Proceedings of the 6th WSEAS Int. Conf. on EVOLUTIONARY COMPUTING, Lisbon, Portugal, June 16-18, 2005 (pp65-69)

Cooperative evolutive concept learning: an empirical study

Filippo Neri
University of Piemonte Orientale
Dipartimento di Scienze e Tecnologie Avanzate
Piazza Ambrosoli 5,
15100 Alessandria AL, Italy

Abstract - An investigation of the results produced by two cooperative learning strategies exploited in the system REGAL is reported. The objective is to produce a more efficient learning system. An extensive description about how to setup suitable experiments is included. It is worthwhile to note that, in principle, these cooperative learning strategies could be applied to a pool of different learning systems.

Keywords: genetic algorithms, concept learning, cooperative learning.

I. Introduction

Concept learning [3] is the task of finding a rule (in a wide sense) that discriminates between positive and negative instances of a given concept. The relevance of concept learning is well characterized by the variety of its fielded applications like prediction of mutagenetic compounds [12], and management of computer systems and networks [13], [16]. Learning concepts means searching large hypothesis spaces. So, the capability to take advantage of effective search becomes a plus.

Approaches based on Genetic Algorithms [8], [5] proved their potentialities on a variety of concept learning tasks [1], [11], [4], [6].

From these efforts it emerged that the main disadvantage of using GAs, with respect to alternative approaches, stays in their high user waiting time and in their high computational cost. A possible way of reducing GA computational cost is to use distributed computation efficiently: possibly by promoting cooperation or competition among the simultaneous evolving populations. This approach is known as cooperative evolution or co-evolution [10], [7], [18], [15], [24].

Hillis [7] studied a host-parasite coevolutive system to develop sorting network. Other researchers exploit co-evolution to decompose complex problem into simpler subproblems at runtime, and then the evolution of several species, each one oriented to a subproblem's solution, is promoted. Periodically, a candidate solution for the problem is assembled from the species' best individuals and evaluated. Finally, the solution evaluation is backpropagated to the existing species through a new problem decomposition that affects their further evolution [10], [18], [15], [24].

In the past, we investigated how the adoption of cooperative learning into the GA-based system REGAL [15] could produce a more efficient learning system. Research on cooperative learning includes also approaches like: boosting [22] and bagging [2]. These techniques combines a pool of classifiers in order to improve their separate classification performances. Generally they exploit re-sampling or weighting of the learning instances in order to acquire different classifiers to be combined, and they are independent from the specific used learning method.

The paper organization follows. In Section 2, REGAL and two cooperative learning strategies are briefly described. In Section 3, the experimental context is analyzed. In Section 4, the results are reported. The conclusion ends the work.

II. The system REGAL

REGAL [4], [15] learns relational disjunctive concept descriptions in a restricted form of First Order Logic by using cooperative evolution. In REGAL an individual is a conjunctive formula (encoded as a fixed length bitstring) and a subset of the individuals in the populations has to be determined to form a disjunctive description for the target concept. For the scope of this work, we concentrate on REGAL's cooperative architecture as a description of the system's other components have already been published.

REGAL's architecture is a network of N processes $GALearners$, coordinated by a $Supervisor$ that imposes cooperation among the evolving populations. Metaphorically speaking, each $GALearner$ realizes a niche, defined by a subset of the learning instances, where some species lives. Each $GALearner_n$ tries to find a description for a subset of the learning instances LS_n by evolving its population. In addition, the $GALearners$ may perform migration (exchange) of individuals. The $Supervisor$ coordinates the distributed learning activity by periodically assigning different subsets of the learning instances to the $GALearners$. The composition

of these subsets depends on the specific cooperative policy used. Two policies of cooperation will be investigated.

III. Two Cooperative Learning Strategies

REGAL's results depend on the emergence of an effective cooperative behavior among its learning processes. As said before, in the system, cooperation is achieved by periodically adjusting the learning sets assigned to each $GALearner$. Thus, the cooperative learning strategy that determines the composition of these learning sets becomes the very responsible for a successful outcome. As no a priori information is available on what is a successful assignment of learning instances, we decided to develop two cooperative learning strategies based on different assumptions.

