Revisited: Machine Intelligence in Heterogeneous Multi Agent Systems

Kaustav Jyoti Borah[1], *Department of Aerospace Engineering, Ryerson University, Canada*
email: kborah@ryerson.ca

Rajashree Talukdar[2], *B. Tech, Department of Computer Science and Engineering, SMIT, India.*

Abstract: Machine learning techniques have been widely applied for solving decision making problems. Machine learning algorithms perform better as compared to other algorithms while dealing with complex environments. The recent development in the area of neural network has enabled reinforcement learning techniques to provide the optimal policies for sophisticated and capable agents. In this paper we would like to explore some algorithms people have applied recently based on interaction of multiple agents and their components. We would like to provide a survey of reinforcement learning techniques to solve complex and real-world scenarios.

1. Introduction

An entity that recognizes its ambience with the help of sensors and uses its effectors to act upon that environment is called an agent [1]. Multi agent systems is a subfield of machine learning is used in many intelligent autonomous systems to make it smarter. To coordinate with other independent agents' behaviour in multiple agent systems environment, ML techniques aims to provide principles for construction of complex systems. When the agents are not dependent of one another in such a way that they can have approach to the environment independently. Hence, they need to embrace new circumstances. Therefore, learning and exploring about the environment demands the inclusion of a learning algorithm for each agent. Some amount of interaction becomes mandatory amongst the different agents in a multi agent system for them to act like a group. However, internet technology becomes useful for the application of software agents which gets high importance in academia and commercial institutes.

Based on the heterogeneity of agent systems, multi agent systems can be further subdivided into two categories 1) Homogeneous and 2) Heterogeneous. Homogeneous MAS include agents that all have the same characteristics and functionalities, while heterogeneous MAS include agents with variety of features that means the heterogeneous systems are composed of many subagents or sub components which are not uniform throughout. It is a distributed system contains many other hardware and software that works together in cooperative fashion to solve a critical problem. Example of Heterogeneous systems are smart grids, computer networks, Stock Market, Power systems distributions in aircraft etc.

A good advantage about multi agent learning is that, the performance of the agent enhances every moment. Multi agent learning systems is known for the exchange information's between their agents and interacting with the environment. All the algorithms in machine learning which were developed can be transferred to settings where there are multiple, interdependent, interacting learning agents [5] . However, they may need alteration to take into account about other agents in the environment [3,4]. In our paper, we focussed on different machine learning techniques. The paper organises as section 3 describes the environments in multi agent system, section 4 describes general heterogeneous system architecture, section 5 machine learning techniques followed by section 6 applications and future work and section 7 is conclusion.

2. Ongoing Research

This research is carried out during my PhD studies so far has been focused primarily on multiagent systems and machine learning.

3. Multi Agent System Environments

Agents can operate in many different types of environments. The main categories are summarised below, in mutually excluding pairs, based on the definitions provided by [30] [31]

1. *Static environments*: Based on agents action, the environment reacts.
2. *Dynamic environments*: If there is no input from an agent still the environment can change, to potentially unknown states.
3.*Fully observable environments*: At each time step, the full state of environment is available.
4.*Partially observable environments*: All the state of environment is not available, only some part will be available.
5. *Deterministic environments*: To get the next state of environment, the agent has to take actions in the current state.
6. *Stochastic environments*: Current state action can lead to another state if needed.
7. *Stationary environment*: a stationary environment does not evolve over time and has a predefined set of states.
8.. *Non-stationary environment*: a non-stationary environment evolves over time and can lead an agent to previously unencountered states.

