2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Mutual Learning: Part II --Reinforcement Learning

Kumpati S. Narendra
Center for Systems Science
Yale University
New Haven, CT 06521, USA
kumpati.narendra@yale.edu

Snehasis Mukhopadhyay
Computer and Information Science
Indiana University Purdue University Indianapolis
Indianapolis, IN 46202, USA
smukhopa@iupui.edu

***Abstract*—The concept of "Mutual Learning" was introduced by the authors in Part I of this paper which was presented at the 2019 ACC. This is the second of the series of papers the authors propose to write on this subject. The principal question addressed in all of them concerns the process by which two agents "learn" from each other. More specifically, the question is how two (or more) agents should share their information to improve their performance.**

In Part I, the concept of mutual learning was introduced and discussed briefly in the context of two deterministic learning automata learning from each other in a static random environment. In this paper, we first provide some reasons why the concept can become complex even in such simple situations, and propose some (weak) necessary conditions for the problem to be well defined. Following this, we consider similar questions which arise when two agents use reinforcement learning in both static and dynamic environments. In particular, in the latter category, mutual learning in Markov Decision Processes (MDP) with finite states is discussed. Simulation results are presented wherever appropriate to complement the theoretical discussions.

Keywords—mutual learning, reinforcement, learning automata, Markov Decision Processes

I. Introduction

Learning theory is a vast field, multidisciplinary in character, in which a variety of different methods have been investigated. Currently it has wide applications in diverse areas including machine learning, multi-agent systems, robotics, and neuroscience. Most of the work on the subject reported in the literature has dealt with a single learner operating in a deterministic or stochastic environment. In part I of this paper (Narendra and Mukhopadhyay, 2019), the authors introduced the concept of "mutual learning" and discussed it in the context of stochastic automata (Narendra and Thathachar, 1989). Even though the term "mutual learning" had already been in use, the paper by Narendra and Mukhopadhyay was the first to suggest that the concept be investigated in a quantitative sense. In contrast to other works, it was suggested that mutual learning be studied as a problem in systems theory, focusing on general questions and issues. The principal question raised is concerned with the manner in which two or more agents who have partially learned about a process or an environment should share their information to improve their decisions.

Rationale for Mutual Learning:

Mutual learning can happen between two humans, a human and a machine, or two machines. The first class of problems can be of interest to researchers in communities such as social psychology. Learning between humans and machines is becoming increasingly important with advances in autonomous technology, and mutual learning between machines will be widely used in the future, when machines performing similar tasks, but trained using different approaches, attempt to cooperate. Problems involving mutual learning are ubiquitous, and range from the relatively simple to those that are extremely complex and difficult (or, sometimes impossible) to formulate analytically. This is due to the fact that the learning procedures used by the two agents may be identical, similar to each other, or vastly different, making the interpretation of different experiences in a single analytical framework quite complex.

Objectives of the Paper: Our principal aim in this paper is to introduce questions for discussion, suggest some preliminary answers, and indicate the major difficulties encountered. In (Narendra and Mukhopadhyay, 2019), several scenarios were proposed in which mutual learning can be studied. These scenarios included optimization, decision-making, identification, pattern recognition, etc.. In this paper, we study, in some detail, mutual learning using deterministic and stochastic reinforcement learning schemes in static environments, as well as dynamic Markov Decision Processes (MDP) with a finite number of states using both direct and

Authorized licensed use limited to: Yale University. Downloaded on September 18,2022 at 19:14:23 UTC from IEEE Xplore. Restrictions apply.

indirect learning methods. Future papers will attempt to extend the approaches to systems described by stochastic nonlinear differential/difference equations. In fact, all the principal learning approaches suggested in the literature appear to fall within the scope of investigations proposed in this paper.

Organization of the Paper: In section 2, the origin of the term Mutual Learning as introduced in this paper is discussed. As stated earlier in this Section, while the term had been used qualitatively by other authors, the suggestion that it be discussed quantitatively is due to the authors. In Section 2 the earlier work is briefly described. Section 3 is devoted to mutual learning in static stochastic environments. The discussion also leads to some simple necessary conditions for problems in mutual learning to be well defined. Finally, in Section 4 mutual learning in Markov decision processes is discussed.

