Unsupervised Learning through Cooperative Data Mining Agents

Herna Viktor
Department of Informatics
School of Information Technology
University of Pretoria
Pretoria, South Africa
hlviktor@hakuna.up.ac.za

Heidi Arndt
Department of Informatics
School of Information Technology
University of Pretoria
Pretoria, South Africa

ABSTRACT

This paper describes a cooperative multi-agent learning system used for discovering concepts hidden in large data repositories. In this approach, two or more learning agents engage in unsupervised inductive learning to analyze the concepts embedded in the data. The agents learn from one another by accessing and internalizing one another's knowledge. Cooperative learning facilitates the transfer of knowledge between multiple agents, thus leading to the creation of a dynamically evolving knowledge repository.

Keywords: Cooperation, multi-strategy learning, unsupervised learning, knowledge-acquisition

1. INTRODUCTION

Agents and multi-agent systems are increasingly being recognized as important models for complex systems comprised of a number of sophisticated components that communicate with one another [10, 9]. The cooperation of multiple diverse agents is an important direction in multi-agent research, and has been investigated by [7, 14, 21], as well as many other research groups.

This paper describes a cooperative multi-agent system used to facilitate unsupervised machine learning from large data repositories by teams of machine learning agents. The system dynamically incorporates new machine learning agents through the use of a knowledge based communication structure. In this approach, an agent forms an initial representation of the concepts contained in the data. These concepts are expressed using a uniform rule-based language and are subsequently placed in each agent's knowledge base. However, the agents do not learn in isolation. Rather, the cooperative incentive structure of the cooperative team approach facilitates the acquisition of new knowledge, while maximizing both the individual agent's and the team's results. Agents learn through direct access to one another's knowledge repositories. Each learning agent assists the others by making its high quality results available to the team. Team member internalize the knowledge of others through a process of data generation, followed by the re-iteration of the learning process. Each agent thus acts as both learner and instructor, autonomously creating an initial knowledge base and aiding the other learners through sharing their insights.

The paper is organized as follows. Section 2 describes the cooperative agents that are used for concept discovery. The section contains a description of the agent architecture and the program followed by each agent. Section 3 concludes the paper and discusses future research directions.

2. COOPERATING AGENTS

The learning agent and multi-agent learning system concepts are based on our perception of how humans learn. Humans do not learn in isolation, but communicate with other intelligent agents (both human and computational) in an environment where they cooperate to learn. They interact in many ways and most of what they achieve is a result of interaction [4]. Weiß(1999) states that a multi-agent learning system offers a natural way to view and characterize learning [20]. A multi-agent learning system thus provides a framework for modeling the computational realization of these interactions and reflects the insight that learning, intelligence and interaction are deeply coupled.

This section introduces a cooperative multi-agent learning system that is used to discover concepts through unsupervised inductive learning. In addition, the knowledge of a human domain expert is also modeled as part of the system [11, 19]. This knowledge is acquired through the use of structured interviews and entered into the system by means of a graphical user interface (GUI).

In the cooperative agent teams approach discussed here, diverse agents fully cooperate by sharing their knowledge with one another. Hence, the agents are able to improve their behavior through the experience gained from learning from the data as well as from others. Note that an implicit assumption of our approach is that the agents are willing and able to communicate and thus act sociable [9]. Also, we assume that all agents act rationally, thus acting to improve their

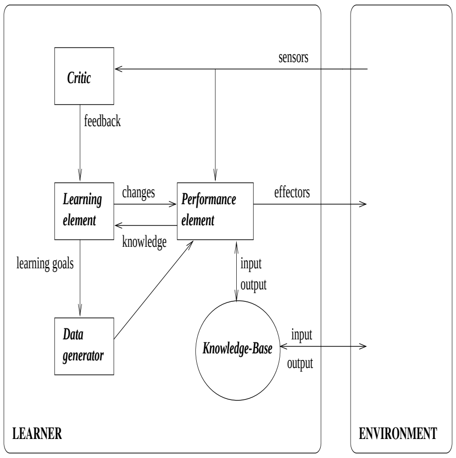

Figure 1: A learning agent architecture

own utility functions [2].

