Intelligent Environments 2021
E. Bashir and M. Luštrek (Eds.)

doi:10.3233/AISE210096

Inverse Reinforcement Learning Through Max-Margin Algorithm

Syed Ihtesham Hussain Shah [a,b,1], Antonio Coronato [b]
[a] *Dept. of ICT and Engineering, University of Parthenope, Italy*
[b] *Institute for high performance computing and networking (ICAR), National Research Council, Italy*

Abstract. Reinforcement Learning (RL) methods provide a solution for decision-making problems under uncertainty. An agent finds a suitable policy through a reward function by interacting with a dynamic environment. However, for complex and large problems it is very difficult to specify and tune the reward function. Inverse Reinforcement Learning (IRL) may mitigate this problem by learning the reward function through expert demonstrations. This work exploits an IRL method named Max-Margin Algorithm (MMA) to learn the reward function for a robotic navigation problem. The learned reward function reveals the demonstrated policy (expert policy) better than all other policies. Results show that this method has better convergence and learned reward functions through the adopted method represents expert behavior more efficiently.

Keywords. Reinforcement Learning, Inverse Reinforcement Learning, Markov Decision Process, Max-Margin Algorithm

1. Introduction

Machine Learning (ML) is an application of Artificial Intelligence (AI) that provides the ability to automatically learn and improve from experience without being explicitly programmed. ML focuses on the development of computer programs that can access data and use it to learn for themselves.
RL is one of the ML paradigms where agents solve problems by acquiring experiences via interactions with the environment [1]. The result of agent's interaction is a policy that can provide solutions to complex tasks without having specific knowledge about the underlying problem [2] in several fields (e.g., healthcare [3]). Due to the generalization capability of RL, it can be applied to more complex scenarios [4]. However, the specification of a reward function in advance is a problem in RL that may cause design difficulties.
Most of the time, researchers are supposed to specify the reward function manually to infer optimal decision [5,6]. However, the reward signals may be extremely scattered under this settings. Most rewards are almost zero. This technique may lead to the difficulty

[1]Corresponding Author: Syed Ihtesham Hussain Shah, Institute for high performance computing and networking (ICAR), National Research Council (CNR), Via Pietro Castellino, 111 Napoli – 80131, Italy. E-mail: ihtesham.shah@icar.cnr.it

of recognition which actions are useful in obtaining the ultimate feedback [7].
IRL can solve this problem where the derivation of the reward function is performed from expert demonstrations and behavior [8]. In the last 20 years, IRL has gained the attention of many researchers in fields of psychology, AI, ML and control theory. In IRL the goal is to learn the underlying rewards function for the expert demonstrations or trajectories.
In this paper, we adopt a technique of maximum margin IRL [9] for finding a best-fitting reward function for expert trajectories. The reward function can be considered as a linear combination of features function and weight vector. Features functions are predefined basis functions and the weight vector is computed by maximizing the margin between both the expert feature expectations and the estimated feature expectations. The solution is acceptable when this margin goes below a predefined threshold value. The result shows that learned reward functions under this scheme are true underlying reward functions for expert trajectories.
The rest of the paper is organized as follows: Related work in emerging and existing fields is discussed in section-2 that presents an overview of the state of the art. Section-3 comprises a brief review of the background and problem formulation, where basics about the Markov Decision Process (MDP) and IRL linear programming are presented. Section-4 introduces our proposed model. This section describes the problem of learning the reward function not explicitly, but through observing an expert demonstration. Experimental setup and results are discussed in section-5. We summarize the paper in section-6 by giving a conclusion and future directions.