4. General Heterogeneous Multi agent architecture

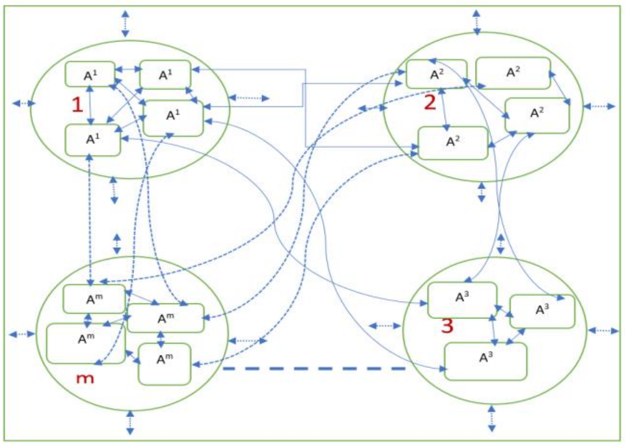

Figure 1: General Heterogeneous Architecture

The figure here shows general multi agent scenario for heterogeneous communicating agents. These agents are communicating to each other to fulfill a global goal. In the figure above, we have “m” group of heterogeneous agents where A= {A^1, A^2, … A^m} are the group of agents. Inside group 1, we assume that all agents are homogeneous. All the arrow mark with blue colours shows a communication link between the agents meaning all agents can communicate.

5. Machine learning Techniques

Artificial Intelligence has a well-known subfield under its domain which is identified widely as Multi Agent systems. It makes a prominent influence on devising and resolving extremely difficult framework problems. In multi agent systems; an agent has no clear definition and therefore as defined previously the agents are considered as entity with goals, action and domain knowledge in the environment. The agents will function as a “behaviour” and although the capability to coordinate the behaviors of autonomous agents is a recent concept yet development in the field is quick by building upon the existing work conducted earlier in the field of Distributed Artificial Intelligence (DAI) [2].

The perspective of Machine learning is mostly based on the type of the reward or return that critic can furnish as a response to learner. The three main techniques [5] that were widely discussed widely previously were 1) Supervised Learning 2) Unsupervised learning and 3) Reinforcement learning. In supervised learning, the correct result of the system is provided by the critic where as in unsupervised learning, no response is guaranteed at all. While in reinforcement learning; actor-critic techniques a newly introduced method is applied to the output of the learner which produced an ultimate return. In all three cases, the system environment or the agents themselves are assumed to provide a learning feedback.

The generality and robustness of multiagent reinforcement learning algorithms has grabbed the attention for its wide usage and these techniques have been applicable in both stationary and non-stationary environments.

5.1 Supervised Learning

The machine learning technique has the ability to learn a function that are based on input output pairs which maps an input to an output .It deduces a function from *labeled training data* consisting of a set of *training example.* Considering each example which can be expressed as a pair consisting of an input and a desired output is one of the main characteristics of Supervised learning. Algorithms based on Supervised learning examine the training data and produces an inferred function, with the help of which new examples can be mapped. The perfect class labels for critical and unknown instances can be determined by an algorithm during the optimal scenario. This can be attained when the learning algorithm generalizes from the training data to unknown situations in a "reasonable" way. There are various complexities that are encountered when multiple agents interact with each other, therefore direct application of supervised learning algorithms are difficult because they typically assume a critic that can confer the agents with the “correct” behavior [5] for a particular specific situation. Supervised learning method was used by Sniezynski [9] for the fish bank game. The interaction in heterogeneous agents’ environment are improved by Garland and Alterman [10] with the use of learning coordinated procedures. William [11] and his team delineated inductive learning methods to understand individual’s attitude’s in the area of semantic webs. A rule induction technique was applied by Gehrke and Wojtusiak [12], in their research for route planning. Learning abilities are incorporated into BDI model by Airiau et al. [13], in which decision tree learning is utilized to enhance plan applicability testing. Table 1 shows few important characteristics of above supervised learners compared to the learning aspects [5]

Table 1: Supervised learners compared to learning aspects [5]

	Cen/Decentralized	Coordination	Learning	Final Goal
Sniezynski [9]	Centralized	No	2-Agents	Selfish
Garland and Alterman [10]	De-Centralized	Yes	All- Agents	Cooperative
Williams [11]	De-Centralized	Yes	All- Agents	Cooperative
Gehrke and Wojtusiak [12]	De-Centralized	Yes	All- Agents	Cooperative
Airiau et al. [13]	De-Centralized	Yes	All- Agents	Cooperative

5.2 Unsupervised Learning

In this case, there is no explicit concepts are given. Many researchers have already reviewed the unsupervised learning algorithms applied to multi agent-based systems and conclude that, this is not a suitable technique for multi agent applications. Main approach in multi agent is to increase the overall performance, but unsupervised learning is supposed to be an aimless approach for this application. Hence minimal research has been witnessed in this area. However, unsupervised learning algorithms were used by the researchers to find a solution to the auxiliary issues that help agent through its learning [5, 14,15].