II. MUTUAL LEARNING – RESEARCH AND APPLICATIONS

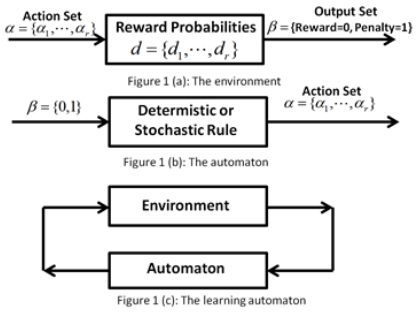
As mentioned in (Narendra and Mukhopadhyay, 2019), the authors introduced the term "Mutual Learning" as one derived from the problem of "Mutual Adaptation" investigated by the first author and his co-workers. However, a survey of the literature revealed to the authors that the term had been studied by others qualitatively in diverse areas. In the following, we summarize some of this work on mutual learning.

Ikemoto et al (2012) discuss an interesting study involving mutual learning and co-adaptation in a human-robot system, inspired by the parenting behavior in humans. In the context of artificial neural networks, Zhang et al (2017) discuss the problem of an ensemble of deep neural networks learning from each other in the context of a classification task. It was concluded that a collection of small neural networks with mutual learning can outperform a "powerful" single teacher network. In machine vision, Nie et al (2018) discuss mutual learning to achieve superior performance between two related, but disparate computer vision tasks, i.e., human parsing and pose estimation. Klein et al (2005) and Rosen-Zvi et al (2002) discuss the problem of synchronization between two neural networks through mutual learning whereby their weights converge to two parallel vectors. They also show how these synchronized parallel vectors can be used for data encryption. Liu et al (2012) discuss the integration of suitable mutual learning in an artificial bee colony optimization algorithm, and demonstrate through simulation studies that improved performance can be obtained by incorporating such mutual learning. Another related research theme is multi-agent learning systems, where the agents focusing on different, disparate subtasks of a complex task cooperate to solve the problem, in a spirit similar to mathematical game theory. Panait and Luke (2005) provide a survey of this somewhat well-established field, highlighting the issues of inter-agent communication, task decomposition, and scalability in such multi-agent systems. In contrast to this theme of multi-agent systems, in mutual learning, the agents are involved in solving the same or similar tasks, and the objective is for them to act as (partial) teachers to each other in order to improve their learning. Further, our objective, in contrast to other works, is to study such mutual learning problems as systems theory problems, focusing on general questions and issues. Our choice of reinforcement learning as the initial framework for studying mutual learning is motivated by the availability of statistical and mathematical learning theories to analyze convergence of reinforcement learning systems, thereby making the mutual learning problem analytically tractable in this framework.

III. MUTUAL LEARNING IN STATIC, STOCHASTIC ENVIRONMENTS – LEARNING AUTOMATA

The complex questions that arise in mutual learning can be illustrated using simple learning automata schemes, described in the earlier paper by the authors. The discussion of such schemes also reveals that some simple conditions need to be satisfied if the learning is to proceed in a rational manner.

A learning automaton consists of an environment E connected in feedback with an automaton A as shown in Figure 1. An environment E (figure 1a) is described by the triple $\{\alpha, d, \beta\}$ where $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ represents the finite input set with r actions α_i, an output set $\beta = \{\beta_1, \beta_2\}$ with $\beta_1 = 1$ representing a reward, and $\beta_2 = 0$ representing a penalty, and $d = \{d_1, d_2, \dots, d_r\}$ a set of unknown reward probabilities d_i = Prob[$\beta(n) = 1 | \alpha(n) = \alpha_i$].



Objective: The overall objective of the learning automaton is to choose the best action αopt among the actions in the action set α. This is accomplished by the automaton making a rational decision after every experiment in the environment.

We first consider a simple approach in which all the actions are chosen with equal probabilities $\frac{1}{r}$. This

Authorized licensed use limited to: Yale University. Downloaded on September 18,2022 at 19:14:23 UTC from IEEE Xplore. Restrictions apply.

involves no learning, and the expected reward will be $d_{av} = \frac{1}{r}\sum_{i=1}^{r} d_i$. Learning is consequently used to achieve a reward greater than the above. When the expected reward is greater than d_{av}, the automaton is said to be expedient. When the learning scheme chooses the best action with probability 1, it is said to be optimal. From a practical standpoint, it is not possible to conduct the experiment an infinite number of times. Only tentative conclusions can be drawn using a finite number of trials in an environment.

Deterministic Automaton: An automaton is said to be deterministic if on the basis of the action α_i and the environmental response $\beta_i \in \{0,1\}$, it chooses the next action using a fixed rule.

Stochastic Automaton: The automaton is said to be stochastic if, on the basis of response of the environment after every trial, it updates the probabilities with which the actions are to be chosen.