2. Related work

RL has experienced growth in attention and interest due to promising results in intelligent environments [10–12] and the areas like: playing AlphaGo [13], controlling systems in robotics [14–16], medical [17], atari [18] and competitive video . A method of investigating challenges posed by reporting procedures, reproducibility and proper experimental techniques through Deep Reinforcement Learning (DRL) is discussed in [19]. Generally, Imitation Learning (IL) is categorized into three types: adversarial imitation learning (AIL) behaviour cloning (BC) and (IRL). Behaviour Cloning (BC) [20] directly maps states to an actions and learns policy through supervised learning. BC can avoid interacting with the environment. However, BC introduces a compounding error without considerable improvement during training.
In IRL [9, 21] the goal is to learn the reward function based on the expert demonstrations. It models the intention and preference of the demonstrator. Maximum Likelihood IRL (MLIRL) [22] estimate the gradient of the likelihood function. It defines that likelihood of the data set can be represented by the product of likelihood of the state-action pairs.
However, as compared with IRL, BC [23] learns policy directly by minimizing the Jensen-Shannon divergence between learned policy and expert policy [24]. Therefore different techniques is utilized to recover both uncertainty and rewards information. In a continuous state and action spaces Gaussian process [25] is utilized. Deep GP model [26] mounds multiple hidden GP layers. It has the ability of learning complicated reward structures with very limited demonstrations. Leveraged Gaussian processes [27] can

learn from both negative and positive demonstrations. Apprenticeship Learning (AL) is to learn a reward function that illuminates the demonstrated policy [28] a margin better than alternative policies.
BC and RL [29,30] are two major methods used now a days in robotics, autonomous car driving and healthcare sector. Dynamic Treatment Regime (DTR) [31–34] oversimplify personalized medicine and treatment is frequently tailored to a patient's dynamic-state. When the Electronic Health Record (EHR) are optimal and plentiful, BC can effectively recover the doctor's policies.

3. Preliminaries

3.1. Markov Decision Process.

MDP is used for decision making (e.g. [35]) and is based on Markov property, which does not consider past information when taking actions and depends on only current state. MDP is a form of tuple (S,A,T,R,γ) [36]. Where $S=(s_1,s_2,s_3,.....s_n)$ represent set of states that exist in the given environment. The agent can move to any of these states based upon the chosen action and transition function. $A=(a_1,a_2,.....a_n)$ is a set of actions that the agent can take in a state at every time step. $T(s_t,a_t,s_(t+1))$ is a transition function which tells the probability of ending up in a state $s_(t+1)$ if you take action a in state s in time step t. $\gamma\in[0,1]$: is a discount factor that takes the value between zero and one. A value of zero gives more weight to immediate rewards and a value close to one gives more weight to long term rewards.

3.2. Reinforcement Learning

The rationale of RL is to interact with the environment without having any prior knowledge and to learn how to achieve a goal as stated in Figure 1. The component that interacts with the environment is called **Agent**. An agent takes an action a_t in a state s_t at time step t and the environment returns the next state and reward. The aim of the RL agent is to learn an optimal **Policy**, which tells which action to take in a state in order to maximize a long term reward. The aggregated reward is estimated as given in Eq. (1).

$$\begin{aligned} V^{\pi}(s) &= E_{\pi}\{R_t|s_t=s\} \\ &= \sum_{a\in A(s)}\pi(s,a)\sum_{s_{t+1}\in S}T^{a}_{ss_{t+1}}\{r(s,a)+\gamma V^{\pi}(s_{t+1})\} \end{aligned} \tag{1}$$

Where V^{π} is the value function to estimate the accumulated reward when you start in state s_t and follow the current policy π. R_t is the reward function which is calculated as given below.

$$R_t = r_{t+1}+\gamma r_{t+2}+\gamma^2 r_{t+3}+.....\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \tag{2}$$

Similarly, the Q value function can be written as:

