IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 30, 2020, accepted November 12, 2020, date of publication November 17, 2020,
date of current version December 2, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3038605

A Gentle Introduction to Reinforcement Learning and Its Application in Different Fields

MUDDASAR NAEEM[1], SYED TAHIR HUSSAIN RIZVI[2], AND ANTONIO CORONATO[3]
[1]Department of Engineering, Universita' Degli Studi di Napoli Parthenope, 80143 Napoli, Italy
[2]Department of Computer Engineering, The University of Lahore, Lahore 54590, Pakistan
[3]ICAR-CNR, 80131 Napoli, Italy

Corresponding author: Muddasar Naeem (muddasar.naeem@uniparthenope.it)

This work is funded by Italian Government through Universita' Degli Studi di Napoli Parthenope, Napoli (grant 33 cycle PhD Scholarship) and partially supported by ASMARA - Applicazioni pilotapost Direttiva 2010/65in realta portualiitaliane della Suite MIELE a supporto delle Authority per ottimiz- zazione della inteRoperabilita nell\u2019intermodalitA.

ABSTRACT Due to the recent progress in Deep Neural Networks, Reinforcement Learning (RL) has become one of the most important and useful technology. It is a learning method where a software agent interacts with an unknown environment, selects actions, and progressively discovers the environment dynamics. RL has been effectively applied in many important areas of real life. This article intends to provide an in-depth introduction of the Markov Decision Process, RL and its algorithms. Moreover, we present a literature review of the application of RL to a variety of fields, including robotics and autonomous control, communication and networking, natural language processing, games and self-organized system, scheduling management and configuration of resources, and computer vision.

INDEX TERMS Artificial intelligence, reinforcement learning, applications, healthcare, robotics, communication, natural language processing, computer vision, resource management, IoT.

I. INTRODUCTION

With the development in computing technology and the inception of new intelligent algorithms, the goal of Artificial Intelligence (AI) has become a step closer. AI is a simulated intelligence on programmable machines and tries to mimic the human brain. Machine Learning (ML) is a sub-field of AI, which concerns with ''the question of how to develop software agents that improve automatically with experience'' [1].

There are mainly three categories of ML.

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

The first category i.e. supervised learning is related to learning from a set of training data of labeled examples that are provided by a domain expert who takes the role of the external supervisor in the learning process. The goal is to enable the learning agent with the capability to generalize its responses to cases not included in the training set. Unsupervised learning, instead, is related to finding hidden patterns and knowledge into a dataset, without any supervision.

The associate editor coordinating the review of this manuscript and approving it for publication was Ahmed A. Zaki Diab.

Finally, Reinforcement Learning (RL) is the type of learning guided by a specific objective. An agent learns by interacting with an unknown environment, typically in a try-and-error way. This is the most common way of learning for a child, who does something and observes what it happens. The agent receives feedback in terms of a reward (or punishment) from the environment; then, it uses this feedback to train itself and collect experience and knowledge about the environment.

Reinforcement Learning problems are related to learning which is the best action to perform, situation-by-situation, in order to maximize the aggregated reward. RL agent has to learn a policy (i.e. a complete mapping between situations and actions) by trying actions out without any domain expert has told it, as in many other forms of machine learning. Another relevant characteristic of a RL problem is that in any situation the agent has to choose between exploiting its current knowledge of the environment (perform an action already tried previously in that situation) or exploring actions never tried before in that situation.

In this paper, we present the widely used RL algorithms in healthcare, robotics and autonomous control, communication and networking, natural language processing, games and self-organized system, scheduling management and

configuration of resources, IoT and computer vision. This work aims to serve as a guideline for those who want to start working in reinforcement learning.

The remaining document is organized as follows; Section II gives an introduction to Markov Decision Processes. Section III discusses several classes of RL algorithms. Finally, we present applications of RL followed by the conclusion.

II. MARKOV DECISION PROCESS

Markov Decision Process (MDP), a reinterpretation of Markov chains is used for the decision-making process in a stochastic environment. The goal of MDP is to give the mapping of optimal actions for each state of an environment. MDP based is on Markovian property which does not consider past information and only the present matters. Prediction of the next state is completely independent of past states. Things or physics of the given environment are stationary, and rules do not change. Chess is a good example where neither rules change nor you need to remember your past moves to play the next move. Moreover, many other applications may be modeled as an MDP, like operation research, control theory, statistics, games, econometrics, AI, robotics, dialogue control, optimal investments, medical tests, logistics bio-reactor control, etc. Next, we define a few terms to completely understand MDP and RL.

A. ENVIRONMENT

We need to define an environment for every RL problem. Defining an environment means to list a set of rules i.e. which action an agent is permitted to select, what states the environment has, what will be the rewards or penalties.

As mentioned previously, for Reinforcement Learning you have to define an environment. Creating an environment essentially means defining a clear set of rules. What actions is an agent allowed to take in this environment? What possible states does this environment have? What are the definitions of Rewards and Penalties?.

The Environment is the world where an agent or software algorithm can interact or move. The input to the environment is an agent's action taken, current state while the environment's output is the next state and reward. The environment can be anything that process and determines an agent's actions and corresponding consequences (reward, resulting state). The environment may be a game, a healthcare setup or a place where an agent lives.

B. STATES [$S = s_0, s_1, \ldots, s_n$]

The set of states, places, positions in a given world/ environment where an agent can reach or visit is called a state. For a controller, it can be pressure or temperature value. For a navigating robot, the state may be a room or a place. A State can be represented by coordinates, by simply numbers or alphabets. the Number of states can be finite and infinite. The non-transitioned out states or states which end the process are called terminal states. When an agent moves from one state to another state, it is called a visit and multiple visits i.e from the start state to any terminal state is known as an episode.

C. ACTIONS [$A = a_0, a_1, \ldots, a_n$]

Actions or set of actions are anything which an agent or a robot can imagine or allowed to do in an environment. For example, in a grid world, an agent may go left, right, down, up or can stay in the current state. In a single horizontal line environment, an agent can move left or right. In general, an agent may opt for any action that is available in a given scenario.

D. TRANSITION MODEL

It describes the rules, dynamics or physics of the given environment. Transition function $T(s, a, s')$ consists of present state, choosen action and new state. It gives probability what will happen if you do an action in a state i.e. probability of moving from state s (current state) to state s' (new state) by taking an action a and getting reward R. So the transition model is independent of past i.e. states and action while dependent on current state, chosen action and resulting state (Markov property).

E. REWARD [R(s)]

The Reward function $R(s)$ returns a numerical value that an agent can get for being in a state after taking an action. Rewards tell whether a state is useful or not such that agent gets a higher reward when moving in useful states and lower reward when moving in undesirable states. Reward acts as feedback for an agent which can be positive or negative. Nature or amount of reward has some effect on actions and policy (definition of policy will be given later). Sometimes minor changes make a big difference in agent behavior. The cumulative future reward also called return is given as:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \ldots + r_\infty \tag{1}$$

where subscript t denotes a specific time step. Here we need to end a series of rewards instead of infinite returns. We are more interested in episodic tasks and cumulative discounted rewards. Episodic tasks are those which terminates after a certain time step or when the agent reaches in a terminal state. We may denote the long term discounted reward as a utility of a state.

F. UTILITY FUNCTION

The agent receives a reward at every state and we can use it to calculate the utility function of states history as defined in equation (2).

$$U_h = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \ldots + \gamma^n R(s_n) \tag{2}$$

γ is discount factor between zero and one. The utility of a single state is defined as follows:

$$U(s) = \sum_{t=0 \to \infty} \gamma^t R(s_t) \tag{3}$$

Now we have two extreme values of γ i.e. 0 and 1. If we set γ equal to 1 it represents a long-sighted algorithm, similar to equation 1 and so all rewards are treated equally irrespective of their arrival time. However, a value of γ equal to 0 means a short-sighted algorithm that gives more weight to immediate rewards. Such algorithm ignores the impact of future reward on total reward and only select actions that are currently best.

Up till now, we have defined an MDP framework. We conclude that the objective for a RL agent is to maximize its aggregated reward by visiting desired states and avoid the undesired states which can be achieved through policy.

G. POLICY

A policy guides an agent which action to choose in a given state. It is a mapping from a set of states to a corresponding set of actions. An optimal policy gives long term optimized reward which an agent could get in a lifetime. From policy, one may infer a plan, but a policy is different from a plan. A plan tells the sequence of actions from the start state to the destination state. A policy tells what to do in whatever situation you are in or which action is best an agent should take in a specific state. This is due to Markovian property which only depends on the present state. So a policy tells how an agent should act. To find optimum policy we need to know about Bellman equation which is presented next.

H. BELLMAN EQUATION

The famous Bellman equation invented in 1953, by Richard Bellman has the most significant importance in solving MDP and RL problems. The Bellman equation can be employed to calculate utility function. It is a recursion for future expected rewards.

$$U(s) = R(s) + \gamma \max_a \sum T(s, a, s')U(s') \tag{4}$$

I. PARTIALLY OBSERVABLE MARKOV DECISION PROCESS (POMDP)

The POMDP is an extension of an MDP that plays an important role when an agent cannot a make direct observation about the underlying state. Like MDP, POMDP has probabilistic transitions between a finite set of states and fellows the Markovian property but not sure which state an agent is in. In other words, POMDP has control over state transitions but does not has complete control over state observability.

III. REINFORCEMENT LEARNING

Humans always try to improve their interaction with the environment based on their previous experience. An artificial agent in the RL setup tries to perform the same behavior. As we have understood the concept of state, action, environment, reward and policy so technically we can define the goal of an RL agent is to search an optimal policy for a given set of states to maximize long-term reward. The general scenario of RL working framework is shown in figure.1. The RL being the most target-oriented branch of ML also incorporate delayed rewards in very complex delayed responded setup

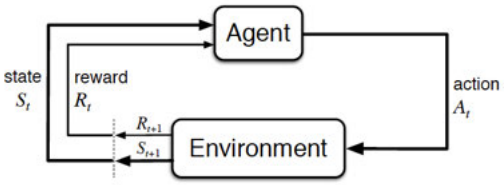

FIGURE 1. The reinforcement learning problem [2].

where it is difficult to identify which action was beneficial over a lot of time slots.

A. MULTI-ARMED BANDITS

The simplest case to solve by RL is multi-armed bandits problems where each action available to an agent has some reward based on probability distribution. For example, multiple one-armed bandits (slot or gambling machines) or one major machine with many slots, where the target is to get the maximum desired measure of reward over a time period by ''gambling on slot-machines (or bandits)'' [3] that have unknown and different expected outcomes. Similarly, the RL agent also tries to obtained highest possible reward over a lot of episodes.

B. EXPLORATION VS EXPLOITATION

A delicate balance between exploitation and exploration is one major problem in RL which does no appear in other machine learning algorithms. In exploitation mode, an agent takes the best action out of already known knowledge while in exploration, an agent may attempt stochastic action to increase its information in order to get more reward. For example, in a slot machine problem, you may try to find new bandits under given probability distribution before finalizing the best actions. But at the same time, this search for the best choice may take you away from maximizing cumulative reward. Similarly, in exploitation, you take the usual path to travel from one place to another but you need to try other possible paths for exploration.

C. VALUE FUNCTION

The state-value function and the utility function defined before, are the same and both are used for the value of a state. In [4] they called this term as the state-value function or simply value function while in [5], it is referred as a utility function. The value function of a state is represented by $V(s)$ and the under policy π is represented by $V^{\pi}(s)$ i.e. outcome by starting in state s and following policy π thereafter as given in (5).

$$V^{\pi}(s) = (R_t|s_t) \tag{5}$$

While $Q^{\pi}(s, a)$ also known as action-value function gives value when starting from state s by taking action a and following π thereafter and is referred as Q-value.

$$Q^{\pi}(s, a) = (R_t|s_t, a_t) \tag{6}$$

TABLE 1. Difference between model-free methods and model based methods.

Model-free	Model-based
1. Model free algorithms e.g. MC Control, SARSA, Q-learning rely on real samples from the environment and do not use generated predictions of next state and next reward to alter behaviour.	1. Model based algorithms like DP use the model's predictions of the next state and reward in order to calculate optimal actions.
2. Model-free approaches are based on habitual conditions and learn through trial-and-error methods.	2. Model-based approaches are well suited for goal-directed decisions and learn through planning.
3. Most state of the art algorithms use model-free RL due to availability of simulators that are able to generate huge amounts of data.	3. Model-based methods are beneficial in applications where we have strict restrictions on the sample complexity.
4. The consequences of actions are predicted by past experiences in case of model-free approach.	4. The consequences of actions are predicted by the structure of the world in case of model-based approach.
5. The values and parameters of Model-free approach change slowly over time due to iterative updating	5. Model-based approaches update its values and parameters very fast
6. Extensive experience is required by model-free approaches	6. Computational requirements are high in case of model-based approaches
7. Strong convergence is guaranteed in case of model-free model.	7. Strong convergence is not guaranteed in case of model-based models.

Note that for the same environment, value function may change with change in policy. The key to solve RL problems, as well as the goal of RL agent, will be to find these value function for a specific problem. The Bellman equation for state and state-action value functions is given next respectively.

$$V^{\pi}(s) = R(s) + \sum(\gamma V^{\pi}(s')) \tag{7}$$

$$Q^{\pi}(s, a) = R(s) + \sum(\gamma Q^{\pi}(s', a')) \tag{8}$$

IV. RL ALGORITHMS

Any of RL algorithm may be policy-based or value-based or a combination of both i.e. actor-critic method. the RL algorithms can also be classified as model free methods e.g. Q-learning and model-based algorithms such as dynamic programming, transition models and return function. In a model-based algorithm, an agent does not rely on trials instead it exploits already learned model. In model-based RL, an agent can make predictions about different states and corresponding rewards after learning.

the Model-free methods depend on trial and error to update their experience and knowledge about the given environment as they do not have knowledge of the transition model and reward function. They have to learn the system dynamics by interacting with the environment over a large number of times. Some examples of the model free algorithms are Monte Carlo (MC) and Temporal Difference (TD).

Table 1 presents a comparison between model-based and model-free techniques [6]–[8] and [9]. The mutual weakness of both approaches can be overcome by hybrid approaches having properties of both [10], [11] and [12].

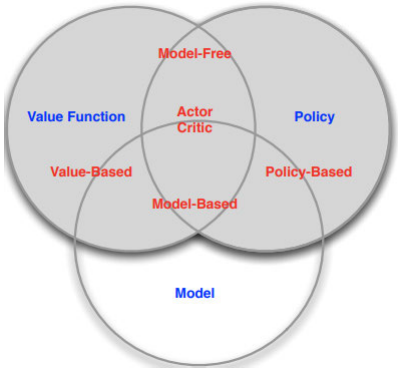

FIGURE 2. Types of reinforcement learning algorithms http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/intro_RL.pdf.

A. DYNAMIC PROGRAMMING

Introduced by Richard Bellman, Dynamic Programming (DP) is a mathematically solid technique to solve optimization problems. The two properties of DP are optimal substructure and overlapping sub-problems and MDP satisfies these two properties. A complex and complicated problem can be sequentially divided into simpler and smaller subproblems. Like humans, it is also easy for a machine to learn stepwise. So, DP searches every possible solution and chooses one which is best at each computation. Similarly, the aim of RL is to search for optimal policies or actions for an agent to act optimally.

Being model-based, the DP algorithm needs the full observable knowledge (transition model of and reward function). So, in some RL problems where the given environment may be modeled as an MDP, DP methods (value iteration or policy iteration) can be utilized to get optimal value function or policy. In other words, DP may be applied for planning in an MDP to solve either a control problem or prediction problem. Policy and value iteration algorithms are presented next. In other words, DP may be applied for planning in an MDP to solve either a control problem or prediction problem as summarized in table 2. Policy and value iteration algorithms are presented next.

TABLE 2. DP algorithms http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/DP.

Problem	Bellman Equation	Algorithm
Prediction	Bellman expectation equation	Iterative policy evaluation
Control	Bellman expectation equation + Greedy policy improvement	Policy iteration
Control	Bellman optimality equation	Value iteration

1) VALUE ITERATION

It starts with a value function randomly and then by using the Bellman equation, improve the value function recursively. When the function reached the optimal value, then get the policy from it i.e. the optimal policy for the given task. The pseudo-code for Value Iteration Algorithm (VIA) as in [2] is given in algorithm 1.

Algorithm 1: Value Iteration Algorithm

Initialize V arbitrarily
Repeat
$\Delta \leftarrow 0$
For each $s \in S$
$v \leftarrow V(s)$
$V(s) \leftarrow max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
$\Delta \leftarrow max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)
output a deterministic policy, π, such that
$\pi(s) = argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$

Primarily, the VIA determines the optimal state value function by improving $V(s)$ estimation iteratively. This method initializes $V(s)$ to random values arbitrary. It continuously performs updates on $V(s)$ and $Q(s, a)$ values function until convergence. VIA guarantees the convergence to the optimal values.

2) POLICY ITERATION

In the policy iteration, our target is learning of an optimal policy by iteratively using the Bellman equation. The policy algorithm has three steps. First, select any random policy and perform policy evaluation. In the second step, improve the policy by using the value function and finally repeat first two steps until convergence. The pseudo-code for Policy Iteration Algorithm (PIA) as in [2] is given in algorithm 2.

Algorithm 2: Policy Iteration Algorithm

1. Initialization
$V(s) \in Rand\pi(s) \in A(s)arbitrarilyforalls \in S$
2. Policy evaluation
Repeat
$\Delta \leftarrow 0$
For each $s \in S$
$v \leftarrow V(s)$
$V(s) \leftarrow max_a \sum_{s',r} p(s', r|s, \pi(s)[r + \gamma V(s')]$
$\Delta \leftarrow max(\Delta, |v - V(s)|)$
3. Policy imporvement
policy -stable $\leftarrow$ true
for each $s \in S$:
$a \leftarrow \pi(s)$
$\pi(s) = argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
if $a \neq \pi(s)$ then policy stable $\leftarrow$ false if
policy-stable then stop and return V and π; else go to 2.

PIA manipulates the policy directly, instead of finding it indirectly through optimal $V(s)$ or $Q(s, a)$. It is a method in RL that assists in learning the optimal policy. A policy that can maximizes the long term discounted reward. This approach is more beneficial, where there are many options to select from, and every choice has its own risks and rewards.

TABLE 3. Working of MC algorithms http://www0.cs. ucl.ac.uk/staff/d.silver/web/Teaching_files/DP.pdf.