First, we analyzed the methods used by well known learning systems (like: AQ [14], C4.5 [20], FOIL [19]) to deal with a large set of instances that cannot be covered by a single conjunctive formula. They all exploit a "divide et impera" policy (also known as "learn one conjunct at a time"): learns a description, remove the instances covered by it from the learning set, and restart the learning on the remaining instances. So we decided to implement a similar policy as a cooperative learning strategy, named Let Seed Expand, that works as follows: when a learner find a description ψ, remove from its learning set all the instances covered by other already found descriptions and not covered by ψ, and let ψ improve. In some sense, this policy realizes a pool of "divide et impera" learners evolving in parallel. A drawback of the "divide et impera" approach is that it causes the learning of a number of descriptions covering few instances (the "small disjuncts" problem) that are usually not very predictive [9]. The reason for such behavior is the sharp reduction of data available for learning in the latest rounds of application of the policy.

We also defined an alternative form of cooperation, named Describe Those Still Uncovered, that forces the learners in dealing as soon as possible with the instances difficult to cover. Essentially, as soon as a promising concept description emerges, the instances not covered by it are included into all the learning sets, whereas each covered instance is inserted into only one learning set. This approach should reduce the probability that "small disjuncts" appear.

The detailed description of the cooperative learning strategies follows.

The Cooperative Learning Strategy *Let Seeds Expand*

The cooperative learning strategy Let Seeds Expand, LSE, has been explicitly designed to allow a parallel learning activity based on the "divide et impera" philosophy: remove the covered instances and learn a description for the remaining ones. Its definition follows:

$CoopLS_{LSE}$(Concept, E, C, ω, $\{LS_n\}$, N)
/* Concept is the current concept description */
/* E is the set of the available concept instances */
/* C is the set of the available non concept instances */
/* ω is the class of the concept instances */
/* $\{LS_n\}$ is the set of niches definitions */
/* N is the number of available $GALearners$ */
LS = $E \cup C$
NotCovered = E - $\cup_{\psi \in Concept}$ PosCov(ψ, LS, ω)
for n=1 *to* N
LS_n = C $\cup$ NotCovered
endfor
π-list = < sort ψ $\in$Concept by decreasing values of $\pi(\psi$, LS, $\omega)$ >
n = 1
while not empty(π-list) *do*
φ = FirstElem(π-list)
π-list = π-list - φ
LS_n = $LS_n \cup$ PosCov(φ,LS,ω)
n = (n + 1) mod N
endwhile
return($\{LS_n\}$)

The procedure $CoopLS_{LSE}$ first determines which learning instances are not covered by the current concept description Concept; these instances will be included into every new niche definition. Afterwards, the extension of one or more conjuncts in Concept is added to a generic niche definition. Roughly speaking, the $CoopLS_{LSE}$ strategy assigns to each $GALearner$ the task of extending (generalizing) the extension of a found description to include the uncovered instances.

The name "Let Seeds Expand" derives from the way the concept description appears: first, some formulas, describing subsets of the learning instances, are found and, then, their extensions grow to include the uncovered instances. Considering the extension of the found concept description, this form of cooperation favors the discovery of formulas having overlapping extensions because a number of the same learning instances appears into several niche definitions.

The Cooperative Learning Strategy *Describe Those Still Uncovered*

A different form of cooperation $CoopLS_{DTSU}$, (Describe Those Still Uncovered), has been designed to help the discovery of descriptions covering the "difficult" instances. As soon as a promising concept description appears the instances not covered can be identified as difficult ones. Thus they get included into all the learning sets to increase their probability of being covered, whereas the covered ones are inserted into only one learning set. The policy definition is:

$CoopLS_{DTSU}$(Concept, E, C, ω, $\{LS_n\}$, N)
/* Concept is the current concept description */
/* E is the set of the available concept instances */
/* C is the set of the available non concept instances */
/* ω is the class of the concept instances */
/* $\{LS_n\}$ is the set of niches' definitions */
/* N is the number of available $GALearners$ */
LS = $E \cup C$
NotCovered = E - $\cup_{\psi \in Concept}$ PosCov(ψ, LS, ω)
for n=1 *to* N
LS_n = C $\cup$ NotCovered
endfor
Assigned = $\emptyset$
π-list = < sort ψ $\in$Concept by decreasing value of $\pi(\psi$, LS, $\omega)$ >
n = 1
while not empty(π-list) *do*
φ = FirstElem(π-list)
π-list = π-list - φ
LS_n = $LS_n \cup$ $\{e | e \in$ PosCov(φ,LS,ω) and $e \notin$ Assigned)$\}$

Assigned = Assigned $\cup LS_n$
n = (n + 1) mod N
endwhile
return($\{LS_n\}$)

First, the procedure $CoopLS_{DTSU}$ includes the learning instances not covered by the current concept description into each new niche definition. After, the $CoopLS_{DTSU}$ strategy orders the formulas in the current concept description Concept according to their π value. Then, the i-th $GALearner$ get the task of learning a description covering the instances not covered by the first i-1 formulas in π-list, plus the instances not covered by Concept.

According to this policy, the learning instances covered by Concept are included into only one niche definition. Instead, those instances not covered by any formula appear in all the niche definitions. As soon as an instance is covered, the number of niches, containing it, drops to one. Considering the extensions of the found concept description, this form of cooperation biases the learning activity towards descriptions that do not cover the same instances, i.e., they tend to have almost non overlapping extensions.

IV. Empirical Qualitative Evaluation

The effectiveness of any concept learning system is primarily evaluated on the basis of its averaged prediction error estimate. However, in order to provide a closer insight in a system behavior, additional measures may be used, such as, for instance, measures accounting for the structure of the acquired concept description. The comparison of REGAL's performances in terms of its average prediction error has already been analyzed [17]. We are here interested in the qualitative evaluation of how cooperation affects the structure of the found concept descriptions. Consequently, we will study REGAL's behavior with and without a cooperative strategy at work and considering the effect of migration. Given all the previous, setting up a suitable experimental context involves dealing with the following three issues:

1) The selection of what characteristics of concept description should be measured. We chose the following ones: (a) its average prediction error (ε) evaluated on a independent set of instances; (b) its complexity (C); (c) the number of conjuncts (NC) in Concept; (d) the maximum (MXC), average (AVC) and minimum (SMC) number of positive examples covered by any conjunct in Concept; (e) and the user waiting time (T), i.e. cpu time of the slowest learners to complete its task. The complexity (C) of a concept description has been defined as the number of conditions (i.e. its number of constants) to be tested in order to verify it.

2) The selection of the learning problem. In order to be able to compare the learned concept descriptions with respect to reasonable target ones, we chose an applicative domain whose (near to) optimal concept descriptions are *a priori* known. These target concept descriptions are characterized by a null predictive error and by a low complexity value.

3) The selection of a set of operative conditions, including parameters' values, under which to run the learning system.

We now discuss issues 2) and 3) in more details.

Characteristics of the Selected Application As applicative domain, we selected a known concept learning dataset: the "Mushrooms"[1] [23] one. This problem is characterized by the absence in its hypothesis spaces of a purely conjunctive concept description and by the existence in its hypothesis spaces of at least a disjunctive concept description. The knowledge about this hypothesis space comes from results appeared in the literature.

From previous experiments, we know that the Mushrooms application admits as good description for the poisonous mushrooms concept that requires 15 conditions to be tested.

Three randomly selected sets of 4000 instances (2000 edible plus 2000 poisonous) have been used as learning sets, while the remaining 4124 instances have been used for testing.

Choosing Proper Experimental Configurations In order to run a GA-based system, a set of parameters such as the population size, the number of generations to be accomplished (in short, the generation number), the crossover probability, the mutation rate, etc. have to be fixed [5]. In general, the results obtained by any GA-based system are sensitive to the chosen values. A system is robust to the parameter variation if a little variation in its parameters values corresponds to a little shift in the "quality" of its results. We analyzed and discussed REGAL's parameter sensitivity in [4].