5.3 Reinforcement Learning

Reinforcement learning stands different from the other two methods mentioned above, in a way that it is a method in which the paradigm of the agent is matched exactly whereas for the supervised and unsupervised learning the learner must be provided with healthy data. Autonomous agent that has no prior knowledge behaviour of the system or of the environment is assigned by reinforcement learning by gradually enhancing its performance based on given returns as the learning task is carried out [15]. Hence, a huge percentage of people working in this area used reinforcement learning algorithms. As stated in [5], the literature available for reinforcement learning algorithms of multi-agent systems can be divided into two different subsets: 1) Estimating value functions-based methods; and 2) Stochastic search-based methods in which behaviours are learnt directly by the agents without involving value functions and it focusses on evolutionary computation [5]. Single and multiple agents’ algorithms [5] are available for learning methods based on estimate value functions. A Markov decision Process also known as MDP is known to the environment in reinforcement learning as most of reinforcement learning algorithms uses dynamic programming approach. There are many algorithms based on reinforcements learning. Comparison of various reinforcement learning algorithms are [5].

Table 2: Comparison of various Reinforcement learning algorithms [5]

Researchers	Cen/Decentralized		Coordination b/w agents	Learning	Final Goal
Bowling and Veloso [22]	D		Yes	All-Agents	Cooperative and Competitive
Barto et al. [19]	C		No	All-Agents	Selfish
Sutton [20]	C		No	All-Agents	Selfish
Moore and Atkeson [21]	C		No	All-Agents	Selfish
Greenwald and Hall [23]	D		Yes	All-Agents	Competive
Kononen [24]	D		Yes	One-Agent	Cooperative
Lagoudakis and Parr [25]	C		No	One-Agent	Selfish
McGlohon and Sen [26]	D		Yes	All-Agents	Cooperative
Qi and Sun [27]	D		Yes	All-Agents	Cooperative
Puterman [17]	C		No	All-Agents	Selfish
Watkins and Dayan [18]	C		No	All-Agents	Selfish
Bertsekas [16]	C		No	All-Agents	Selfish

Table 3: Standard algorithms for Reinforcement learning [5]

Algorithm	Description	Model	Policy	Action Space	State Space	Operator
Monte Carlo	Every visit to monte Carlo	Model Free	Off	Discrete	Discrete	Sample-Means
Q Learning	State Action reward state	Model Free	Off	Discrete	Discrete	Q-Value
SARSA	State action reward state action	Model Free	On	Discrete	Discrete	Q-Value

5.3.1 Monte-Carlo Method

Monte–Carlo Method [32] obtains value function producing the episode repeatedly and it keeps a note about the average return at each state or each state action-pair. Thus, the calculation of the state value function is carried out as follows:

$$V_{\pi}^{MC}(s) = \lim_{i\to+\infty} E\left[r^{i}(s_t)\middle|s_t = s,\pi\right] \quad (1)$$

Where $r^i(s^t)$ symbolizes observed returns at state s^t in episode i**th**. Similarly, the value function of state action pair is given as follows:

$$Q_{\pi}^{MC}(s,a) = \lim_{i \to +\infty} E\left[r^{i}(s_t,a_t)\middle| s_t = s, a_t = a, \pi\right] \qquad (2)$$

MC method is known as model free as prior knowledge about transitional probabilities is not mandatory in this method.

