for each class. Consequently, in order to form a solution to the classification problem, a complete set of rules has to be selected from the solutions population, which must therefore contain rules for every outcome.

Representation. A rule is considered to be a first logic entity in conjunctive form, i.e.:

$$\text{if } (a_1 = v_1) \wedge (a_2 = v_2) \wedge ... \wedge (a_n = v_n) \text{ then class } k$$

where $a_1, a_2, , ..., a_n$ are the attributes, $v_1, v_2, , ..., v_n$ are the values in their domain of definition and $k = 1, 2, ..., p$.

As a result, an individual has the form, i.e. $c = (c_1, c_2, ..., c_n \mid k)$, $k = 1, 2, ..., p$.

Initialization. As previously stated, at least p rules have to be obtained. Recombination will take place only between individuals with the same outcome, therefore, the size of the solutions population has to be of at least $2p$ individuals, i.e. differentiating two individuals per class. However, we only state here the minimum size of the rules population; a higher number of individuals would obviously bring a better covering of the search space.

Fitness evaluation. For each individual and for each sample from the tests population, we have to construct a *history* of the scores they obtained during encounters. The actual fitness evaluation of each individual/sample will be equal to the sum of all scores in their history.

The main question is: How are the scores given? When an encounter between a rule and a sample takes place, the distance (we used the same normalized Manhattan distance as before) between them is computed. The task for the rules is to be as similar as possible to the samples in the training set, therefore the aim is to minimize the distance between them and the samples from the training set with the same outcome. The score that is attached to a rule is given by the negative value of the distance; the maximum score a rule aims to attain is thus 0, meaning that the rule is identical to the sample it encountered.

Conversely, for a sample in the tests population, we attach the actual value of the distance between it and the rule that it met. As rules get closer to a certain pattern of samples, they will subsequently encounter other samples that have larger fitness values (and as a consequence higher chances of being selected for encounters) because they are very different from those rules. Thus, new fitter samples are continuously selected in order to adapt the rules so that they will resemble them too, i.e. evolve the solutions according to a high variety of tests.

Immediately after the initialization of the rules population, the fitness evaluations for both the rules and the samples have to be computed. In this respect, for each rule, a sample with the same outcome as its label is randomly selected and the encounter takes place: This has to be performed for a number of times equal to the history length. Each individual will thus posses a history and, as a result, an evaluation. In a similar manner, for a number of times equal to the history length, each sample from the training set will be considered and random individuals with the same outcome are selected in order to form the encounters that will complete their histories.

After the initial fitness calculation, each time an encounter takes place, solutions and samples are chosen from each population by means of ranking selection. As encounters take place only between solutions and samples with the same outcome, they will occur separately and in turn for each class (Algorithm 2). After the encounter, the new score is added to the history queue and the oldest score is removed, such that the same history length is maintained.

An individual that is obtained after the variation operators is evaluated as follows: for a number of times equal to the history length, a sample with the same outcome as its own is selected using ranking selection and encounters take place. Its fitness may now be computed by summing all the scores in the history. It is then included in the population by replacing the individual with the worst fitness evaluation.

Algorithm 2 The competitive coevolution approach to classification

```
t ← 0;
randomly initialize solutions population Pop_Sol(t);
create history and evaluate individuals in Pop_Sol(t);
create history and evaluate individuals in Pop_Test(t);
while termination condition = false do
  t ← t + 1;
  for j = 1, 2, ..., number of classes do
    for i = 1, 2, ..., number of encounters do
      select solution labelled by j from Pop_Sol(t − 1);
      select test labelled by j from Pop_Test(t − 1);
      obtain result from encounter between solution and test;
      update history and evaluation of solution with -result;
      update history and evaluation of test with +result;
    end for
    select two solutions with class j from Pop_Sol(t − 1);
    apply variation operators to obtain one offspring;
    evaluate offspring;
    insert offspring into Pop_Sol(t);
  end for
end while
```

Selection and variation operators. In our experiments we only tried the ranking scheme, as it is usually advised in the general framework of competitive coevolution.

As regards the variation operators, intermediate recombination and mutation with normal perturbation were employed. Recombination takes place only between individuals within the same class and, therefore, the offspring inherits the outcome of the parents. The mutation operator does not apply to the class gene.

We cannot imagine any obstacle for using any other recombination or mutation operators [1].

Remark: Variation and replacement take place for every class in turn (Algorithm 2).

Stop condition. Competitive techniques usually require more iterations than a canonical EA, as in each generation there is only one new descendant that enters the population. However, in our approach for classification, we apply the variation operators for individuals of every class in the same generation, a change that makes several individuals (descendants), i.e. p instances (one for each class), enter the population within that iteration.

The stop condition we used refers to a fixed number of generations.

3.2 Competitive coevolution parameters

There are two important parameters related to the competitive coevolution technique: The history length and the number of encounters that take place within one generation. The larger the values for both of them, the more accurate the fitness evaluation is for an individual/sample. Unfortunately, together with the raise in the values for either of the two, the runtime of the algorithm also increases.