Mutual Learning: Let two automata operate in the same environment. If one of the automata has performed a very large number of trials, it essentially knows most of the details about the environment, and, in particular, the best action (based on the probabilities computed). Such an automaton may be considered a teacher. However, our interest is in automata which have performed the experiments only a finite number of times, and want to improve their decisions based on the information received from the other.

Interaction of the Automata: The following questions become relevant while considering mutual learning:

a) How often do the automata communicate with each other – only once, a finite number of times, periodically, or, at random intervals? What information do they communicate to each other? (Communication)

b) Do both the automata use the same algorithm, similar algorithms, or very different algorithms? In the latter two cases, how do they interpret each other's communicated information? (Translation)

c) How much credence do the two automata have in each other, based on past interactions? (Trust)

d) Do the automata share the same (or, similar) goals? (Goal Congruence)

3.1 Simulation Experiments with Learning Automata

In this subsection we include a few simple simulation studies of mutual learning of increasing complexity in static environments. These include deterministic and stochastic learning schemes described earlier.

Simulation Study 1: Tsetlin Automata: In this study we consider the simple case of two Tsetlin automata learning from each other. This is shown in Figure 2.

α_1 and α_2 are two actions which have unknown reward probabilities of 0.9 and 0.2 respectively in a static environment. It is seen that an automaton starting in state φ_1(i.e., choosing action α_1) stays in the same state with probability 0.9, while one starting in state φ_2 switches to state φ_1 with probability 0.8. (Note that at any instant only one action is chosen so that the count of the other action remains constant as the number of trials increases). Figures 3(a) and 3(b) show the the number of times the two actions α_1 and α_2 are chosen for the two automata. Three different cases can be considered:

(i) Both automata have performed the same number of trials

(ii) The first automaton has performed a large number of trials, while the second has performed only a few, i.e., $n_1 \gg n_2$.

(iii) $n_1 \ll n_2$.

In view of the great disparity between the reward probabilities of the two actions, both automata conclude that α_1 is the better action after the number of trials exceeds 7. When $n_1 \gg n_2$ $(n_1 = 10, n_2 = 3)$, the first automaton concludes that α_1 is the better action. After communication between the two automata, the second automaton "learns" from the first that α_1 is the better action (i.e., the first automaton acts as a teacher for the second).

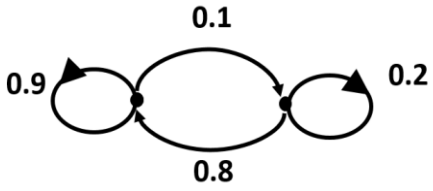
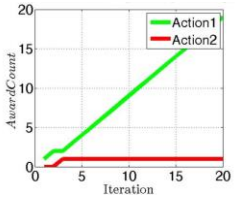
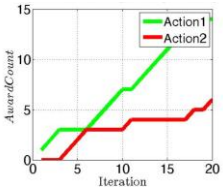


Fig. 2: A scheme of the transition probabilities.



(a) Scheme 1.



(b) Scheme 2.

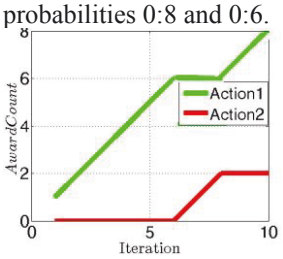
Fig. 3: Results of simulation 1.

Simulation Study 2: Stochastic Automata: A stochastic automaton using the L_{RI} scheme was simulated in the same environment, and the convergence of the scheme was studied. The convergence was found to be substantially slower. However, it is evident after detailed

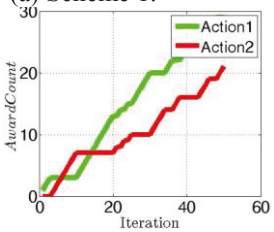
Authorized licensed use limited to: Yale University. Downloaded on September 18,2022 at 19:14:23 UTC from IEEE Xplore. Restrictions apply.

exchange of information regarding trials and rewards, that the superior action was selected.

Simulation Study 3: The same experiment carried out in Simulation 1, with two simple deterministic schemes was replaced with an environment with reward probabilities 0:8 and 0:6.



(a) Scheme 1.



(b) Scheme 2.

Fig. 4: Results of simulation 3.

