Learning for Distributed Artificial Intelligence Systems

Michael L. Dowell and Ronald D. Bonnell

Department of Electrical & Computer Engineering
University of South Carolina, Columbia, SC 29208

1. Introduction

1.1 Single Agent Learning

Over the last four decades, machine learning's primary interest has been single agent learning. In general, single agent learning involves improving the performance or increasing the knowledge of a single agent [5]. An improvement in performance or an increase in knowledge allows the agent to solve past problems with better quality or efficiency. An increase in knowledge may also allow the agent to solve new problems. An increase in performance is not necessarily due to an increase in knowledge. It may be brought about simply by rearranging the existing knowledge or utilizing it in a different manner. In addition, new knowledge may not be employed immediately but may be accumulated for future use.

Single agent learning systems may be classified according to their underlying learning strategies. These strategies are ordered according to the amount of inferencing or the degree of knowledge transformation required by the learning system. This order also reflects the increasing amount of effort required by the learning system and the decreasing effort required by the teacher. These strategies are separated into the following six categories [10]:

1. **Rote Learning** - This strategy does not require the learning system to transform or infer knowledge. It includes learning by imitation, simple memorization and learning by being programmed. In this context, a system may simply memorize previous solutions and recall them when confronted with the same problem.

2. **Learning from Instruction** - This strategy, also called learning by being told, requires the learning system to select and transform knowledge into a usable form and then integrate it into the existing knowledge of the system. It includes learning from teachers and learning by using books, publications and other types of instruction.

3. **Learning by Deduction** - Using this strategy, the learning system derives new facts from existing information or knowledge by employing deductive inference.

These truth-preserving inferences include transforming knowledge into more effective forms and determining important new facts or consequences. Explanation-based Learning is an example of deductive learning.

4. **Learning by Analogy** - This form requires the learning system to transform and supplement its existing knowledge from one domain or problem area into new domain or problem areas.This strategy requires more inferencing by the learning system than previous strategies. Relevant knowledge must be found in the system's existing knowledge by using induction strategies. This knowledge must then be transformed or mapped to the new problem using deductive inference strategies.

5. **Learning from Examples** - This strategy, also called concept acquisition, requires the learning system to induce general class or concept descriptions from examples and counter-examples of a concept. Since the learning system does not have prior or analogous knowledge of the concept area, the amount of inferencing is greater than both learning by deduction and analogy.

6. **Learning from Observation and Discovery** - Using this strategy, the learning system must either induce class descriptions from observing the environment or manipulate the environment to acquire class descriptions or concepts. This unsupervised form of learning requires the greatest amount of inferencing among all of the different forms of learning.

1.2 Multiple Agent Learning

Distributed artificial intelligence (DAI) systems solve problems using multiple, cooperative agents. In these systems, control and information are often distributed among the agents. This reduces the complexity of each agent and allows agents to work in parallel and increases problem solving speed. In addition, a DAI system can continue to operate even if some of its agents cease to operate. This behavior allows the system to degrade gracefully in the event of failure of any of its parts. Also, each agent has resource limitations which could limit the ability of a single agent system to solve large, complex problems. Allowing multiple agents to work on these types of problems may be the only way to realistically solve them.

In general, multiple agent learning involves improving the performance of the group of agents as a whole or increasing the domain knowledge of the group. It also includes increasing communication knowledge. An increase in communication knowledge can lead to an increase in performance by allowing the agents to communicate in a more efficient manner.

In the context of improving the performance of a group of agents, allowing individual agents to improve their performance may not be enough to improve the performance of the group. To apply learning to the overall group performance, the agents need to adapt and learn to work with the each other. Indeed, the agents may not need to learn more about the domain, as in the traditional sense of machine learning, to improve group performance. In fact, to improve the performance of the group, the agents may only need to learn to work together and not necessarily improve their individual performance. In addition, not all of the agents must be able to learn or adapt to allow the group to improve.

This paper will examine the learning potential for different dimensions of distributed artificial intelligence systems. This exploratory study will be concerned with adapting and learning at the knowledge and organizational levels. Several existing systems will be examined and classified according to the dimensions for learning. This paper will not examine general dimensions for DAI, but only those dimensions that can be used for examining learning in a DAI system.