The following observation is noteworthy. Data mining concerns the extraction of knowledge from large data repositories [8]. The use of an environment where the data are shared between the agents can thus lead to unacceptably high interlearner communication costs [2]. To avoid this pitfall, each agent in our system maintains its own data set and communicates with the others through the use of a rule-based query and reporting facility, as discuss in Section 2.2.

The research presented in this paper extends the original cooperative inductive learning team (CILT) system as described in [17] and [18], where the learning task concerned supervised inductive learning.

2.1 Agent architecture

The agent architecture consists of five components, as depicted in Figure 1. The role of the five components are as follows [1].

- The learning element contains the unsupervised learning algorithm that is used to form the concept descriptions, i.e. the rules, that describe the problem domain. These concepts are formed through observation of the data repository.

  Note that any one of a large number of currently available learning algorithms can be used. This is due to the fact that the agent architecture is not restricted to a specific concept description language or machine learning algorithm. The results of an algorithm should, however, be represented in, or translatable into disjunctive normal form (DNF) rules.

  Currently, the system contains four types of agents [1, 17]. These agents' learning elements include one of the following algorithms: the CN2 rule induction program [3], the C4.5 decision tree algorithm [15] and the ANNSER rule extraction from neural network program [17]. A PA agent represents the computational modeling of a domain expert's knowledge [16].

  The supervised induction algorithms was adapted for unsupervised learning, following the approach as described by Langley (1998): Given k attributes, the algorithm executes k times, with a different attribute as so-called predicted attribute or concept class. This results in k classifiers, each designed to predict the attribute as a function of the other attributes. The attribute with the strongest and most interesting prediction are chosen as predictive attribute.

- The performance element controls, monitors and guides the learning process. It accepts training examples from the environment and presents them to the learning element or critic. It also controls the execution of knowledge base queries and initiates the different learning phases within the agent learning program.

- The critic evaluates the current knowledge of the agents, as expressed by a set of DNF rules. The critic uses the evaluation function of the CN2 rule induction program [3] to calculate three performance measures, i.e. the accuracy, interestingness and confidence. These measures are defined in Section 2.2.

- The knowledge base is used to store the current knowledge of the agents, i.e. the rules describing the concepts together with performance measures. It is used for interlearner communication, where agents share their knowledge by querying one another's knowledge bases.

  After a team of agents has made their initial commitment to cooperate, they need to determine how they will interact [6]. Communication is possible when the following two conditions hold. Firstly, the knowledge contained in the knowledge bases are represented in a uniform way, i.e. by means of DNF rules. Secondly, uniform performance measures are used to access the individual's knowledge. When adhering to these condition, an agent is able to interpret and determine the quality of the knowledge as acquired by itself or from other team members.

- The data generator is used to generate new data examples that are used to improve and focus each agent's search. The examples are based on the results obtained from communication with other team members.

Recall that the current knowledge of a human domain expert may be used to construct a *personal assistant* (PA) agent, which contains a knowledge base reflecting the current domain expertise [13]. The inclusion of the PA agent into the learning team implies that the experts' background knowledge regarding the domain are explicitly modeled as part of the multi-agent learning system. This agent thus represents the expert's current view of the problem domain. Interested readers are referred to [16] for an overview of the PA agent construction approach and [19] which describes the results obtained when applying this approach to a real-world case study.

2.2 Agent program

Prior to learning, the original data set is divided into a training set, test set and validation set. Each agent uses its own

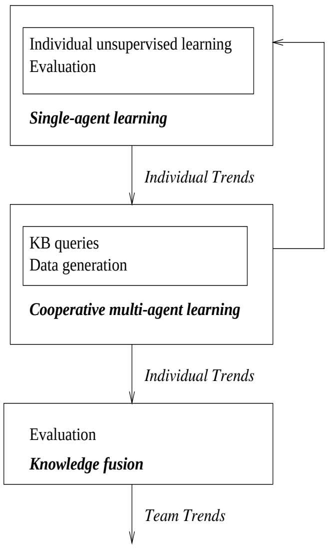

Figure 2: The three phased agents program

copy of the training set to form an initial description of the concepts contained in the data. The test set is used to evaluate the results of initial single agent learning and the cooperative learning process. The quality of the final results is evaluated using the validation set. The agent program, as depicted in Figure 2, consists of a number of episodes, completed in three phases.

