Artificial Intelligence Research and Development
A. Cortés et al. (Eds.)

doi:10.3233/FAIA220359

Mutual Information Weighing for Probabilistic Movement Primitives

Adrià COLOMÉ and Carme TORRAS
Institut de Robòtica i Informàtica Industrial, CSIC-UPC

Abstract. Reinforcement Learning (RL) of trajectory data has been used in several fields, and it is of relevance in robot motion learning, in which sampled trajectories are run and their outcome is evaluated with a reward value. The responsibility on the performance of a task can be associated to the trajectory as a whole, or distributed throughout its points (timesteps). In this work, we present a novel method for attributing the responsibility of the rewards to each timestep separately by using Mutual Information (MI) to bias the model fitting of a trajectory.

Keywords. Reinforcement Learning, Motion Learning, Mutual Information

1. Introduction

In the application of AI to robot motion learning, it is a common approach to use a policy represented as a set of parameters that, applied to a parametrized family of functions, define the robot motion at every timestep. These policy parameters are then perturbed in order to generate new motions that are evaluated by a reward function. Then, a so-called Policy Search (PS) [2,3] method obtains a new policy (set of parameters) from the samples and their measured performance. In generating these new samples, two approaches were defined in [2]: step-based and episodic-based.

In step-based sampling, a perturbation of the policy is generated at every timestep of the trajectory and so each trajectory is given a score (reward) value at every timestep. On the contrary, episode-based sampling perturbs the parameters of a whole trajectory and runs it. Whilst the step-based approach can provide more information when timestep performance can be measured, its application to fields such as robotics can become dangerous, as it induces high-frequency perturbations on the reference motion. This is why, in robotics, step-based methods have been applied to episode-based samples, keeping the idea of associating a reward to each timestep of a sampled trajectory, but with a smoother profile. As an example of such methods, in Policy Improvement with Path Integrals (PI2) [4], the authors define a cost-to-go associated to each sample in order to accordingly give importance to each timestep of each trajectory. This cost-to-go is then associated with a probability used as a weight for updating the policy parameters. This method assigns a responsibility to each timestep by using the time remaining for the trajectory and its deviation from the mean. However, there is no necessary correlation between the deviation from the mean and the reward. A trajectory could present a lot of variability that is completely unrelated to its performance. Therefore, in this work, we explore a way to distribute responsibility by using an indicator of how two variables (deviation from the

mean at current timestep vs whole trajectory reward) are related, namely Mutual Information (MI) [1]. By using MI, we obtain relative weights with which to bias the fitting of the updated policy, i.e, the trajectory stochastic parametrization. In this paper, we firstly present preliminary concepts in Sec. 2, followed by the proposed methodology in Sec. 3 and the conclusions in Sec. 4.

2. Preliminaries

2.1. Mutual Information

In this work, we will use the MI between two variables and its discretization. In general, the mutual information between two random variables is a measure of their dependence, or how much information is obtained from one variable after observing the other, and is calculated as $MI(X;Y) = \sum_Y \sum_X P(x,y)\log\left(\frac{P(x,y)}{P(x)P(y)}\right)$. In our applications, we neither have discrete distributions nor have such analytical expressions so as to calculate the MI. Therefore, we partition the data in bins of an equal number of data points and the discrete version is used. Using such bins will prevent the method from outliers and non-linearities.

2.2. Probabilistic Movement Primitives (ProMPs)

For motion characterization, Movement Primitives (MP) are a popular approach in order to easily encode a trajectory. Amongst the most used MPs, Probabilistic Movement Primitives (ProMPs) [5] present advantages such as that they can fit the time-dependent variability of the motion, and probability operations such as product and conditioning can be used on them. Given a number of basis functions per DoF, N_f, ProMPs use a time-dependent matrix $\Phi_t = [\phi_t^1; ...; \phi_t^{Nf}]$ to encode position, ϕ_t being the vector of normalized kernel basis functions (e.g., uniformly distributed Gaussian basis function over time). Thus, the position and velocity state vector $\mathbf{y}_t$ can be represented as $\mathbf{y}_t = \Phi_t^T \omega + \varepsilon_y$, where $\varepsilon_y \sim \mathcal{N}(0, \Sigma_y)$ is a zero-mean Gaussian noise and the weights ω are also treated as random variables with a distribution $p(\omega) = \mathcal{N}(\omega|\mu_\omega, \Sigma_\omega)$. Subsequently, the parameters of the distribution $\theta = \{\mu_\omega, \Sigma_\omega, \Sigma_y\}$, Σ_y being the state covariance, are fitted by means of a maximum likelihood estimate, i.e., we compute the sample mean and the sample covariance of ω. Then the probability of observing a trajectory τ can be expressed as the product of all timestep probabilities: $p(\tau;\theta) = \prod_t \int \mathcal{N}(\mathbf{y}_t|\Phi_t^T \omega, \Sigma_y)\mathcal{N}(\omega|\mu_\omega, \Sigma_\omega)d\omega$.

