Collaborative Learning with Dirichlet Process Clustering for Rapid Online Adaptation in Robotics

Runjun Mao

August 12, 2023

Abstract

Robots in deployment can face unforeseeable distortion in their motion due to unseen environment or hardware failures. Model-based online adaptation learns this distortion from the interactions with the current environment and corrects its motions accordingly. However, this can be very challenging as it is hard to select the most suitable model and its hyper-parameters for the current dynamics with little prior knowledge. If we could leverage the plentiful historical real-world interactions, we may build statistical models for the common types of dynamics faced by the robot, hence providing guidance for adaptations. This paper developed a method to enable fast online adaptation for the robot by grouping up historical interaction data into clusters. The historical data can be collected from several robots of the same design performing different tasks in different environments, hence allowing learning in a collaborative manner. The method takes an non-parametric approach of novelty by modelling the data distribution with infinite mixture of Gaussian Processes and conducting clustering using Dirichlet Processes. This achieves unbiased training while ensures high efficiency in terms of both data and computation during deployment. The experiment is conducted on a simulated robot to compare with the previous work for online adaptation. The method accurately groups up the collaboratively collected data without any knowledge of the source of each data, and it offers a significant improvement over the baseline in any previously encountered environments. As for unseen environments, it can also efficiently update the clusterings with only a limited number of interactions to provide assistance for future tasks.

1 Introduction

Locomotion controls of legged robots are typically too complex to be designed manually. In recent years, deep Reinforcement-Learning (deep RL) has made huge success in game playing [1, 2] and is regarded promising for application in robotics. However, training such well performing agents requires enormous amounts of interactions with the environment, meaning that this can only be done in simulation where we can model the interaction and collect the training data much faster than in real-time. This raises a problem for application in robotics since there is always a gap between simulation and reality, and the policy trained in simulation could suffer from this sim-to-real transfer. Another problem is that any unforeseen changes to the environment dynamics could lead to severely degraded performance. These deep RL trained agents require extra fine-tuning or even retaining to retain their capabilities. For example, the team of OpenAI Five [3] had to conduct three additional amendments in the architecture followed by extra trainings to incorporate small version updates in Dota 2 during the development of the agent. For game-playing agents, these drawbacks are tolerable as there is no sim-to-real problem, and the environment is overall stationary. For robotics however, sim-to-real must be considered, and the environment dynamics is typically non-stationary as the robot can be travelling between different terrain, carrying different payloads or suffering from damages in its hardware.

It would be better if we may learn the true dynamics from the interactions with the current dynamics and correct our policy accordingly. This is a model-based reinforcement learning (MBRL) solution that allows robots to learn policies with lesser interactions by learning a dynamics model and use it optimize the policy [4, 5, 6]. However, the amount of data required to learn a model typically scales exponentially with the dimensionality of the input space [7], hence it is not suitable for online adaptation. The promising approach to address this is to use a repertoire-based method, where we learn a repertoire of elementary policies (e.g., one policy for turning left, one for moving forward, etc.) using quality diversity optimisation [8] to fully cover the task space (e.g., the 2D displacement we want the robot to make). The repertoire-based method is a hierarchical control method that keeps a variety of elementary policies and treats each policy as an action. We then give each elementary policy a behaviour descriptor (BD) to quantify the way it acts. This enables us to skip the low-level states and actions (i.e., joint encoders and torques for all motors) and to work with the high-level behaviour space that has much smaller dimensionality. During the adaptation, we learn the distortion (called transformation model) in the behaviour space from real-world interactions. Thanks to the reduced dimensionality, the transformation model can be learned in real-time. Instead of optimising the policy with this learned model, we use a repertoire-based control that first predicts the outcomes for each elementary policy in the repertoire and then simply chooses the one with the predicted outcome that is most aligned with the current goal. This is made available for keeping a repertoire of policies rather than a single policy, so that the robot can adapt to the distortion by finding the corresponding policies that compensate the distortion (for example the robot aims to go forward and distortion is left, then a policy originally aiming right-front can be used for going forward instead).

The repertoire-based method has achieved very impressing results in task-solving [9] and online damage recovery [10]. But this method heavily relies on having a good model that is both data efficient and suitable for modelling the distortion. The most commonly used model is Gaussian Process (GP) [11]. The prior mean function and the kernel are the most important factors of GP. If no prior knowledge is given about the dynamics, these have to be selected manually according to experience. If the distortion is very large (like broken legs) or complex, the GPs can be very inefficient or even misleading. If we have data of real-world interactions across several different environments, we can build a GP for each of the situations and then teach the robot to identify the one that mostly explains its current situation. However, such real-world data are expensive to collect and could lead to covariate shift (the collected data distribution is different from that during deployment). A promising solution that is to leverage historical data of real-world interactions collected during deployment. After performing each task, the interaction data can be uploaded to an archive system where they will be grouped into clusters based on their dynamics. Thus, the data doesn't have to be collected on purpose, and the acquired data distribution is completely unbiased. This design allows the robots to learn to adapt in a collaborative manner. If we have multiple robots performing different tasks in different environments, we can quickly get an overview of the commonly encountered dynamics and select the corresponding GPs to cover each cluster. If a robot encounters a new environment, the interaction data will be uploaded and analysed, and then all the robot will be able to quickly adapt to this environment.

There are many challenges needs to be addressed to enable this collaborative learning. First, traditional clustering algorithms like k-nearest neighbours [12] and k-means [13] need to specify the number of clusters. While in our case, this number is not even fixed as we might observe more clusters as we collect more data. Second, we know that each task corresponds to a single environment, and hence we are essentially clustering the tasks instead of clustering the data points. However, since some tasks are tougher than others, different tasks generate data of different sizes. As a result, we will be clustering data of inconsistent sizes. Third, since there is a large repertoire of policies, only a small subset of them will be executed while performing each task. Hence, it is very likely that the data collected from two tasks have no overlapping policies, making it hard to determine whether they have similar dynamics and whether they should be grouped together. Moreover, after we have found the clusters, the data in each cluster can never cover the entire repertoire. Hence it is difficult to determine the prior mean for the policies that are not present in this cluster. This paper aims to solve all the above problems by taking an non-parametric clustering method and using a uniquely designed prior mean function.

References

[1] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[2] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[3] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David

Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[4] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

[5] Konstantinos Chatzilygeroudis, Roberto Rama, Rituraj Kaushik, Dorian Goepp, Vassilis Vassiliades, and Jean-Baptiste Mouret. Black-box data-efficient policy search for robotics. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 51–58. IEEE, 2017.

[6] Rituraj Kaushik, Konstantinos Chatzilygeroudis, and Jean-Baptiste Mouret. Multi-objective model-based policy search for data-efficient learning with sparse rewards. In *Conference on Robot Learning*, pages 839–855. PMLR, 2018.

[7] E Keogh and A Mueen. Curse of dimensionality. encyclopedia of machine learning. c. sammut and gi webb, eds, 2010.

[8] Antoine Cully and Yiannis Demiris. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation*, 22(2):245–259, 2017.

[9] Miguel Duarte, Jorge Gomes, Sancho Moura Oliveira, and Anders Lyhne Christensen. Evolution of repertoire-based control for robots with complex locomotor systems. *IEEE Transactions on Evolutionary Computation*, 22(2):314–328, 2017.

[10] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, and Jean-Baptiste Mouret. Reset-free trial-and-error learning for robot damage recovery. *Robotics and Autonomous Systems*, 100:236–250, 2018.

[11] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[12] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[13] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.