The value for the number of encounters parameter directly depends on the population size of the two species: If there are many individuals in any of the populations, then a high value for the number of encounters have to be set in order to update the fitness evaluations of a great amount of the individuals.

A value that is too small for the history length parameter could make the fitness evaluation of an individual/sample change too drastically after each encounter and thus the fitness evaluation would not objectively reflect the quality of the individual/sample in contrast to the other population.

3.3 Test stage. Rules application

After the evolutionary process stops, one rule for every class is selected and these are applied to the test set. For each sample in the test set, the dissimilarity to each of the rules is computed. The found outcome of the sample is taken from the rule it resembles the most.

4 Experiments

Two data sets concerning benchmark classification problems coming from the University of California at Irvine (UCI) Repository of Machine Learning Databases[1], i.e. Wisconsin breast cancer diagnosis and iris recognition, are selected for reasons of validation and comparison. Besides, the former is a two-class instance, while the latter represents a multi-class task, which should reveal whether the classification algorithms remain flexible and feasible with some increase in the number of outcomes.

Finally, a real-world data set, courtesy of the University Hospital in Craiova, Romania, is also considered with the purpose of testing and application on an unpredictable environment that is usually associated with raw data. For all grounds mentioned above, the selection of test problems certainly contains a variety of situations that is necessary for the objective validation of proposed competitive approach in application to classification.

In all conducted experiments, for each parameter setting, the training set is formed of randomly picked samples and the test set contains the rest of the samples. In order to prove the stability of the approaches, each reported average result is obtained after 30 runs of the algorithm.

The experiments section is organized as follows: A small description of each data set is first outlined; it is then continued with the results obtained for our competitive algorithm. The section closes by undertaking a comparison to the accuracies obtained by the cooperative counterpart on the same problems.

4.1 Data sets description

The significant information on the each of the considered classification tasks is given in the following lines.

Breast cancer diagnosis. The data set contains 699 observations on nine discrete cytological factors and reflects a chronological grouping of the data. The objective is to identify whether a sample corresponds either to a benign or a malignant tumour. Class distribution is 65.5% for benign and 34.5% for malignant. There are 16 missing values for attribute 6; we replaced them by the average value for that attribute.

Iris plants classification. There are 150 samples with three possible classes pertaining to this data set. Each sample consists of four attributes which denote the length and width for petals and sepals of the iris flowers. Samples are equally distributed among classes.

[1] Available at http://www.ics.uci.edu/∼mlearn/MLRepository.html

Hepatic cancer early diagnosis. Hepatocellular carcinoma (HCC/hepatoma) is the primary malignancy (cancer) of the liver that ranks fifth in frequency among all malignancies in the world. In patients with a higher suspicion of HCC, the best method of diagnosis involves a scan of the abdomen, but only at a high cost. A cheap and effective alternative consists in detecting small or subtle increases for serum enzymes levels. Consequently, based on a set of fourteen significant serum enzymes, a group of 299 individuals and two possible outcomes (HCC and non-HCC), we aim to provide an efficient computational means of checking the consistency of decision making in the early detection of HCC at significantly low expense.

4.2 Competitive classification validation

Pre-experimental planning: Only the two benchmark data sets from the UCI repository were used for preliminary experiments. The first observation in these tests refers to the high amount of time necessary for the algorithm to run: The explanation lies in the fact that the tests population is very large and, at the beginning of the evolutionary process, all samples are evaluated. This means that for each sample, for a number of times equal to the history length, an individual is selected and an encounter takes place between the two, assigning a score to the sample.

We also noticed at this stage the importance of the two competitive coevolution parameters: history length and the number of encounters. They also significantly influence the runtime of the program that implements the algorithm. However, a more exact evaluation of an individual or sample is obtained if the history length value is large and, on the other hand, a high value for the number of encounters updates the evaluations of individuals/samples.

Task: It will be investigated if the competitive classification technique can perform as well as the cooperative approach.

Setup: The values for the parameters were manually tuned and are indicated in Table 1.

In the current experiment, in order to enhance the speed of the algorithm, we tried to reduce the population size as much as possible: Less individuals means they will have more encounters with samples from the other species and their fitness will be updated very often.

None of the considered data sets was normalized.

Results: The average results that were obtained after 30 runs by applying the competitive classification technique are illustrated in Table 2.

Observations: The average results show that the competitive approach is significantly weaker than the cooperative one (Table 2). Not only the final results prove that this approach is much poorer than the cooperative one, but there is also a great difference as concerns runtime: To make an objective comparison, we measured the average runtime for the same breast cancer data set, by using the competitive approach with the parameters indicated in Table 1.