First-Visit MC Policy Evaluation	Every–Visit MC Policy Evaluation
1. Increment counter when state s is visited first time in an episode $N(s) \longleftarrow N(s) + 1$ 2. Increment total return $S(s) \longleftarrow S(s) + 1$ 3. Estimate the value by using mean return $V(s) = S(s)/N(s)$ 4. By law of large numbers $V(s) \Longrightarrow V\pi(s)$ as N(s) $\Longrightarrow \infty$	1. Increment counter when state s is visited every time in an episode $N(s) \longleftarrow N(s) + 1$ 2. Increment total return $S(s) \longleftarrow S(s) + 1$ 3. Estimate the value by using mean return $V(s) = S(s)/N(s)$ 4. By law of large numbers $V(s) \Longrightarrow V\pi(s)$ as N(s) $\Longrightarrow \infty$

B. MONTE CARLO METHODS

the Monte Carlo (MC) technique utilizes randomness to solve a problem. MC is a model free method that learns from complete episode (no bootstrapping) using the idea of mean return. MC approach may be divided into First-visit MC and every-visit MC. Former is the average of the returns as given in equation (1) by following the first visit to a state s over a period of episodes whereas in every-visit MC, the average is based on all visits to a state s. Pseudo codes of both algorithms are given in table 3.

We may use the MC method for prediction and control estimation and we say it passive and active RL respectively. In passive RL, the goal of the agent is to learn the value function by using a policy while in active RL, the agent tries to find optimal policy when interacting with the given environment. Advantages of MC methods over that of DP are these:

-It may be applied with sample models (simulations).

-MC methods are easy to implement and for environments with small subsets of states.

-MC finds solutions which are optima, via direct interaction with the world.

The Utility and value functions in the MC method are calculated by using equations (3) and (6) but with expectation operator as rewritten in equations (13) and (10) respectively.

$$U(s) = \mathbb{E} \sum_{t=0 \rightarrow \infty} \gamma^t R(s_t) \tag{9}$$

$$Q^{\pi}(s, a) = \mathbb{E}(R_t|s_t, a_t) \tag{10}$$

The True utility of state in equation (13) is guaranteed to converge when we calculate and save returns for a large number of times according to "the law of large numbers" [13]. From equation (13), we can calculate the utility of states which may be used for prediction, but for control problem, we need the utility of each action. The state-action value function as given in equation (10) stores the utilities against a specific action executed in a state. Initially, all state-action pairs are filled the same way as evaluation in the policy iteration algorithm and then we improve the policy greedily as given in equation (11).

$$\pi(s) = \text{argmax}_a Q(s, a) \tag{11}$$

It is important in MC control problems, to select start states randomly along with actions having non zero probability. Although, MC approaches can reach to optimal policy without knowledge of transition model, reward function and given policy but the MC methods have to wait till the completion of an episode to update value function. This is a limitation of MC method and solution to this limitation is Temporal Difference methods where the update can be done after one step.

C. TEMPORAL DIFFERENCE METHODS

In the previous section, we noted a serious problem with MC methods i.e. to wait for an update until an episode ends. ''In this section, the temporal difference method would be discussed that is categorized a model-free RL approach and will come to know how TD can solve this problem. TD technique learns by bootstrapping [14] from the estimation of the present value function.

Generally, TD is employed for prediction of a quantity that depends on the next values but in the RL problem, TD is applied for a prediction about aggregated reward. TD is identical to DP as it does updates on the basis of current estimate and identical to the MC technique when sampling [15]. The generic rule for the TD method is given below:

$$\text{New estimate} \leftarrow \text{Old estimate} + \alpha\,[\textit{Target} - \text{Old estimate}] \tag{12}$$

On the right side of equation (12), the difference part is the estimation error δ and the target is a real value. The goal is to reduce δ. The symbol α is known as the learning rate or step size between zero and one. The RL agent takes into account more recent reward for $\alpha = 1$ and learns nothing when $\alpha = 0$. In the last section, we calculated the utility function as the expected return of a state as given in equation (13) and so is the target calculated in equation (14).

$$V(s) = \mathbb{E} \sum_{t=0\rightarrow\infty} \gamma^t R(s_t) \tag{13}$$

$$\text{Target} = \mathbb{E} \sum_{t=0\rightarrow\infty} \gamma^t R(s_t) \tag{14}$$

But in TD, we need to update the utility function after every visit instead of all visits till the episode ends as we did in MC method. However, in TD, we do not have all rewards and have only estimated utilities and rewards at $t + 1$. So here comes the concept of bootstrapping by which we can get new estimates by using estimates. We can explain this by the next two equations as follows:

$$\text{Target} = \mathbb{E}\left[r_{t+1} + \gamma\left(r_{t+2} + \gamma r_{t+3} + \ldots..\gamma^{k-1} r_{t+k+1}\right)\right] \tag{15}$$

$$\text{Target} = \mathbb{E}\left[r_{t+1} + \gamma U\,(s_{t+1})\right] \tag{16}$$

So we are left with two known quantities and we can write the final update rule as fellows:

$$V(s_t) \leftarrow V(s_t) + \alpha\,[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \tag{17}$$

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{18}$$

Using equation (18), we can rewrite equation (17) as follows:

$$V(s_t) = V(s_t) + \alpha\delta_t e_t(s_t) \tag{19}$$

the TD technique in equation (17) also known as $TD(0)$ waits only till next update. TD form a target immediately at $t + 1$ perform an update based on current estimate $V(s_{t+1})$ and observed reward r_{t+1} as given in (17). To consider the previous states we need to use eligibility traces which is short term memory procedure to save track of visited states in previous steps and it is defined as below:

$$e_t(s) = \gamma\lambda et - 1(s) \quad \text{if } s \neq s_t$$
$$\gamma\lambda et - 1(s) + 1 \quad \text{if } s = s_t$$

The parameter $\lambda \in [0, 1]$ is the decay parameter and known as the accumulating trace or trace decay. By using eligibility traces, we may define weight to update each visited state. We have TD(0) when $\lambda = 0$ and updates are performed only on immediately visited states while for TD(1) with $\lambda = 1$, all the visited states are updated equally. Eligibility traces are more useful for environments with large state space and sparse reward. a few advantages and disadvantages of MC and TD methods are given in the table 4. Q-learning and SARSA are popular TD methods which are being presented next.

TABLE 4. Difference between MC and TD.

MC	TD
1. Learning only at the end of episode	1. Learning at every step
2. Only learns from complete sequences	2. Can learn from incomplete sequences
3. MC only works for terminating (episodic) environments	3. TD works in non-terminating (continuing) environments.
4. High variance, zero bias	4. Low variance, some bias
5. Convergence, $V(s) \Longrightarrow V\pi(s)$ as experience $\Longrightarrow \infty$	5. Convergence, $V(s) \Longrightarrow V\pi(s)$ as experience $\Longrightarrow \infty$
6. More effective in non-Markov environments	6. More efficient in Markov environments

1) SARSA

State-Action-Reward-State-Action (SARSA) is introduced in [16] as a modification of Q-learning because of resemblance to Q-learning, Sutton in [4] called it SARSA. SARSA is an on-line learning technique where an agent interacts with the given environment and performs policy update based on actions selected. The action value function Q is updated by an error and adjustment is made by the α (learning rate) as given in 20.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{20}$$

The pseudo-code of SARSA is given in algorithm 3 wherein first step agent takes action and moves a step onward and observes the reward in the second step along with a new state and action. Then in the third step, SARSA updates the Q function as in algorithm 3 and in the last step,

Algorithm 3: SARSA Algorithm

1. Initialize
Q arbitrarily
Q (terminal) =0
Repeat
initialize s
choose $a \in \epsilon - greedy$
Repeat
take action a, observe r, s'
choose $a' \in \epsilon - greedily$
$Q(s_t, a_t) \leftarrow$
$Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}] - Q(s_t, a_t)$
$s \leftarrow s'$
$a \leftarrow a'$
s is terminal
until convergence

the policy is updated at every visit by taking the action with maximum Q value.

The convergence in SARSA maybe speeds up by employing eligibility traces to state-action pairs as performed on states in $TD(\lambda)$. The eligibility traces are updated in $SARSA(\lambda)$ as given below:

$$e_t(s,a) = \gamma\lambda et - 1(s,a) + 1 \quad \text{if } s = s_t \text{ and } a = a_t$$
$$e_t(s,a) = \gamma\lambda et - 1(s,a) \quad \text{otherwise} \tag{21}$$

And the resulting update rule for the SARSA algorithm by using trace is:

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha\delta_t e_t(s,a) \quad \text{for all } s \in S \tag{22}$$

Convergence in SARSA is guaranteed when all the pairs (state-action) are observed for infinite times [5]. To consider all states and actions, we may use $\epsilon-$ greedy policy which randomly takes the action with small probability ϵ and otherwise selects action with high values as described below:

$$\pi(s) = \text{argmax}_a\, Q(s,a) \quad \text{if } \sigma > \epsilon$$
$$a \approx A(s) \quad \text{if } \sigma \leq \epsilon \tag{23}$$

where $0 \leq \sigma \leq 1$. If the value of epsilon is high then SARSA converges slowly as it does more exploration while with a small value, visit of all state-action pairs is not guarantee and this is a central exploitation-exploration dilemma in RL setup. However, in most of the cases good choice for epsilon value is 0.1.

2) Q-LEARNING

Q-learning is an off-policy forward learning and model TD algorithm [17] for control. Its update rule is given as:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \sigma\left[r_{t+1} + \gamma \max_a Q(s_t, a_t)\right] \tag{24}$$

Q-learning method learns to find an optimal policy by observation which is to say off-policy learning [18]. Its pseudo-code is given in algorithm 4.

Algorithm 4: Q-Learning Algorithm

1. Initialize
Q arbitrarily
Q (terminal) =0
Repeat
initialize s
Repeat
choose $a' \in \epsilon - greedily$
take action a, observe r, s'
$Q(s_t, a_t) \leftarrow$
$Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}] - Q(s_t, a_t)$
$s \leftarrow s'$
s is terminal
until convergence

In the Q-learning algorithm, the future action a' is taken using greedy policy i.e. choose an action which has maximum Q-value of the next state. The convergence in Q-learning may be speed up by using eligibility traces which are updated in Q-learning(λ) as follows:

$$e_t(s,a) = Iss_t.Iaa_t + \gamma\lambda et - 1(s,a) \quad \text{if } Q_{t-1} = \max_a Q_{t-1}(s_t, a)$$
$$e_t(s,a) = 0 \quad \text{otherwise} \tag{25}$$

Iss_t is used here for identity indication. For $s = s_t$, the Iss_t and Iaa_t are equal to 1. The δ for Q-learning is defined as:

$$\delta_s = r_{t+1} + r \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t) \tag{26}$$

The resulting update rule for the Q-learning algorithm by using trace is:

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha\delta_t e_t(s,a) \quad \text{for all } s \in S \tag{27}$$

The difference between Q-learning and SARSA is shown in table 5.

TABLE 5. Difference between Q-LEARNING and SARSA.

Q-learning	SARSA
1. Move one step selecting a_t from $u(s_t)$	1. Move one step selecting a_t from $\pi(s_t)$
2. Observe r_{t+1}, s_{t+1}	2. Observe $r_{t+1}, s_{t+1}, a_{t+1}$
3. Update the state-action function $Q(s_t, a_t)$	3. Update the state-action function $Q(s_t, a_t)$
4. (optional) Update the policy $\pi(s_t) \longleftarrow \text{argmax}_a(s_t, a_t)$	4. Update the policy $\pi(s_t) \longleftarrow \text{argmax}_a Q(s_t, a_t)$

D. ACTOR-CRITIC ALGORITHM

Actor-Critic (AC) [19] is a hybrid approach based on both policy and value function [20]. The value function is estimated by critic as described by equation (28) while the policy is updated by actor based on the feedback from critic. The AC method may be used for small as well as for large state-action spaces. For problems having small state and action space, the critic is estimated through Q-function

Algorithm 5: Actor-Critic Algorithm [2]

Initialization
Rewards for state-action pairs $R_{s,a}$
$\gamma = 0.9$
initialize s
1. select a_t in s_t
2. Get s_{t+1}
3. Get R_{s_t,a_t}
4. Update state s_t utility function (critic)
$\mathrm{U}(s_t) \leftarrow U(s_t) + \alpha[r_{t+1}) + \gamma U(s_{t+1)-U(s_t)]}$
5. Update the probability of a_t using error (actor)
$\delta = r_{t+1} + \gamma U(s_{t+1)-U(s_t)}$

while the Boltzmann policy or $\epsilon-$ greedy is used for actor policy estimation.

$$V^{\pi}\left(s'\right) = \sum_{a} \pi\left(s,a\right) \sum_{s'} p\left(s'|s,a\right)\left[R\left(s,s',a\right) + \gamma V^{\pi}\left(s'\right)\right] \quad (28)$$

$$Q^{\pi}\left(s,a\right) = \sum_{s'} p\left(s'|s,a\right)\left[R\left(s,s',a\right) + \gamma Q^{\pi}\left(s',a'\right)\right] \quad (29)$$

The working flow of the actor critic algorithm is given in figure 3 and pseudo-code in algorithm 5.

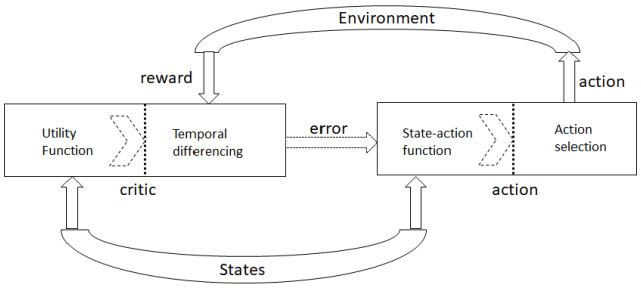

FIGURE 3. Actor-critic algorithm architecture.

The critic part of AC does estimation of the value function which may be the Q value (action-value) or the V value (state-value). While the actor performs updates on the policy distribution as guided by the critic (for example with policy gradients). If both the actor and critic functions are parameterized with NN then it is called Q Actor Critic. The other useful variants of AC are Advantage Actor Critic (A2C) and Asynchronous Advantage Actor Critic (A3C). These two methods have potential applications in recent achievements of RL.

E. BAYESIAN METHODS

Bayesian RL methods may be a natural optimization solution to exploration-exploitation dilemma due to their ability to capture uncertainty by probability distribution [21]. Myopic value of information [17], policy gradient, POMDP discretization, upper confidence bound, Bayesian sparse sampling, BEETLE and Thompson sampling [22] are some of the famous methods that are used for Bayesian approximations.

F. DEEP Q NETWORK

Deep Q-Network (DQN) is considered as the main utilizer of Neural Network (NN) [2]. Main focus of this algorithm was to deal with the different environmental variables having large state space. Variable estimation is the main goal to be addressed. The equation (30) focuses on deploying Q-learning method for updating variables using NN.

$$r_j + \gamma \max_{a'} Q(\phi_{j+1}, a', \theta^{-}) \quad (30)$$

where, the ϕ is used for state s, while θ is a parameter used in the NN. RL based scenarios having high dimensional values can be boosted up with the help of Neural Network (NN). Newly arriving unpredicted states can be approximated with the help of DQN. The behavior of functions based on active values, when approximated with the help of nonlinear function provides models that are totally divergent from normal values and behaves unstably. NN based non-linear actions can be approximated in the same way.

Following are the issues that are directly related to such behavior:

- Co-related sequence of data samples.
- Alteration in Q-values results into quick changes in designed policy.
- Data switching from one location to another location is possible.
- Back propagation of Naïve Q-Learning gradient may result into unstable solutions.

The Deep Q-Network (DQN) method is presented in algorithm 6. Experience Replay keeps experiences which contain action, corresponding state transitions and resulting rewards and mini-batches are made for the update of NN. Q value estimation requires model that has instability based parameters. NN based Production of models results into estimation of Q-value function. For this purpose, two NN are utilized normally and named after Q-value and other is named after target Q-Network.

Value estimation obtained from operations are almost identical to the values obtained from target Q-Network. In this computational structure, target function based parameters are adjusted with respect to target functions and these computational values are updated on current network during execution of each step. Therefore, functionality of loss function can be described as the error of squared network Q-Value and target Q-value. As a result, fluctuation of desired operation becomes less dramatic and provides smooth and stabilized training.

From past few years, deep reinforcement learning has attracted the attention of many researchers due to its ability to learn from the actions and to be able to work in continuously changing dynamic environments [23], [24] and [25]. Controlling a robotic manipulator like a helicopter or playing a computer game are few applications of deep RL.

Algorithm 6: Deep-Q Network Algorithm

Setup RM
Initialize Q(s,a)
Initialize target
repeat
  Initialize sequence $s_1 = x_1$ and prepocessed sequence $\theta_1 = \theta(s_1)$
  repeat
    Either choose randomly a_i with probability ϵ or choose $a_i = argmax_a(Q(\phi(s_t), a; \theta))$
    Get r_{t+1} and x_{t+1}
    $s_{t+1} = (s_t, a_t, x_{t+1})$
    $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_{t+1}, \phi_{t+1})$ in RM
    Sample random minibatch of transitions $(\phi_t, a_t, r_{t+1}, \phi_{t+1})$ from RM
    if *episode ends at step k+1* **then**
      $y_k = r_k$
    else
      $y_k = r_k + \gamma \max_{a'} Q(\phi_{k+1}, a', \theta^-)$
    end
    Have a gradient descent step on (y_k - Q(ϕ_k, a_k, θ))
    Every C steps set $Q^* = Q$
  until *t = 1, T*;
until *Episode = 1, M*;

TABLE 6. Difference between deep Q learning and deep recurrent Q learning.

Deep Q Learning	Deep Recurrent Q Learning
1. This approach cannot handle past dependencies as agents in this network cannot handle older things than a fixed size.	1. This approach can memorize and handle past dependencies.
2. This network is easier and faster to train.	2. This network is trickier and slower to train.
3. This network is not suitable for partially observable environments as more than one observations are required to understand the current state.	3. It is suited to deal with partially observable environments and can estimate parameters in an environment like in a 3-dimensional video game.
4. Can be used for those applications or games that require short term planning like Traffic Light Control.	4.Can be used for those applications or games that require long term planning like 3D games.

A lot of deep reinforcement learning models have been proposed, comparison of two widely used networks are given in table 6.

V. APPLICATIONS

This section presents some important applications of reinforcement learning in healthcare, robotics and autonomous control, communication and networking, natural language processing, games and self-organized system, scheduling management and configuration of resources, IoT and computer vision.

A. HEALTHCARE

Usage of machine learning has showed extraordinary results in the field of health care. Researchers initially focused on usage of machine learning algorithms to diagnose the conditions or forecasting the outcomes of treatments. These tasks are also important but determining the best treatment policy to use for a specific patient is a challenging task and cannot be performed using traditional approaches. Reinforcement learning is a promising candidate for solving such problems of health care.

Over the past years, RL has found important applications in the healthcare sector [26]. For example, a few works on the treatment of Epilepsy disease by using RL are: ([27], [28] and [29]). Cancer is a more threatening disease in human life. Use of RL in cancer research is not new and have been used in cancer detection, classification of tumors and diagnosis. Many useful contribution of RL in the treatment of cancer in recent years include: ([30]–[35] and [36]). We have also the seen use of RL in the treatment of schizophrenia, anemia, and sepsis e.g. [37], ([38]–[41] and [42]) and ([43]–[46] and [47]) respectively.

