Combining Efforts Towards an Improved Classification: an application to Bioinformatics

Luciana Fernandes Schroeder[1], Ana Lúcia C. Bazzan[1],
Rodrigo Martínez Béjar[2]

[1] Instituto de Informática - Universidade Federal do Rio Grande do Sul
Av. Bento Gonçalves, 9500 – CP. 15064
91.501-970 Porto Alegre, RS, Brazil
{luciana, bazzan}@inf.ufrgs.br

[2] Departamento de Ingeniería de la Información y las Comunicaciones
Facultad de Informática - Universidad de Murcia
Campus de Espinardo
Espinardo 30071
Murcia Spain
rodrigo@dif.um.es

Abstract. Very few works exist on Multi-Agent systems to improve symbolic learning through knowledge exchange. The motivation of this work is to mimic human beings interaction in order to reach better solutions. This aims at supporting a recent practice in Data Mining which is the use of collaborative systems. These systems can be based on agents which interact with each other and with the environment, cooperating to solve a problem. This article proposes an architecture for such an environment which combines different symbolic Machine Learning algorithms encapsulated in agents that collaborate to improve their knowledge. We use this environment along with a method called Rescaling to improve knowledge discovery.

Keywords: Multi-Agent Systems, Data Mining, Bioinformatics, Machine Learning.

Topics: Multi-Agent Systems and Distributed AI
Machine Learning, Knowledge Discovery and Data Mining

1 Introduction

The motivation of this work is the application of a multi-agent system for improving symbolic learning through knowledge sharing. A recent practice in Data Mining is the use of collaborative multi-agent systems. These systems are usually based on agents which interact with each other and with the environment, cooperating to solve a problem.

Data Mining is a powerful technique that makes use of Machine Learning algorithms for knowledge extraction. However**,** no algorithm can be the best choice in all possible domains. Each algorithm contains an explicit or implicit bias that leads it to prefer certain generalizations over others [7]: the strong point of one can be the other's weakness. Therefore, different Machine Learning techniques applied to the same dataset hardly generate the same result [21]. For example, figure 1 shows the result of two different Machine Learning algorithms (algorithm A and B) applied to the same dataset with two distinct concepts, x and y. The A tool constructed an accurate model for concept x and a weak description for concept y. On the other hand, the B too builds a precise model for concept y and failed in the concept x description. In general, the combination of inductors increases the accuracy by reducing the bias. This integration aims at overcoming limitations of individual techniques through hybridization or fusion of various techniques. These ideas have lead to the emergence of many different kinds of system architectures.

Our aim is to evaluate the possibility of improving the classification with the MASKS (**M**ulti-**A**gent **S**ystem based on **K**nowledge **S**haring) environment, which combines inductors in a multi-agent system with autonomy to improve individual models through knowledge sharing. Besides, we describe the environment architecture and compare the first results driven from its use.

This paper describes a data mining effort concerning molecular biology data. The data was gathered from the P53 [11] database gene mutations in human tumors and cell lines. In order to improve the general accuracy, a method called Rescaling [12] was applied to the data prior the MASKS environment.

The paper is organized as follows. Section 2 describes some related work regarding Data Mining on biological data and Data Mining using multi-agent systems. Section 3 gives an overview on agents. Section 4 describes the MASKS environment architecture. Section 5 illustrates the application of this environment using a portion of P53 database. Section 6 discusses future work, and Section 7 concludes the paper.

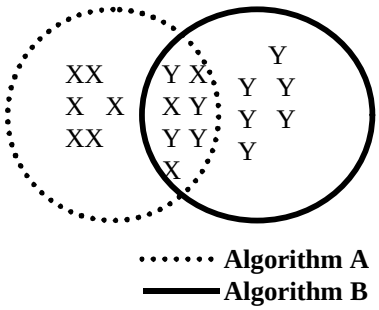

Figura 1 - Learning bias of algorithm A and B

2 Related Work

The application of Data Mining techniques in biological databases is one of the most exciting activities in modern biology because there is much unexplored knowledge in the data. There are many works describing fair good results obtained from its use in molecular biology domain. In [3], two clustering techniques (k-means algorithm and hierarchic grouping) are matched to assemble genes with similar phenotypes; in [19], the Open Reading Frames (ORF or DNA sequences which contain nucleotides marking the beginning of a gene but not its end) of a yeast are classified in diverse functional categories; in [8], a Bayesian system is used to anticipate the place where the proteins became active in the cell; in [2], a decision network is constructed to classify the proteins in soluble or not soluble; and in [15], the C4.5 algorithm is used to generate rules for automatic annotation of Keywords regarding proteins.