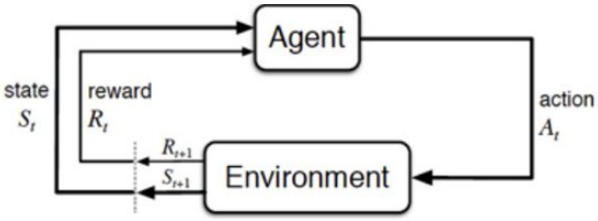

Figure 1. The Reinforcement Learning Problem

$$Q^{\pi}(s,a) = \sum_{\pi}\{R_t | s_t = s, a_t = a\} \tag{3}$$

Q function is the expected aggregated return when you take action a in state s under the current policy π for all state-action pairs afterwards. The goal is to find an optimal policy π^*. The state value function (V) or the state-action value function (Q) for optimal policy π^* using Bellman optimality equations are given below:

$$V^*(s) = \max_a \sum_{s_{t+1} \in S} T^a_{ss_{t+1}}(R^a_{ss_{t+1}} + \gamma V^*(s_{t+1})) \tag{4}$$

$$Q^*(s,a) = \sum_{s_{t+1} \in S} T^a_{ss_{t+1}}(R^a_{ss_{t+1}} + \gamma \max_{at+1} V^*(s_{t+1}, a_{t+1})) \tag{5}$$

Eqs. (4) and (5) explains the target of the optimal policy i.e. is to learn the policy that provides the maximum reward return and it is also considered as the fundamental strategy for several IRL methods.

3.3. Inverse Reinforcement Learning (IRL)

IRL is the problem of learning the preferences of an expert (agent) by observing its behaviour and avoiding the manual specification of a reward function. IRL flips the RL problem and attempts to extract the reward function from the observed behaviour of an agent. We have some expert trajectories that are supposed to generate the optimal policy. Therefore, the goal here is to find the reward function that is being implicitly optimised by the optimal policy π^*.
IRL is expressed as an optimization problem and Linear Programming (LP) [37] can be adopted to solve it by considering three types of state spaces. These state spaces are discussed next.

3.3.1. LP for finite state space

First, consider the optimal policy π^* for a MDP with a finite state space S and the policy transition matrix $T^{\neg\pi}_{n*n}$. For each state only one action provides an optimal solution from the set of K possible actions and all other $k-1$ actions are non-policy actions hence non policy transition matrix are $T^{\neg\pi} = (T^1, \ldots T^{k-1})$. An action is considered optimal only when the reward function satisfies the following equation.

$$(T^{\neg\pi} - T^{\pi})(I - \gamma T^{\pi})^{-1} R0 \tag{6}$$

3.3.2. LP for infinite state space

In infinite state spaces, transition probabilities can not be expressed in terms of matrix. The expected value based on sampled subset $s_0 \subset S$ is estimated and it is guaranteed that actions through the estimated policy always generate a lower expected value than the expert's policy actions.

$$\forall_{s_0 \in S}, \forall_{a \in A} : E_{\acute{s}T(\acute{s}|s,\pi(s)}\{V^{\pi}(\acute{s})\} \geq E_{\acute{s}\,T(\acute{s}|s,a)}\{V^{\pi}(\acute{s})\} \tag{7}$$

3.3.3. LP with sampled trajectories

It deals with the scenario where the knowledge about an exact policy is not given. An optimal policy based on some unknown reward function generates these trajectories. However, the goal is to calculate the empirical value of a trajectory rather than the expected value of the policy. We assume that there is a vector of features ϕ over states and by using the current estimated reward function $\hat{R}$ the corresponding value estimate $\hat{V}_i(s)$ for each value function can be calculated as:

$$\hat{V}_i(S) = \sum_{s_j \in S} \gamma^j \phi_i(s_j) \tag{8}$$

It is also considered that the optimal expert policy π^* always generates an higher empirical value $\hat{V}(\pi^*)$ than any other policy π^i.

$$\hat{V}(\pi^*) \geq \hat{V}(\pi^i) \tag{9}$$

4. System Model

Given the expert trajectories τ^E, the goal is to find the reward function that represents the best explanation of the expert behavior.