RL and deep learning architectures have been used effectively in analyzing medical images obtained from positron emission tomography, CT scan, magnetic resonance, radiography, ultrasound, and microscope for segmentation, detection and classification. Few examples on use of RL in medical imaging are: [48]–[54]. Moreover, ([55]–[59] and [60]) highlight effective implementation of RL for the diagnostic systems.

Now a days, clinicians are more keen to personalized treatment of patients [61]. There are many important contributions of RL which make personalized treatment more realistic and gettable objective such as: ([62]–[68] and [69]).

RL algorithms are commonly used in dialogue system and chat-bots. Some of the good applications of RL in dialogue systems are ([70]–[83] and [84]). Similarly, there is potentiality of RL to be used in control systems. Few RL based control systems deployed in healthcare industry are: ([85]–[93] and [94]).

The Rehabilitation process is an important part of the treatment and we can see a few works of RL which provide assistance in different types of rehabilitation i.e. motor rehabilitation ([95], [96]; cognitive rehabilitation [97]–[99] and gait rehabilitation [100] and [101]) respectively. RL may also be applied for risk management the during nuclear medical examination at hospitals [102] and healthcare examinations centers ([103] and [104]). Some studies using DL for cognitive impaired people are found in [105] and [106].

Another critical aspect of the treatment process is the use of medicines at home as advised by the clinician. But post-hospitalization treatment at home may be challenging for elderly persons and patients having cognitive or physical disabilities which lead to medication error. In Europe, most of the drugs taken by patients are done completely wrong, causing the death of 195,000 per year according to a report of

'Cittadinanzattiva-Patient Rights Tribunal'. Recently, some works have been presented to assist patients at home during treatment process and minimize medication errors by using RL and deep learning [107]–[109].

B. ROBOTICS AND AUTONOMOUS CONTROL

Many types of problems in autonomous control and robotics may be model as reinforcement learning problem ([110]–[112] and [113]). Trial and error mechanism of RL architecture helps a robot and control system to autonomously learn an optimal behavior by interacting with the environment ([114] and [115]).

Some of the RL applications to solve robotic problems by using model-based algorithms include ([116]–[118] and [119]). These works were proposed to enable a robot for the penalty kick, navigation task, vision-based mobile robot docking task and the task of obstacle avoidance respectively.

Robotic applications using RL model free methods are ([120]–[127] and [128]). The RL method used in [120] trains a dynamical system motor primitives for pouring water. A robot is controlled by a human teacher to reach a task in [121]. Gaussian process regression is applied in [122], [128] and [123] to learn grasping. a humanoid robot is enabled to learn a pouring task in [124] and balancing a ball in a beam in [125] respectively. The 'Brainstorner Tribots' introduced in [126] has won 'Robocup 2006 Midsize league' and can learn different skills like: penalty shots, defenses, dribbling, interception, kicking, motor speed and position control. A dog robot is trained to learn transporting a ball in [127]. The work of [129] solves the cart pole task in less than twenty seconds while a visually driven block-stacking problem is learned in [130].

Soft robots have been getting significant attention in the industrial sector in the last two decades due to their superior efficiency and accuracy. Deep RL has been widely used for different manipulation tasks such as: reaching, door opening, picking and dropping ([131]–[135]).

Deep RL has been successfully and effectively used in soft robotic navigation to assist many control systems and robots to perform functionality autonomously, for example, autonomous driving tasks. Few important studies related to use of Deep RL methods on navigation are: [136]–[144].

Moreover, balancing a two-wheel robot [145], learning helicopter flight [146], selecting striking movements in a table tennis [147], a robot dog learns a jumping behavior and driving a radio-controlled car [148], setting the meta actions for dart throwing [149] and learning of biped walk patterns [150], [151] are all applications of RL in robotic and autonomous control systems.

C. COMMUNICATION AND NETWORKING

Reinforcement learning, especially deep RL can be potentially applied to solve many challenges and problems related to communication and networking [152]. Modern networks such as Unmanned Aerial Vehicle (UAV), Heterogeneous Networks (HeNets) and Internet of Things (IoT) may be autonomous, ad-hoc and decentralized if network entities i.e. UAVs, mobile users and IoT devices are able to take autonomous decisions e.g. data rate selection, spectrum access, base station association and transmit power control. In recent years, deep RL has been applied in communications and networks successfully to achieve the desired objective.

Applications of deep RL method related to interference management and multiple radio access are analyzed in [153]. Methods to address network access and physical layer modulation problems are reviewed in [154] while [155] discuss deep RL algorithms traffic network control. Potential uses of deep RL in network cyber security are surveyed in [156].

Adaptive rate control, joint user association, spectrum access and dynamic spectrum access are the important issues which may be addressed by the use of deep RL. Deep Q Network (DQN) architecture with Long Short-Term Memory (LSTM) [157], DQN using feed-forward neural network [158], DQN with LSTM and actor-critic [159], DQN with RNN and CNN [160]; and DQN with actor-critic [161] are few examples related to the adaptive rate control issue. The network access issue is considered in ([162]–[170]).

A deep Q-learning method is presented in [171] for efficient allocation of energy resources in green wireless networks. The method has also been applied for improvement in the performance of Vehicular ad doc Networks (VN) ([172]–[174]). The same deep Q-learning technique for VN then is generalized in [175] for smart city applications. The two use cases of smart cities by using generalized deep Q-learning are referred in [176] and [177]. A joint design of computing, caching and communications in VN to minimize cost is proposed in [178].

Caching and offloading are among the key processes of information centering networking. Modeling, optimization and wireless proactive caching are discussed in: [171]–[176], [179]–[185]. Different policies using deep RL to address data and computation offloading are proposed in: [186]–[191].

Moreover, RL algorithms may be used to protect networks from attacks e.g. cyber-physical attack, jamming attack and denial of service. DQN is commonly used to address network security issues. Few examples of DQN in UAV, IoT and other networks are: [192]–[199]. Connectivity preservation is another important parameter of any network. Actor-critic method and DQN have been applied in communication networks to maintain connectivity. Few examples on application of RL related to preserving connectivity are: [57], [200]–[202].

In addition, few other miscellaneous applications of RL related to communications and networking include user localization ([72], [203], [204]); direction of arrival estimation [205]; data collection ([206]–[210]); power control ([211]–[213]); signal detection [214] and traffic engineering and routing ([214]–[221]).

D. NATURAL LANGUAGE PROCESSING

This section discusses the potential applications of reinforcement learning in Natural Language Processing (NLP). NLP is the process of extracting content out of a raw and unstructured

text. Although unsupervised learning methods are good but they need a lot of data to acquire something meaningful and consequently to extract the patterns out of the text. RL can be the better method for NLP since RL based agent learns the behavior of a trainer in a simulated world through trial and error.

Many interesting tasks related to NLP like: text games [222], language to program execution [223], extraction of information [224], sentiment analysis [225], knowledge graph reasoning [226], automatic query reformulation [227], information retrival [228], summarization [229], [230] and semantic parsing [231] where RL can contribute significantly.

Initially, RL was used mainly for synthetic language [232] and small corpa [233] but advancement in representation learning ([234]–[236]) makes it possible to design models that obtain knowledge from text corpa. The acquired knowledge then may be integrated into subsequent decision-making tasks. Few examples of transfer knowledge from word representation to downstream language tasks are: [237]–[241].

The literature related to applications of RL in NLP maybe divided into language-assisted RL and language conditional RL [242]. In the former case, the language is used to aid learning while the interaction with the specific language in later case, is initiated by the task formulation at hand itself. In some scenario, both categories may be used together [243], [244].

Language-assisted RL then further may be categorized into language for structuring policies and language for communication knowledge. To discuss works related to the first branch, a neural architecture for visual questions and answers proposes by [245] is applied to RL problems in [246] and [244]. Few other works on structuring policies are [247]–[249] and [250] while some articles concerning the communicating knowledge include [251], [252] and [253].

Integration of natural language in language conditional RL is unavoidable and can be discussed as instruction following RL agents. There is lot of work related to this specific area, for example: [254], [254]–[260] and [261]. Furthermore, RL agents can be used for optimization by inducing reward with the use of instructions such as: [262]–[265] and [266].

Some of the recent works related to the text generation in NLP are [267]–[269] and [270]. Machine translators based on new deep learning techniques have been achieving dominance over conventional machine translators. Some of the examples of modern machine learning translators can be found in [271]–[274] and [275].

E. AUTONOMOUS IoT

The integration of autonomous control system and Internet of Things (IoT) results in autonomous IoT (AIoT) systems. Reinforcement learning algorithms introduce ambient intelligence into AIoT systems through the provision of solutions to the closed-loop tasks of processing the sensory data to make control decisions [276]. AIoT is relatively a new trend and it has not been studied adequately. We next present some recent works on the applications of RL in autonomous IoT system.

An IoT-enabled mobile robot to monitor plant health using Q-learning and CNN based method is proposed in [277]. Q-learning based global routing agent is presented in [278] to maintain a balance between the lifetime of the network and overall delay. Q-learning also used for optimization of energy consumption and the delay in [279]. AIOT systems for energy storage management and energy trading process are discussed in [280], [281] and ([282]–[287]) respectively. Some other applications of Q-learning for AIOT may be seen in ([288]–[298]).

RL actor-critic have been widely used in AIOT systems to learn stochastic polices mostly for continuous state or action space problems and also for discrete state problems in few cases. Moreover, actor-critic method may be used to train deep RL models with fewer computational resources and samples. Some examples on use of RL actor-critic methods in AIOT systems are: ([299]–[307]).

Deep Q-Networks (DQN)algorithms are most frequently adopted in AIOT systems in recent years. A few applications of DQN and Double DQN (DDQN) in IoT communications systems are: [213], [308]–[310]and [311]; in IoT Cloud/Fog/Edge computing are: [312]–[315] and [316]; in autonomous IoT robotics are: [277], [317]–[319] and [320]; in IoT smart vehicles are: [174], [321] and [292] and in smart grids are: [322]–[324] and [325] respectively.

F. COMPUTER VISION

Computer vision is an analysis of the process in which computer machine gets understanding from videos and images. More precisely, it is used for image analysis i.e. recognition, scene understanding and classification, motion analysis, visual control and integration with natural language processing. Image processing involves solving decision-making tasks and some recent works are witnessed of successful use of reinforcement learning to such problems.

Deep Q Networks attracted the attention of many researchers for solving image processing tasks but their usage was limited to simple applications. Introduction of pixel-wise rewards in Reinforcement learning (PixelRL) makes it a better choice to solve tasks like image restoration, color enhancement and image denoising [326]. In PixelRL approach, reinforcement learning (RL) is combined with state-of-the-art image processing techniques like Convolutional Neural Networks (CNN) to solve real-time complex computer vision problems. Scheduling of important tasks or finding a shortest path between two points in images are few more examples where CNN extracts the features from images and RL learns the optimized way to proceed and perform scheduled task [326], [327] and [328].

Reading maps [329], face hallucination [330], view planning [331] and semantic parsing [332] are other applications of RL in computer vision. We can also see the connection

between generative adversarial networks, RL such as actor critic algorithm and inverse RL in [333] and [334] respectively.

A tree-structure RL method for sequential object searching, attention aware deep RL algorithm to recognition in video, multi agent RL for searching joint object, policy search for detection of visual object, a hierarchical framework for image classification, POMDP and deep RL methods for motion analysis of the object, have been proposed in [335]–[341] and [342] respectively.

G. GAMES AND SELF-ORGANIZED SYSTEMS

Reinforcement learning is playing a vital contribution in creating efficient algorithms for video games [343] and [344]. A lot of works are going on to train agents to interact with their surroundings using automatic feature engineering to outperform the human intelligence and achieved super intelligence [345] and [346].

RL recently outperformed the human intelligence and achieved super intelligence in strategic games [347]. These strategical features can also be useful in self-organized systems (like cellular networks etc) to improve their performances [348].

Reinforcement learning is revolutionizing almost every category of games, from classic arcade games having two-dimensional perfect information [349], [350] and [351] to Real-time strategic games having three-dimensional imperfect information [352], [353] and [354] from single-agent 3D games like TORCS [355] to multi-agent 3D-games like Quake III [354].

This progress is only becoming possible due to the publicly available research platforms [356]–[358] and [359] and competitions [360] and [349] that are making a huge contribution in the development of artificial intelligence in games.

Deep Mind of Google introduces a framework for writing games and evaluating the performance of artificial intelligence on a variety of benchmark games [361]. A software platform, ViZDoom, is proposed in [362] for machine learning research that uses raw visual information for learning and reduces the need of high-level information.

These platforms provide full control of the environment to train an agent or bot for different situations. These strategical features can also be useful in self-organized systems (like cellular networks etc) to improve their performances [347] and [348].

DeepMind's StarCraft II [363] and Facebook's TorchCraft [364] Learning Environment respectively provide interfaces to the StarCraft II and StarCraft real-time strategy games, presenting challenges in long term planning and micromanagement. To provide more flexible environments, (a). DeepMind Lab was developed on top of the Quake III Arena first-person shooter engine [365], (b) Microsoft's Project Malmo exposed an interface to the Minecraft sandbox game [366]. Both these environments give customizable platforms in 3D environments to RL agents.

H. SCHEDULING, MANAGEMENT AND CONFIGURATION OF RESOURCES

Reinforcement learning plays an important role in scheduling, allocation and management of limited resources ([367], [368] and [369]). These resources can be computational power of processing units, available storage or network bandwidth, etc. The RL algorithms are useful to configure the network resources efficiently and to optimize the routing capabilities. ([370] and [371]).

Optimal resource allocation is key network capacity, system enhancement and coverage. Resource allocation for radio access network and space communication by using deep Q-learning architecture are proposed in [372] and [373] respectively. Applications of RL on the use of network slicing for resource management are reviewed in [374]. Deep Q-learning method is used in [375] for optimization of power control at the based station which provides a better prospect then the conventional power allocation methods e.g factional programming [376]. A hierarchical deep RL architecture for power management and resource allocation in cloud computing is presented in [377] and the proposed method was validated with Google cluster traces.

Efficient user scheduling in multi-user Multiple Input Multiple Output (MIMO) and massive MIMO systems are very important for better utilization of available resources at base station [378]–[380] and [381]. The RL policy gradient algorithms studied in [382], are used for user scheduling in [383] and for resource management of computer systems in [384] respectively and shows superior performance. RL actor-critic algorithm also used for scheduling in the real time data flow processing system in [385] and [386]. Scheduling for resource management to address multi-resource cluster with policy gradient using deep RL proposed in [387] and the proposed method was validated with simulation.

RL algorithms have been applied for actively detecting localizing objects in scenes [388], multiple object searching [389], visual tracking [390],selection of the best descriptor out of given set of features and then choosing best classifier from given set [391] and [392], processing of monocular images to predict collision-free motor commands [393] and predicting prominent features out of large scale visual features using fast classifier [394].

I. MISCELLANEOUS APPLICATIONS

RL is an exciting machine learning field and the future of applied AI in many areas. It is not only limited to only above mentioned sectors. [395] and [396] can be use to study education-related RL applications. Similarly, some applications related to finance and business management are: [397]–[403] and [404]–[407] respectively.

Q-learning based applications in the small grid are [293] and [408] and used on electric power system control and for the decision process. RL applications addressing industrial problems are discussed in [409] and [410]. Works of [411], [412] and [37] discuss the role of RL related to security

and privacy. Moreover, RL has potential uses in intelligent transportation such as self driving cars [413], [413] and [414].

J. CHALLENGES

Reinforcement learning is showing outstanding results in real-world scenarios, however, there are a lot of challenges and difficulties that make implementation of reinforcement learning difficult [110], [415], [416] and [417]. By identifying the challenges and addressing the issues, more applicable RL approaches can be developed for real-world scenarios. Few of the challenges and difficulties are discussed below:

Efficiency of samples is an important parameter that needs to be considered in reinforcement learning [415]. If the data samples are limited in numbers in case of a real-world scenario, the learning algorithm would have very low exploration of the environment from training data as the information and states are not fully covered in used samples. To handle this issue, the efficiency of data should be good enough or a reinforcement learning approach should be sample-efficient to learn the environment from limited amount of data [418] and [419].

There can be a large number of action and state spaces in complex real-world scenarios that can make the implementation of a RL approach difficult [415] and [420]. A lot of approaches have been proposed to divide such high-dimensional states among different candidates to reduce the complexity of a given task [421].

Satisfying the safety guidelines is also an important constraint and challenge that should be considered before using a specific decision-making algorithm or approach [415] and [422]. Following safety constraints during the operation phase as well as in the exploratory phase are important for reinforcement learning approaches, so a system cannot destroy itself or and its environment in a real-world scenario.

As mentioned earlier, non-stationary environments are a critical issue for reinforcement learning [415]. Most of the real-world applications have partial observability and the complete information related to environment and conditions is not available, therefore, this type of tasks is difficult to handle. Furthermore, slow response by agents in a computationally-intensive application and system delays also affect the realization of reinforcement learning [415].

Bias (B) and variance in RL is a significant issue that may affect significantly the results of a learning process [423] refer also to how well the reinforcement signal represents the actual reward function for the given environment [424].

Let us focus on the value function (analogues considerations hold for the action value function). Being $v_\pi(S_t)$ the actual value function, iterative methods estimate $V_\pi(S_t)$ by iterating the following formula:

$$V_\pi(S_t) \leftarrow V_\pi(S_t) + \alpha(Tg - V_\pi(S_t)) \tag{31}$$

On every update, the estimate gets closer to the Target (Tg), but there are different ways to define the Target depending on the method adopted.

In MC method, as discussed in section IV-B, the Target is defined as the return G_t:

$$Tg = G_t = \sum_{j=0}^{T} \gamma^j R_{t+1+j} \tag{32}$$

Therefore, the estimate of the value function is updated at the end of the episode. In contrast, in TD methods the Target is obtained without waiting the end of the episode, but bootstrapping; i.e., adding the immediate reward to the estimate of the future rewards:

$$Tg = R_{t+1} + \gamma V_\pi(S_{t+1}) \tag{33}$$

The bias of an estimator Θ of a function θ is defined as the difference between the estimator's expected value $\mathbb{E}[\Theta]$ and the true value of the function being estimated. Therefore

$$B = \mathbb{E}[Tg] - v_\pi(S_t) \tag{34}$$

In case of MC method, this equation becomes $\mathbb{E}[G_t] - v_\pi(S_t)$, but in RL $v_\pi(S_t) = \mathbb{E}[G_t]$ by definition, consequently, MC method is unbiased. In contrast, in TD method, the Target depends on $V_\pi(S_{t+1})$, which is an estimate of the value function at time $t+1$, thus it is biased.