However, for convergence to take places this method is based on two vital assumptions [32]

1) it has a large number of episodes and

2) visiting every state and every action should be carried out for a large number of times. Making this ‘’exploration’’ a feasible one, usage of e-greedy strategy in policy improvement should be made:

$$R_S: \pi \to \pi' = \psi'(s) = \left\{ \begin{array}{l} 1 - \epsilon + \frac{\epsilon}{|\Delta_{\Pi}(s)|}, a_i = a_j \Lambda j = argmax\ Q_{\Pi}(s.a_k) \\ \\ \frac{\epsilon}{|\Delta_{\Pi}(s)|},\ \forall a_i \in \Delta_{\pi} \wedge a_i \neq a_j \end{array} \right\} \qquad (3)$$

Where $|\Delta_{\pi}(s)|$ depicts number of candidate action taken in state s and $0<\epsilon<1$. Usually the division of MC algorithm are made into two groups: on-policy and off-policy. In on-policy method application of policy π is carried out for both evaluation and exploration and purpose. Therefore, the policy π should be stochastic or soft. But off-policy uses different policy $\pi' \neq \pi$ in order to produce the episodes and hence π can be deterministic. Off-policy method is more attractive due to its simplicity, but the stability of on-policy method is more when dealing with continuous state space problems and when applying it together with function approximator such as neural networks.

5.3.2 Temporal Difference Method

TD (Temporal Difference) is a model free approach that learns from its experiences[32]. This method in order to make an update doesn’t wait for the episode to get over. Every step is updated within the episode by leveraging 1-step Bellman equation and therefore a faster convergence is possibly feasible for this method:

$$U_1: V^{i}(s_t) \leftarrow \mathrm{a}V^{i-1}(s_t) + (1-\alpha)(r_{i+1} + \gamma V^{i-1}(s_{t+1}) \qquad (4)$$

Where α is a step size parameter and 0< α <1. It makes use of previously calculated values V^{i-1} for updating of the current one’s V^{i}, which is known as bootstrapping method, which has the advantage of learning fast over the the non-bootstrapping methods in most of the cases. TD learning is also divided into two major categories: On- policy TD learning (Sarsa) and Off-policy TD learning(Q-learning). In Sarsa algorithm estimates value function of state action pair based on:

$$U_2 : Q^i(s_i, a_t) \leftarrow \alpha Q^{i-1}(s_{t,}a_t) + (1-\alpha)(r_{t+1} + \gamma Q^{i-1}(s_{t+1}, a_{t+1}) \qquad (5)$$

On the other hand, single step optimality bellman equation is used by Q-learning to carry out the update, i.e., direct approximation of value function of optimal policy is done by Q-learning:

$$U_3 : Q^i(s_t, a_t) \leftarrow \alpha Q^{i-1}\left(s_{t,}a_t\right) + (1-\alpha)(r_{t+1} + \gamma \max_{a_{t+1}^j} Q^{i-1}\,(s_{i+1}, a_{t+1}^j)\,) \qquad (6)$$

It can be noticed that the operator max in update rule substitutes for a deterministic policy and this strongly explains why Q-learning is off-policy.

Tabular structure is used by these two methods in order to store the value function of each state or each function pair. However, for solving complicated problems which consists of a number of stages it becomes insufficient due to lack of memory. Therefore, actor-critic (AC) method is introduced to overcome these limitations. AC includes two memory structure for an agent [32]: actor structure is utilized for selection of appropriate action in accordance to the observed state and transfer the same to critic structure for evaluation. Critic structure uses the following equation as a TD error to decide the future tendency of a selected action.

$$\delta(a_t) = \beta\left(r_{t+1} + \gamma V_{(s_{t+1})}\right) - (1-\beta)V_{(S_t)} \qquad (7)$$

5.3.3 Q learning

In order to learn agent systems or sub-systems, Q learning is widely used reinforcement learning based approach. The final objective of a Q learning is to evaluate a state of action that is so called policy that results in maximum utility for the agent. After one action is executed there will be reward for each policy that is generated. In simple word, the environment is explored in which a reward is observed by experimenting the different actions such that the agent learns in the process. In a finite markov decision process, an optimal action selection can be identified by Q learning provided indefinite time for exploration is given and a party random policy. The result of Q-learning, i.e., the policy, may be observed as a table in which a numeric value is assigned to each state–action pair (s, a), that gives an estimate of the (possibly long-term) reward to be received when implementing a in s. After receiving a reward, an agent performs the required numerical value updating of the state–action pair, based on the reward and on the evaluated best reward to be obtained in the new state. Now with time, the agent is capable of enhancing its estimates of the rewards to be obtained for all state–action pairs. The reader can find the Q learning algorithm in more details [7,8]

There are some advantages of Q-learning which includes 1) Absolute convergence toward the optimum 2) Natural applicability to agent systems because of the coupling of learning and exploration. However, it has drawbacks that includes 1) Designing numerical suitable reward function can be a nontrivial task. 2) Convergence to optimal and 3) No explanation is given for any action preferences in learning the result.