During the first phase, the agents engage in single-agent learning. An initial set of concept descriptions is created from the training data set and the quality of the results is evaluated against the test set.

Cooperative multi-agent learning follows next. The participating agents form a learning team that cooperate by granting other members access to their own knowledge bases. The goal of multi-agent cooperation is twofold, i.e. to improve the individual results as well as the team results.

Thirdly, the results of cooperative learning are evaluated against the validation set, and the final individual results are fused into a team knowledge base. This knowledge repository provides a holistic view of the knowledge acquired by the team as a whole.

2.2.1 *Single-agent learning*

During single-agent learning, each agent is supplied with a training data set that is used to induce a set of concept descriptions. Figure 3 depicts the single-agent learning process. Each agent's learning element is used to form concept descriptions without interacting with the other agents. The results of individual learning are presented in DNF format. The process is controlled by the performance element, which guides the learning element through parameter adaptation.

The critic evaluates each set of concept descriptions against the training and test set to measure the agent's quantitative performance [12]. The critic calculates the overall accuracy

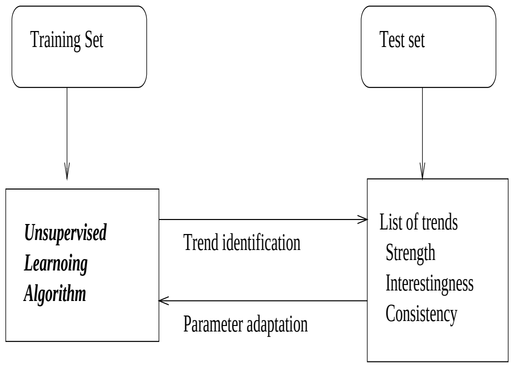

Figure 3: Initial single agent learning

of the set of concept descriptions, together with the following performance measures:

- The **accuracy** of a concept description is defined as the total number of positive examples t_{pos} covered (or matched) by the concept description divided by the total number of examples, denoted by $t_{pos} + t_{neg}$, where t_{neg} are those examples that were identified incorrectly. For example, consider a concept description R_1 which covers 14 examples, identifying 10 examples correctly and 4 examples incorrectly. The accuracy of the concept description is 71.4%. A concept description is strong when the calculated accuracy is higher than the average positive coverage as calculated over all the concept descriptions in the rule sets of the participating agents.

- The **interestingness**, which is based on the contents of a confusion matrix, denotes the positive coverage of a concept description over the different predicted concepts. For example, consider a training set that has 754 examples belonging to concept A, 66 examples of concept B and 173 examples that describe concept C. Assume that concept description R_1 covers 95 examples of concept A, two examples describing concept B and two describing concept C. The percentage coverages for each (13%, 3% and 1% respectively) varies considerably and indicates that rule R_1 is a common concept description describing concept A. This makes this concept description interesting and worthwhile to investigate further. Note that the highest percentage positive coverage must be that of the assigned class. That is, if we have two concepts A and B, with interestingness measures of 14% and 12% respectively, then A are selected. That is, the most profitable concept description is deemed interesting.

- The **consistency** of a concept description is determined by evaluating the accuracy of a concept description against a number of data sets. For example, assume that the accuracy of concept description R_1 is 96% against the training set, 100% against the test set and 93.3% on the validation set. For example, the discrepancy value between the test and validation sets is calculated as

$$|S_{ts} - S_{vs}|, \tag{1}$$

where S_{ts} and S_{vs} denote the accuracies against the test and validation sets respectively. For this example, |100 - 93.3| = 6.7. Low discrepancy values indicate a consistent concept description. This measure indicates that a concept description is not just a chance occurrence within one data set, but occurs with similar accuracy in all three data sets.

The single-agent phase is continued until the best rule set, i.e. the one with the highest overall performance measures, is selected and placed in the agent's knowledge base.

2.2.2 Cooperative learning

Cooperative learning involves the improvement of the results of the individual agents through active cooperation of all team members. This is accomplished by removing low quality rules and generating new, related high quality rules. The cooperative learning process is described next.