In the case of Deep RL, the value estimate is often obtained using a deep NN, making things worse. For example, in DQN the Q-estimates are computed using an old copy of the network (a ''target'' network), that will provide ''older'' Q-estimates, with a very specific kind of bias, relating to the belief of an outdated model.

Concerning the variance of an estimator, it measures the noise introduced by stochastic events. In case of MC method, the Target (i.e., G_t) is the sum of all rewards collected until the end of the episode. It depends both on the sequence of all actions taken and on possible actual trajectories. In particular, taking action A_t from S_t, the agent may experiment different trajectories resulting even in very different cumulative rewards. Therefore, the variance may be high. In contrast, in TD method the $Target_{TD} = R_{t+1} + \gamma V_\pi(S_{t+1})$ has much less stochastic components. Consequently, the degree of variance is minor.

Many techniques attempt to mitigate the negative effect of too much bias or too much variance in the reward signal. Few of them are Proximal Policy Optimization (PPO), asynchronous advantage AC, trust region policy optimization, and others [425]. MC or TD can give good performance for simple environments but for complex environments $n - step$ bootstrapping can boost learning significantly at the expense of an extra hyperparameter n.

A combination of online and offline environments is applied for a bias-variance trade-off in [426]. Previous works [427] and [428] have handled the issues involved in balancing bias and variance. The most common methods are to reduce the variance of an estimate while keeping the bias unchanged. The baselines of these studies [427] and [428], are of the policy gradient [429], which utilizes an actor, who defines

the policy, and a critic, who provides a more reduced variance reward structure to update the actor.

A DL model having high bias value oversimplifies the model that causes a model to miss relevant information and features, leading to high error on both training and testing data. While, a model having high variance value does not generalize the unseen data, causing a model to perform well on training data but shows high error rate on testing data. An optimal balance is required between these two values that would never underfit or overfit the required DL model. A number of techniques have been proposed by researchers to estimate and control the bias-variance trade-off in a DL model [430] and [431].

K. LESSON LEARNED

While discussing applications of RL techniques, we noted few issues that should addressed in future. At the same time RL methods have strong points in the development of intelligent systems applications. Next we list some learned lessons.

A reward function definition is important to obtain advantages of RL methods. Its choice can be non-trivial as long as it is crucial to consider the time scale e.g. the balance between a small immediate gain or a large future benefit and the expert knowledge. Reward may be learned by use of IRL techniques if trajectories of expert choices are present.

An implicit balance between exploitation i.e. selecting already known rewarding actions and exploration i.e. to search new actions which may benifit more, is a major issue for any RL algorithm especially in large states environments. $\epsilon-$Greedy, Thompson sampling [22], information gain [432] and optimistic exploration [433] can provide solution to exploration-exploitation dilemma.

Some problems like healthcare domain may have partial observation of environments. Conventional RL methods assume environments are fully observable i.e Markov environments. MDP can be generalized to Partially Observable MDP (POMDP) by restricting the available state information as for example done in [434] and [435].

Dealing with non-stationary environments is another problems in machine learning. Conventional machine learning methods need periodic retraining to mitigate with non-stationary environments. However, online RL techniques can cope this issue partially and adapt the policy to a dynamic non-stationary environment by granting some exploration, unless the changes happen often.

In the real world, environments may be non-stationary and demand sequential decision-making under uncertainty [436], [437] and [438]. The non-stationary rewards and transition probabilities make these types of applications a difficult domain to handle. However, reinforcement learning performs well in case of sequential decision-making under uncertainty for example by giving more weight to recent rewards than to long-past rewards [2], [439], [440] and [441].

Some RL methods depends on a sequential try and error procedures that few times can be time consuming, in particular Deep RL algorithms. The scientific community recently has worked with the industry achieved results on this issue. A common solution is to have different agents perform learning individually or learn separate parts of the problem and then extract a common knowledge.

In many real-time applications, a major issue is the availability of large datasets for training. Often, the available amount of data is not sufficient to train properly the models. However, limited amounts of real-time data can be expanded into a large set of realistic data using state-of-the-art techniques and networks like generative adversarial networks (GANs) that generate data useful for training synthetically [442]–[445], and [446].

Synthetic data generation using Generative Adversarial Networks is preferable as compared to other augmentation techniques [447] as RL agents can be exposed to a broad range of extreme conditions by combining both real and synthetic data that allows the model to deal with unpredicted rare events [442], [448] and [449]. Unlike other augmentation techniques, GANs also eliminate data set biases in generated data for proper training of a machine learning model and make synthetic data indistinguishable from real data [450] and [451]. Others solutions are using of transfer learning [452] for the training of a model over larger datasets and then use a pre-trained model on small datasets.

VI. DISCUSSION

Convergence of RL methods may depend on the selection of the hyperparameters such as example discount factor (gamma), epsilon for exploration, eligibility traces (Lambda), and on the sequence of averaging steps. Convergence theorems can find bounds for hyperparameters. More recommendations on convergence may be studied in the article ''Convergence of reinforcement learning algorithms and acceleration of learning'' [453].

According to [454] ''policy gradient methods perform gradient ascent on the objective function, and the convergence is guaranteed. Value-based RL algorithms, instead, minimize the ''Bellman error'' of fit but, in the worst case, it is not guaranteed to converge in the nonlinear cases. Differently, model-based RL techniques minimizes the error of fit and the model is guaranteed to converge.''

RL algorithms may guarantee convergence to the optimal policy when an agent can adequately experiment and the given environment is Markovian. However, in multi-agent cases, multi RL agents perform learning in a shared environment, that may not be an MDP model. Therefore, in multi agent systems the optimal policy of an RL agent not only depends on the environment, but also on the policies of the other agents. This scenario arises naturally in many applications, such as: traffic light control, auctions, distributed control, economics, telecommunications, robotics, etc. The multi-agent RL is applied in these fields because control is inherently decentralized or the complexity of the applications. In these systems it is crucial that RL agents are able to find solutions by competing and/or coordinating with other RL agents [455].

We have been seeing the enormous growth, exciting new techniques, breakthroughs and many real-life applications. We expect and believe to witness even more, much faster and efficient. As a result, this report is not complete yet regarding width and depth but we have attempted to summarize the important applications of reinforcement learning and its variants in various fields.

In this article, we have introduced basic concepts of RL and its algorithms in an easy and simple way. Then we have summarized RL applications in healthcare, robotics and autonomous control, communication and networking, natural language processing, games and self-organized system, scheduling management and configuration of resources, autonomous IoT, and computer vision.

At the end, it is important to revisit the comprehensive target of all of present articles: the development of AI, in particular, RL based systems that are able to interact with the given environment and learn. Interaction with the world around is the advantage and disadvantage of RL at the same time. Whilst having many challenges in an attempt to get understanding of our continuously changing and complex world, RL permits us to select how we explore it. In effect, RL provide the agent with the ability to do experiments to get a better understanding of the environment, enabling the agent to learn even complex and high-level causal relationships [456].

The availability of physics engines and high-quality visual renderers now encourage us to move forward in this direction, with research works that are trying to learn intuitive models of physics in visual world [457]. Presence of challenges in the real world will be possible, but steady development in being done in agents that learn the fundamental principles of the environment through interaction and action. Conceivably, then, we are not too far away from RL based AI systems that interact, observe, learn and act in more human-like ways in more complex environments.

VII. CONCLUSION

Reinforcement learning provides a technically and mathematically solid solution for optimal decision making in many challenging tasks having multi-dimensional, noisy data, complex nonlinear dynamics, sequential decision procedures with delayed rewards. This work intends to present a comprehensive report of RL applications to many important areas including healthcare, robotics, communications and networking, natural language processing, internet of things, computer vision, games and scheduling.

REFERENCES

[1] T. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.
[2] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, vol. 2, 2nd ed. Cambridge, MA, USA: MIT Press, 2015.
[3] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2, pp. 235–256, 2002.
[4] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[5] N. C. Russell, J. M. Malik, and D. D. Edwards, *Artificial Intelligence: A Modern Approach*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 2003.
[6] Q. J. Huys, A. Cruickshank, and P. Seriès, "Reward-based learning, model-based and model-free," in *Encyclopedia of Computational Neuroscienc*. New York, NY, USA: Springer, 2015, pp. 2634–2641.
[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: http://arxiv.org/abs/1509.02971
[8] Y. Duan, "Benchmarking deep reinforcement learning for continuous control," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.
[9] Z. Kurth-Nelson and A. D. Redish, "Temporal-difference reinforcement learning with distributed representations," *PLoS ONE*, vol. 4, no. 10, p. e7362, Oct. 2009.
[10] B. Miranda, W. M. N. Malalasekera, T. E. Behrens, P. Dayan, and S. W. Kennerley, "Combined model-free and model-sensitive reinforcement learning in non-human primates," *PLOS Comput. Biol.*, vol. 16, no. 6, Jun. 2020, Art. no. e1007944.
[11] B. Kurdi, S. J. Gershman, and M. R. Banaji, "Model-free and model-based learning processes in the updating of explicit and implicit evaluations," *Proc. Nat. Acad. Sci. USA*, vol. 116, no. 13, pp. 6035–6044, Mar. 2019.
[12] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, and S. Levine, "Combining model-based and model-free updates for trajectory-centric reinforcement learning," 2017, *arXiv:1703.03078*. [Online]. Available: http://arxiv.org/abs/1703.03078
[13] S. Grimmett, "Probability and random processes," Tech. Rep., 1992, vol. 2.
[14] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, Aug. 1988.
[15] A. Barto and R. S. Sutton, *Reinforcement Learning: An Introduction*. 1998.
[16] G. A. Rummery and M. Niranjan, *On-Line Q-Learning Using Connectionist Systems*. 1994.
[17] C. Watkins, "Learning from delayed rewards," Tech. Rep., 1989, vol. 3, no. 1.
[18] F. Melo, "Convergence of Q-learning: A simple proof," Inst. Syst. Robot., Instituto Superior Técnico, Lisboa, Portugal, Tech. Rep.
[19] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1008–1014.
[20] P. Massimiliano, "A developmental model of trust in humanoid robot," Univ. Plymouth, Plymouth, U.K., Tech. Rep., 2018.
[21] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 681–688. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104482.3104568
[22] M. Strens, "A Bayesian framework for reinforcement learning," Tech. Rep., 2000.
[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*. [Online]. Available: http://arxiv.org/abs/1312.5602
[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[25] M. Hausknecht and P. Stone, "Deep recurrent Q-Learning for partially observable MDPs," 2015, *arXiv:1507.06527*. [Online]. Available: http://arxiv.org/abs/1507.06527
[26] A. Coronato, M. Naeem, G. De Pietro, and G. Paragliola, "Reinforcement learning for intelligent healthcare applications: A survey," *Artif. Intell. Med.*, vol. 109, Sep. 2020, Art. no. 101964. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S093336572031229X, doi: 10.1016/j.artmed.2020.101964.
[27] A. Guez, R. Vincent, M. Avoli, and J. Pineau, "Adaptive treatment of epilepsy via batch-mode reinforcement learning," in *Proc. AAAI*, 2008, pp. 1671–1678.
[28] T. Högberg, "Widening bottlenecks in drug discovery glimpses from drug discovery technology europe 2005," *Drug Discovery Today*, vol. 10, no. 12, pp. 820–822, Jun. 2005.

[29] *Challenge and Opportunity on the Critical Path to New Medical Products*, U.S. Dept. Health Hum. Services Food Drug Admin., 2004.

[30] K. Humphrey. (2017). *Using Reinforcement Learning to Personalize Dosing Strategies in a Simulated Cancer Trial With High Dimensional Data*. [Online]. Available: http://hdl.handle.net/10150/625341

[31] Y. Zhao, M. R. Kosorok, and D. Zeng, "Reinforcement learning design for cancer clinical trials," *Statist. Med.*, vol. 28, no. 26, pp. 3294–3315, Nov. 2009.

[32] C. J. Watkins and P. Dayan, "Technical note: Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992, doi: 10.1023/A:1022676722315.

[33] Y. Zhao, D. Zeng, M. A. Socinski, and M. R. Kosorok, "Reinforcement learning strategies for clinical trials in nonsmall cell lung cancer," *Biometrics*, vol. 67, no. 4, pp. 1422–1433, Dec. 2011, doi: 10.1111/j.1541-0420.2011.01572.x.

[34] Y. Liu, B. Logan, N. Liu, Z. Xu, J. Tang, and Y. Wang, "Deep reinforcement learning for dynamic treatment regimes on medical registry data," in *Proc. IEEE Int. Conf. Healthcare Inform. (ICHI)*, Aug. 2017, pp. 380–385, doi: 10.1109/ICHI.2017.45.

[35] I. El Naqa, M. Feng, L. Bazzi, J. Dow, K. C. Cuneo, M. M. Matuszak, K. K. Brock, K. Suresh, M. Schipper, T. S. Lawrence, and R. K. T. Haken, "Reinforcement learning strategies for decision making in knowledge-based adaptive radiation therapy: Application in liver cancer," *Int. J. Radiat. Oncol. Biol. Phys.*, vol. 96, no. 2, p. S45, Oct. 2016, doi: 10.1016/j.ijrobp.2016.06.119.

[36] A. Jalalimanesh, H. S. Haghighi, A. Ahmadi, H. Hejazian, and M. Soltani, "Multi-objective optimization of radiotherapy: Distributed Q-learning and agent-based simulation," *J. Experim. Theor. Artif. Intell.*, vol. 29, no. 5, pp. 1071–1086, Sep. 2017.

[37] S. M. Shortreed, E. Laber, D. J. Lizotte, T. S. Stroup, J. Pineau, and S. A. Murphy, "Informing sequential clinical decision-making through reinforcement learning: An empirical study," *Mach. Learn.*, vol. 84, nos. 1–2, pp. 109–136, Dec. 2010, doi: 10.1007/s10994-010-5229-0.

[38] P. Escandell-Montero, M. Chermisi, J. M. Martínez-Martínez, J. Gómez-Sanchis, C. Barbieri, E. Soria-Olivas, F. Mari, J. Vila-Francés, A. Stopper, E. Gatti, and J. D. Martín-Guerrero, "Optimization of anemia treatment in hemodialysis patients via reinforcement learning," *Artif. Intell. Med.*, vol. 62, no. 1, pp. 47–60, Sep. 2014.

[39] E. B. Laber, K. A. Linn, and L. A. Stefanski, "Interactive model building for Q-learning," *Biometrika*, vol. 101, no. 4, pp. 831–847, Oct. 2014, doi: 10.1093/biomet/asu043.

[40] I. Hochberg, G. Feraru, M. Kozdoba, S. Mannor, M. Tennenholtz, and E. Yom-Tov, "A reinforcement learning system to encourage physical activity in diabetes patients," *CoRR*, 2016. [Online]. Available: http://arxiv.org/abs/1605.04070

[41] Y. Wang, H. Fu, and D. Zeng, "Learning optimal personalized treatment rules in consideration of benefit and risk: With an application to treating type 2 diabetes patients with insulin therapies," *J. Amer. Stat. Assoc.*, vol. 113, no. 521, pp. 1–13, Jan. 2018, doi: 10.1080/01621459.2017.1303386.

[42] M. Kranz, A. Möller, N. Hammerla, S. Diewald, T. Plötz, P. Olivier, and L. Roalter, "The mobile fitness coach: Towards individualized skill assessment using personalized mobile devices," *Pervas. Mobile Comput.*, vol. 9, no. 2, pp. 203–215, Apr. 2013, doi: 10.1016/j.pmcj.2012.06.002.

[43] M. K. Anthony, C. Gordon, and L. A. Celi, "Optimising the management of severe infections in intensive care with reinforcement learning," *J. Exp. Theor. Artif. Intell.*, 2016.

[44] W. Weng, M. Gao, Z. He, S. Yan, and P. Szolovits, "Representation and reinforcement learning for personalized glycemic control in septic patients," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1712.00654

[45] M. Komorowski, L. A. Celi, O. Badawi, A. C. Gordon, and A. A. Faisal, "The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care," *Nature Med.*, vol. 24, no. 11, pp. 1716–1720, Nov. 2018. [Online]. Available: https://www.nature.com/articles/s41591-018-0213-5

[46] S. Saria, "Individualized sepsis treatment using reinforcement learning," *Nature Med.*, vol. 24, no. 11, pp. 1641–1642, Nov. 2018. [Online]. Available: https://www.nature.com/articles/s41591-018-0253-x

[47] A. Raghu, M. Komorowski, L. A. Celi, P. Szolovits, and M. Ghassemi, "Continuous state-space models for optimal sepsis treatment—A deep reinforcement learning approach," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1705.08422

[48] J. Peng and B. Bhanu, "Closed-loop object recognition using reinforcement learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 2, pp. 139–154, Feb. 1998, doi: 10.1109/34.659932.

[49] M. Shokri and H. Tizhoosh, "Using reinforcement learning for image thresholding," in *Proc. Can. Conf. Elect. Comput. Eng.*, May 2003, pp. 1231–1234, doi: 10.1109/CCECE.2003.1226121.

[50] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama, "A reinforcement learning framework for medical image segmentation," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2006, pp. 511–517, doi: 10.1109/IJCNN.2006.246725.

[51] B. Netto, S. Magalhaes, C. Leite, V. Rodrigues, A. Correa, A. C. de Paiva, and A. de A. Neto, "Application on reinforcement learning for diagnosis based on medical image," in *Reinforcement Learning*. Jan. 2008, doi: 10.5772/5291.

[52] F. Sahba, H. R. Tizhoosh, and M. M. Salama, "Application of reinforcement learning for segmentation of transrectal ultrasound images," *BMC Med. Imag.*, vol. 8, no. 1, p. 8, Apr. 2008, doi: 10.1186/1471-2342-8-8.

[53] G. Maicas, G. Carneiro, A. P. Bradley, J. C. Nascimento, and I. Reid, "Deep reinforcement learning for active breast lesion detection from DCE-MRI," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, 2017, pp. 665–673.

[54] S. Kluckner. (2018). *Medical Scanner Teaches Itself to Optimize Clinical Protocols and Image Acquisition*. [Online]. Available: http://www.freepatentsonline.com/20180032841.pdf

[55] R. McPherson and M. Pincus, *Henry's Clinical Diagnosis and Management by Laboratory Methods*. Amsterdam, The Netherlands: Elsevier, 2017.

[56] D. Thapa, I.-S. Jung, and G.-N. Wang, "RL based decision support system for u-healthcare environment," in *Reinforcement Learning*. Jan. 2008, doi: 10.5772/5292.

[57] Y. Ling, S. A. Hasan, V. Datla, A. Qadir, K. Lee, J. Liu, and O. Farri, "Diagnostic inferencing via improving clinical concept extraction with deep reinforcement learning: A preliminary study," in *Proc. Mach. Learn. Healthcare Conf.*, vol. 68, Aug. 2017, pp. 271–285.