A recent practice in Data Mining is the use of collaborative multi-agent systems. Some examples are given in [20, 22]. JAM [20] is a multi-agent system for mining distributed data. There are two agent types - the learner and the meta-learner. The learner has a Machine Learning algorithm - each learner applies its technique separately and brings the result to be combined by the meta-learner. The CILT system [22] is based on agents with different Machine Learning algorithms that collaborate with each other to improve the classification task. Due to this collaboration, the agents generate new data and add it to the training file, which is further presented to the agents for the sake of classification improvement.

As for the use of Multi-agent systems, in [6] a prototype is described aiming at automating the annotation of a virus sequence. This work is based on Multi-agent information gathering: search, filtering, integration, analysis, and presentation of the data to the user.

3 Overview on Agents

For decades Artificial Intelligence has focused on intelligent problem-solving concerning a single entity, be it a robot or a human being, an expert system or a vehicle. However, none of theses entities can be treated in an isolated fashion. In no way they may be regarded as a single unit, complex or simple. To be able to open a door, a robot has to accomplish at least two main tasks: image processing and planning, both being closely affected by the presence of other dynamic entities in the environment. Besides functional distribution of tasks among entities, the spatial distribution of several components of the system has also been one of the main motivations for developing distributed problem-solving frameworks.

In real world applications, those entities and components may not be considered as a whole, since the complexity may increase to intractable levels. However, decomposing the tasks and ultimate goals of those entities implies that, in the majority of cases, each part has to interact with others. How helpful and efficient these interactions might be has, in fact, motivated the first researchers in the field which is now called Distributed Artificial Intelligence (DAÍ).

DAÍ is commonly associated with the term “society of agents”, meaning a network of problem-solvers where each is autonomous and has particular abilities, but cannot solve the overall problem individually due to a lack of resources, information, or expertise.

There is no widely accepted definition for agent, the term itself is related to the Latin word “agere”, or “to do”. In this work we assume that an intelligent agent is software that can take independent actions on behalf of a user’s goals, without explicit intervention by this user.

4 The Agent-Based Cooperative Learning

The MASKS environment groups different symbolic inductors encapsulated in agents in order to classify data. The model is build over a statistical analysis of a number of instances that describe the predetermined categories or concepts. If the model is considered acceptable, then it is used to classify future instances.

The MASKS environment defines the methods of interaction between the participants and the rules exchanged between the agents. The environment contains the learning problem to be solved by its members.

The environment goal is to improve the individual result and to make sure that all the learners benefit from the interaction. The cooperative agent-based environment success is measured by the improvement of the average accuracy in the knowledge base of each agent.

The environment architecture has one component called Splitter, which separates the input set in two disjoints samples: one for learning and the other for final evaluation. This component is optional.

4.1. Agent Architecture

The agent architecture consists of five components depicted in Figure 2.

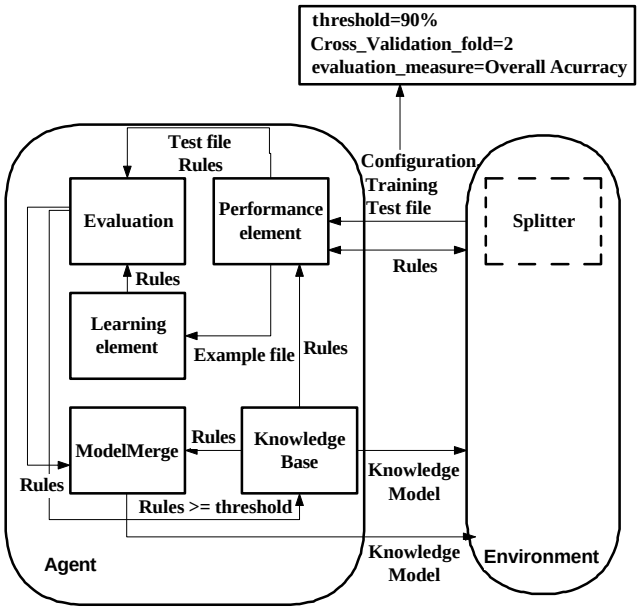

Figure 1 - Agent architecture

- Learning element: contains the rule induction technique. In the present work, the learning element contains three of these techniques: CN2 [4], C4.5 [18] and PART [10]. CN2 and PART approaches extract rules to describe the concepts contained in the data while C4.5 produces a decision tree.