6. Applications and Future work

One major application in this area is development of an algorithm which can perform very well in airport ground handling management system. In this communication the ground handling fleet management problem [28] at airports is taken into consideration with the objective of enhancing aircraft service at arrival and departure terminals while we consider the ground cost issues. After collaborative discussion between ground management, airport authorities and airline, a machine learning technique will address to solve this multi agent-based problem.

Another example involves prediction in the stock market. Portfolio management in the stock trading [29] results in a successful handling and presents a theory-based foundation for a stock trading system. The overall portfolio management tasks include extracting the user profiles, collecting information on initial portfolio position of user, on behalf of the user it observes the environment and suggesting better decisions, so the investment goals of the user meets the requirements , and making decision suggestions to meet the investment goals of the user. Based on the requirement analysis, Davis [29] presented a framework for a Multi-Agent System for Stock Trading (MASST). The primary issues addressed , it provides with different information sources and by interacting the agents and providing decision-making for investors in the stock market. The candidate agents are identified and also the tasks that the agents perform. Agent interaction and exchange of information and knowledge between agent also has been described.

7. Conclusion

This research addressed some machine learning techniques for agents and multi agent systems [5]. Furthermore, we also have provided two examples for reader as an application of multi agent-based systems that are being developed at Ryerson University, Toronto, Canada. While machine learning methods applied to multi agent systems still require more attention to prove their practicality. ML for multi agents is still a relatively new research area, and there are lot of open issues that require further development, some of them have already been mentioned.

Acknowledgement

I would like to thank my wife Priyanka Talukdar, research scholar, department of Civil Engineering of IIT- Guwahati (India) for her valuable suggestions in shaping this paper. This survey was funded by NSERC (Natural Sciences and Engineering Research Council) Canada and my supervisor in Ryerson University, Canada.

References

[1] Shoham, Y., and Leyton-Brown, K. *Multiagent Systems Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press. 2009.

[2] Peter Stone and Manuela Veloso. “Multiagent Systems: A Survey from a Machine Learning Perspective”, *Autonomous Robotics* volume 8, number 3. July 2008.

[3] Yang, Z., and Shi, X. “An agent-based immune evolutionary learning algorithm and its application”, In *Proceedings of the Intelligent Control and Automation (WCICA)*. 5008-5013. 2014.

[4] Qu, S., Jian, R., Chu, T., Wang, J., and Tan, T. “Computational Reasoning and Learning for Smart Manufacturing Under Realistic Conditions”, In *Proceedings of the Behavior, Economic and Social Computing (BESC) Conferences*, 1-8. 2014.

[5] Khaled M. Khalil, Mohamed Abdelaziz , Taymour T. Nazmy and Abdel- Badeeh M. Salem, “Machine Learning Algorithms for multi Agent systems” Proceedings of the International conference on Intelligent Information processing, Security and Advanced Communication- IPAC’15 2015.

[6] Yoad Lewenberg, 2017 “Machine Learning Techniques for Multiagent Systems” Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17) pp. 5185-5186, 2017.

[7] Mitchell, T. *Machine Learning*. New York: McGraw-Hill, 1997.

[8] Kaebling, L.P., Littman, M.L., & Moore, A.W. “Reinforcement learning: a survey”, *Journal of Artificial Intelligence Research, 4*. 1996.

[9] Sniezynski, B. “Supervised Rule Learning and Reinforcement Learning in A Multi-Agent System for the Fish Banks Game”, *Theory and Novel Applications of Machine Learning*. 2009.

[10] Garland, A., and Alterman, A. “Autonomous agents that learn to better coordinate”, *Autonomous Agents and Multi-Agent Systems* 8, 267-301. 2004.