[58] R. Poolla, "A reinforcement learning approach to obtain treatment strategies in sequential medical decision problems," Scholars Commons, Univ. South Florida, Tampa, FL, USA, Tech. Rep., 2003.

[59] Y. Ling, S. A. Hasan, V. V. Datla, A. Qadir, K. Lee, J. Liu, and O. Farri, "Learning to diagnose: Assimilating clinical narratives using deep reinforcement learning," in *Proc. 8th Int. Joint Conf. Natural Lang. Process.*, 2017, pp. 895–905.

[60] G. Yauney and P. Shah, "Reinforcement learning with action-derived rewards for chenotherapy and clinical trial using dosing regimen selection," in *Proc. Mach. Learn. Healthcare Conf.*, 2018, pp. 161–226.

[61] M. Pirmohamed, S. James, S. Meakin, C. Green, A. K. Scott, T. J. Walley, K. Farrar, B. K. Park, and A. M. Breckenridge, "Adverse drug reactions as cause of admission to hospital: Prospective analysis of 18 820 patients," *BMJ*, vol. 329, no. 7456, pp. 15–19, Jul. 2004, doi: 10.1136/bmj.329.7456.15.

[62] S. Schleidgen, C. Klingler, T. Bertram, W. H. Rogowski, and G. Marckmann, "What is personalized medicine: Sharpening a vague term based on a systematic literature review," *BMC Med. Ethics*, vol. 14, no. 1, p. 55, Dec. 2013, doi: 10.1186/1472-6939-14-55.

[63] L. G. Jaimes, M. Llofriu, and A. Raij, "CALMA, an algorithm framework for mobile just in time interventions," in *Proc. SoutheastCon*, Apr. 2015, pp. 1–5, doi: 10.1109/SECON.2015.7133041.

[64] E. M. Shakshuki, M. Reid, and T. R. Sheltami, "An adaptive user interface in healthcare," *Procedia Comput. Sci.*, vol. 56, pp. 49–58, 2015, doi: 10.1016/j.procs.2015.07.182.

[65] S. Nemati, M. M. Ghassemi, and G. D. Clifford, "Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach," in *Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Aug. 2016, pp. 2978–2981, doi: 10.1109/EMBC.2016.7591355.

[66] R. Padmanabhan, N. Meskin, and W. M. Haddad, "Learning-based control of cancer chemotherapy treatment," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 15127–15132, Jul. 2017, doi: 10.1016/j.ifacol.2017.08.2247.

[67] R. Padmanabhan, N. Meskin, and W. M. Haddad, "Reinforcement learning-based control of drug dosing for cancer chemotherapy treatment," *Math. Biosci.*, vol. 293, pp. 11–20, Nov. 2017, doi: 10.1016/j.mbs.2017.08.004.

[68] H.-H. Tseng, Y. Luo, S. Cui, J.-T. Chien, R. K. T. Haken, and I. E. Naqa, "Deep reinforcement learning for automated radiation adaptation in lung cancer," *Med. Phys.*, vol. 44, no. 12, pp. 6690–6705, Nov. 2017, doi: 10.1002/mp.12625.
[69] B. K. Petersen, J. Yang, W. S. Grathwohl, C. Cockrell, C. Santiago, G. An, and D. M. Faissol, "Precision medicine as a control problem: Using simulation and deep reinforcement learning to discover adaptive, personalized multi-cytokine therapy for sepsis," *CoRR*, 2018.
[70] A. Raghu, M. Komorowski, I. Ahmed, L. A. Celi, P. Szolovits, and M. Ghassemi, "Deep reinforcement learning for sepsis treatment," *CoRR*, 2017.
[71] J. Daniel and H. M. James. (2014). *Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. [Online]. Available: http://www.cs.colorado.edu/ martin/SLP/Updates/1.pdf
[72] D. Kim, C. Breslin, P. Tsiakoulis, M. Gasi, M. Henderson, and S. Young. (2014). *Inverse Reinforcement Learning for Micro-Turn Management*. [Online]. Available: http://mi.eng.cam.ac.uk/~sjy/papers/kgbt14.pdf
[73] A. Gravano and J. Hirschberg, "Turn-taking cues in task-oriented dialogue," *Comput. Speech Lang.*, vol. 25, no. 3, pp. 601–634, Jul. 2011, doi: 10.1016/j.csl.2010.10.003.
[74] D. DeVault, K. Sagae, and D. Traum, "Incremental interpretation and prediction of utterance meaning for interactive dialogue," *Dialogue Discourse*, vol. 2, no. 1, pp. 143–170, May 2011, doi: 10.5087/dad.2011.107.
[75] D. David, K. Sagae, and D. Traum, "Detecting the status of a predictive incremental speech understanding model for real time decision making in a spoken dialogue system," in *Proc. 12th Annu. Conf. Int. Speech Commun. Assoc.*, Aug. 2011, pp. 1–5.
[76] T. Baumann and D. Schlangen, "Predicting the micro-timing of user input for an incremental spoken dialogue system that completes a user's ongoing turn," Tech. Rep., 2011, pp. 120–129.
[77] T. Visser, D. Traum, D. DeVault, and R. O. den Akker, "A model for incremental grounding in spoken dialogue systems," *J. Multimodal User Interfaces*, vol. 8, no. 1, pp. 61–73, Mar 2014.
[78] H. Chinaei and B. Chaib-Draa, "Dialogue strategy learning in healthcare: A systematic approach for learning dialogue models from data," in *Proc. 5th Workshop Speech Lang. Process. Assistive Technol.*, 2014, pp. 13–19.
[79] T. Hiraoka, G. Neubig, S. Sakti, T. Toda, and S. Nakamura, "Reinforcement learning of cooperative persuasive dialogue policies using framing," in *Proc. 25th Int. Conf. Comput. Linguistics*, 2014, pp. 1706–1717.
[80] T. Yamagishi, *Trust: The Evolutinary Game of Mind and Society*. Springer, 2011.
[81] S. Ono, T. Obo, L. C. Kiong, and N. Kubota, "Robot communication based on relational trust model," in *Proc. 41st Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Nov. 2015, pp. 005335–005338, doi: 10.1109/IECON.2015.7392941.
[82] P. Odom and S. Natarajan, "Active advice seeking for inverse reinforcement learning," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2016, pp. 512–520. [Online]. Available: http://dl.acm.org/citation.cfm?id=2936924.2936998
[83] G. Kunapuli, P. Odom, J. W. Shavlik, and S. Natarajan, "Guiding autonomous agents to better behaviors through human advice," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, pp. 409–418.
[84] C. Hamidreza and B. Chaib-Draa, *Building Dialogue POMDPs from Expert Dialogue: An End-to-End Approach*. 2016.
[85] K. M. Jagodnik, P. S. Thomas, A. J. van den Bogert, M. S. Branicky, and R. F. Kirsch, "Training an actor-critic reinforcement learning controller for arm movement using human-generated rewards," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 10, pp. 1892–1905, Oct. 2017.
[86] E. M. Shakshuki, M. Reid, and T. R. Sheltami, "Dynamic healthcare interface for patients," *Procedia Comput. Sci.*, vol. 63, pp. 356–365, Jan. 2015.
[87] R. D. Vincenti, "Reinforcement learning in models of adaptive medical treatment strategies," McGill Univ., Montreal, QC, Canada, Tech. Rep., 2014.
[88] R. Istepanian, N. Philip, and M. Martini, "Medical QoS provision based on reinforcement learning in ultrasound streaming over 3.5G wireless systems," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 566–574, May 2009.
[89] P. M. Pilarski, M. R. Dawson, T. Degris, J. P. Carey, and R. S. Sutton, "Dynamic switching and real-time machine learning for improved human control of assistive biomedical robots," in *Proc. 4th IEEE RAS & EMBS Int. Conf. Biomed. Robot. Biomechatronics (BioRob)*, Jun. 2012, pp. 296–302.
[90] G. Czibula, I. M. Bocicor, and I.-G. Czibula, "Temporal ordering of cancer microarray data through a reinforcement learning based approach," *PLoS ONE*, vol. 8, no. 4, 2013, Art. no. e60883.
[91] R. Kamal, C. S. Hong, and M. J. Choi, "Autonomic learning through stochastic games for rational allocation of scarce medical resources," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2013, pp. 852–855.
[92] E. Lee, M. S. Lavieri, M. L. Volk, and Y. Xu, "Applying reinforcement learning techniques to detect hepatocellular carcinoma under limited screening capacity," *Health Care Manage. Sci.*, vol. 18, no. 3, pp. 363–375, Sep. 2015, doi: 10.1007/s10729-014-9304-0.
[93] N. Prasad, L. Cheng, C. Chivers, M. Draugelis, and B. E. Engelhardt, "A reinforcement learning approach to weaning of mechanical ventilation in intensive care units," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1704.06300
[94] M. K. Bothe, L. Dickens, K. Reichel, A. Tellmann, B. Ellger, M. Westphal, and A. A. Faisal, "The use of reinforcement learning algorithms to meet the challenges of an artificial pancreas," *Expert Rev. Med. Devices*, vol. 10, no. 5, pp. 661–673, Sep. 2013.
[95] M. K. Holden, "Virtual environments for motor rehabilitation: Review," *CyberPsychol. Behav.*, vol. 8, no. 3, pp. 187–211, Jun. 2005, doi: 10.1089/cpb.2005.8.187.
[96] H. Thieme, J. Mehrholz, M. Pohl, J. Behrens, and C. Dohle, "Mirror therapy for improving motor function after stroke," *J. Neurol. Sci.*, vol. 333, p. e573, Oct. 2013.
[97] M. M. Sohlberg and C. A. Mateer, *Cognitive Rehabilitation: An Integrative Neuropsychological Approach*. 2017.
[98] Y. Bogdanova, M. K. Yee, V. T. Ho, and K. D. Cicerone, "Computerized cognitive rehabilitation of attention and executive function in acquired brain injury: A systematic review," *J. Head Trauma Rehabil.*, vol. 31, no. 6, pp. 419–433, 2016.
[99] M. Zhu, Z. Zhang, J. P. Hirdes, and P. Stolee, "Using machine learning algorithms to guide rehabilitation planning for home care clients," *BMC Med. Informat. Decis. Making*, vol. 7, no. 1, p. 41, Dec. 2007.
[100] M. Bortole, A. Venkatakrishnan, F. Zhu, J. C. Moreno, G. E. Francisco, J. L. Pons, and J. L. Contreras-Vidal, "The h2 robotic exoskeleton for gait rehabilitation after stroke: Early findings from a clinical study," *J. Neuroeng. Rehabil.*, vol. 12, no. 1, p. 54, Jun. 2015.
[101] E. Swinnen, D. Beckwée, R. Meeusen, J.-P. Baeyens, and E. Kerckhofs, "Does robot-assisted gait rehabilitation improve balance in stroke patients? A systematic review," *Topics Stroke Rehabil.*, vol. 21, no. 2, pp. 87–100, Mar. 2014.
[102] A. Coronato and A. Cuzzocrea, "An innovative risk assessment methodology for medical information systems," *IEEE Trans. Knowl. Data Eng.*, early access, Sep. 11, 2020, doi: 10.1109/TKDE.2020.3023553.
[103] G. Paragliola and M. Naeem, "Risk management for nuclear medical department using reinforcement learning algorithms," *J. Reliable Intell. Environ.*, vol. 5, no. 2, pp. 105–113, Jul. 2019.
[104] G. Paragliola, A. Coronato, M. Naeem, and G. De Pietro, "A reinforcement learning-based approach for the risk management of e-Health environments: A case study," in *Proc. 14th Int. Conf. Signal-Image Technol. Internet-Based Syst. (SITIS)*, Las Plamas, Spain, Nov. 2018, pp. 26–29.
[105] A. Coronato and G. De Pietro, "Situation awareness in applications of ambient assisted living for cognitive impaired people," *Mobile Netw. Appl.*, vol. 18, no. 3, pp. 444–453, Jun. 2013.
[106] G. Paragliola and A. Coronato, "Gait anomaly detection of subjects with Parkinson's disease using a deep time series-based approach," *IEEE Access*, vol. 6, pp. 73280–73292, 2018, doi: 10.1109/ACCESS.2018.2882245.
[107] A. Coronato and M. Naeem, "A reinforcement learning based intelligent system for the healthcare treatment assistance of patients with disabilities," in *Proc. 16th Int. Symp. Pervas. Syst., Algorithms, Netw. (I-SPAN)*, 2019, pp. 15–28.
[108] M. Naeem, G. Paragliola, A. Coronato, and G. De-Pietro, "A CNN based monitoring system to minimize medication errors during treatment process at home," in *Proc. 3rd Int. Conf. Appl. Intell. Syst.*, Las Palmas de Gran Canaria, Spain, Jan. 2020, pp. 1–5.
[109] M. Naeem, A. Coronato, and G. Paragliola, "Adaptive treatment assisting system for patients using machine learning," in *Proc. Int. Workshop Cogn. Neural Syst. (CNS)*, Granada, Spain, Oct. 2019, pp. 460–465.
[110] P. Kormushev, S. Calinon, and D. Caldwell, "Reinforcement learning in robotics: Applications and real-world challenges," *Robotics*, vol. 2, no. 3, pp. 122–148, Jul. 2013.

[111] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.
[112] F. Guenter, M. Hersch, S. Calinon, and A. Billard, "Reinforcement learning for imitating constrained reaching movements," *Adv. Robot.*, vol. 21, no. 13, pp. 1521–1544, Jan. 2007.
[113] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, Sep. 2013.
[114] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Singapore, May 2017, pp. 3389–3396.
[115] P. Wang, C. Chan, and A. de La Fortelle, "A reinforcement learning based approach for automated lane change maneuvers," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Changshu, China, Jun./Jul. 2018, pp. 1379–1384.
[116] T. Hester, M. Quinlan, and P. Stone, "A real-time model-based reinforcement learning architecture for robot control," *CoRR*, 2011.
[117] B. Bakker, V. Zhumatiy, G. Gruener, and J. Schmidhuber, "Quasi-online reinforcement learning for robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2006, pp. 2997–3002.
[118] T. Martinez-Marin and T. Duckett, "Fast reinforcement learning for vision-guided mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Barcelona, Spain, Apr. 2005, pp. 4170–4175.
[119] C. Touzet, "Neural reinforcement learning for behaviour synthesis," *Robot. Auton. Syst.*, vol. 22, no. 3, pp. 251–281, 1997.
[120] M. Tamosiunaite, B. Nemec, A. Ude, and F. Wörgötter, "Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives," *Robot. Auton. Syst.*, vol. 59, no. 11, pp. 910–922, Nov. 2011.
[121] S. Bitzer, M. Howard, and S. Vijayakumar, "Using dimensionality reduction to exploit constraints in reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 3219–3225.
[122] K. Gräve, J. Stückler, and S. Behnke, "Learning motion skills from expert demonstrations and own experience using Gaussian process regression," in *Proc. ISR*, 2010, pp. 1–8.
[123] O. Kroemer, R. Detry, J. Piater, and J. Peters, "Active learning using mean shift optimization for robot grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2009, pp. 2610–2615.
[124] B. Nemec, M. Tamošiunaite, F. Wörgötter, and A. Ude, "Task adaptation through exploration and action sequencing," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, Dec. 2009, pp. 610–616.
[125] B. Nemec, M. Zorko, and L. Zlajpah, "Learning of a ball-in-a-cup playing robot," in *Proc. 19th Int. Workshop Robot. Alpe-Adria-Danube Region (RAAD)*, Jun. 2010, pp. 297–301.
[126] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange, "Reinforcement learning for robot soccer," *Auton. Robots*, vol. 27, no. 1, pp. 55–73, Jul. 2009.
[127] V. Soni and S. P. Singh, "Reinforcement learning of hierarchical skills on the Sony Aibo robot," in *Proc. Int. Conf. Develop. Learn.*, 2006, pp. 1–6.
[128] D. Lizotte, T. Wang, M. Bowling, and D. Schuurmans, "Automatic gait optimization with Gaussian process regression," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2007, pp. 944–949.
[129] M. P. Deisenroth and C. E. Rasmussen, "A model-based and data-efficient approach to policy search," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 465–472.
[130] M. P. Deisenroth, C. E. Rasmussen, and D. Fox, "Learning to control a low-cost manipulator using data-efficient reinforcement learning," in *Robotics: Science and Systems*. 2007.
[131] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: http://arxiv.org/abs/1509.02971
[132] V. Mnih, A. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[133] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.
[134] N. Heess, D. Tb, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. Ali Eslami, M. Riedmiller, and D. Silver, "Emergence of locomotion behaviours in rich environments," 2017, *arXiv:1707.02286*. [Online]. Available: http://arxiv.org/abs/1707.02286
[135] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3389–3396.
[136] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2017, pp. 3357–3364.
[137] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[138] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: An interactive 3D environment for visual AI," 2017, *arXiv:1712.05474*. [Online]. Available: http://arxiv.org/abs/1712.05474
[139] T. Schaul, D. Horgan, K. Gregor, and D. Silver, "Universal value function approximators," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1312–1320.
[140] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2017, *arXiv:1703.10593*. [Online]. Available: http://arxiv.org/abs/1703.10593
[141] A. Khan, C. Zhang, N. Atanasov, K. Karydis, V. Kumar, and D. D. Lee, "Memory augmented control networks," 2017, *arXiv:1709.05706*. [Online]. Available: http://arxiv.org/abs/1709.05706
[142] J. Bruce, N. Suenderhauf, P. Mirowski, R. Hadsell, and M. Milford, "One-shot reinforcement learning for robot navigation with interactive replay," 2017, *arXiv:1711.10137*. [Online]. Available: http://arxiv.org/abs/1711.10137
[143] D. S. Chaplot, E. Parisotto, and R. Salakhutdinov, "Active neural localization," 2018, *arXiv:1801.08214*. [Online]. Available: http://arxiv.org/abs/1801.08214
[144] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," 2018, *arXiv:1803.00653*. [Online]. Available: http://arxiv.org/abs/1803.00653
[145] N. Vlassis, M. Toussaint, G. Kontes, and S. Piperidis, "Learning model-free robot control by a Monte Carlo EM algorithm," *Auton. Robots*, vol. 27, no. 2, pp. 123–130, Aug. 2009.
[146] J. A. Bagnell and J. G. Schneider, "Autonomous helicopter control using reinforcement learning policy search methods," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2001, pp. 1615–1620.
[147] J. Peters, K. Muelling, and Y. Altun, "Relative entropy policy search," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2010, pp. 1607–1612.
[148] J. Z. Kolter and Y. N. Andrew, "Policy search via the signed derivative," in *Robotics: Science and Systems*. 2009.
[149] J. Kober and J. Peters, "Policy search for motor primitives in robotics," *Mach. Learn.*, vol. 84, nos. 1–2, pp. 171–203., 2010.
[150] R. Tedrake, T. W. Zhang, and H. S. Seung, "Stochastic policy gradient reinforcement learning on a simple 3D biped," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep./Oct. 2004, pp. 2849–2854.
[151] R. Tedrake, T. W. Zhang, and H. S. Seung, "Learning to walk in 20 minutes," in *Proc. Yale Workshop Adaptive Learn. Syst.*, 2005, pp. 1939–1412.
[152] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3133–3174, May 2019.
[153] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," Oct. 2017, pp. 1–98, *arXiv:1710.02913*. [Online]. Available: http://arxiv.org/abs/1710.02913
[154] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2595–2621, Jun. 2018.
[155] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward Tomorrow's intelligent network traffic control systems," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2432–2455, May 2017.
[156] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018.
[157] M. Gadaleta, F. Chiariotti, M. Rossi, and A. Zanella, "D-DASH: A deep Q-learning framework for DASH video streaming," *IEEE Trans. Cognit. Commun. Netw.*, vol. 3, no. 4, pp. 703–718, Dec. 2017.
[158] P. Victor Rodrigues Ferreira, R. Paffenroth, A. M. Wyglinski, T. M. Hackett, S. G. Bilén, R. C. Reinhart, and D. J. Mortensen, "Multiobjective reinforcement learning for cognitive satellite communications using deep neural network ensembles," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 5, pp. 1030–1041, May 2018.