In this work, we used our usual parameter setting as reported in Table I. The population size and the generation number were chosen after some exploratory runs which allowed to determine a sufficiently small value. A migration rate of 0.5 means that half of one population migrates toward other GAs.

V. REGAL with or without using a cooperative strategy

The experiments reported in this section aims to study what kind of descriptions are learnt and what computational cost is involved when no cooperation or some cooperation policy is exploited. A set of basic configurations has been selected to act as a baseline. The following configurations, corresponding to the parameter settings appearing in Table I, have been considered:

CONF1 (16 GAlearners and μ = 0.0) - A basic distributed approach: 16 $GA\_Learners$, each one evolving a population of 100 individuals. No cooperative strategy to coordinate the learners. This means that every learner exploits the whole learning set.

CONF2 (16 GA learners and μ = 0.5) - As CONF1 plus migration of individuals among the $GA\_learners$.

Plus CONF1 and CONF2 exploiting one cooperative policy.

[1]The problem consists in recognizing mushrooms from the Agaricus and Lepiota families as Edible (the firsts) and Poisonous (the seconds). The dataset contains 8124 instances, 4208 of edible mushrooms and 3916 of poisonous ones. Each instance is described by a vector of 22 discrete attributes, each of which can assume from 2 to more than 6 different values. By defining a predicate for each <attribute, value> pair, the language template for this application could be coded as a bitstring of 126 bits.

TABLE I

REGAL'S CONFIGURATIONS USED IN THIS WORK.

Parameter	Value
Population size	1600
Number of GA learner	16
Crossover probability p_c	0.6
Mutation probability p_m	0.0001
Migration rate μ	0.0 or 0.5
Generation limit	200
Generation gap	0.9
Cooperation	None/LSE/DTSU

In Table II, the results obtained are reported. The leftmost column of the table shows the configuration's identifier. The other columns of the table contains the parameters already described plus the 'Cons & Compl' field that summarizes whether the learned concept description is complete and consistent on the learning set. Finally, the rows, with the value "Target", report the features of the target concept. For each configuration settings three runs have been performed. The reported error rate is an average over the three runs. Instead the other values are the real values of the description found in the experiment with the median error rate.

The experimental findings can be summarized as follows:
A) In CONF1, the maintenance of genetic diversity is mainly deferred to the *locality* of the evolution: each $GA\_Learner$ only affects the evolution of its population. When migration of individuals occurs (CONF2), genetic diversity across population tends to reduce. Thus letting individuals, describing (part of) their parents' original niches, merge and favoring the appearance of general descriptions. In turn, this biases the learning system toward the discovery of overfit concept description [21] that may decrease the classification performances as observable when passing from CONF1 to CONF2 in the experiments. In addition, migration increases the computational cost of a factor proportional to the number of exchanged individuals. This is due to the double evaluation migrating individuals are subjected to in the leaving and incoming niche. A minor point to be investigated during the system's reimplementation would be to reduce this computational overhead.

Let us evaluate now the contribution of cooperation to REGAL's performances:
1) both forms of cooperation allow to learn good concept descriptions.
2) The effect of migration of individuals is not very evident from the point of view of the error rate but a decrease in the solution complexity is observable when it is used.
3) Quite surprisingly using a cooperative strategy does not significantly increase the system running cost. The reason may be that the evolving populations tend to converge toward simple descriptions at an earlier generation than when no cooperation is present.

In summary, it seems that both cooperative policies perform reasonably well across a variety of system's configurations. Of course, additional study is needed in order to confirm or discard these latter conclusions.

TABLE II

REGAL LEARNING THE "POISONOUS MUSHROOMS" CONCEPT.