Firstly, each agent evaluates its own knowledge base to identify those concept descriptions with performance measures that fall below some predetermined threshold values. Usually, the threshold values are set to the average values as produced by the team. This follows from the educational principle, where an above average result is considered to be adequate [4]. These threshold values can also be reset after consultation with domain experts. Two threshold values are of importance.

- The *accuracy* threshold is set to the average accuracy of the concept descriptions identified by the team members. For example, consider a team consisting of two agents A and B. Assume that agent A produced three concept descriptions with accuracy 8%, 10% and 14% respectively. Similarly, agent B produced two concept descriptions with accuracies 10% and 12%. The accuracy threshold is thus set to 10.8%.
- The *interestingness* threshold is set to the average interestingness of the concept descriptions produced by the agents. This value is calculated using the same approach as followed when calculating the accuracy threshold.

A concept description is considered to be of a high quality if both thresholds are reached, i.e. when it is considered to be both strong and interesting. Low quality concept descriptions are removed from the agent's knowledge base.

Next, each agent *queries* the other agents to obtain related concept descriptions that are of a high quality, as depicted in Figure 4. Following Viktor (1999), related concept descriptions fall into three categories, namely concept descriptions that overlap with one another, subsume one another or correspond to one another. A low quality concept description that can be identified as a misconception or being in conflict with a high quality concept description is removed from an agent's knowledge base. The related high quality concept descriptions are placed on a NewRule list, to be used for further learning.

The data generator generates a new training set that is based on the concept descriptions in the NewRule list, as follows [18]. The data generation process uses the number of examples describing a particular concept, as found in the original training set, when producing a new set of examples. Assume that the training set contains E_1 training examples describing concept C_1. The training set contains m concepts, $C_1, \ldots, C_m$, where $m \geq 1$. *For example, consider an employee profile data set which contains the predictive attribute organization type. This attribute represents four concepts, namely private organization, technical university, science council and university. Assume that the original training set contains 500 examples concerning private organizations (C_1). The additional examples describe the other three types of organizations, as follows: 300 examples for technical universities (C_2), 200 examples concerning science councils (C_3) and 422 examples describing universities (C_4).*

Consider a concept description R_1 that describes concept C_1 and covers a number of examples $[x_1, x_i]$. Also, assume that rule R_1 covered $x_1 \subseteq E_1$ training examples for concept C_1 and x_j training examples that were not of concept C_1. These x_j examples describe one or more of the other concepts C_i, where $i \neq 1$, contained in the data set. *For example, assume a concept description covers 100 training examples. The training set contains 80 examples concerning private organization (C_1) whilst the remaining 20 examples describe the other three types of organizations, i.e. ten examples describe technical universities (C_2), five examples regard science councils (C_3) and five examples concern universities (C_4).*

The data generator produces new training examples that contain x_1 occurrences for concept C_1 and $x_k \subseteq x_j$ occurrences for each of the other concepts C_i, where rule R_1 covers x_k examples of concept C_i. The examples are added to the original training set. *The data generator thus proceed to generate 100 new examples using the information regarding the number of occurrences, i.e. 80 (C_1), ten (C_2), five (C_3) and five (C_4) respectively. These examples are added to the original training set.*

Even though the new training set is biased towards the concept description R_1, care is taken to ensure that, when constructing the new training set, the relevant knowledge in the original training set is maintained. This is achieved in two ways [18]. Firstly, the new training set contains the original training data together with the newly generated set. In this way, the data in the original data set are augmented with the new knowledge obtained from the high quality rule R_1. Secondly, the original distribution of the attribute values in the original training set are used when constructing the new training set. For Boolean and nominal valued attributes, the number of occurrences of the attribute to each concept is determined. When considering continuous valued attributes, the mean values, variances and the original range of attribute values are determined for each concept. The new training set combines this distribution information with the knowledge as contained in the rule [18].