[159] S. Chinchali, "Cellular network traffic scheduling with deep reinforcement learning," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2018, pp. 1–9.
[160] T. Huang, R.-X. Zhang, C. Zhou, and L. Sun, "QARC: Video quality aware rate control for real-time video streaming based on deep reinforcement learning," in *Proc. ACM Multimedia Conf. Multimedia Conf. (MM)*, Oct. 2018, pp. 1208–1216.
[161] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proc. Conf. ACM Special Interest Group Data Commun.*, Aug. 2017, pp. 197–210.
[162] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–7.
[163] N. Zhao, Y.-C. Liang, D. Niyato, Y. Pei, and Y. Jiang, "Deep reinforcement learning for user association and resource allocation in heterogeneous networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.
[164] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access," in *Proc. Int. Conf. Comput., Netw. Commun.*, 2017, pp. 257–265.
[165] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2375–2385, Aug. 2018.
[166] M. Chu, H. Li, X. Liao, and S. Cui, "Reinforcement learning-based multiaccess control and battery prediction with energy harvesting in IoT systems," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2009–2020, Apr. 2019.
[167] H. Ye and G. Y. Li, "Deep reinforcement learning for resource allocation in V2V communications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.
[168] U. Challita, L. Dong, and W. Saad, "Proactive resource management for LTE in unlicensed spectrum: A deep learning perspective," Dec. 2017, pp. 1–29, *arXiv:1702.07031*. [Online]. Available: http://arxiv.org/abs/1702.07031
[169] S. Liu, X. Hu, and W. Wang, "Deep reinforcement learning based dynamic channel allocation algorithm in multibeam satellite systems," *IEEE Access*, vol. 6, pp. 15733–15742, 2018.
[170] M. Chen, W. Saad, and C. Yin, "Liquid state machine learning for resource allocation in a network of cache-enabled LTE-U UAVs," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.
[171] Y. He, Z. Zhang, and Y. Zhang, "A big data deep reinforcement learning approach to next generation green wireless networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.
[172] Y. He, C. Liang, Z. Zhang, F. R. Yu, N. Zhao, H. Yin, and Y. Zhang, "Resource allocation in software-defined and information-centric vehicular networks with mobile edge computing," in *Proc. IEEE 86th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2017, pp. 1–5.
[173] Y. He, F. R. Yu, N. Zhao, H. Yin, and A. Boukerche, "Deep reinforcement learning (DRL)-based resource management in software-defined and virtualized vehicular ad hoc networks," in *Proc. 6th ACM Symp. Develop. Anal. Intell. VehicularNetworks Appl. (DIVANet)*, 2017, pp. 47–54.
[174] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.
[175] Y. He, F. R. Yu, N. Zhao, V. C. M. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 31–37, Dec. 2017.
[176] Y. He, F. R. Yu, N. Zhao, and H. Yin, "Secure social networks in 5G systems with mobile edge computing, caching, and device-to-device communications," *IEEE Wireless Commun.*, vol. 25, no. 3, pp. 103–109, Jun. 2018.
[177] Y. He, C. Liang, F. R. Yu, and Z. Han, "Trust-based social networks with computing, caching and communications: A deep reinforcement learning approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 66–79, Jan. 2020.
[178] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10190–10203, Nov. 2018.
[179] C. Zhong, M. C. Gursoy, and S. Velipasalar, "A deep reinforcement learning-based framework for content caching," in *Proc. 52nd Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2018, pp. 1–6.
[180] M. Schaarschmidt, F. Gessert, V. Dalibard, and E. Yoneki, "Learning runtime parameters in computer systems with delayed experience injection," Oct. 2016, pp. 1–10, *arXiv:1610.09903*. [Online]. Available: http://arxiv.org/abs/1610.09903
[181] Y. He and S. Hu, "Cache-enabled wireless networks with opportunistic interference alignment," Jun. 2017, pp. 1–6, *arXiv:1706.09024*. [Online]. Available: http://arxiv.org/abs/1706.09024
[182] Y. He, C. Liang, F. R. Yu, N. Zhao, and H. Yin, "Optimization of cache-enabled opportunistic interference alignment wireless networks: A big data deep reinforcement learning approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[183] Y. He, Z. Zhang, F. R. Yu, N. Zhao, H. Yin, V. C. M. Leung, and Y. Zhang, "Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10433–10445, Nov. 2017.
[184] X. He, K. Wang, H. Huang, T. Miyazaki, Y. Wang, and S. Guo, "Green resource allocation based on deep reinforcement learning in content-centric IoT," *IEEE Trans. Emerg. Topics Comput.*, vol. 8, no. 3, pp. 781–796, Jul. 2020.
[185] M. Chen, W. Saad, and C. Yin, "Echo-liquid state deep learning for 360° content transmission and caching in wireless VR networks with cellular-connected UAVs," Apr. 2018, pp. 1–10, *arXiv:1804.03284*. [Online]. Available: http://arxiv.org/abs/1804.03284
[186] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019.
[187] C. Zhang, Z. Liu, B. Gu, K. Yamori, and Y. Tanaka, "A deep reinforcement learning based approach for cost- and energy-aware multi-flow mobile data offloading," *IEICE Trans. Commun.*, vol. E101.B, no. 7, pp. 1625–1634, 2018.
[188] Z. Tang, X. Zhou, F. Zhang, W. Jia, and W. Zhao, "Migration modeling and learning algorithms for containers in fog computing," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 712–725, Sep. 2019.
[189] X. Wan, G. Sheng, Y. Li, L. Xiao, and X. Du, "Reinforcement learning based mobile offloading for cloud-based malware detection," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.
[190] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Performance optimization in mobile-edge computing via deep reinforcement learning," in *Proc. IEEE 88th Veh. Technol. Conf. (VTC-Fall)*, Aug. 2018, pp. 1–6.
[191] L. Ji, G. Hui, L. Tiejun, and L. Yueming, "Deep reinforcement learning based computation offloading and resource allocation for MEC," in *Proc. IEEE WCNC*, Apr. 2018, pp. 1–5.
[192] A. Ferdowsi and W. Saad, "Deep learning for signal authentication and security in massive Internet-of-Things systems," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1371–1387, Feb. 2019.
[193] G. Han, L. Xiao, and H. V. Poor, "Two-dimensional anti-jamming communication based on deep reinforcement learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 2087–2091.
[194] Y. Chen, Y. Li, D. Xu, and L. Xiao, "DQN-based power control for IoT transmission against jamming," in *Proc. IEEE 87th Veh. Technol. Conf. (VTC Spring)*, Jun. 2018, pp. 1–5.
[195] A. Ferdowsi, U. Challita, W. Saad, and N. B. Mandayam, "Robust deep reinforcement learning for security and safety in autonomous vehicle systems," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 307–312.
[196] L. Xiao, Donghua, Jiang, X. Wan, W. Su, and Y. Tang, "Anti-jamming underwater transmission with mobility and learning," *IEEE Commun. Lett.*, vol. 22, no. 3, pp. 542–545, Mar. 2018.
[197] X. Lu, L. Xiao, C. Dai, and H. Dai, "UAV-aided cellular communications with deep reinforcement learning against jamming," Apr. 2019, pp. 1–6, *arXiv:1805.06628*. [Online]. Available: http://arxiv.org/abs/1805.06628
[198] X. Liu, Y. Xu, L. Jia, Q. Wu, and A. Anpalagan, "Anti-jamming communications using spectrum waterfall: A deep reinforcement learning approach," *IEEE Commun. Lett.*, vol. 22, no. 5, pp. 998–1001, May 2018.
[199] L. Xiao, C. Xie, M. Min, and W. Zhuang, "User-centric view of unmanned aerial vehicle transmission against smart attacks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3420–3430, Apr. 2018.
[200] W. Huang, Y. Wang, and X. Yi, "A deep reinforcement learning approach to preserve connectivity for multi-robot systems," in *Proc. 10th Int. Congr. Image Signal Process., Biomed. Eng. Informat. (CISP-BMEI)*, Oct. 2017, pp. 1–7.

[201] C. Wang, J. Wang, X. Zhang, and X. Zhang, "Autonomous navigation of UAV in large-scale unknown complex environment with deep reinforcement learning," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Nov. 2017, pp. 858–862.

[202] M. Faris and E. Brian, "Deep Q-learning for self-organizing networks fault management and radio performance improvement," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, 2018, pp. 1457–1461.

[203] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, Jan. 2017.

[204] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, "Deep convolutional neural networks for massive MIMO fingerprint-based positioning," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–6.

[205] H. Huang, J. Yang, H. Huang, Y. Song, and G. Gui, "Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8549–8560, Sep. 2018.

[206] X. Li, J. Fang, W. Cheng, H. Duan, Z. Chen, and H. Li, "Intelligent power control for spectrum sharing in cognitive radios: A deep reinforcement learning approach," *IEEE Access*, vol. 6, pp. 25463–25473, 2018.

[207] Y. Zhang, B. Song, and P. Zhang, "Social behavior study under pervasive social networking based on decentralized deep reinforcement learning," *J. Netw. Comput. Appl.*, vol. 86, pp. 72–81, May 2017.

[208] L. Wang, W. Liu, D. Zhang, Y. Wang, E. Wang, and Y. Yang, "Cell selection with deep reinforcement learning in sparse mobile crowdsensing," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 1543–1546.

[209] L. Xiao, Y. Li, G. Han, H. Dai, and H. V. Poor, "A secure mobile crowdsensing game with deep reinforcement learning," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 1, pp. 35–47, Jan. 2018.

[210] B. Zhang, C. H. Liu, J. Tang, Z. Xu, J. Ma, and W. Wang, "Learning-based energy-efficient data collection by unmanned vehicles in smart cities," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1666–1676, Apr. 2018.

[211] L. Wang, D. Zhang, A. Pathak, C. Chen, H. Xiong, D. Yang, and Y. Wang, "CCS-TA: Quality-guaranteed online task allocation in compressive crowdsensing," in *Proc. ACM Int. Joint Conf. Pervas. Ubiquitous Comput. (UbiComp)*, 2015, pp. 683–694.

[212] A. Zappone, M. Debbah, and Z. Altman, "Online energy-efficient power control in wireless networks by deep neural networks," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.

[213] T. Oda, R. Obukata, M. Ikeda, L. Barolli, and M. Takizawa, "Design and implementation of a simulation system based on deep Q-network for mobile actor node control in wireless sensor and actor networks," in *Proc. 31st Int. Conf. Adv. Inf. Netw. Appl. Workshops (WAINA)*, Mar. 2017, pp. 195–200.

[214] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

[215] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proc. 1st Int. ICST Conf. Simulation Tools Techn. Commun. Netw. Syst.*, 2008, p. 60.

[216] U. Challita, W. Saad, and C. Bettstetter, "Deep reinforcement learning for interference-aware path planning of cellular-connected UAVs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[217] U. Challita, W. Saad, and C. Bettstetter, "Cellular-connected UAVs over 5G: Deep reinforcement learning for interference management," Jan. 2018, pp. 1–30, *arXiv:1801.05500*. [Online]. Available: http://arxiv.org/abs/1801.05500

[218] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," 2018, *arXiv:1801.05757*. [Online]. Available: http://arxiv.org/abs/1801.05757

[219] G. Stampa, M. Arias, D. Sanchez-Charles, V. Muntes-Mulero, and A. Cabellos, "A deep-reinforcement learning approach for software-defined networking routing optimization," Sep. 2017, pp. 1–3, *arXiv:1709.07080*. [Online]. Available: http://arxiv.org/abs/1709.07080

[220] K. Wehrle, M. Günes, and J. Gross, "The NS-3 network simulator," in *Modeling and Tools for Network Simulation*. Heidelberg, Germany: Springer, 2010.

[221] M. Roughan, "Simplifying the synthesis of Internet traffic matrices," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 93–96, Oct. 2005.

[222] J. He, J. Chen, X. He, J. Gao, L. Li, L. Deng, and M. Ostendorf, "Deep reinforcement learning with a natural language action space," in *Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2016, pp. 1621–1630.

[223] K. Guu, P. Pasupat, E. Z. Liu, and P. Liang, "From language to programs: Bridging reinforcement learning and maximum marginal likelihood," in *Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2015, pp. 1051–1062.

[224] K. Narasimhan, A. Yala, and R. Barzilay, "Improving information extraction by acquiring external evidence with reinforcement learning," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2016, pp. 2355–2365.

[225] A. Radford, R. Jozefowicz, and I. Sutskever, "Learning to generate reviews and discovering sentiment," 2017, *arXiv:1704.01444*. [Online]. Available: https://arxiv.org/abs/1704.01444

[226] W. Xiong, T. Hoang, and W. Y. Wang, "A reinforcement learning method for knowledge graph reasoning," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2017, pp. 564–573.

[227] R. Nogueira and K. Cho, "Task-oriented query reformulation with reinforcement learning," 2017, *arXiv:1704.04572*. [Online]. Available: https://arxiv.org/abs/1704.04572

[228] B. Mitra and N. Craswell, "Neural models for information retrieval," 2017, *arXiv:1705.01509*. [Online]. Available: https://arxiv.org/abs/1705.01509

[229] X. Zhang and M. Lapata, "Sentence simplification with deep reinforcement learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 584–594.

[230] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," 2017, *arXiv:1705.04304*. [Online]. Available: https://arxiv.org/abs/1705.04304

[231] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, "Neural symbolic machines: Learning semantic parsers on freebase with weak supervision," in *Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2017, pp. 23–33.

[232] K. M. Hermann, F. Hill, S. Green, F. Wang, R. Faulkner, H. Soyer, D. Szepesvari, W. Marian Czarnecki, M. Jaderberg, D. Teplyashin, M. Wainwright, C. Apps, D. Hassabis, and P. Blunsom, "Grounded language learning in a simulated 3D world," 2017, *arXiv:1706.06551*. [Online]. Available: http://arxiv.org/abs/1706.06551

[233] M. Janner, K. Narasimhan, and R. Barzilay, "Representation learning for grounded spatial reasoning," in *Proc. TACL*, 2018, pp. 49–61.

[234] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proc. NAACL*, 2018, pp. 2227–2237.

[235] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805

[236] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[237] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov, "A deep visual-semantic embedding model," in *Proc. NIPS*, 2013, pp. 2121–2129.

[238] Y. Goldberg, "Assessing bert's syntactic abilities. Conference on robtics research," *CoRR*, 2019.

[239] M. E. Peters, M. Neumann, L. Zettlemoyer, and W. T. Yih, "Dissecting contextual word embeddings: Architecture and representation," in *Proc. EMNLP*, 2018, pp. 1499–1509.

[240] R. Socher, M. Ganjoo, C. D. Manning, and A. Y. Ng, "Zero-shot learning through cross-modal transfer," in *Proc. NIPS*, 2013, pp. 935–943.

[241] I. Tenney, P. Xia, B. Chen, A. Wang, A. Poliak, R. T. McCoy, N. Kim, B. V. Durme, S. Bowman, D. Das, and E. Pavlick, "What do you learn from context? Probing for sentence structure in contextualized word representations," in *Proc. ICLR*, 2019.

[242] J. Luketina, N. Nardelli, G. Farquhar, J. Foerster, J. Andreas, E. Grefenstette, S. Whiteson, and T. Rocktäschel, "A survey of reinforcement learning informed by natural language," 2019, *arXiv:1906.03926*. [Online]. Available: https://arxiv.org/abs/1906.03926

[243] P. Goyal, S. Niekum, and R. J. Mooney, "Using natural language for reward shaping in reinforcement learning," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 1–10.

[244] D. Bahdanau, F. Hill, J. Leike, E. Hughes, A. Hosseini, P. Kohli, and E. Grefenstette, "Learning to understand goal specifications by modelling reward," in *Proc. ICLR*, 2019.