$$\begin{aligned} \tau^E &= [(s_1^1, a_1^1, s_2^1, a_2^1, .., s_d^1, a_d^1), (s_1^2, a_1^2, s_2^2, a_2^2, .., s_d^2, a_d^2), ...] \\ &= [\tau^1, \tau^2, \tau^3 ...] \end{aligned} \tag{10}$$

The rewards function can be considered as a linear combination of **k** known, fixed and bounded basis functions ϕ.

$$R(s,a) = w_1\phi_1(s,a) + w_2\phi_2(s,a) + \ldots\ldots + w_k\phi_k(s,a) \tag{11}$$

$$R(s,a) = \sum_i \{w_i \phi_i(s,a)\} \tag{12}$$

Where feature function $\phi_i : S \rightarrow \mathbf{R}$ and weights $w_i \in \mathbf{R}$. The policy π is defined as mapping states to an action and the value of a policy V is the sum of all the discounted rewards that is possible to collect by following that policy as given in Eq. (8). Expectation of a value function V^{π} is defined as:

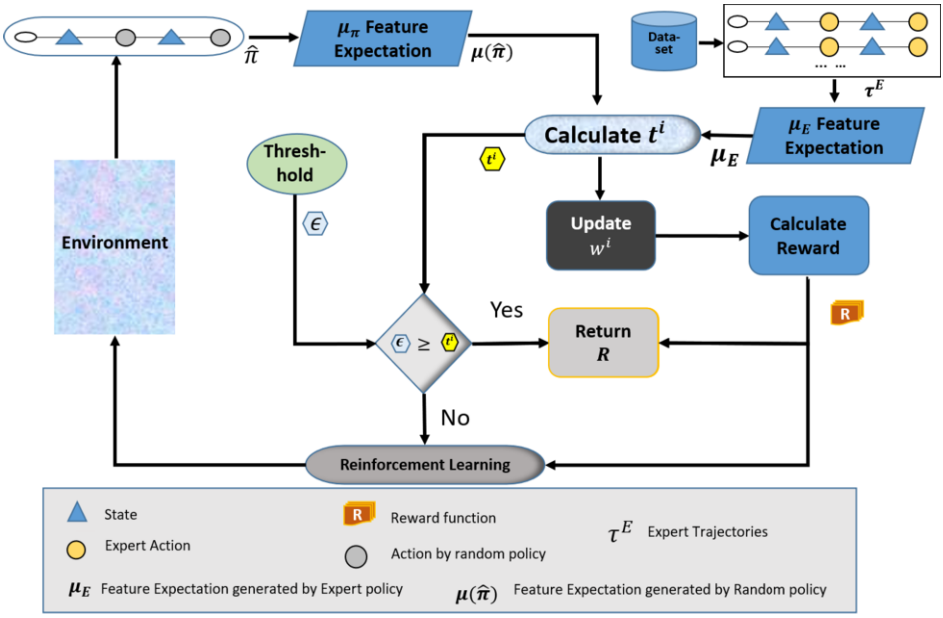

Figure 2. Proposed Layout for Max-Margin Algorithm

$$E[V^{\pi}(s_0)] = E[\sum_{t=1}^{\infty} {}^{t}R(s_t)|\pi] \tag{13}$$

$$= E[\sum_{t=1}^{\infty} {}^{t}w.\phi(s_t)|\pi] \tag{14}$$

$$= w\,.\,E[\sum_{t=1}^{\infty} {}^{t}\phi(s_t)|\pi] \tag{15}$$

$$= w\,.\,\mu_\pi \tag{16}$$

Algorithm-1 is used to find a weight vector w that minimizes the difference between the expert feature expectations μ_E and the estimated feature expectations $\mu_{\hat{\pi}}$. Let consider an MDP without reward MDP/R. Given the expert trajectory τ^E, we calculate expert's feature expectations μ_E. We also estimate the feature expectation $\mu(\hat{\pi})$ specifically by exploring a random policy $\hat{\pi}$. To find a policy $\hat{\pi}$ such that $||\mu_E - \mu\hat{\pi}||_2 \leq \varepsilon$ [38]:

$$E[\sum_{t=1}^{\infty} {}^{t}R(s_t)|\pi_E] - E[\sum_{t=1}^{\infty} {}^{t}R(s_t)|\hat{\pi}] \tag{17}$$