- Performance element: controls, monitors and guides the learning element progress. It is responsible for the data input/output between the agent itself and between the agent and the environment.
- Knowledge Base element: stores the rules generated by the learning element approved by the Evaluation element.
- ModelMerge element: has a role similar to a knowledge base. The ModelMerge element stores other agents' best rules together with the agent's own rules.
- Evaluation element: calculates the rules reliability. This is the element that decides whether or not to store the rules produced by the Learning element or those delivered by the Performance element.

4.2. Individual Learning

As learning happens in two stages, the first one is dedicated to the individual learning. The input here is the pre-processed training set and the configuration set. After that, the agent applies its rules inductor to the examples.

The objective of the individual learning is to create an individual domain model. This model is composed of rules approved by the Evaluation component in order to achieve a compact result.

As soon as the individual learning is over, the rules created are evaluated using the test file (data that had not been used to generate the model). The Evaluation element measures the quality of each rule by executing the CN2 rule evaluation function, and stores those that are equal or better than the threshold (informed in the configuration file) to the agent knowledge base. This function estimates the rule accuracy by applying the Laplace expected error estimate (Formula 1).

LaplaceAccuracy = (TP + 1)/(TP + FP + K) (Formula 1)

The formula depends on TP (true positives which means the number of examples correctly covered by the rule), FP (false positives which means the number of examples wrongly covered by the rule) and K (the number of classes in the domain).

Most of the time, the individual learning stage produces a rule set with well described concepts along with poor described ones. This happens due to the algorithm heuristics applied to the data for extracting knowledge.

Each Machine Learning algorithm that induces symbolic classifiers makes use of a proper syntax to describe the induced model. Since in the cooperative learning stage the agents will look for better rules, it is necessary to have them transformed into the same format. The transformation process takes place after the agent applies its algorithm to the training file, prior the evaluation step.

The format adopted in this work is called PBM [17] which look like this: *if* <*condition*> *then* <*concept* = C_i>. The PBM format has a library that converts some of the most common Machine Learning symbolic algorithms to its proper format. The transformation process followed by the agents is depicted in Figure 3.

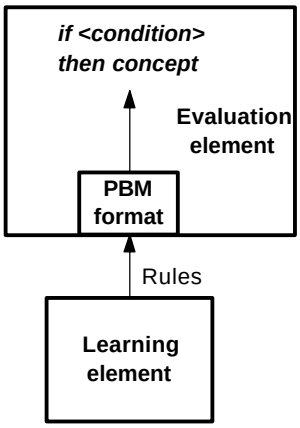

Figure 2 - Transformation process

4.3. Cooperative Learning

The goal of the cooperative learning is to improve the quality of the result. This stage input consists of the knowledge bases that store the knowledge obtained during the individual learning.

The cooperative learning consists of two further steps. During the first one, a agent queries other agents' knowledge bases. The first agent to start the interaction is the one that got the poorest overall accuracy (measured by the number of examples correctly classified by the agent's model). The agent searches for its equivalent rules with better quality. The rules that fill this requisite are added to the agent ModelMerge component. Each agent repeats this process from the poorer to the richer overall accuracy. We say that a rule is equivalent to another one when the two describe the same concept and the attributes used for them overlap. This way, a high quality rule is added to the agent's ModelMerge component either when it is similar, or overlaps, subsumes, or is in conflict with a low quality rule. For example, consider the rules R1 and R2 that describe concept C. The R1 rule contains the attribute-value test for attributes x and y, while the R2 rule includes tests for attributes x and z [22]. We than say that these two rules are related or equivalent.

When the communication ends, the rules taken from the knowledge base that were not changed are copied in the ModelMerge component. At this moment, each agent has two distinct models about the problem domain. It is necessary to evaluate the newest one which is stored in the ModelMerge component.