[245] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," in *Proc. CVPR*, 2016, pp. 39–48.
[246] A. Das, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Neural modular control for embodied question answering," in *Proc. CORL*, 2018, pp. 53–62.
[247] J. Andreas, D. Klein, and S. Levine, "Modular multitask reinforcement learning with policy sketches," in *Proc. ICML*, 2017, pp. 166–175.
[248] H. Hu, D. Yarats, Q. Gong, Y. Tian, and M. Lewis, "Hierarchical decision making by generating and following natural language instructions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 10025–10034.
[249] Y. Tian, Q. Gong, W. Shang, Y. Wu, and C. L. Zitnick, "An extensive, lightweight and flexible research platform for real-time strategy games," in *Proc. NIPS*, 2017, pp. 2659–2669.
[250] T. Shu, C. Xion, and R. Socher, "Hierarchical and interpretable skill acquisition in multi-task reinforcement learning," in *Proc. ICLR*, 2018, pp. 1–14.
[251] S. R. K. Branavan, D. Silver, and R. Barzilay, "Learning to win by reading manuals in a Monte–Carlo framework," *J. Artif. Intell. Res.*, vol. 43, pp. 661–704, Apr. 2012.
[252] J. Eisenstein, J. Clarke, D. Goldwasser, and D. Roth, "Reading to learn: Constructing features from semantic abstracts," in *Proc. ACL*, 2009, pp. 958–967.
[253] K. Narasimhan, R. Barzilay, and T. Jaakkola, "Grounding language for transfer in deep reinforcement learning," *J. Artif. Intell. Res.*, vol. 63, pp. 849–874, Dec. 2018.
[254] M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the talk: Connecting language, knowledge, and action in route instructions," in *Proc. AAAI*, 2006, p. 4.
[255] T. Kollar, S. Tellex, D. Roy, and N. Roy, "Toward understanding natural language directions," in *Proc. 5th ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI)*, Mar. 2010, pp. 259–266.
[256] D. S. Chaplot, K. M. Sathyendra, R. K. Pasumarthi, D. Rajagopal, and R. Salakhutdinov, "Gated attention architectures for task-oriented language grounding," in *Proc. AAAI*, 2018.
[257] Y. Artzi and L. Zettlemoyer, "Weakly supervised learning of semantic parsers for mapping instructions to actions," in *Proc. ACL*, 2013.
[258] H. Chen, A. Shur, D. Misra, N. Snavely, and Y. Artzi, "Touchdown: Natural language navigation and spatial reasoning in visual street environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 12538–12547.
[259] D. Misra, J. Langford, and Y. Artzi, "Mapping instructions and visual observations to actions with reinforcement learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017.
[260] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel, "Interpreting visually-grounded navigation instructions in real environments," in *Proc. CVPR*, 2018.
[261] D. L. Chen and R. J. Mooney, "Learning to interpret natural language navigation instructions from observations," in *Proc. AAAI*, 2011.
[262] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy, "Understanding natural language commands for robotic navigation and mobile manipulation," Tech. Rep.
[263] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI*, 2008.
[264] R. Agarwal, C. Liang, D. Schuurmans, and M. Norouzi, "Learning to generalize from sparse and underspecified rewards," 2019, *arXiv:1902.07198*. [Online]. Available: http://arxiv.org/abs/1902.07198
[265] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. NIPS*, 2016, pp. 4565–4573.
[266] M. Babes, V. Marivate, K. Subramanian, and M. L. Littman, "Apprenticeship learning about multiple intentions," in *Proc. ICML*, 2011, pp. 1–8.
[267] J. Li, W. Monroe, and D. Jurafsky, "Learning to decode for future success," 2017, *arXiv:1701.06549*. [Online]. Available: https://arxiv.org/abs/1701.06549
[268] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 2672–2680.
[269] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2017, pp. 1–7.
[270] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–16.
[271] M. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," 2017, *arXiv:1705.03122*. [Online]. Available: https://arxiv.org/abs/1705.03122
[272] D. Monroe, "Deep learning takes on translation," *Commun. ACM*, vol. 60, no. 6, pp. 12–14, May 2017.
[273] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 3104–3112.
[274] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.
[275] J. Wu, J. B. Tenenbaum, and P. Kohli, "Neural scene de-rendering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 699–707.
[276] L. Lei, Y. Tan, S. Liu, K. Zhang, K. Zheng, and X. Shen, "Deep reinforcement learning for autonomous Internet of Things: Model, applications and challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1722–1760, 3rd Quart., 2020.
[277] M. Saravanan, P. S. Kumar, and A. Sharma, "IoT enabled indoor autonomous mobile robot using CNN and Q-learning," in *Proc. IEEE Int. Conf. Ind. 4.0, Artif. Intell., Commun. Technol. (IAICT)*, Jul. 2019, pp. 7–13.
[278] G. Künzel, G. P. Cainelli, I. Müller, and C. E. Pereira, "Weight adjustments in a routing algorithm for wireless sensor and actuator networks using q-learning," *IFAC-PapersOnLine*, vol. 51, no. 10, pp. 58–63, 2018.
[279] X. Liu, Z. Qin, and Y. Gao, "Resource allocation for edge computing in IoT networks via reinforcement learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[280] X. Qiu, T. A. Nguyen, and M. L. Crow, "Heterogeneous energy storage optimization for microgrids," *IEEE Trans. Smart Grid*, vol. 7, no. 3, pp. 1453–1461, May 2016.
[281] B. Mbuwir, F. Ruelens, F. Spiessens, and G. Deconinck, "Reinforcement learning-based battery energy management in a solar microgrid," *Energy-Open*, vol. 2, no. 4, p. 36, 2017.
[282] Y. Lim and H.-M. Kim, "Strategic bidding using reinforcement learning for load shedding in microgrids," *Comput. Electr. Eng.*, vol. 40, no. 5, pp. 1439–1446, Jul. 2014.
[283] X. Xiao, C. Dai, Y. Li, C. Zhou, and L. Xiao, "Energy trading game for microgrids using reinforcement learning," in *Proc. Int. Conf. Game Theory Netw.* Cham, Switzerland: Springer, 2017, pp. 131–140.
[284] X.-G. Kim, Y. Zhang, M. van der Schaar, and J.-W. Lee, "Dynamic pricing for smart grid with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr./May 2014, pp. 640–645.
[285] B.-G. Kim, Y. Zhang, M. van der Schaar, and J.-W. Lee, "Dynamic pricing and energy consumption scheduling with reinforcement learning," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2187–2198, Sep. 2016.
[286] P. P. Reddy and M. M. Veloso, "Strategy learning for autonomous agents in smart grid markets," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011.
[287] E. Foruzan, L.-K. Soh, and S. Asgarpoor, "Reinforcement learning approach for optimal distributed energy management in a microgrid," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 5749–5758, Sep. 2018.
[288] A. Yu, R. Palefsky-Smith, and R. Bedi, "Deep reinforcement learning for simulated autonomous vehicle control," Course Project Rep., 2016, pp. 1–7.
[289] M. Vitelli and A. Nayebi, "CARMA: A deep reinforcement learning approach to autonomous driving," Stanford Univ., Stanford, CA, USA, Tech. Rep., 2016.
[290] B. Mirchevska, C. Pek, M. Werling, M. Althoff, and J. Boedecker, "High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2156–2162.
[291] M. K. Pal, R. Bhati, A. Sharma, S. K. Kaul, S. Anand, and P. B. Sujit, "A reinforcement learning approach to jointly adapt vehicular communications and planning for optimized driving," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 3287–3293.
[292] R. F. Atallah, C. M. Assi, and M. J. Khabbaz, "Scheduling the operation of a connected vehicular network using deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1669–1682, May 2018.
[293] Z. Wen, D. O'Neill, and H. Maei, "Optimal demand response using device-based reinforcement learning," *IEEE Trans. Smart Grid*, vol. 6, no. 5, pp. 2312–2324, Sep. 2015.

[294] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra, "Residential demand response using reinforcement learning," in *Proc. 1st IEEE Int. Conf. Smart Grid Commun.*, Oct. 2010, pp. 409–414.

[295] T. J. Walsh, A. Nouri, L. Li, and M. L. Littman, "Learning and planning in environments with delayed feedback," *Auton. Agents Multi-Agent Syst.*, vol. 18, no. 1, p. 83, 2009.

[296] R. Lu, S. H. Hong, and X. Zhang, "A dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach," *Appl. Energy*, vol. 220, pp. 220–230, Jun. 2018.

[297] R. Lu and S. H. Hong, "Incentive-based demand response for smart grid with reinforcement learning and deep neural network," *Appl. Energy*, vol. 236, pp. 937–949, Feb. 2019.

[298] H. Liu, S. Liu, and K. Zheng, "A reinforcement learning-based resource allocation scheme for cloud robotics," *IEEE Access*, vol. 6, pp. 17215–17222, 2018.

[299] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchainempowered mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8050–8062, Aug. 2019.

[300] X. Cheng, F. Lyu, W. Quan, C. Zhou, H. He, W. Shi, and X. Shen, "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, May 2019.

[301] H. Zhu, Y. Cao, X. Wei, W. Wang, T. Jiang, and S. Jin, "Caching transient data for Internet of Things: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2074–2083, Apr. 2019.

[302] Y. Wei, F. R. Yu, M. Song, and Z. Han, "Joint optimization of caching, computing, and radio resources for fog-enabled IoT using natural actor–critic deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2061–2073, Apr. 2019.

[303] P. Wang and C.-Y. Chan, "Formulation of deep reinforcement learning architecture toward autonomous driving for on-ramp merge," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–6.

[304] G. Sartoretti, W. P. Y. Wu, T. S. Kumar, S. Koenig, and H. Choset, "Distributed reinforcement learning for multi-robot decentralized collective construction," in *Distributed Autonomous Robotic Systems*. Cham, Switzerland: Springer, 2019, pp. 35–49.

[305] Q. Qi, J. Wang, Z. Ma, H. Sun, Y. Cao, L. Zhang, and J. Liao, "Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4192–4203, May 2019.

[306] Q. Qi and Z. Ma, "Vehicular edge computing via deep reinforcement learning," 2018, *arXiv:1901.04290*. [Online]. Available: http://arxiv.org/abs/1901.04290

[307] G. K. Venayagamoorthy, R. K. Sharma, P. K. Gautam, and A. Ahmadi, "Dynamic energy management system for a smart microgrid," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 8, pp. 1643–1656, Aug. 2016.

[308] M. Kwon, J. Lee, and H. Park, "Intelligent IoT connectivity: Deep reinforcement learning approach," *IEEE Sensors J.*, vol. 20, no. 5, pp. 2782–2791, Mar. 2020.

[309] Y. Su, X. Lu, Y. Zhao, L. Huang, and X. Du, "Cooperative communications with relay selection based on deep reinforcement learning in wireless sensor networks," *IEEE Sensors J.*, vol. 19, no. 20, pp. 9561–9569, Oct. 2019.

[310] A. S. Leong, A. Ramaswamy, D. E. Quevedo, H. Karl, and L. Shi, "Deep reinforcement learning for wireless sensor scheduling in cyber-physical systems," 2018, *arXiv:1809.05149*. [Online]. Available: http://arxiv.org/abs/1809.05149

[311] D. Li, S. Xu, and J. Zhao, "Partially observable double DQN based IoT scheduling for energy harvesting," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2019, pp. 1–6.

[312] L. Huang, X. Feng, C. Zhang, L. Qian, and Y. Wu, "Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing," *Digit. Commun. Netw.*, vol. 5, no. 1, pp. 10–17, Feb. 2019.

[313] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019.

[314] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for IoT devices with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.

[315] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, early access, Mar. 4, 2019, doi: 10.1109/TETC.2019.2902661.

[316] J. Ren, H. Wang, T. Hou, S. Zheng, and C. Tang, "Federated learning-based computation offloading optimization in edge computingsupported Internet of Things," *IEEE Access*, vol. 7, pp. 69194–69201, 2019.

[317] H. Sasaki, T. Horiuchi, and S. Kato, "A study on vision-based mobile robot learning by deep Q-network," in *Proc. 56th Annu. Conf. Soc. Instrum. Control Eng. Jpn. (SICE)*, Sep. 2017, pp. 799–804.

[318] J. Xin, H. Zhao, D. Liu, and M. Li, "Application of deep reinforcement learning in mobile robot path planning," in *Proc. Chin. Automat. Congr. (CAC)*, Oct. 2017, pp. 7112–7116.

[319] T. Yasuda and K. Ohkura, "Collective behavior acquisition of real robotic swarms using deep reinforcement learning," in *Proc. 2nd IEEE Int. Conf. Robot. Comput. (IRC)*, Jan. 2018, pp. 179–180.

[320] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," 2019, *arXiv:1901.06455*. [Online]. Available: http://arxiv.org/abs/1901.06455

[321] M. A. Khamis and W. Gomaa, "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework," *Eng. Appl. Artif. Intell.*, vol. 29, pp. 134–151, Mar. 2014.

[322] V. Francois-Lavet, D. Taralla, D. Ernst, and R. Fonteneau, "Deep reinforcement learning solutions for energy microgrids management," in *Proc. Eur. Workshop Reinforcement Learn. (EWRL)*, 2016.

[323] Y. Ji, J. Wang, J. Xu, X. Fang, and H. Zhang, "Real-time energy management of a microgrid using deep reinforcement learning," *Energies*, vol. 12, no. 12, p. 2291, Jun. 2019.

[324] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg, "On-line building energy optimization using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3698–3708, Jul. 2019.

[325] L. Xiao, X. Xiao, C. Dai, M. Pengy, L. Wang, and H. V. Poor, "Reinforcement learning-based energy trading for microgrids," 2018, *arXiv:1801.06285*. [Online]. Available: http://arxiv.org/abs/1801.06285

[326] R. Furuta, N. Inoue, and T. Yamasaki, "PixelRL: Fully convolutional network with reinforcement learning for image processing," *IEEE Trans. Multimedia*, vol. 22, no. 7, pp. 1704–1719, Jul. 2020.

[327] S. Woo, J. Yeon, M. Ji, I.-C. Moon, and J. Park, "Deep reinforcement learning with fully convolutional neural network to solve an earthwork scheduling problem," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Miyazaki, Japan, Oct. 2018, pp. 4236–4242, doi: 10.1109/SMC.2018.00717.

[328] A. M. Hafiz, "Image classification by reinforcement learning with two-state Q-learning," 2020, *arXiv:2007.01298*. [Online]. Available: http://arxiv.org/abs/2007.01298

[329] G. Brunner, O. Richter, Y. Wang, and R. Wattenhofer, "Teaching a machine to read maps with deep reinforcement learning," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2018.

[330] Q. Cao, L. Lin, Y. Shi, X. Liang, and G. Li, "Attention-aware face hallucination via deep reinforcement learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 690–698.

[331] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*.

[332] F. Liu, S. Li, L. Zhang, C. Zhou, R. Ye, Y. Wang, and J. Lu, "3DCNN-DQN-RNN: A deep reinforcement learning framework for semantic parsing of large-scale 3D point clouds," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5678–5687.

[333] D. Pfau and O. Vinyals, "Connecting generative adversarial networks and actor-critic methods," 2016, *arXiv:1610.01945*. [Online]. Available: https://arxiv.org/abs/1610.01945

[334] C. Finn, P. Christiano, P. Abbeel, and S. Levine, "A connection between gans, inverse reinforcement learning, and energy-based models," in *Proc. Workshop Adversarial Training (NIPS)*.

[335] Z. Jie, X. Liang, J. Feng, X. Jin, W. F. Lu, and S. Yan, "Tree-structured reinforcement learning for sequential object localization," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 127–135.

[336] Y. Rao, J. Lu, and J. Zhou, "Attention-aware deep reinforcement learning for video face recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3931–3940.

[337] X. Kong, B. Xin, Y. Wang, and G. Hua, "Collaborative deep reinforcement learning for joint object search," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1695–1704.
[338] S. Mathe, A. Pirinen, and C. Sminchisescu, "Reinforcement learning for visual object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2894–2902.
[339] S. Welleck, J. Mao, K. Cho, and Z. Zhang, "Saliency-based sequential image attention with multiset prediction," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5173–5183.
[340] J. Supancic, III, and D. Ramanan, "Tracking as online decision-making: Learning a policy from streaming videos with reinforcement learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 322–331.
[341] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2711–2720.
[342] N. Rhinehart and K. M. Kitani, "First-person activity forecasting with online inverse reinforcement learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3696–3705.
[343] A. Jeerige, D. Bein, and A. Verma, "Comparison of deep reinforcement learning approaches for intelligent game playing," in *Proc. 9th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Las Vegas, NV, USA, 2019, pp. 0366–0371.
[344] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," 2019, *arXiv:1912.10944*. [Online]. Available: https://arxiv.org/abs/1912.10944
[345] C. Amato and G. Shani, "High-level reinforcement learning in strategy games," in *Proc. 9th Int. Conf. Auton. Agents Multiagent Syst. (AAMAS)*, Richland, SC, USA, vol. 1, 2010, pp. 75–82.
[346] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, pp. 354–359, Oct. 2017.
[347] M. N. U. Islam and A. Mitschele-Thiel, "Reinforcement learning strategies for self-organized coverage and capacity optimization," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 2818–2823.
[348] J. Moysena and L. Giupponi, "From 4G to 5G: Self-organized network management meets machine learning," *Comput. Commun.*, vol. 129, pp. 248–268, Sep. 2018.
[349] Y. Aytar, T. Pfaff, D. Budden, T. Paine, Z. Wang, and N. de Freitas, "Playing hard exploration games by watching YouTube," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2930–2941.
[350] B. Christopher *et al.*, "Dota 2 with large scale deep reinforcement learning," 2019, *arXiv:1912.06680*. [Online]. Available: https://arxiv.org/abs/1912.06680
[351] Z. Tang, D. Zhao, Y. Zhu, and P. Guo, "Reinforcement learning for build-order production in StarCraft II," in *Proc. 8th Int. Conf. Inf. Sci. Technol. (ICIST)*, Jun. 2018, pp. 153–158.
[352] K. Shao, D. Zhao, N. Li, and Y. Zhu, "Learning battles in vizdoom via deep reinforcement learning," in *Proc. IEEE Conf. Comput. Intell. Games*, Aug. 2018, pp. 1–4.
[353] T. Chen, G. Shahar, Z. Tom, J. M. Daniel, and M. Shie, "A deep hierarchical approach to lifelong learning in minecraft," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 1553–1561.
[354] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castaneda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, and T. Graepel, "Human-level performance in first-person multiplayer games with population-based deep reinforcement learning," *CoRR*, 2018.
[355] M. Sahisnu, L. Bing, W. Shuai, Z. Yingxuan, L. Lifeng, and L. Jian, "Action permissibility in deep reinforcement learning and application to autonomous driving," in *Proc. ACM SIGKDD Conf. Knowl. Discovery Data Mining*, 2018, pp. 1–10.
[356] G. B. Marc, N. Yavar, V. Joel, and H. B. Michael, "The arcade learning environment: An evaluation platform for general agents," *J. Artif. Intell. Res.*, vol. 47, pp. 253–279, Jun. 2013.
[357] J. Arthur, B. Vincent-Pierre, G. Y. V. Esh, H. Hunter, M. Marwan, and L. Danny, "Unity: A general platform for intelligent agentst," *CoRR*, 2018.
[358] J. Matthew, H. Katja, H. Tim, and B. David, "The Malmo platform for artificial intelligence experimentation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 4246–4247.
[359] M. C. Machado, M. G. Bellemare, E. Talvitie, J. Veness, M. Hausknecht, and M. Bowling, "Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents," *J. Artif. Intell. Res.*, vol. 61, pp. 523–562, Mar. 2018.
[360] R. R. Torrado, P. Bontrager, J. Togelius, J. Liu, and D. Perez-Liebana, "Deep reinforcement learning for general video game AI," in *Proc. IEEE Conf. Comput. Intell. Games (CIG)*, Maastricht, The Netherlands, Aug. 2018, pp. 1–8.
[361] M. Lanctot *et al.*, "OpenSpiel: A framework for reinforcement learning in games," 2019, *arXiv:1908.09453*. [Online]. Available: http://arxiv.org/abs/1908.09453
[362] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaskowski, "ViZDoom: A doom-based AI research platform for visual reinforcement learning," in *Proc. IEEE Conf. Comput. Intell. Games (CIG)*, Sep. 2016, pp. 1–8.
[363] O. Vinyals *et al.*, "StarCraft II: A new challenge for reinforcement learning," 2017, *arXiv:1708.04782*. [Online]. Available: https://arxiv.org/abs/1708.04782
[364] G. Synnaeve, N. Nardelli, A. Auvolat, S. Chintala, T. Lacroix, Z. Lin, F. Richoux, and N. Usunier, "TorchCraft: A library for machine learning research on real-time strategy games," 2016, *arXiv:1611.00625*. [Online]. Available: http://arxiv.org/abs/1611.00625
[365] C. Beattie *et al.*, "DeepMind lab," 2016, *arXiv:1612.03801*. [Online]. Available: http://arxiv.org/abs/1612.03801
[366] M. Johnson, K. Hofmann, T. Hutton, and D. Bignell, "The malmo platform for artificial intelligence experimentation," in *Proc. IJCAI*, 2016, pp. 4246–4247.
[367] W. Zhang and T. G. Dietterich, "A reinforcement learning approach to job-shop scheduling," in *Proc. 14th Int. Joint Conf. Artif. Intell. (IJCAI)*, vol. 2. San Francisco, CA, USA: Morgan Kaufmann Publishers, 1995, pp. 1114–1120.
[368] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. 15th ACM Workshop Hot Topics Netw. (HotNets)*, New York, NY, USA, 2016, pp. 50–56.
[369] G. Tesauro, "Online resource allocation using decompositional reinforcement learning," in *Proc. AAAI*, 2005, pp. 886–891.
[370] J. A. Boyan and M. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in *Proc. 6th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers, 1993, pp. 671–678.
[371] M. L. Littman and J. A. Boyan, "A distributed reinforcement learning scheme for network routing," in *Proc. Int. Workshop Appl. Neural Netw. Telecommun.*, 1993, pp. 45–51.
[372] Z. Xu, Y. Wang, J. Tang, J. Wang, and M. C. Gursoy, "A deep reinforcement learning based framework for power-efficient resource allocation in cloud RANs," in *Proc. IEEE ICC*, May 2017, pp. 1–6.
[373] T. M. Hackett, S. G. Bilén, P. V. R. Ferreira, A. M. Wyglinski, and R. C. Reinhart, "Implementation of a space communications cognitive engine," in *Proc. Cogn. Commun. Aerosp. Appl. Workshop (CCAA)*, 2017, pp. 1–7.
[374] R. Li, Z. Zhao, Q. Sun, C.-L. I, C. Yang, X. Chen, M. Zhao, and H. Zhang, "Deep reinforcement learning for resource management in network slicing," Nov. 2018, pp. 1–12, *arXiv:1805.06591*. [Online]. Available: http://arxiv.org/abs/1805.06591
[375] Y. S. Nasir and D. Guo, "Deep reinforcement learning for distributed dynamic power allocation in wireless networks," *arXiv:1808.00490*, Apr. 2019, pp. 1–12. [Online]. Available: https://arxiv.org/abs/1808.00490
[376] K. Shen and W. Yu, "Fractional programming for communication systems—Part I: Power control and beamforming," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2616–2630, May 2018.
[377] N. Liu, Z. Li, Z. Xu, J. Xu, S. Lin, Q. Qiu, J. Tang, and Y. Wang, "A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning," in *Proc. 37th IEEE Int. Conf. Distrib. Comput. (ICDCS)*, Jun. 2017, pp. 372–382.
[378] M. Naeem, S. Bashir, Z. Ullah, and A. A. Syed, "A near optimal scheduling algorithm for efficient radio resource management in multi-user MIMO systems," *Wireless Pers. Commun.*, vol. 106, pp. 1411–1427, Mar. 2019, doi: 10.1007/s1127.
[379] M. Naeem, S. Bashir, A. A. Syed, and U. Khan, "Performance analysis of scheduling algorithms for multiuser MIMO systems," in *Proc. IEEE 12th Int. Conf. Appl. Sci. Technol.*, Islamabad, Pakistan, Jan. 2016.