2. Classification for Multiple Agent Learning

Multiple agent learning systems may be classified according to their underlying group learning strategies. Each individual agent may still use any of the single agent learning strategies. These learning strategies can be separated into four proposed categories:

1. **Control Learning** - Learning and adapting to work with other agents involves adjusting the control of each agent's problem solving plan or agenda. Different tasks may have to be solved in a specific sequence. If the tasks are assigned to separate agents, the agents must work together to solve the tasks. Learning which agents are typically assigned different types of tasks will allow each agent to select other agents to work with on different tasks. Teams can be formed based on the type of task to be solved. Some of the issues involved are the type, immediacy and importance of task, as well as each agent's task solving ability, capability, reliability and past task assignments. Each team member's plan would be adjusted according to the other agent's plans.

2. **Organization Learning** - Learning what type of information and knowledge each agent possesses allows for an increase in performance by specifying the long term responsibilities of each agent. By assigning different agents different responsibilities, the group of agents can improve group performance by providing a global strategy [6].

3. **Communication Learning** - Learning what type of information, knowledge, reliability and capability each agent possesses allows for an increase in performance by allowing improved communication. Directly addressing the best agent for needed information or knowledge allows for more efficient communication among the agents.

4. **Group Observation and Discovery Learning** - Individual agents incorporate different information and knowledge. Combining this differing information and knowledge may assist in the process of learning new class descriptions or concepts that could not have been learned by the agents separately [11].

3. Dimensions for Multiple Agent Learning

Learning for DAI systems can be divided along two issues: learning about individual agents and learning about groups of agents. The information learned about the individual agents can be used in learning about groups. To learn about individual agents, some method of measuring a single agent's performance must be possible. Likewise, to learn about groups of agents, some method of measuring a group's performance must be possible.

3.1 Learning about Individual Agents

Learning and modeling the knowledge each agent possesses, which includes the current knowledge base and current data the agent senses from its surroundings, allows agents to directly query the correct agent instead of broadcasting a query to all agents.

Agents may be able to solve none, one, many or all types of tasks. By learning which tasks each agent can solve, task allocation becomes simpler.

Agents also have different problem solving capabilities. By learning how many tasks each agent can accomplish, over or underloading of agents can be avoided.

Depending on the tasks assigned and the agent's ability, different non-local information will be needed by different agents. Learning what non-local information each agent will need for each type of task allows an estimate of the communication and performance cost for assigning a task to an agent. Information can include partial solutions. By realizing the partial solutions necessary to solve a task, the control of different agents can be adjusted to assist in efficient planning of tasks.

Agents will also differ in the quality of their solutions and their reliability.
Important tasks can be assigned to better agents.

- Agent Knowledge
  - Knowledge Base
  - Data
- Agent Task Solving
  - Ability
  - Capability
  - Control
- Agent Non-Local   Information
- Agent Quality
  - Solution Quality
  - Reliability

Figure 1: Information about Agents

3.2 Learning about Groups of Agents

By learning each agent's knowledge and task solving ability, an organization of agents can be developed. For example, for one type of problem, a team of agents can be brought together based on the individual agent's knowledge, data and task solving ability. In addition, with the ability to measure the performance of a group of agents, different team performances could be compared to determine the best team for a specific type of problem. The performance could be measured based on solution time, solution quality or communication overhead.

In every organization, agents must communicate with other agents. Without any information about other agents, an agent must broadcast its queries. Once information about other agents' knowledge, data and task solving ability is learned, selective or on-demand communication can be used. This new protocol can help lower communication costs. In fact, if enough information is known about other agents, such as task assignments and non-local information needed, agents can anticipate other agents' needs and send them unsolicited information to further lower communication costs.

Learning the optimal amount of cooperation between agents is best done at the group level since it may be difficult for an agent to measure or estimate its affect on other agents. Measuring the performance of a group while changing the amount of cooperation for some agents within the group allows the effect of agents upon other agents to be determined.

4. Examples

4.1 The MINDS System

The MINDS(**M**ultiple **I**ntelligent **N**ode **D**ocument **S**ervers) system retrieves documents in an environment of distributed workstations [7,10]. The MINDS system learns individual user's interests and preferences plus the distribution pattern of documents across the network of workstations to improve the search sequence of the document base. The performance of the system improves as more and more documents are retrieved.

Each user has metaknowledge about the distribution of relevant documents based on keywords and users. This metaknowledge consists of a user list. For each user, a list of keyword and certainty factor pairs are stored. The certainty factor represents a measure of the possibility of finding relevant documents, characterized by the keyword, in the user's collection of documents. These certainty factors are used for ordering the search sequence for relevant documents in the document base.

In this system, the agents' organization is dynamically configured based on information received during previous document searches. The searching sequences are based on knowledge of other users' document collections.

