State of the Art on: Configurable Markov Decision Processes

Guglielmo Manneschi, guglielmo.manneschi@mail.polimi.it

1. Introduction to the research topic

Machine Learning (ML) is the field of Artificial Intelligence aimed at the development of algorithms that learn to perform specific tasks from data. These algorithms aim at increasing certain measures of performance by using statistical models to identify patterns in the data. The kind of tasks usually involved are predictions, classifications and decisions.

Reinforcement Learning (RL) is the branch of ML focused on decision making, i.e., on agents that learn to choose actions in a real or simulated environment, to achieve certain goals. The word *reinforcement* refers to the psychological attitude of humans and other animals to increase or strengthen the response to a certain stimulus when a positive reward is received. The same behavioural modification is what RL uses to make an artificial agent mimic the learning process of animals.

The most prestigious conferences related to the field of Machine Learning are NeurIPS (Neural Information Processing Systems) and ICML (International Conference on Machine Learning). In the last few years the attendence to these conferences has increased significantly, proving the growing interest in the field not only from the academic community but also from industry. Other top conferences, still relevant, but with a wide spectrum on the AI field, are AAAI (AAAI Conference on Artificial Intelligence) and IJCAI (International Joint Conference on Artificial Intelligence). Relevant journals are the *Journal of Machine Learning Research* (Microtome), *Transactions on Pattern Analysis and Machine Intelligence* (IEEE) and *Machine Learning* (Springer).

1.1. Preliminaries

1.1.1 Markov Decision Processes

The theoretical framework used to represent the typical learning scenario in Reinforcement Learning (RL) [17] is based on Markov Decision Processes (MDPs) [14]. An MDP is a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma, \mu)$ composed by a set of states $\mathcal{S}$ that represents what the agent perceives of the environment, a set of actions $\mathcal{A}$ that the agent can do, a probability distribution $P(s'|s,a)$ on the new state of the environment s' given the previous state s and the chosen action a, a reward function $R(s,a)$ that is the expected immediate reward that the agent receives when it performs action a on state s. Additionally there is an initial state distribution $\mu(s)$ that gives the probability of s being the first state of an episode, and a discount factor γ which indicates how much the agent gives importance to future rewards versus immediate rewards, i.e., how much it is far-sighted.

The agent chooses each action following a probability distribution given by a (possibly stochastic) policy $\pi(a|s)$, that depends on the current state s. The reward signal is the reinforce mechanism of the learning process, it is a measure of how much the agent is doing well on a certain task, and the goal of the agent is to change its policy so to maximize the future rewards.

The performance of a policy is evaluated through the *expected return*, i.e. the expected discounted sum of the rewards r_t collected along a trajectory τ (i.e. a sequence of state-action pairs), where the expectation is taken over the probability density function $p_\pi(\tau)$ of observing the trajectory τ given the policy π:

$$J^\pi = E_{\tau \sim p_\pi(\tau)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right],$$

which can also be expressed as:

$$J^\pi = \frac{1}{1-\gamma} \int_{\mathcal{S}} d^\pi(s) \int_{\mathcal{A}} \pi(a|s) R(s,a) \mathrm{d}a \, \mathrm{d}s,$$

where $d^\pi(s)$ is the γ-discounted state distribution [16], defined recursively as:

$$d^\pi(s) = (1-\gamma)\mu(s) + \gamma \int_{\mathcal{S}} d^\pi(s') P^\pi(s'|s) ds',$$

and P^π is the state kernel function defined as $P^\pi(s'|s) = \int_{\mathcal{A}} \pi(a|s) P(s'|s,a) da$. Solving an MDP consists in finding a policy π^* that maximizes J^π.

1.1.2 Tools

Reinforcement Learning algorithms are often written in Python, a high-level interpreted programming language which has become very popular for its simplicity. The mathematical part can be implemented using the scientific library SciPy, that offers an efficient interface to deal with linear algebra and other types of problems.

Many toolkits have been developed to speed up the implementation of these algorithms, one that is becoming a standard in the field is OpenAI Gym [4]. It provides a vast collection of environments that expose a common interface, to let researchers test their algorithms on many control problems and compare the performances. Specific libraries that directly implement RL algorithms are Baselines [7] and Garage [8].

To deal with powerful function approximations such as neural networks, there are many frameworks available such as Tensorflow [1], Caffe [9], and Torch [6]. The advantage in using such libraries is that they provide tools to easily optimize the parameters of the model with gradient-based approaches, without the need of computing gradients explicitly.

1.2. Research topic

Configurable Markov Decision Processes (Conf-MDPs) [10] are an extension of MDPs that add the possibility to configure the environment. We can imagine the presence of a supervisor modifying the probability distributions $P(s'|s,a)$ that define the model's dynamics or the initial states distribution.

A possible application in the task of driving a car, is to change how the car reacts to the agent's actions, such as increasing the maximum speed or modifying the aerodynamical properties. Another example is the interaction between a student and an automatic teaching system, in which the student or an external entity can change the difficulties of the questions or the speed at which the concepts are presented. Finally, the design of a street network, in which the configuration of semaphores' transition times could reduce the journey time of the drivers (agents).

Conf-MDP is a novel approach in RL problems, based on the observation that many real-world problems allow the configuration of some environmental parameters. They allow to model problems in a way that cannot be done with the traditional frameworks. Letting the agent configure the environment as part of its policy is in fact inadequate, since the configuration activity can possibly be carried out by an external supervisor. A multi-agent approach would not be suited as well, since the supervisor should act at a different level than the learning agent.