[11] Williams, A. “Learning to share meaning in a multi-agent system”, *Autonomous Agents and Multi-Agent Systems* 8, 165-193. 2004.

[12] Gehrke, J. D., and Wojtusiak, J. “Traffic Prediction for Agent Route Planning”, In *Proceedings of the International Conference on Computational Science*, 692-701. 2008.

[13] Airiau, S., Padham, L., Sardina, S., and Sen, S. “Incorporating Learning in BDI Agents”, In *Adaptive Learning Agents and Multi- Agent Systems Workshop (ALAMAS+ALAg-08)*, 2008.

[14] Kiselev, A. A self-organizing multi-agent system for online unsupervised learning in complex dynamic environments. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 1808-1809, 2008.

[15] Sadeghlou, M., Akbarzadeh-T, M. R., and Naghibi-S, M. B. “Dynamic agent-based reward shaping for multi-agent systems”, In *Proceedings of the Iraniance Conference on Intelligent Systems (ICIS)*, 1-6, 2014.

[16] Bertsekas, D. P. *Dynamic Programming and Optimal control*, 2nd Ed., Athena Scientific, 2001.

[17] Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st Ed., Wiley, 2008.

[18] Watkins, C. J. C. H., and Dayan, P. Q-learning. *Machine Learning* 8, 279-292, 1992.

[19] Barto, A. G., Sutton, R. S., and Anderson, C. W. "Neuronlike adaptive elements that can solve difficult learning control problems", *IEEE Transactions on Systems, Man, and Cybernetics* 5, 843-846, 1983, 1992.

[20] Sutton, R. S. "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming", In *Proceedings of the Seventh International Conference on Machin Learning (ICML-90)*, Austin, US, 216-224, 1990.

[21] Moore, A. W., and Atkeson, C. G. "Prioritized sweeping: Reinforcement learning with less data and less time", *Machine Learning* 13, 103-130, 1993.

[22] Bowling, M., and Veloso, M. "Multiagent learning using a variable learning rate", *Artificial Intelligence* 136, 215-250, 2002.

[23] Greenwald, A., and Hall, K. "Correlated-Q learning", In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-03)*, Washington, US, 242-249, 2003.

[24] Kononen, V. "Gradient descent for symmetric and asymmetric multiagent reinforcement learning", *Web Intelligence and Agent Systems* 3, 17-30, 2005.

[25] Lagoudakis, M. G., and Parr, R. "Least-squares policy iteration", *Machine Learning Research* 4, 1107-1149, 2003.

[26] McGlohon, M., and Sen, S. "Learning to Cooperate in Multi- Agent Systems by Combining Q-Learning and Evolutionary Strategy", *In Proceedings of the World Conference on Lateral Computing*, 2004.

[27] Qi, D., and Sun, R. "A multi-agent system integrating reinforcement learning, bidding and genetic algorithms", *Web Intelligence and Agent Systems* 1, 187-202, 2003.

[28] Salma Fitouri Trabelsi, carlos Alberto Nunes Cosenza, Luis Gustavo Zelaya Cruz, Felix Mora-Camino "AN operational approach for ground handling management at airports with imperfect information" 19th International Conference on Industrial Engineering and Operations Management, Jul, Valladolid, Spain, 2013.

[29] Darryl Davis, Yuan Luo and Kecheng Liu, "A Multi-Agent Framework for Stock Trading" School of Computing, Staffordshire University, Stafford ST18 0DG, UK, Department of Computer Science, University of Hull, HU6 7RX, UK 2000/8

[30] Andrei Marinescu " Prediction-Based Multi-Agent Reinforcement Learning for Inherently Non-Stationary Environments" PhD thesis, Computer Science, University of Dublin, Trinity College, 2016.

[31] Russell, S. and Norvig, P. Artificial Intelligence: A Modern Approach. Prentice Hall, 2003.

[32] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi, " Deep Reinforcement Learning for Multi-Agent Systems: A Review of Challenges, Solutions and Applications" retrieved from arXiv:1812.11794v2 [cs.LG] 6 Feb 2019.