The evolution of the rewards obtained as a function of the trials are shown for both the schemes in Figure 4. At the end of 10 trials, the learning scheme 1 has 9 rewards and 1 reward with actions a1 and a2 while the second scheme has 6 and 4 respectively. Assuming a threshold of 0.8 for the reward probability, scheme 1 concludes that action a1 is the best action. But scheme 2 has tobase its action on the information obtained from scheme 1. Hence, ”learning” in this case is entirely on the part of scheme 2 which also chooses a1 as the best action.

IV. MUTUAL LEARNING IN DYNAMIC, STOCHASTIC ENVIRONMENTS – REINFORCEMENT LEARNING

4.1 The Markov Decision Problem

A Markov Decision Process (MDP) is defined by the following quantities:

- The State Space S, a finite set,
- The action set A, a finite set, listing all actions available to the agent in any state
- A state transition probability function P : S X S X A → (0, 1)
- An immediate payoff function R : S X S X A → [0; 1] where 0 corresponds to no reward and 1 corresponds to reward.

The objective of the agent is to maximize the overall discounted reward, i.e., the objective is not merely to maximize the reward of the next transition, but to maximize the reward over all future transitions from an initial state.

4.2 Conventional Reinforcement Learning

Reinforcement learning aims to find the optimal decision (or, decision rule) in feedback with an unknown and uncertain teacher or environment, on the basis of a qualitative and noisy on-line performance feedback provided by the teacher in the form of a reinforcement signal. There exists a large variety of learning models and algorithms (Kaelbling, Littman, and Moore, 1996; Sutton and Barto, 1998) depending on the assumptions on the teacher or environment. One of the earliest reinforcement learning models is the learning automaton discussed in the previous section.

While the basic learning automaton model was proposed for a stationary but unknown environment (constant reward probabilities), more recent and advanced models have focused on nonstationary and dynamic environments. Such models deal with the case where the state of the environment evolves dynamically based on the agent’s actions, and a Markov Decision Process (MDP) model (for discrete state case) or a stochastic nonlinear difference equation model (for continuous state case) is used to describe the environment. The objective in this dynamic formulation is once again, to learn an optimal policy. However, in this case it is to maximize the long-term (over a finite or infinite horizon) possibly discounted reward or pay-off received by the agent. Many different methods based on Bellman’s dynamic programming (for discrete-state MDPs) and Hamilton-Jacobi-Bellman (HJB) equation (Al-Tamimi, Lewis, and Abu-Khalaf, 2008) or its linear version Riccati equation (for continuous-state environments) have been proposed to learn the optimal policies in such stochastic, unknown, dynamic environments.

Model-free (Direct) and Model-based (Indirect) Learning: The large classes of algorithms available for various reinforcement learning models can be broadly categorized into two classes: Model-free and Model-based methods. Model-free methods (such as linear reward-inaction algorithm for learning automata, and temporal difference (Sutton, 1988) and Q-learning (Watkins and Dayan, 1992) algorithms for MDPs aim to directly learn the optimal policy without attempting to learn a model or estimate of the environment. Model-based methods such as Adaptive Dynamic Programming (Barto, Bradtke, and Singh, 1995) and Approximate Dynamic Programming (Powell, 2007), on the other hand, maintain and update a model of the environment, and it is this model which is used to compute the optimal policy.

4.3 Mutual Reinforcement Learning in an MDP

The Markov Decision Problem was briefly described in sections 4.1 and 4.2. It was also mentioned there that both direct (model-free) and indirect (model-based) approaches have been proposed in the literature for solving the reinforcement learning problem in MDPs. In the former (e.g., in Temporal difference or Q-learning algorithms), a value function (a function of state in temporal difference, and a function of state-action pairs

Authorized licensed use limited to: Yale University. Downloaded on September 18,2022 at 19:14:23 UTC from IEEE Xplore. Restrictions apply.

in Q-learning) is directly updated on the basis of the reinforcements received, without taking recourse to estimating a model of the underlying MDP.

The mutual learning problem, then, reduces to how to combine the knowledge structures of two agents, possibly exploring different state space regions and/or following different trajectories, developing independent estimates (of value functions or of MDP parameters).

Mutual Learning in MDPs using Direct Methods: Given two value functions $V_1(x)$ and $V_2(x)$ corresponding to the two agents using the temporal difference algorithm, each agent can use a weighted combination of the two as the new value function.

$$V(x) = \frac{w_1(x)}{(w_1(x) + w_2(x))} V_1(x) + \frac{w_2(x)}{(w_1(x) + w_2(x))} V_2(x)$$

where the relative magnitudes of $w_i(x)$ at a particular state will determine which agent acts as the teacher and which agent the student. In a simplified scenario, $w_i(x)$ is binary valued, i.e., 1 for the teacher (which agent is the teacher is appropriately determined, as explained below), and 0 for the student at a particular state x. In the case when the agents are using Q-learning, the value functions V(x) is replaced by the Q-values Q(x,a) for both the agents.