Algorithm 1 Max-margin Algorithm

Given: Expert trajectories τ^E generated by behavior policies π^E, discount factor γ, termination criteria ε,
Initialize: Feature matrix ϕ, number of iteration, $n = \infty$, Expert feature expectation $\mu_E = E[\sum_{t=1}^{\infty} {}^t R(s_t)|\pi_E]$
Randomly pick some policy $\hat{\pi}^0$.
while $(t \geq \varepsilon)$ **do**
    Compute t^i by using Eq. (20) and let the w^i be the value to attain this maximum.
    Compute Reward function $R(w^{(i)})^T \phi$.
    Using RL Algo. find policy $\hat{\pi}^{(i)}$ for this Reward
    Estimate Feature expectation $\mu(\hat{\pi}) = E[\sum_{t=1}^{\infty} \gamma^t \phi(s_t)|\hat{\pi}]$
    Set $i = +1$.
end while
Return: $R = (w^{(i)})^T \phi$

$$= |w^t \mu_E - w^t \mu(\hat{\pi})| \tag{18}$$

$$\begin{aligned} &\leq ||w^t||_2 \, ||\mu(\hat{\pi}) - \mu_E||_2 \\ &\leq \varepsilon \end{aligned} \tag{19}$$

Where $w : (||w||_1 \leq 1)$.
Maximum values of w that minimize the distance between the expert feature expectations and the estimated feature expectations are the true values for w [39].

$$t^i = \max_{w:||w||_2 \leq 1} \min_{j \in \{0..(i-1)\}} w^T (||\mu(\hat{\pi}) - \mu_E||_2) \tag{20}$$

The value of w generates the reward function through dot multiplication with the basis function ϕ.
The optimal policy for this reward function is computed through a RL algorithm. The estimated feature expectations for this kind of policy is calculated and compared with the expert feature expectations. We want to explore a policy which minimizes the euclidean distance stated in Eq. (20). A reward function for such a policy will be the true under laying reward function for the expert policy.
If the algorithm terminates with $t^{(n+1)} \leq \varepsilon$ then it means that there is at-least one policy from the set whose performance is as good as the expert's policy minus ε. For a solution μ_E is separated from μ^i by a margin of at most ε.

$$w^T \mu^i \geq w^T \mu_E - \varepsilon, \ \forall w \ with \ ||w||_2 \leq 1 \tag{21}$$

$$\min ||\mu_E - \mu^i||_2 \leq \varepsilon \tag{22}$$

In this section we have introduced an approach of max-margin algorithm. Layout of proposed model is presented in Figure 2. At the initial stage, some policies are randomly