[380] M. Naeem, S. Bashir, U. Khan, and A. A. Syed, "Modified SINR based user selection for MU-MIMO systems," in *Proc. IEEE 6h Int. Conf. Inf. Commun. Technol.*, Karachi, Pakistan, Dec. 2015, pp. 1–4.

[381] M. Naeem, U. Khan, S. Bashir, and A. A. Syed, "Modified leakage based user scheduling for MU-MIMO systems," in *Proc. IEEE 13th Int. Conf. Frontiers Inf. Technol. (FIT)*, Islamabad, Pakistan, Dec. 2015, pp. 1–4.

[382] R. A. Sutton, D. McAllester, and Y. M. S. Singh, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. 12th Int. Conf. Neural Inf. Process. Syst.*, 1999, pp. 1057–1063.

[383] Y. Yang, Y. Li, K. Li, S. Zhao, R. Chen, J. Wang, and S. Ci, "DECCO: Deep-learning enabled coverage and capacity optimization for massive MIMO systems," *IEEE Access*, vol. 6, pp. 23361–23371, 2018.

[384] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. 15th ACM Workshop Hot Topics Netw.*, 2016, pp. 50–56.

[385] T. Li, Z. Xu, J. Tang, and Y. Wang, "Model-free control for distributed stream data processing using deep reinforcement learning," *Proc. VLDB Endowment*, vol. 11, no. 6, pp. 705–718, Feb. 2018.

[386] G. D. Arnold, "Deep reinforcement learning in large discrete action spaces," Apr. 2016, pp. 1–11, *arXiv:1512.07679*. [Online]. Available: https://arxiv.org/abs/1512.07679

[387] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. ACM Workshop Hot Topics Netw. (HotNets)*, 2016, pp. 50–56.

[388] J. C. Caicedo and S. Lazebnik, "Active object localization with deep reinforcement learning," 2015, *arXiv:1511.06015*. [Online]. Available: http://arxiv.org/abs/1511.06015

[389] X. Kong, B. Xin, Y. Wang, and G. Hua, "Collaborative deep reinforcement learning for joint object search," 2017, pp. 1–11, *arXiv:1702.05573v1*. [Online]. Available: https://arxiv.org/abs/1702.05573v1

[390] D. Zhang, H. Maei, X. Wang, and Y.-F. Wang, "Deep reinforcement learning for visual object tracking in videos," 2017, pp. 1–10, *arXiv:1701.08936v2*. [Online]. Available: https://arxiv.org/abs/1701.08936v2

[391] M. Pinol, A. Sappa, A. Lopez, and R. Toledo, "Feature selection based on reinforcement learning for objectrecognition," in *Proc. Adapt. Learn. Agent Workshop*, Valencia, Spain, Jun. 2012, pp. 4–8.

[392] M. Pinol, "Reinforcement learning of visual descriptors for object recognition," Ph.D. dissertation, Univ. Autonomade Barcelona, Bellaterra, Spain, 2014, pp. 1–109.

[393] F. Sadegh and S. Levine, "CAD2RL: Real single-image flight without a single real image," 2017, pp. 1–12, *arXiv:1611.04201v4*. [Online]. Available: https://arxiv.org/abs/1611.04201v4

[394] C. Craye, D. Filliat, and J.-F. Goudou, "Environment exploration for object-based visual saliency learning," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Stockholm, Sweden, May 2016, pp. 2303–2309.

[395] Y.-E. Liu, T. Mandel, E. Brunskill, and Z. Popovic, "Trading off scientific knowledge and user learning with multi-armed bandits," in *Proc. Educ. Data Mining (EDM)*, 2014, pp. 161–168.

[396] T. Mandel, Y. E. Liu, S. Levine, E. Brunskill, and Z. Popovic, "Offline policy evaluation across representations with applications to educational games," in *Proc. Int. Conf. Auton. Agents Multiagent Syst. (AAMAS)*, 2014, pp. 1077–1084.

[397] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machinelearning algorithms," *J. Banking Finance*, vol. 34, pp. 2767–2787, Nov. 2010.

[398] C. Phua, V. Lee, K. Smith, and R. Gayler, "A comprehensive survey of data mining-based fraud detection research," 2010, *arXiv:1009.6119*. [Online]. Available: https://arxiv.org/abs/1009.6119

[399] J. C. Hull, *Options, Futures, and Other Derivatives*, 9th ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2014.

[400] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 653–664, Mar. 2016.

[401] M. W. Brandt, A. Goyal, P. Santa-Clara, and J. R. Stroud, "A simulation approach to dynamic portfolio choice with an application to learning about return predictability," *Rev. Financial Stud.*, vol. 18, no. 3, pp. 831–873, 2005.

[402] A. W. Lo, "The adaptive markets hypothesis: Market efficiency from an evolutionary perspective," *J. Portfolio Manage.*, vol. 30, pp. 15–29, 2004.

[403] L. Prashanth, C. Jie, M. Fu, S. Marcus, and C. Szepesvári, "Cumulative prospect theory meets reinforcement learning: Prediction and control," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1406–1415.

[404] D. Silver, L. Newnham, D. Barker, S. Weller, and J. McFall, "Concurrent reinforcement learning from customer interactions," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 924–932.

[405] G. Theocharous, P. S. Thomas, and M. Ghavamzadeh, "Personalized ad recommendation systems for life-time value optimization with guarantees," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2015, pp. 1–7.

[406] X. Li, L. Li, J. Gao, X. He, J. Chen, L. Deng, and J. He, "Recurrent reinforcement learning: A hybrid approach," 2015, *arXiv:1509.03044*. [Online]. Available: https://arxiv.org/abs/1509.03044

[407] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. Int. World Wide Web Conf. (WWW)*, 2010, pp. 661–670.

[408] F. Ruelens, B. J. Claessens, S. Vandael, B. D. Schutter, R. Babuska, and R. Belmans, "Residential demand response of thermostatically controlled loads using batch reinforcement learning," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2149–2159, Sep. 2017.

[409] A. Surana, S. Sarkar, and K. K. Reddy, "Guided deep reinforcement learning for additive manufacturing control application," in *Proc. NIPS Deep Reinforcement Learn. Workshop*, 2016.

[410] D. Hein, S. Depeweg, M. Tokic, S. Udluft, A. Hentschel, T. A. Runkler, and V. Sterzing, "A benchmark environment motivated by industrial control problems," in *Proc. IEEE Symp. Adapt. Dyn. Program. Reinforcement Learn. (IEEE ADPRL)*, 2017.

[411] B. Balle, M. Gomrokchi, and D. Precup, "Differentially private policy evaluation," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 2130–2138.

[412] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proc. ACM CCS Workshop Artif. Intell. Secur.*, 2017, pp. 3–14.

[413] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*. [Online]. Available: https://arxiv.org/abs/1604.07316

[414] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, and U. Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," 2017, *arXiv:1704.07911*. [Online]. Available: https://arxiv.org/abs/1704.07911

[415] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," 2019, *arXiv:1904.12901*. [Online]. Available: https://arxiv.org/abs/1904.12901

[416] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[417] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester, "An empirical investigation of the challenges of real-world reinforcement learning," 2020, *arXiv:2003.11881*. [Online]. Available: https://arxiv.org/abs/2003.11881

[418] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," 2017, *arXiv:1703.03400*. [Online]. Available: https://arxiv.org/abs/1703.03400

[419] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4754–4765.

[420] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for YouTube recommendations," in *Proc. 10th ACM Conf. Rec. Syst.*, 2016, pp. 191–198.

[421] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, and B. Coppin, "Deep reinforcement learning in large discrete action spaces," 2016, *arXiv:1512.07679*. [Online]. Available: https://arxiv.org/abs/1512.07679

[422] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," 2018, *arXiv:1801.08757*. [Online]. Available: https://arxiv.org/abs/1801.08757

[423] M. Botvinick, S. Ritter, J. X. Wang, ZebKurth-Nelson, C. Blundell, and D. Hassabis, "Reinforcement learning, fast and slow," *Trends Cognit. Sci.*, vol. 23, no. 5, pp. 408–422, May 2019.

[424] H. Hachiya, T. Akiyama, M. Sugiayma, and J. Peters, "Adaptive importance sampling for value function approximation in off-policy reinforcement learning," *Neural Netw.*, vol. 22, no. 10, pp. 1399–1410, Dec. 2009.

[425] C. Kim, "Deep reinforcement learning by balancing offline monte carlo and online temporal difference use based on environment experiences," *Symmetry* vol. 12, no. 10, p. 1685, 2020, doi: 10.3390/sym12101685.

[426] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, PP. 78–87, 2012.
[427] S. Gu, T. Lillicrap, Z. Ghahramani, R. Turner, B. Schlkopf, and S. Levine, "Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, San Diego, CA, USA, 2017, pp. 3846–3855.
[428] G. Chen, "Merging deterministic policy gradient estimations with varied bias-variance tradeoff for effective deep reinforcement learning," 2019, *arXiv:1911.10527*. [Online]. Available: http://arxiv.org/abs/1911.10527
[429] V. Mnih, A. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2016, pp. 1928–1937.
[430] A. Koppel, A. S. Bedi, and K. Rajawat, "Controlling the bias-variance tradeoff via coherent risk for robust learning with kernels," in *Proc. Amer. Control Conf. (ACC)*, Philadelphia, PA, USA, Jul. 2019, pp. 3519–3525, doi: 10.23919/ACC.2019.8814879.
[431] Z. Yang, Y. Yu, C. You, J. Steinhardt, and Y. Ma, "Rethinking bias-variance trade-off for generalization of neural networks," 2020, *arXiv:2002.11328*. [Online]. Available: https://arxiv.org/abs/2002.11328
[432] H. Mohamed and D. J. Rezende, "Variational information maximisation for intrinsically motivated reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2125–2133.
[433] P. Auer and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, nos. 2–3, pp. 235–256, 2002.
[434] N. Steimle and B. T. Denton, "Markov decision processes for screening and treatment of chronic diseases," in *Markov Decision Processes in Practice*. 2017.
[435] S. S. Jaakkola and M. Jordan, "Reinforcement learning algorithm for partially observable Markov decision problems," in *Proc. Adv. Neural Inf. Process. Syst.*, 1995, pp. 345–352.
[436] S. Padakandla, K. J. Prabuchandran, and S. Bhatnagar, "Reinforcement learning in non-stationary environments," 2019, *arXiv:1905.03970*. [Online]. Available: http://arxiv.org/abs/1905.03970
[437] T. Y. Tang, S. Egerton, and N. Kubota, "Reinforcement learning in non-stationary environments: An intrinsically motivated stress based memory retrieval performance (SBMRP) model," in *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, Beijing, China, Jul. 2014, pp. 1728–1735, doi: 10.1109/FUZZ-IEEE.2014.6891757.
[438] D. de Oliveira, "Reinforcement learning based control of traffic lights in non-stationary environments: A case study in a microscopic simulator," in *Proc. EUMAS*, 2006.
[439] B. C. Da Silva, E. W. Basso, A. L. C. Bazzan, and P. M. Engel, "Dealing with non-stationary environments using context detection," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 217–224.
[440] S. Abdallah and M. Kaisers, "Addressing environment non-stationarity by repeating Q-learning updates," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1582–1612, 2016.
[441] V. Lomonaco, K. Desai, E. Culurciello, and D. Maltoni, "Continual reinforcement learning in 3D non-stationary environments," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 248–249.
[442] A. T. Z. Kasgari, W. Saad, M. Mozaffari, and H. V. Poor, "Experienced deep reinforcement learning with generative adversarial networks (GANs) for model-free ultra reliable low latency communication," 2019, *arXiv:1911.03264*. [Online]. Available: http://arxiv.org/abs/1911.03264
[443] X. Yi, E. Walia, and P. Babyn, "Generative adversarial network in medical imaging: A review," *Med. Image Anal.*, vol. 58, Dec. 2019, Art. no. 101552.
[444] A. Antoniou, A. Storkey, and H. Edwards, "Data augmentation generative adversarial networks," 2017, *arXiv:1711.04340*. [Online]. Available: https://arxiv.org/abs/1711.04340
[445] S. Shao, P. Wang, and R. Yan, "Generative adversarial networks for data augmentation in machine fault diagnosis," *Comput. Ind.*, vol. 106, pp. 85–93, Apr. 2019.
[446] M. A. Tanner and W. H. Wong, "The calculation of posterior distributions by data augmentation," *J. Amer. Stat. Assoc.*, vol. 82, no. 398, pp. 528–540, Jun. 1987.
[447] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Process. Lett.*, vol. 24, no. 3, pp. 279–283, Mar. 2017.
[448] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, p. 60, Dec. 2019.
[449] W. Zhang, X. Li, X.-D. Jia, H. Ma, Z. Luo, and X. Li, "Machinery fault diagnosis with imbalanced data using deep generative adversarial networks," *Measurement*, vol. 152, Feb. 2020, Art. no. 107377.
[450] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5907–5915.
[451] W. Xiong, W. Luo, L. Ma, W. Liu, and J. Luo, "Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2364–2373.
[452] P. A. T. Haarnoja, H. Tang, and S. Levine, "Reinforcement learning with deep energy-based policies," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1352–1361.
[453] A. Potapov and M. K. Ali, "Convergence of reinforcement learning algorithms and acceleration of learning," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 67, no. 2, Feb. 2003, Art. no. 026706.
[454] S. Liu, K. Y. Ngiam, and M. Feng, "Deep reinforcement learning for clinical decision support: A brief survey," Jul. 2019, *arXiv:1907.09475v1*. [Online]. Available: https://arxiv.org/abs/1907.09475v1
[455] A. Nowe, P. Vrancx, and Y.-M. De Hauwere, "Game theory and multi-agent reinforcement learning," in *Reinforcement Learning* (Adaptation, Learning, and Optimization), vol. 12. Berlin, Germany: Springer, 2012.
[456] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *IEEE Signal Process. Mag.*
[457] M. Denil, P. Agrawal, T. D. Kulkarni, T. Erez, P. Battaglia, and N. de Freitas, "Learning to perform physics experiments via deep reinforcement learning," in *Proc. ICLR*, 2017, pp. 1–15.

MUDDASAR NAEEM received the master's degree in telecommunication and networks from IQRA University Islamabad in 2015. He was a Visiting Lecturer with Quaid-i-Azam University Islamabad. He is currently a Researcher Scholar with the University of Naples Parthenope, Naples, Italy. His research interests include application of reinforcement learning and deep learning in healthcare.

SYED TAHIR HUSSAIN RIZVI received the Ph.D. degree in computer and control engineering from the Politecnico di Torino, Torino, Italy, in 2018. He is currently an Assistant Professor of computer engineering with The University of Lahore, Lahore, Pakistan. His research interests include the efficient realization of algorithms on embedded systems and machine learning.

ANTONIO CORONATO is curremtly Head of the Research Group on Self-Learning and Self-Adaptive Systems, Institute for High Performance Computing and Networking, National Research Council of Italy. He is also the Co-founder and the Co-Editor in Chief of the *Journal of Reliable Intelligent Environments*. He has participated to the organization of many international workshops and conferences, including several editions of Intelligent Environments. He has also edited Special Issues of *ACM Transactions on Autonomous Systems*, *ACM Transactions on Embedded Systems*, IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS and other journals.