4.2 The Learning Contract Net

The next example is an extension of Smith and Davis' Contract Net [3,4]. The Contract Net's motivation was to allow opportunistic, adaptive task allocation among a group of agents. Each agent can assume two different roles: manager and contractor. A manager monitors task execution and processes task results. A contractor executes tasks.

An agent decomposes a problem into tasks. The agent then assumes the role of manager and announces each task by broadcasting a task announcement to the entire group of agents. The other agents then evaluate these announcements in relationship to their own capabilities and submit bids on the tasks each is able to solve. The manager then selects one or several agents and informs the successful bidders through an award message. The selected agents then assume the role of contractor and execute the task. A task awarded to a contractor in this manner may also be decomposed into several tasks and subsequently the contractor becomes a manager of these new tasks.

Initially, no information is known about any of the agents in the Contract Net. However, as problems are solved, information about agents can be learned. This proposal is called the Learning Contract Net.

After a problem is decomposed into tasks and the tasks are announced, agents submit bids on the tasks each is able to solve. These bids allow the system to learn each agent's task solving ability. Each time a task is announced, a record of each agent's bid is recorded for the particular task type. In subsequent task announcements, the announcement is not broadcast to all agents but to only a small group that has responded to similar announcements in earlier problems. When new agents become active, they receive every type of announcement until their abilities are learned.

After tasks have been awarded, each agent's capability can be determined by comparing its performance and task bidding behavior to its task load. Some agents may not bid for new tasks after being awarded one; others may continue to bid for new tasks. In addition, the performance of an agent may dramatically decrease when it becomes overloaded. By recording an agent's performance compared to the type of task assigned, the ability of the agent can be measured. In addition, by recording an agent's performance compared to the number of tasks assigned, the agent's capability can be also be measured. This is similar to Shaw and Whinston's task awarding payoff system [11]. From learning each agent's task solving ability and capability, the system might eventually switch from task announcement and bidding to task assignment based on agent ability and load.

Initially, as tasks are being solved, each agent will have to broadcast queries for information to all agents. By noting which agents respond to certain queries, agents can slowly learn which agents to query for different types of information.

By learning each agent's knowledge and data along with task ability and capability, an group organization could be formed to solve specific problems. Initially, the group of agents would be organized as a decentralized market as described by Malone [8]. Once the functional ability of each agent has been learned, different types of organizations could be attempted and the performance of each measured. In this sense, the system would start with a completely decentralized organization with connections between every agent and evolve into either a centralized market, a product or functional hierarchy. The agents would start as autonomous agents and eventually move to master/slave relationships.

4.3 Shaw and Whinston

Shaw and Whinston also propose an extension to the Contract Net [11]. In their system, when an agent is awarded a task, it receives a payoff proportional to its bid. The payoff increases the successful agent's strength. Likewise, the managing agent that awarded the bid decreases its strength by the amount of the payoff. Strength records a measure of an agent's past performance. In addition, an agent's strength affects its ability to bid for tasks in the future. An agent's strength, as well as its task specialization and readiness, are used in determining its bid for a specific task. As a result, stronger agents, those having successfully completed more tasks, are increasingly favored in the bidding process.

Their system also uses genetic operators to mutate existing agents or produce new agents from two existing agents. Weaker agents are mutated or transformed to incorporate useful characteristics and capabilities from other, stronger agents. Whereas new agents are produced by combining characteristics and capabilities from two existing agents.

Each agent's characteristics and capabilities are represented in the form of chromosomes. The chromosomes are implemented using a string of 0's, meaning the agent is not capable of performing the operation, and 1's, meaning the agent is capable of performing the operation. The specific operation is denoted by its position in the chromosome. While it appears there is no increase in new knowledge, an improvement in performance is brought about by rearranging the existing knowledge and utilizing it in a different manner.

4.4 The Knos Environment

The Knos(**Kn**owledge Acquisition, Dissemination and Manipulation **O**bjects) system is an object-oriented environment that supports object migration among workstations and object adaptation and learning [12].

A Kno is an object that has a specific structure and behavior usually inherited from its class definition. The structure refers to the information a Kno contains while the behavior of a Kno is determined by the operations it can perform. An operation is composed of a set of production rules. In addition, Knos can be created and destroyed. The difference between an agent in a typical DAI system and a Kno is the fact that a Kno can move from one context to another. While more than one context can exist on a workstation, typically different contexts are associated with different workstations. For the purposes of this paper, a Kno will be considered a typical agent similar to other agents in the preceding sections.