Mutual Learning in MDPs using Indirect Methods: In this case, detailed counts of the state transitions are maintained by the reinforcement learning algorithm itself, and the experiences of the two agents can be easily combined (such as by means of simple aggregation) to determine the current estimates of the MDP parameters. Such parameters can then be used to estimate the optimal policy using certainty equivalence principles.

In the following, we further clarify the mutual learning algorithm by considering the four issues of Trust, Communication, Goal Congruence, and Translation, of mutual learning mentioned earlier in the paper.

Trust: As explained earlier in this section, for simplicity we assume Trust $w_i(x)$ of each agent to be binary valued (i.e., 1 for the teacher and 0 for the student). The identity of the teacher and the student can be determined using various heuristic rules. We propose a rule based on a measure of uncertainty in the experience of an agent. In particular, each agent maintains counts $NI_i(x)$ and $ND_i(x)$ corresponding to the number of times its value function, using its own experience, was incremented and decremented respectively. Then, whichever agent j has the highest absolute value of $(NI_j(x) - ND_j(x))$ at a particular x is the teacher at that x.

Communication: The amount of communication between the two agents is determined both by the frequency of communication as well as the information communicated in each communicating instant. This, in turn, guides the design of the mutual learning algorithm. For example, we may assume that communication (and, hence mutual learning) takes place once every T units of time, interspersed with self-learning (i.e., learning from one's own experiences). We can then experiment with different values of T. Further, we may assume that, during each communication, the two agents only communicate the values of current states (and the corresponding state's count variables NI and ND). This is in contrast to communicating the information for all states which would allow for the possibility of updating the values of all states through mutual learning.

Goal Congruence: In most of the simulation studies reported later, it is assumed that the payoff functions of the two agents are the same. This is to avoid the possibilities that the two agents may have different optimal policies in which case mutual learning will in fact hinder the learning by each agent. For example, with inverted payoff functions of the two agents, mutual learning may indeed have undesirable effects (e.g., lack of convergence, or wrong convergence).

Translation: For most of our discussions on mutual learning, we assume that the two agents are using the same learning algorithm. The problem of mutual learning may become considerably more difficult (and in fact, possibly ill-posed), if the two agents use two different learning algorithms. For example, consider the case of agent 1 using a direct learning algorithm (such as Q-learning), while agent 2 uses an indirect learning method (such as adaptive dynamic programming). Translation from the state of the agent using the indirect method to that of the agent using direct method is fairly straightforward using Bellman equations. However, without any additional constraints, the reverse is indeed ill-posed, due to the non-uniqueness of model parameters (transition probability and immediate reward matrices) that result in a given set of state values.

4.4 Simulation Experiment for an MDP

While both Direct and Indirect methods were described in the previous section, due to space limitations, only the Direct method is used in the following simulation study.

The Problem: An MDP has four states and two actions in each state. The transition probabilities for action 1 are:[0.5 0.5 0 0; 0.4 0.5 0.1 0; 0 0.5 0.3 0.2; 0 0 0.5 0.5]. The transition probabilities for action 2 are: [0.6 0.4 0 0; 0.4 0.6 0 0; 0 0 0.6 0.4; 0 0 0.4 0.6]. The rewards in the four states are : [0.1 0.1 0.9 0.9].

The Simulation: We first implement just 1 agent with Q-learning. This corresponds to using only conventional reinforcement learning without any mutual learning. The Q-values for all state action pairs are initialized to 0, and the initial state is chosen to be s1 (the first state). The Q values of the state action pair are then updated using the standard Q-learning algorithm. (The actions are chosen in state s, according to the Boltzmann distribution, with probabilities:

$Exp\ (Q(s,a)/T)\ /\sum_b Exp(Q(s,b)/T)$

Authorized licensed use limited to: Yale University. Downloaded on September 18,2022 at 19:14:23 UTC from IEEE Xplore. Restrictions apply.

where T is the temperature parameter which is initially a large value (say, 1000) and is reduced with k (time steps) as say, $1000/\sqrt{k}$).

We next implement two agents using identical Q-learning algorithms. The two agents communicating their information to each other would correspond to mutual learning. Agent 1 starts in state s1, while agent 2 starts in state s4, and both agents update their Q values with a mutual learning interval of 5 trials.