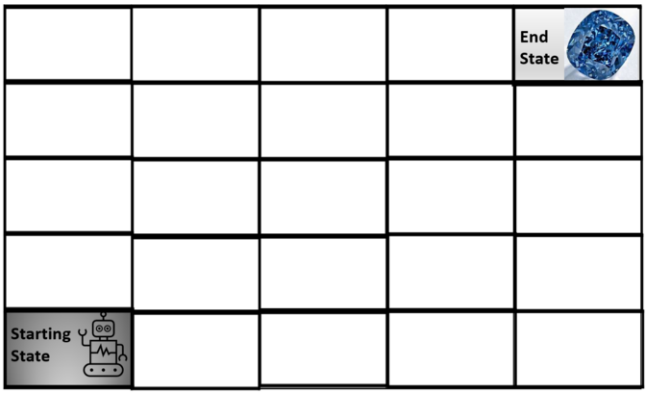

Figure 3. A sample gridworld problem

picked and feature expectation is calculated. The value of the threshold t^i is then computed and compared with a predefined value of ε. In case the value of t^i is not satisfactory, then the weight vector w^i is updated. The weight vector defines the values of the underlying reward function using dot multiplication with feature vector ϕ.
The optimal policy for the current reward function is estimated by utilizing the RL algorithm. The feature expectations for this policy are calculated and found the new value of t^i as in the previous step. Updating weight vector and learning optimal policy for the current reward function through RL is carried out repeatedly until it is found a policy whose feature expectations μ^i are nearly similar to the expert feature expectations μ_E and satisfies the equation given bellow:

$$t^i \leq \varepsilon \tag{23}$$

This policy achieves performance very close to those of the expert's policy. Therefore, the reward function that is used to estimate such an optimal policy through RL is considered the true underlying reward function for the expert trajectories.

5. Experiment

In this section, we report the experiment that we have conducted to evaluate the adopted model. First we have tested the algorithm on a gridword example followed by quantitative and qualitative studies.
A gridworld is a 2-D decision-making example for robotic navigation as stated in [40]. We have considered a 5x5 gridworld for our experiment as shown in Figure 3. The robot has to reach its destination, which is represented by the diamond. Except from the edge states, an agent can take four possible actions at each state including up, $down$, $left$ and $right$ with 30% chance of moving randomly. The grid is divided into non-overlapping regions and for each cell in the region there is a feature $\phi_i(s)$. We have generated different expert trajectories for these non-overlapping regions and hence calculated the expert feature expectation μ_E. On the other hand, some initial policies are also generated randomly and computed the estimated feature expectations.
The value of γ is 0.09 and the threshold is equal to 0.01. The adopted approach of MMA

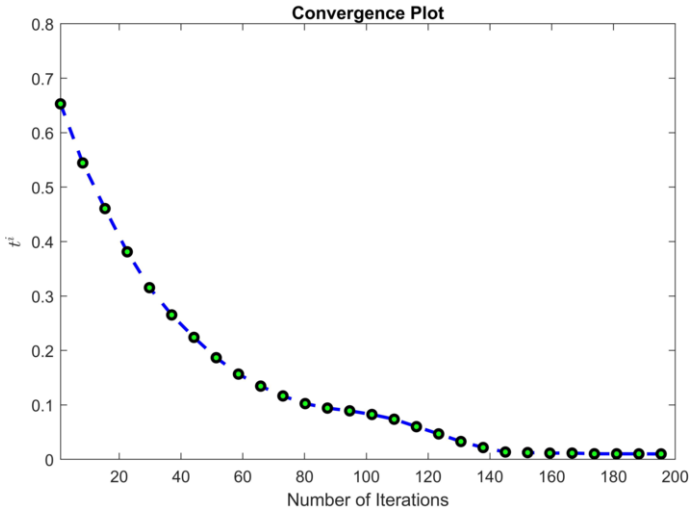

Figure 4. Difference between expert feature expectations and estimated feature expectations along with No. of iterations

is tested to recover the underlying reward function for the given expert trajectories. The difference between the expert feature expectations and the estimated feature expectations is calculated according to Eq. (20). The values of t along with the number of iterations are plotted in Figure 4. It can be seen that the algorithm convergences after a few iterations. The results for initial rewards and recovered rewards are shown in Figure 5 with different colours at each state.

The groundtruth reward represents the initial reward distribution for the gridworld. It can be seen that all states except the end state have zero reward initially. The top right corner (yellow box) represents the end state with the highest reward.