The agent incorporates the one (ModelMerge or Knowledge Base) that covered the highest number of instances from the test file and this becomes the final model (for that agent). The output generated by the environment is the best agent model at all.

4.4. Environment Implementation

The MASKS environment provides the definition of the parent class (Figure 4). Each Machine Learning algorithm is defined as a subclass of the parent class. The parent class provides the definitions that make the communication between the learner agents possible, and between the environment and its population of agents.

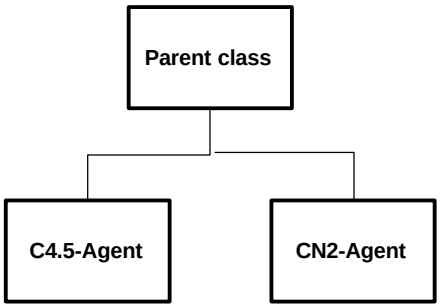

Figure 3 - Class Hierarchy of MASKS environment

After describing the approach used to formulate the agent-based cooperative learning environment, we next show an application to the bioinformatics domain.

5 Application of the Environment

This section describes the application of the environment as introduced in Section 4, to P53 database [11]. This database contains data about mutated p53 gene from cancer patients. This gene encodes a cancer-inhibiting property. The inactivation of p53 by mutation leads to the development of human cancer. The aim of the IARC p53 mutation database is to provide a tool to classify, sort, retrieve, compare and analyze these mutations in order to generate hypotheses on the natural history of human cancer.

The data collection was done in April 2002 by means of the International Agency for Research on Cancer (IARC) web site (http://www.ibiblio.org/dnam/des_p53.htm). This version (June 1997) contains 6271 records organized into 26 attributes.

The MASKS environment consisted of three agents, namely CN2-Agent, C4.5-Agent and PART-Agent. Each agent received the training set and executed the inductive learning step, followed by the evaluation against the test set. From the original database, we chose 4 types of cancer and 13 attributes that presented significance for the experiment. Therefore our data was reduced to 2080 records, divided into Test and Training set with 696 and 1384 lines respectively.

The result was that the set of rules produced by the agents was quite poor. Table 1 shows the initial rule set accuracies and sizes of each agent evaluated against the test set. The rule set produced by the CN2-Agent had the highest overall accuracy (45.8%) followed by the PART-Agent (45.68%). The C4.5-Agent got the worst results because it produced a large number of rules and a low overall accuracy (43.82%). The meager results inspired us to use the Rescaling method prior the MASKS environment.

Rescaling consists of applying algorithms in sequence where the output of an algorithm is used as input to the next. The aim would be to use the estimated probabilities determined from a learning algorithm along with the original data as input to a second algorithm (in this case it will be the MASKS environment).

In order to do it, both files (Training and Test files) were presented to the Naive Bayes algorithm [9]. This algorithm makes use of the Bayes theorem to calculate the probability of each class that appears in the training set. The model generated by the Naive Bayes was used to compute the probability of whether or not the example belongs to the class that appears as its label.

The second stage, model validation, follows the individual model construction. It consists in translating each rule into the PBM format (Table 6) and then measuring its quality against the test file. The rules produced by them had an accuracy increase comparing with the first mining. From overall accuracy around 45% (Table 1) we have got 53% (Table 4). Again the CN2-Agent had the highest overall accuracy (54%), followed by the PART-Agent (52.87%) and C4.5-Agent with 51.72%. The rule format induced by each agent is shown in table 5.

When the communication (explained in Section 3.3) begins, the agent with the poorest overall accuracy (C4.5-Agent) starts by asking its colleagues if at least one of them has rules for class "Breast carcinoma" (for instance). If yes, it asks for its quality. If the agent notices a superior quality in it, then it is necessary to check if the rules are related (explained in Section 3.3). If yes the agent adds the new rule to its ModelMerge component.

For example, Agent-CN2 rule for the "Breast carcinoma" cancer has a higher accuracy (according LaPlace depicted in Section 4.2), than the one generated by Agent-C4.5. As each agent has the goal of improving their knowledge by acquiring better rules, in the "Breast carcinoma" case, Agent-C4.5 adds the rule to its ModelMerge component. All agents repeat this process until there are no more related rules to be exchanged.