CoopLS	μ	T	C	ND	MXD	SMD	AVG	e[%]	Cons & Compl
CONF1									
None	0.0	76	7	2	1946	1139	1542	2	No
LSE	0.0	72	21	4	1946	62	946	0	Yes
DTSU	0.0	73	63	5	1946	329	1064	0	Yes
CONF2									
None	0.5	97	14	2	1946	1161	1553	4	No
LSE	0.5	103	16	3	1946	414	1089	0	Yes
DTSU	0.5	99	42	4	1946	317	1063	0	Yes
Target		-	15	3	1946	197	1096	0	Yes

VI. CONCLUSION

Investigations of two cooperative learning strategies has been reported. We believe that a distributed genetic base learner able to exploit these two cooperative strategies may acquire satisfactory concept descriptions across a range of applications. Additional experimentation, required to confirm or discard this claims, is in progress.

REFERENCES

[1] K. A. De Jong, W. M. Spears, and F. D. Gordon. Using genetic algorithms for concept learning. *Machine Learning*, 13:161–188, 1993.

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

[3] T.G. Dietterich and R.S. Michalski. A comparative review of selected methods for learning from examples. In J.G. Carbonell, R.S. Michalski, and T. Mitchell, editors, *Machine Learning, an Artificial Intelligence Approach*. Morgan Kaufmann, 1983.

[4] A. Giordana and F. Neri. Search-intensive concept induction. *Evolutionary Computation*, 3 (4):375–416, 1995.

[5] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Ma, 1989.

[6] J. Hekanaho. Background knowledge in ga-based concept learning. In *13th International Conference on Machine Learning*, pages 234–242, Bari, Italy, 1996.

[7] W. Daniel Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, volume X, pages 313–324. Addison-Wesley, Santa Fe Institute, New Mexico, USA, 1990 1992.

[8] J. H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, Mi, 1975.

[9] R. Holte, L. Acker, and B. Porter. Concept learning and the problem of small disjuncts. In *11th International Joint Conference on Artificial Intelligence*, pages 813–818, Detroit, MI, 1989.

[10] P. Husbands and F. Mill. A theoretical investigation of a parallel genetic algorithm. In *Fourth International Conference on Genetic Algorithms*, pages 264–270, Fairfax, VA, 1991. Morgan Kaufmann.

[11] C.Z. Janikow. A knowledge intensive genetic algorithm for supervised learning. *Machine Learning*, 13:198–228, 1993.

[12] R. S. King, S. Muggleton, R. A. Lewis, and M. J. E. Sternberg. Theories for mutagenecity: a study in first order and feature based induction. *Artificial Intelligence*, 74, 1995.

[13] W. Lee, S. Stolfo, and K. W. Mok. Mining audit data to build intrusion detection models. In *Knowledge discovery in databases 1998*, pages 66–72, Fairfax, VA, 1998.

[14] R. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *Fifth National Conference on Artificial Intelligence*, pages 1041–1045, Philadelphia, PA, 1986.

[15] F. Neri. *First Order Logic Concept Learning by means of a Distributed Genetic Algorithm*. PhD thesis, Department of Computer Science. University of Torino, Italy, 1997.

[16] F. Neri. Comparing local search with respect to genetic evolution to detect intrusions in computer networks. In *Congress on Evolutionary Computation 2000*, pages 512–517, IEEE Press, 2000.

[17] F. Neri and L. Saitta. Exploring the power of genetic search in learning symbolic classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-18:1135–1142, 1996.

[18] M. Potter. *The Design and Analysis of a Computational Model of Cooperative Coevolution*. PhD thesis, Department of Computer Science. George Mason University, VA, 1997.

[19] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[20] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, California, 1993.

[21] R. Quinlan. Oversearching and layered search in empirical learning. In *International Conference on Machine Learning*, Lake Tahoe, CA, 1995.

[22] R. E. Schapire. A brief introduction to boosting. pages 1401–1406, 1999.

[23] J. S. Schlimmer. Concept acquisition through representational adjustement. Technical Report TR 87-19, Dept. of Information and Computer Science, University of Californina, Irvine, CA, 1987.

[24] J. L. Shapiro. Does data-mod co-evolution improve generalization performances of evolving learners? *Lecture Notes in Computer Science*, LNCS 1498:540–549, 1998.