On the other hand, after having run successfully the adopted approach, the value of the

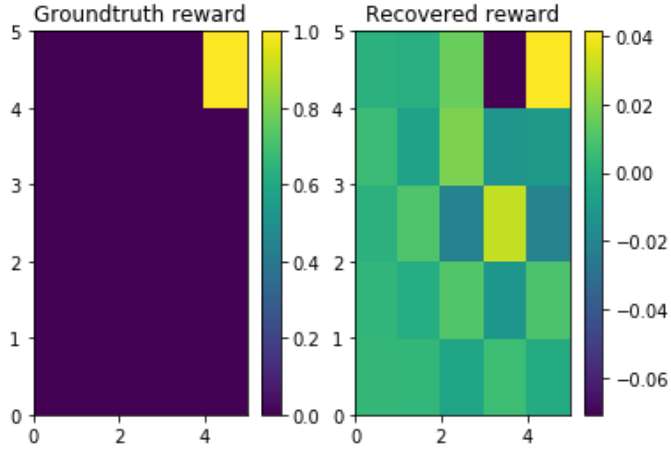

Figure 5. Groundtruth rewards represents the initial reward and after successful implementation of max-margin algorithm under-laying rewards are recovered

recovered rewards at each state is represented beside the "Groundtruth reward" and the scale of colors indicates the behavior and preference for the expert at each state.

6. Conclusion

In this paper, we have adopted a max-margin technique that is based on IRL. It focuses on determining the true underlying reward function for some given demonstrations. The MMA approach assumed that the reward function is shaped as a linear function of known features. This recovered reward function helps RL and MDP problems to mimic the expert behavior in an efficient way. Results achived for robotic navigation in the gridworld problem show that the MMA provides fast convergence.

List of Acronyms

AI Artificial Intelligence
MDP Markov Decision Process
RL Reinforcement Learning
DRL Deep Reinforcement Learning
DTR Dynamic Treatment Regime
IRL Inverse Reinforcement Learning
IL Imitation Learning
AL Apprenticeship Learning
ML Machine Learning
BC Behaviour Cloning
MLIRL Maximum Likelihood IRL
MMA Max-Margin Algorithm
LP Linear Programming

Acknowledgment

This work is partly supported by AMICO project which has received funding from the National Programs (PON) of the Italian Ministry of Education, Universities and Research (MIUR): code ARS0100900 (Decreen.1989, 26 July 2018)

References

[1] M. Naeem, S. T. H. Rizvi, and A. Coronato, "A gentle introduction to reinforcement learning and its application in different fields," *IEEE Access*, vol. 8, pp. 209 320–209 344, 2020.
[2] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.
[3] A. Coronato, M. Naeem, G. De Pietro, and G. Paragliola, "Reinforcement learning for intelligent healthcare applications: A survey," *Artificial Intelligence in Medicine*, vol. 109, p. 101964, 2020.
[4] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[5] 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 review of medical devices*, vol. 10, no. 5, pp. 661–673, 2013.

[6] S. A. Murphy, "Optimal dynamic treatment regimes," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 65, no. 2, pp. 331–355, 2003.

[7] 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," *arXiv preprint arXiv:1705.08422*, 2017.

[8] S. Zhifei and E. M. Joo, "A survey of inverse reinforcement learning techniques," vol. 5, no. 3, pp. 293–311, 2012.

[9] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[10] P. Ribino and C. Lodato, "A norm compliance approach for open and goal-directed intelligent systems," *Complexity*, vol. 2019, 2019.

[11] C. Di Napoli, P. Ribino, and L. Serino, "Customisable assistive plans as dynamic composition of services with normed-qos," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–26, 2021.

[12] P. Ribino and C. Lodato, "A distributed fuzzy system for dangerous events real-time alerting," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 11, pp. 4263–4282, 2019.

[13] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[14] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, and T. Erez, "Learning continuous control policies by stochastic value gradients," *arXiv preprint arXiv:1510.09142*, 2015.

[15] P. Ribino, M. Bonomolo, C. Lodato, and G. Vitale, "A humanoid social robot based approach for indoor environment quality monitoring and well-being improvement," *International Journal of Social Robotics*, pp. 1–20, 2020.