Table 1. Parameter values for the competitive coevolution approach

	Breast Cancer	Iris	Hepatic Cancer
Evolutionary Parameters			
Population size	100	50	100
Mutation probability	0.5	0.5	0.4
Mutation strength	8	1	10
Generations	100	300	150
Competitive Coevolution Parameters			
History length	30	30	30
Number of encounters	20	30	20

Table 2. Average results after 30 runs for the competitive coevolution approach compared to its cooperative counterpart

Data set	Competition	Cooperation
Breast cancer	92.9%	94.5%
Iris	91.1%	95.4%
Hepatic cancer	84.7%	90.5%4

The obtained value was of around 480 seconds, that is almost 13 times slower than the cooperative approach with a large number of collaborators.

The standard deviation of the results have also appeared to be significantly higher, which indicates the fact that this technique is not as stable as the cooperative approach. Nevertheless, it has to be stated that in an objective judgement, there are not too many perturbations that appear in 100, or even 300 generations, since only two individuals per class recombine during one generation and mutation is applied solely to the obtained offspring. To conclude, at least 1000 generations would probably be necessary for the variation operators to considerably change the population. That would, on the other hand, slow down the program even more.

Discussion: The obtained results and the large runtime of the competitive technique indicate that it is not still a viable alternative to the previous cooperative approach. Note however that this is only the first time the competitive approach for classification is proposed and we believe that it represents a good starting point, as there is definitely potential within this technique as well. To outline some ideas for future research concerning the competitive technique for classification, perhaps a preprocessing technique step could be first applied to the training data in order to substantially reduce them. In conjunction with that (or by itself), we presume that employing a chunking technique in order to pick only small parts from the training set and use them as the static species could significantly improve runtime (maybe even the accuracy). Then, after the

rules are specialized on the selected samples, the tests species could bring new ones, while the dynamic population of rules could resume the evolution.

A important enhancement could be brought if the very good rules that are evolved at a certain point could be blocked for further modifications: Make one such individual a *tabu rule* and maybe move it in a rules archive that will be applied when the termination condition is reached. In the way the technique is now built, these good rules have the highest chances to be selected over and over again, therefore modified many times, maybe for the worse.

In the end of the evolutionary run, the best rule of each class in the final population was taken and the entire formed set was applied to the test data. Obviously, a different way of choosing the rules could be imagined, e.g. take several rules for one class or apply an archive variant as suggested above.

5 Conclusions

The presented competitive classification technique brings an interesting and dynamic perspective of targeting classification. However, the method in its current state did not prove to be as efficient as the cooperative approach, probably due to the fact that the latter has been more extensively tested in the past. Conversely, an important advantage of the competitive scenario over the cooperative alternative is that the raise in the number of classes does not target the automatic increase in the number of populations and thus complicate the evolutionary system. Further study and enhancement will complete the creation of a general coevolutionary framework that can provide new insights and successes into the demanding and crucial field of classification.

References

[1] **A. E. Eiben, J. E. Smith** :- Introduction to Evolutionary Computing. Springer-Verlag, Berlin Heidelberg New York, 2003

[2] **H. Juille, J. B. Pollack**:- Dynamics of Co-evolutionary Learning, Proceedings of the 4th International Conference on Simulation of Adaptive Behavior, 1996, 526-534

[3] **H. Juille, J. B. Pollack**:- Coevolutionary Learning: a Case Study, Proceedings of the 15th International Conference on Machine Learning, 1998, 251-259

[4] **J. Paredis**:- Steps towards Co-evolutionary Classification Neural Networks, Artificial Life IV, 1994, 102-108

[5] **J. Paredis**:- Coevolutionary Constraint Satisfaction, Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, LNCS, 866, 1994, 46-55

[6] **J. Paredis, R. Westra**:- Coevolutionary Computation for Path Planning, Proceedings EUFIT, 1, 1997, 394-398

[7] **J. Paredis**:- Coevolving Cellular Automata: Be Aware of the Red Queen! Proceedings of the 7th International Conference on Genetic Algorithms, 1997, 393-399

[8] **J. Paredis**:- Constraint Satisfaction with Coevolution, In D. Corne, M. Dorigo, F. Glover: New Ideas in Optimization, McGraw-Hill, London, 1999, 359-366

[9] **J. Paredis**:- Coevolutionary Algorithms, www.evalife.dk/ToEC2002/publications/paredis_coevo_survey_HEC.ps

[10] **E. V. Siegel**:- Competitively Evolving Decision Trees Against Fixed Training Cases for Natural Language Processing, Advances in Genetic Programming, 1994, 409-423

[11] **C. Stoean, D. Dumitrescu, M. Preuss, R. Stoean**:- Cooperative Coevolution for Classification, Bio-Inspired Computing: Theory and Applications, 2006, 289-298

[12] **C. Stoean, M. Preuss, D. Dumitrescu, R. Stoean**:- Cooperative Evolution of Rules for Classification, IEEE Post-proceedings Symbolic and Numeric Algorithms for Scientific Computing, 2006, 317-322

[13] **C. Stoean, R. Stoean, E. El-Darzi**:- Breast Cancer Diagnosis by Means of Cooperative Coevolution, 3rd ACM International Conference on Intelligent Computing and Information Systems, 2007, 493-497