Cooperative learning thus improved the quality of the knowledge contained in the knowledge base of each agent. Table 5 shows the evaluation of the three agents before and after cooperation. The collaborative learning stage led to an improvement of each agent model. The PART-Agent received 8 rules from the others agents – 3 rules from CN2-Agent and 5 from C4.5-Agent. The CN2-Agent also received 8 rules - 4 from PART-Agent and 4 from C4.5-Agent. The last agent, C4.5 was the one to receive a higher number of rules: 6 rules from the CN2-Agent and 10 from the PART-Agent. The resultant model given by the environment was the one from CN2-Agent.

Table 1 – Overall Rule set Accuracies and Rule set sizes of each agent against the test set

CN2		C4.5		PART	
Accuracy	**Rules**	**Accuracy**	**Rules**	**Accuracy**	**Rules**
45.8%	266	43.82%	750	45.68%	104

Table 2 - Rule generated for the cancer Breast carcinoma

CN2	C4.5	PART
IF CPG_SITE = Yes AND Probabilidade_S < 0.51 AND 0.49 < Probabilidade_N < 0.50 THEN class = Breast carcinoma	Probabilidade_S <= 0.6422 | Probabilidade_S <= 0.519 | | CODON_MUT = ATG: Breast carcinoma	Probabilidade_S <= 0.519 AND CODON_MUT=ATG: Breast carcinoma

Table 3 - PBM format example of the cancer Breast carcinoma

PBM Format		
CN2	**C4.5**	**PART**
IF CPG_SITE = Yes AND Probabilidade_S < 0.51 AND 0.49 < Probabilidade_N < 0.50 THEN CLASS = Breast carcinoma	IF Probabilidade_S <= 0.6422 AND Probabilidade_S <= 0.519 AND CODON_MUT=AAA: THEN CLASS = Breast carcinoma	IF Probabilidade_S <= 0.519 AND CODON_MUT= ATG THEN CLASS = Breast carcinoma

Table 4 - Overall Rule set Accuracies and Rule set sizes of each agents against the test set with Probabilities added to it

CN2		C4.5		PART	
Accuracy	**Rules**	**Accuracy**	**Rules**	**Accuracy**	**Rules**
54%	217	51.72%	372	52.87%	149

Table 5 - Total Result

CN2		C4.5		PART	
Before Cooperation	**After cooperation**	**Before cooperation**	**After cooperation**	**Before cooperation**	**After cooperation**
54%	55%	51.72%	53.72	52.87%	53.87

6 FUTURE WORK

The present version of the MASKS environment is still under construction (communication is just simulated).

The communication among the agents will be implemented using KQML. Actually, we will develop the communication using the SACI Tool (Simple Agent Communication Infrastructure) [13] because it provides a transparent way of doing it following the KQML specification. SACI enables distributed agents to communicate in an easy way. The message content will be expressed in the SQL language once the agents' knowledge bases will be constructed using MySQL [16]. A message example is shown below.

(ask-one
:ontology ML-ontology
:language SQL
:receiver CN2-agent
:sender C4.5-agent
:reply-with q1
:content "Select rules from CN2-agent.knowledge_base where concept = "Breast carcinoma")

Initially, four agents will be part of the MASKS environment – Agent-C4.5, Agent-CN2, Agent-Ripper [5] and Agent-T2 [1]. These Machine Learning algorithms are found in the MLC++ library. The MLC++ [14] library was developed by Stanford University to facilitate the Machine Learning algorithm use.

7 CONCLUSION

The target approach - intersection of distributed artificial intelligence and machine learning – provides a promising technology to address the complexity of modern information environments.

This paper describes the architecture of the MASKS environment that consists of several learning agents to induce rules from training examples. Agents cooperate to improve their knowledge by sharing it with others to achieve better results. The main goal of this architecture is to preserve the learning bias of each Machine Learning algorithm, just improving the misleading rules by agent cooperation.

We have described an application in bioinformatics whose data was obtained from the P53 database. Although the results were poor from the knowledge discovery point of view, we were able to highly improve the accuracy by joining techniques – Rescaling method along with MASKS environment.