Knos can communicate among themselves by passing messages. A message may be addressed to a specific Kno or broadcast to all Knos. A message can be deleted from the blackboard after it has been read or after a certain amount of time has elapsed, or it can remain on the blackboard indefinitely. An expiration time allows more control over communication activity.

Knos can also exist across several different contexts associated with separate workstations. These are termed *complex Knos* and can be used for coordinating tasks among different workstations.

Finally, a Kno can learn from other Knos by receiving operations from them and incorporating the new operations into its existing ones. A Kno can also forget or delete an existing operation from its existing ones. In conjunction with this learning ability, a Kno can manipulate and monitor another Kno's behavior. This allows a Kno to add a new operation to another Kno, cause the Kno to execute the new operation and then monitor the Kno. In essence, the controlling Kno can teach and observe the other Kno. It can also create an instance of itself (self-replicating) and observe the effects of adding new rules on this child Kno before deciding if it should incorporate the new rule into itself.

5. Conclusion

The most promising area of research for learning in distributed artificial intelligence systems is improving task allocation and communication. Learning each agent's task solving abilities and capabilities allows better matching between tasks and agents. Learning each agent's knowledge, data and task assignments allows lower communication costs. Also, the ability of the agents to adapt and learn to work with the other agents allows the system to maintain optimal performance even when new agents join or old agents leave the group.

6. Bibliography

1. Carbonell, Jamie G. and Langley, Pat Machine Learning Tutorial from Seventh National Conference on Artificial Intelligence, 1988.

2. Carbonell, Tom M., Michalski, Ryszard S. and Mitchell, Tom M., eds. Machine Learning: Volume 1. Palo Alto, CA: Tioga Pub. Co., 1983.

1. 3. Davis, Randall and Smith, Reid G. "Frameworks for Cooperation in Distributed Problem Solving" in Readings in Distributed Artificial Intelligence. Bond, Alan H. and Gasser, Les, eds. San Matie, CA: Morgan Kaufmann Pub., Inc., 1988.

2. 4. Davis, Randall and Smith, Reid G."Negotiation as a Metaphor for Distributed Problem Solving" in Readings in Distributed Artificial Intelligence. Bond, Alan H. and Gasser, Les, eds. San Matie, CA: Morgan Kaufmann Pub., Inc., 1988.

5. Decker, Keith S. "Distributed Problem-Solving Techniques: A Survey" in IEEE Transactions on Systems, Man,and Cybernetics, Vol. SMC-17, No. 5, September/October 1987.

6. Dietterich, Thomas G. "Learning at the Knowledge Level" in Machine Learning 1:287-315. Boston : Kluwer Academic Pub., 1986.

7. Durfee, Edmund H., Lesser, Victor R. and Corkill, Daniel D. "Coherent Cooperation Among Communicating Problem Solvers" in Readings in Distributed Artificial Intelligence. Bond, Alan H. and Gasser, Les, eds. San Matie, CA: Morgan Kaufmann Pub., Inc., 1988.

8. Huhns, Michael N., ed. Distributed Artificial Intelligence. Los Altos, CA: Morgan Kaufmann Pub., Inc., 1987.

9. Malone, Thomas W. "Modeling Coordination in Organizations and Markets" in Readings in Distributed Artificial Intelligence. Bond, Alan H. and Gasser, Les, eds. San Matie, CA: Morgan Kaufmann Pub., Inc., 1988.

10. Michalski, Ryszsard S. "Learning Strategies and Automated Knowledge Acquisition: An Overview" in Computational Models of Learning. Bolc, L., ed. Springer-Verlang, 1987.

11. Mukhopadhyay, Uttam, Stephens, Larry M., Huhns, Michael N. and Bonnell, Ronald D. "An Intelligent System for Document Retrieval in Distributed Office Environments" in Readings in Distributed Artificial Intelligence . Bond, Alan H. and Gasser, Les, eds. San Matie, CA: Morgan Kaufmann Pub., Inc., 1988.

12. Shaw, Michael J. and Whinston, Andrew B. "Learning and Adaptation in Distributed Intelligence Systems" in Distributed Artificial Intelligence: Volume II . Gasser, Les and Huhns, Michael N., eds. San Matie, CA: Morgan Kaufmann Pub., Inc., 1989.

13. Tsichritzis, D., Fiume, E., Gibbs, S. and Nierstrasz O. "KNOs: KNowledge Acquisition, Dissemination, and Manipulation Objects" in ACM Transactions on Office Information Systems, Vol. 5 No. 1, January 1987.