When the environment changes, the optimal policy for the agent will change as well, so this could not only allow the selection of the models that offer a faster convergence of learning algorithms or a higher performance, but also provide insights on the learning process and on the agent capabilities. In fact, they would be useful in the identification of the policy space of an observed agent, that is the space of its possible bahaviours. This analysis is deeply tied to Imitation Learning, since it would both make use of Behavioural Cloning, the algorithms that aim at replicating the policy used by an observed agent by only observing its trajectories, and improve Inverse Reinforcement Learning, a set of methods to retrieve the reward function used by the agent.

2. Main related works

2.1. Classification of the main related works

Conf-MDP can be placed inside a group of more general research fields. A classification of the main related works can be done using two criteria, similarity of frameworks on one side and affinity of applications on the other.

The first and most recent result in the literature to propose the presented research topic is [10]. It provides an algorithm that guarantees a monotonic performance improvement in the search for a model-policy pair when the model space is known. Some open issues are the extension to a partially unknown environment, and the use of policy search methods.

Similar topics in the literature consider the environment no longer as a fixed and unknown entity, but rather as partially known or changing over time. One case is environments with imprecise probabilities, where the transition function is specified with probability distributions, and another is non-stationary environments.

Changing the environment multiple times would make the agent find different optimal policies, and if we have access only to its trajectories, such as if we are observing an expert, these trajectories may be used to gather information on the learning process, such as the reward function used.

The main research fields related to this problem is Inverse Reinforcement Learning. IRL aims at recovering the reward function that motivated the agent behaviour given only a set of trajectories. Having the reward function also allows to recover the original policy. The advantage of these techniques, with respect to Behavioural Cloning is that they allow to find a representation of what motivates the agent, instead of an exact formulation of its policy, and hence allow from one side to reconstruct the policy after the retrieval of the reward function, and from the other to obtain a representation of its possible behaviours independent of one particular environment.

2.2. Brief description of the main related works

In [10] the first formulation of the presented research problem can be found. Given a policy space Π and a model space $\mathcal{P}$, the performance measure $J_\mu^{P,\pi}$ is defined taking into account both the agent's policy π and the selected environment configuration P. In this way, it is possible to optimize the parameters to find the model and the policy that give the highest performance. The algorithm constitutes a safe approach in this search of a policy-model couple, i.e., it ensures a monotonic improvement on the performance via a dissimilarity penalization that let only updates to a near region of the search space.

Other studies in the literature worked on similar frameworks, for instance in which the environment dynamics is not fixed or is partially under control of the agent. Markov Decision Processes with imprecise probabilities [15, 18, 5], consider an uncertainty over the transition model. Non-stationary environments [3] take into account that the transition probabilities change over time. Both these approaches, however, don't admit the possibility to dynamically change the environment.

About IRL, the first approaches appeared in the literature require an interaction of the agent with the environment [12, 2]. The limits of these techniques is that they require to solve the "forward problem" of computing the optimal policy at each iteration of the algorithms. Recently, a different approach emerged to solve this issue, for instance in [13], where the algorithm is based on gradient minimization and does not need to solve the forward problem, it requires, however, a representation of the policy space of the agent.

2.3. Discussion

The main open issues on Conf-MDPs are the unexplored possibilities that the configuration of the environment could offer. As already noted, the framework can be extended to include the cases in which the environment is not fully known, and in which a policy search method is used. Both directions would allow getting a broader set of methods to achieve higher performance in this context.

From the insights provided by [11], configuring the environment could be used in IRL to help identify the state-features space compatible with the expert's trajectories and its unknown gradient.

Another interesting research direction could be that of using the model configuration to understand the policy space of an agent. Since the various configurations would make the agent change its optimal behaviour, observing the various policies in multiple environments it may be possible to shape its policy space.

References

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)* (2016), pp. 265–283.

[2] Abbeel, P., and Ng, A. Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning* (2004), ACM, p. 1.

[3] Bowerman, B. L. Nonstationary markov decision processes and related topics in nonstationary markov chains.

[4] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016.

[5] Bueno, T. P., Mauá, D. D., Barros, L. N., and Cozman, F. G. Modeling markov decision processes with imprecise probabilities using probabilistic logic programming. In *Proceedings of the Tenth International Symposium on Imprecise Probability: Theories and Applications* (2017), pp. 49–60.

[6] Collobert, R., Kavukcuoglu, K., and Farabet, C. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop* (2011).

[7] Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. `https://github.com/openai/baselines`, 2017.

[8] Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning* (2016), pp. 1329–1338.

[9] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093* (2014).

[10] Metelli, A. M., Mutti, M., and Restelli, M. Configurable markov decision processes. In *35th International Conference on Machine Learning* (2018), vol. 80, PMLR, pp. 3491–3500.

[11] Metelli, A. M., Pirotta, M., and Restelli, M. Compatible reward inverse reinforcement learning. In *Advances in Neural Information Processing Systems* (2017), pp. 2050–2059.

[12] Ng, A. Y., Russell, S. J., et al. Algorithms for inverse reinforcement learning. In *Icml* (2000), vol. 1, p. 2.

[13] Pirotta, M., and Restelli, M. Inverse reinforcement learning through policy gradient minimization. In *Thirtieth AAAI Conference on Artificial Intelligence* (2016).

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

[15] Satia, J. K., and Lave Jr, R. E. Markovian decision processes with uncertain transition probabilities. *Operations Research 21*, 3 (1973), 728–740.

[16] Sutton, R. S., M. D. A. S. S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems* (2000).

[17] Sutton, R. S., and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

[18] White III, C. C., and Eldeib, H. K. Markov decision processes with imprecise transition probabilities. *Operations Research 42*, 4 (1994), 739–749.