References

1. Auer P., Holte R. and Maass W. "Theory and applications of agnostic pac-learning with small decision trees". In *Proc. of the 12th International Machine Learning Conference*, Morgan Kaufmann, (1995).
2. Bertone, P.; Kluger, Y.; Lan, N.; Zheng, D.; Christendat, D.; Yee, A.; Edwards, A.M.; Arrowsmith, C.H.; Montelione, G.T.; Gerstein, M.B. "SPINE: An integrated tracking database and data mining approach for identifying feasible targets in high-throughput structural proteomics", Nucleic Acids Res., vol. 29, no. 13, pp. 2884-2898, (2001).
3. Clare A., King R. "Knowledge Discovery in Multi-Label Phenotype Data", In European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), (2001).
4. Clark P., Niblett T. "The CN2 Induction Algorithm". In Machine Learning Journal, 3, pp 261-283, (1989).
5. Cohen, W. W. "Fast effective rule induction". In *Proc. of the 12th International Machine Learning Conference*, p. 115:123, San Francisco, CA. Morgan Kaufman, (1995).
6. Decker, K. "A Multi-Agent System for Automated Genomic Annotation". In *Proc. of the* Int. Conf. Autonomous Agents. Montreal, (2001).
7. Dietterich, T.G. "Limitations on inductive learning" (extended abstract), (1997). ftp://ftp.cs.orst.edu/pub/tgd/papers.
8. Drawid A., Gerstein M. "A Bayesian system integrating expression data with sequence patterns for localizing proteins: compresensive application to the yeast genome", *J. Mol. Biol*., vol. 301, pp. 1059-1075, (2000).
9. Duda, Richard and Hart, Peter. "Pattern Classification and Scene Analysis". Wiley, New York, (1973).
10. Frank, Eibe and Witten, I. "Generating Accurate Rule Sets Without Global Optimization", In Shavlik, J., ed., Machine Learning: Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers, San Francisco, CA, (1998).
11. Hainaut, P et al. "IARC Database of P53 gene mutations in human tumors and cell lines: updated compilation, revised formats and new visualisation tools", *Nucleic Acids Res*., vol. 26, no. 1, pp. 205-213, (1998).
12. Henery R. "Combining Classification Procedures" in *Machine Learning and Statistics*. The Interface. Ed. Nakhaeizadeh, C. Taylor, John Wiley & Sons, Inc. (1997).
13. Hübner, J. F. & Sichman, J. S. "Saci Programming Guide", version 0.8, (2001).
14. Kohavi, R. & Sommerfield, D. "MLC++ Machine Learning library in C++", (1996), http://www.sgi.com/Technology/mlc.
15. Kretschmann, E.; Fleischmann, W.; Apweiler, R. "Automatic rule generation for protein annotation with the C4.5 data mining algorithm applied on Swiss-prot". *Bioinformatics*, vol. 17(10), p. 920:926, (2001).
16. MySQL. "The MySQL server", 2000. http://www.mysql.com.
17. Prati, R. C. Baranauskas, J. A. & Monard, M. C. "A Proposal for Unification of the Concept Representation Language to Symbolic Machine Learning Algorithms". Technical Report 137, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/RT_137.ps.zip (in Portuguese).

18. Quinlan, J.R. “C4.5: Programs for Machine Learning”, Morgan Kafmann: San Mateo/CA, (1994).
19. Ross-Macdonald, P.; Coelho, P.S.; Roemer, T.; Agarwal, S.; Kumar, A. et al. “Large-scale analysis of the yeast genome by transposon tagging and gene disruption”, *Nature*, vol. 402, pp. 413-418, (1999).
20. Stolfo, S; Prodromidis, A.; Tselepis, S.; Lee, W.; Fan, D. “JAM: Java Agents for Meta-Learning over Distributed Databases”, In David Heckerman, Heikki Mannila, Daryl Pregibon, and Ramasarny Uthurusamy, editors, *The Third International Conference on Knowledge Discovery & Data Mining*. AAAI Press, (1997).
21. Viktor, H. and H Arndt, “Combining data mining and human expertise for making decisions, sense and policies”, Journal of Systems and Information Technology, 4(2), pp.33-56.
22. Viktor, H. “The CILT multi-agent learning system”, *South African Computer Journal (SACJ)*, 24, pp.171-181, (1999).