Results: Due to space limitations, only Agent 1's average reward per time step is shown in Figure 5 with and without mutual learning. It is evident that the average reward increases faster for agent 1 in the second case with mutual learning, than in the first case when it is working alone.

Possible Explanation: For some of the states that Agent 1 visits, Agent 2 has a more reliable estimate of the Q-values than Agent 1. In those states, Agent 2 acts as a teacher to Agent 1, which improves Agent 1's Q-values. This, in turn, gets reflected in a higher average reward with mutual learning.

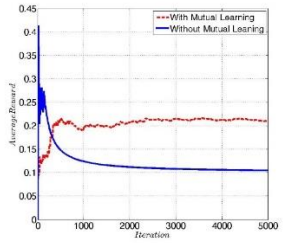


Figure 5: Mutual reinforcement learning

V. COMMENTS AND CONCLUSIONS

The paper introduces the concept of "mutual learning" in which two (or more) agents which "learn" in a random static or dynamic environment, attempt to improve themselves based on the information provided by the other(s). The essential difference between what is proposed in the current paper and past work is the suggestion that the problems should be formulated within a mathematical framework and that the changes in each participant should be quantified in some fashion. The authors believe that the paper will give rise to interesting and meaningful discussions in the systems community concerning mutual learning.

ACKNOWLEDGMENT

The research reported here was supported by the National Science Foundation under grant numbers 1930601 (to Yale) and 1930606 (to IUPUI).The authors are very grateful to Kasra Esfandiari for his help with the simulation studies.

REFERENCES

[1] Al-Tamimi, A., Lewis, F. L., & Abu-Khalaf, M. (2008). Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *38*(4), 943-949.

[2] Atkeson, C. G., & Santamaria, J. C. (1997, April). A comparison of direct and model-based reinforcement learning. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on* (Vol. 4, pp. 3557-3564). IEEE.

[3] Barto, A., & Anandan, P. (1985). Pattern-recognizing stochastic learning automata. *Systems, Man and Cybernetics, IEEE Transactions on*, (3), 360-375.

[4] Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, *72*(1), 81-138.

[5] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, 4, 237-285.

[6] Klein, E., Mislovaty, R., Kanter, I., Ruttor, A. and Kinzel, W., 2005. Synchronization of neural networks by mutual learning and its application to cryptography. In *Advances in Neural Information Processing Systems* (pp. 689-696).

[7] Liu, Y., Ling, X., Liang, Y. and Liu, G., 2012. Improved artificial bee colony algorithm with mutual learning. *Journal of Systems Engineering and Electronics*, *23*(2), pp.265-275.

[8] Narendra, K. S., & Mukhopadhyay, S. (1991). Associative learning in random environments using neural networks. Neural Networks, IEEE Transactions on, 2(1), 20-31.

[9] Narendra, Kumpati S., Mukhopadhyay, Snehasis, "Mutual Learning: Part I – Learning Automata", Accepteed, American Control Conference, 2019

[10] Narendra, K. S., & Thathachar, M. A. (1989). Learning automata. Prentice-Hall.

[11] Nie, X., Feng, J. and Yan, S., 2018. Mutual learning to adapt for joint human parsing and pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 502-517).

[12] Powell, W. B. (2007). Approximate Dynamic Programming: Solving the curses of dimensionality (Vol. 703). John Wiley & Sons.

[13] Rosen-Zvi, M., Klein, E., Kanter, I. and Kinzel, W., 2002. Mutual learning in a tree parity machine and its application to cryptography. *Physical Review E*, *66*(6), p.066135.

[14] Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. Machine learning, 3(1), 9-44.

[15] Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction (Vol. 1, No. 1). Cambridge: MIT press.

[16] Tilak, O., Martin, R., & Mukhopadhyay, S. (2011). Decentralized indirect methods for learning automata games. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 41(5), 1213-1223.

[17] Tilak, O., & Mukhopadhyay, S. (2011). Partially decentralized reinforcement learning in finite, multiagent Markov decision processes. AI Communications, 24(4), 293-309.

[18] Watkins, C. J., & Dayan, P. (1992). Q-learning. Machine learning, 8(3-4), 279-292.

[19] Zhang, Y., Xiang, T., Hospedales, T.M. and Lu, H., 2017. Deep mutual learning. *arXiv preprint arXiv:1706.00*

Authorized licensed use limited to: Yale University. Downloaded on September 18,2022 at 19:14:23 UTC from IEEE Xplore. Restrictions apply.