[16] M. Bonomolo, P. Ribino, and G. Vitale, "Explainable post-occupancy evaluation using a humanoid robot," *Applied Sciences*, vol. 10, no. 21, p. 7906, 2020.

[17] C. Lodato and P. Ribino, "A novel vision-enhancing technology for low-vision impairments," *Journal of medical systems*, vol. 42, no. 12, pp. 1–13, 2018.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[19] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[20] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural computation*, vol. 3, no. 1, pp. 88–97, 1991.

[21] Y. Gao, J. Peters, A. Tsourdos, S. Zhifei, and E. M. Joo, "A survey of inverse reinforcement learning techniques," *International Journal of Intelligent Computing and Cybernetics*, 2012.

[22] H. Ratia, L. Montesano, and R. Martinez-Cantin, "On the Performance of Maximum Likelihood Inverse Reinforcement Learning," in *arXiv preprint arXiv*, 2012, p. 1202.1558.

[23] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in neural information processing systems*, 2016, pp. 4565–4573.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[25] Z.-J. Jin, H. Qian, and M.-L. Zhu, "Gaussian processes in inverse reinforcement learning," in *2010 International Conference on Machine Learning and Cybernetics*, vol. 1. IEEE, 2010, pp. 225–230.

[26] M. Jin, A. Damianou, P. Abbeel, and C. Spanos, "Inverse reinforcement learning via deep gaussian process," *arXiv preprint arXiv:1512.08065*, 2015.

[27] K. Lee, S. Choi, and S. Oh, "Inverse reinforcement learning with leveraged gaussian processes," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3907–3912.

[28] J. A. Bagnell, N. Ratliff, and M. Zinkevich, "Maximum margin planning," in *Proceedings of the International Conference on Machine Learning (ICML)*. Citeseer, 2006.

[29] L. Wang, W. Zhang, X. He, and H. Zha, "Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation," in *Proceedings of the 24th ACM SIGKDD International*

Conference on Knowledge Discovery & Data Mining, 2018, pp. 2447–2456.

[30] Y. Zhang, R. Chen, J. Tang, W. F. Stewart, and J. Sun, "Leap: Learning to prescribe effective and safe treatment combinations for multimorbidity," in *proceedings of the 23rd ACM SIGKDD international conference on knowledge Discovery and data Mining*, 2017, pp. 1315–1324.

[31] J. M. Robins, "Causal inference from complex longitudinal data," in *Latent variable modeling and applications to causality*. Springer, 1997, pp. 69–117.

[32] J. Robins, "A new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect," *Mathematical modelling*, vol. 7, no. 9-12, pp. 1393–1512, 1986.

[33] J. M. Robins, "Correcting for non-compliance in randomized trials using structural nested mean models," *Communications in Statistics-Theory and methods*, vol. 23, no. 8, pp. 2379–2412, 1994.

[34] S. M. Coghlan Jr, M. J. Connerton, N. H. Ringler, D. J. Stewart, and J. V. Mead, "Survival and growth responses of juvenile salmonines stocked in eastern lake ontario tributaries," *Transactions of the American Fisheries Society*, vol. 136, no. 1, pp. 56–71, 2007.

[35] A. Coronato and A. Cuzzocrea, "An innovative risk assessment methodology for medical information systems," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020.

[36] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[37] A. Y. Ng, S. J. Russell *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, vol. 1, 2000, p. 2.

[38] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 729–736.

[39] D. Choi, T.-H. An, K. Ahn, and J. Choi, "Future trajectory prediction via rnn and maximum margin inverse reinforcement learning," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2018, pp. 125–130.

[40] P. Crook and G. Hayes, "Learning in a state of confusion: Perceptual aliasing in grid world navigation," *Towards Intelligent Mobile Robots*, vol. 4, 2003.