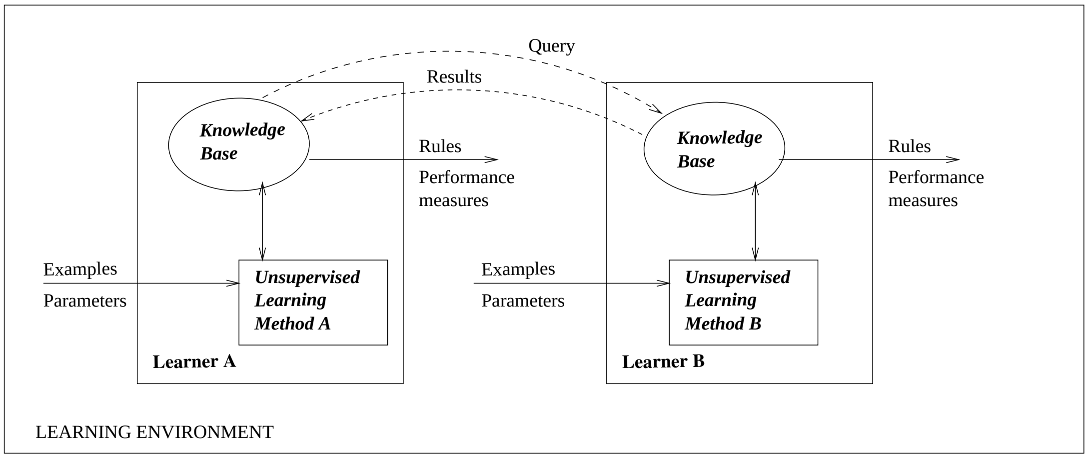

Figure 4: Two cooperating agents

Experimental results, as reported in [18] and [19], indicate that the data generation process works well when applied to a real-world data repository. However, note that the biased nature of the new data set can lead to the formation of rules that do not accurately reflect the knowledge as contained in the original data set. That is, the use of biased training examples may lead to overfitting. At the completion of cooperative learning, the final concept descriptions are evaluated against the validation set. If overfitting did occur, the results of this evaluation will be poor. The low quality of such rules will thus indicate that the current knowledge does not reflect the original data set. Such results should be discarded, since it indicates that a skew representation of the data has been formed. Here, the use of a second, independent *specialization test set* may prove worthwhile. This data set can be used to avoid overfitting during the cooperative learning phase, by independent evaluation of the rules throughout the cooperative learning process.

At the completion of the data generation process, the agent re-iterates the single-agent learning phase and proceeds to generate new concept descriptions from the newly constructed training set. All newly created high quality concept descriptions are added to the individual knowledge base. The system thus follows an incremental learning approach where each agent's knowledge base are improved by adding the high quality results of various iterations. The cooperative learning phase continues until no new rules can be found or a predetermined period of time has elapsed. Next, duplicate concept descriptions are removed. Any further simplification, for example the combination of rules, is dependent on user needs.

2.2.3 *Knowledge fusion*

The final phase involves the assessment of the resulting sets of concept descriptions against a previously unseen validation set. The performance measures are stored in each individual knowledge base. During the final stage, the knowledge bases of the agents are fused into a team knowledge base. This holistic knowledge base represents the knowledge acquired by the system as a whole.

This section described the cooperative multi-agent learning approach used to discover concept descriptions from a data repository. Note that, for illustrative purposes, the description assumed that the learning environment remained static during the three phased learning process [9]. That is, no new data sets were inserted into the learning environment. Rather, each agent *internally* generated new training examples to be used for further training. However, this approach is easily extendable to an environment in which new training data are added to the environment. Recall that the performance element of each autonomous agent monitors the environment. In this scenario, the three phased learning program should thus be re-iterated to be used for further training and the knowledge base updated accordingly.

3. CONCLUSION

Agents and multi-agent systems are recognized as holding promise for improving our understanding of technological and social interactions [5, 9]. The combination of heterogeneous learning agents into a cooperative multi-agent learning system harnesses the power of the individual agents for the benefit of the team. This paper described a multi-agent system consisting of cooperating learning agents used to find the concept descriptions within large data repositories. The approach described in this paper concerns unsupervised inductive learning used for data mining. Here, the learning agents learn to identify the concepts contained in a data repository.

An application of our approach to a real-world **employee profile** data set is described in [19]. In these experiments, a multi-agent learning team consisting of three heterogeneous agents was constructed. The team consisted of the CN2 rule induction program [3], the C4.5 decision tree generation program [15] and a PA learner representing the computational modeling of a domain expert's knowledge [16]. The initial results, as described in [19], are promising and show that the both individual agents and the team benefit from the cooperation with others. Current and future research will focus on the incorporation of other types of agents, including neural network agents, into the team.

A major advantage of our approach is the expandability thereof by the inclusion of other types of agents due to the uniform knowledge based communication scheme. However, the possible reduction of interlearner communication costs should be further explored, especially when expanding our approach to include numerous other learning algorithms. As the number of agents in a system increases, the number of potential interactions that a particular agent should consider tend to grow exponentially [2]. The investigation of communication cost reduction methods is especially important for the further extension of our approach to provide for the possibility for an agent belonging to more than one team.

4. REFERENCES

[1] H. Arndt. *A multi-agent learning system for analyzing the context embedded in qualitative data for decision making.* Submitted Dissertation, University of Pretoria, Pretoria, South Africa, 2001.

[2] C. Brooks, E. Durfee, and A. Armstrong. An introduction to congregating in multiagent systems. In *Proceedings of the International Conference on Multi-agent Systems (ICMAS-00)*, pages 79–86, Boston, USA, July 2000.

[3] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine learning*, 3, 1989.

[4] C. de Villiers. *Computer supported cooperative learning.* D Comm dissertaion, University of Pretoria, Pretoria, South Africa, 1996.

[5] E. Durfee. Preface. In *Proceedings of the International Conference on Multi-agent Systems (ICMAS-00)*, Boston, USA, July 2000. ICMAS.

[6] B. Grosz. What's needed to build team players? In *Proceedings of the International Conference on Multi-agent Systems (ICMAS-00)*, pages 3–4, Boston, USA, July 2000. ICMAS.

[7] A. Haddadi. *Communication and Cooperation in Agent Systems (LNAI Volume 1056).* Springer Verlag, Berlin: Germany, 1996.

[8] J. Han and M. Kamber. *Data mining: concepts and techniques.* Morgan Kaufmann Publishers, San Francisco, USA, 2000.

[9] M. Huhns and M. Singh. Agents and muri-agent systems: themes, approaches and challenges. In *Readings in Agents*, pages 1–23. Morgan Kaufmann Publishers, 1998.

[10] N. Jennings and M. Wooldridge. *Agent technology: Foundations, applications and markets.* Springer Verlag, Berlin, Germany, 1998.

[11] J. Kim and Gill. Deriving expectation to aid knwoledge base creation. *Twelfth workshop on knowledge acquisition, modeling and management*, 1999.

[12] P. Langley. *Elements of machine learning.* Morgan Kaufmann Publishers, California, USA, 1996.

[13] S. Laufmann. Agent software for near-term success in distributed applicatons. *Agent technologies¿ Fournations, applications and markets*, pages 49–69, 1998.

[14] H. Levesque, P. Cohen, and M. Huhns. On acting together. In *Proceedings of the Eight National Conference on Artificial Intelligence*, pages 104: 1 – 69, Boston, USA, 1990.

[15] R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, California, USA, 1994.

[16] H. L. Viktor. Combining humans and machines into a cooperative learning environment. *International Journal on Continuous Engineering and Life-long Learning*, 1999.

[17] H. L. Viktor. *Learning by cooperation: An approach to rule induction and knowledge fusion.* PhD Dissertation, University of Stellenbosch, Stellenbosch, South Africa, 1999.

[18] H. L. Viktor. Learning to cooperate: knowledge discovery through intelligent learning agents. In *Proceedings of the International Conference on Multi-agent Systems (ICMAS-00)*, pages 88–89. ICMAS, July 2000.

[19] H. L. Viktor and H. Arndt. Data mining in practice: from data to knowledge. *International Journal of Computers, Systems and Signals*, December 2000.

[20] G. Weiß. *Multi-agent systems.* MIT Press, Massachusetts, USA, 1999.

[21] M. Wooldridge and N. Jennings. Cooperative problem solving. *Journal of Logic and Computation*, 9(4):563–594, 1999.