3. Methodology

We assume we have a set of N_k trajectories with dimension D, and composed on N_t timesteps each. Therefore, y_{td}^k will correspond to the d-dimension on the $t-$th timestep of the $k-$th trajectory. Associated with each trajectory, we have a reward R_k that is a performance indicator. For each timestep t, we will compute the mutual information $MI(\{y_{td}^k\}_{k\in K}, \{R^k\}_{k\in K})$. Then, we can use an Expectation Maximization-based approach with ProMPs, such as in [6]. However, in [6] the authors considered that the reward for each trajectory has to be associated to the trajectory as a whole. In this paper, we ap-

ply the same method with the following differences: The previous reference use a linear dimensionality reduction in order to reduce the state space's dimension. Here, we omit this part, despite it would be straight-forward to add it if necessary. In this paper, we compute the mutual information between the rewards of the demonstrations and the variability of trajectories and use it to weigh data. Additionally, in order to penalize higher mutual information values in low variance parts of the trajectory, we use the term $\eta_t = \text{trace}(\Sigma_y) \cdot \text{MI}(\{R_k\}_{k=1..Nk}, \{y_{tk}\}_{k=1..Nk})$. By adding a factor of the trace of the variance at each time-step, we enforce that the term η_t is higher when the trajectory values at that timestep present a high correlation with the reward outcome, in addition to these trajectory values also presenting a higher variability.

As we also know the rewards R_k (performance of each trajectory), we used Relative Entropy Policy Search (REPS) in order to convert them to trajectory weights d_k but, differently from [6], we assign each data point $\mathbf{y}_{tk}$ an importance equal to $\delta_{k,t} = \eta_t \cdot d_k$.

3.1. Expectation-Maximization

In this section, we will point out to the main differences with [6] in the formulation. The Expectation Maximization method consists of two steps: First, we evaluate the probability of the model parameters given each trajectory $\mathbf{Y}_k$ (vectorized as a column vector), $p(\mathbf{Y}^k|\omega) = \mathcal{N}(\mathbf{Y}^k|\Psi^T, \mathbf{I}_{N_t} \otimes \Sigma_y)$, where $\otimes$ is the kronecker product and Ψ^T is the matrix of concatenated kernel functions ϕ_t for different dimensions, i.e., $\Psi^T = [I_d \otimes \Phi_1; ...; I_d \otimes \Phi_{N_t}]$. By operating and using Bayes' rule, we can obtain that $p(\omega|\mathbf{Y}^k) = \mathcal{N}(\mu_k, \Sigma_k)$, with

$$\mu_k = \mu_\omega + \Sigma_\omega \Psi \left(\mathbf{I}_{N_t} \otimes \Sigma_y + \Psi^T \Sigma_\omega \Psi\right)^{-1} (\mathbf{Y}^k - \Psi^T \mu_\omega)$$
$$\Sigma_k = \Sigma_\omega - \Sigma_\omega \Psi \left(\mathbf{I}_{N_t} \otimes \Sigma_y + \Psi^T \Sigma_\omega \Psi\right)^{-1} \Psi^T \Sigma_\omega.$$

We will then compute these μ_k, Σ_k, and use them on the Maximization step, where we maximize the weighed expectation of the log-likelihood function in order to obtain the new parameters θ: $L = \sum_{k=1}^{N_k} \sum_{t=1}^{N_y} \delta_{tk} \mathbb{E}_{\omega|\mathbf{y}_t^k;\theta^{old}} \left[\log\left(p(\omega, \mathbf{y}_t^k; \theta)\right)\right]$. By solving this equation (see [6]), we obtain that, at each iteration, the new policy parameters are:

$$\mu_\omega = \left(\sum_{k=1}^{N_k} \sum_{t=1}^{N_t} \delta_{tk}\right)^{-1} \sum_{k=1}^{N_k} \sum_{t=1}^{N_t} \delta_{tk} \mu_k$$
$$\Sigma_\omega = \left(\sum_{k=1}^{N_k} \sum_{t=1}^{N_t} \delta_{tk}\right)^{-1} \sum_{k=1}^{N_k} \sum_{t=1}^{N_t} \delta_{tk} \left[\Sigma_k + (\mu_\omega - \mu_k)(\mu_\omega - \mu_k)^T\right],$$
$$\Sigma_y = \left(\sum_{k=1}^{N_k} \sum_{t=1}^{N_t} \delta_{tk}\right)^{-1} \sum_{k=1}^{N_k} \sum_{t=1}^{N_t} \delta_{tk} [\Psi_t^T \Sigma_k \Psi_t \Omega^T + (\mathbf{y}_t^k - \Psi_t^T \mu_k)(\mathbf{y}_t^k - \Psi_t^T \mu_k)^T].$$

3.2. Results

In order to test this approach, we generated a set of random time-dependent trajectories with high variance around a viapoint and on the endpoint (see Fig. 1 left with the trajectories and their respective mean and variance as a shaded area), and low variance elsewhere. We defined a reward function for each trajectory that penalizes its distance from the desired via-point (in red), i.e., the reward is determined by how far the trajectory passes by a certain via-point. Then, we applied the method in [6] with $r = d$ in the paper, and our method by using REPS [7] to convert trajectory weights to rewards. The

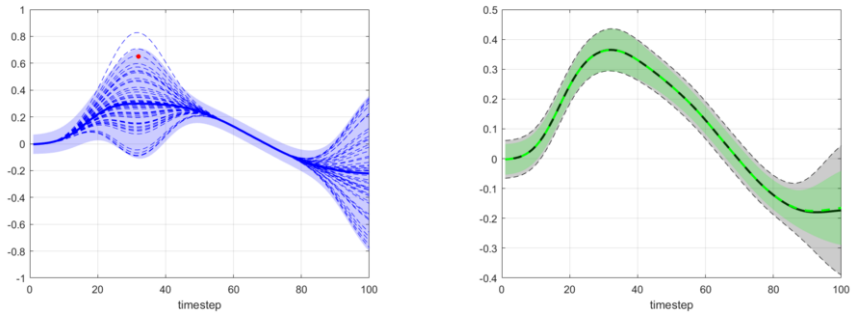

Figure 1. Left: data trajectories used for the proof of concept experiment and their fitting with least-squares. The red dot marks the desired via-point and the reward associated to each trajectory is the distance from that viapoint. On the right plot, the proposed method (in black) against the reference [6]. We see that, while both methods provide a similar result in the area where the reward is decided, at the end of the trajectory, the proposed method allows for more variance, given the fact that the end position does not affect the reward. Note different scales in the vertical axis.

results of fitting both methods are shown in Fig. 1 right, where the mean of both methods (ours in black, [6] in green) is similar, but the variance at the endpoint is larger in our method. This is mostly because the MI between the reward values and the endpoint of the trajectories is small, therefore our method requires less precision and does not give importance on the final point.

4. Conclusions

In this ongoing work paper, we devised a methodology for re-weighing data in order to better distribute the responsibility of the reward throughout a trajectory. The preliminary results in a proof of concept task displayed in Fig. 1 how the method is relieving of responsibility those parts of the trajectory that do not provide a gain to the reward, and therefore allow for more variance. In other words, relating the importance of a trajectory data-point to how much it correlates with the overall performance at that point in the trajectory allows for a better fitting of the trajectory.

References

[1] Cover TM, Thomas JA. Entropy, relative entropy and mutual information. In: Elements of Information Theory. John Wiley & Sons, Inc. ISBN 9780471241959, 2001.

[2] Deisenroth MP, Neumann G, Peters G. A Survey on Policy Search for Robotics. Foundations and Trends in Robotics. 2(1-2): 1-141, 2013.

[3] Chatzilygeroudis K, Vassiliades V, Stulp F, Calinon S, Mouret JB. A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Trans. on Robotics. 36(2): 328-347, 2020.

[4] Stulp F, Theodorou EA, Schaal S. Reinforcement Learning With Sequences of Motion Primitives for Robust Manipulation. IEEE Transactions on Robotics 28(6):1360-1370, 2012.

[5] A. Paraschos, G Neumann, C. Daniel, and J. Peters, "Probabilistic movement primitives". *In Advances in NIPS*, Cambridge, MA: MIT Press., 2013.

[6] A. Colomé, G. Neumann, J. Peters and C. Torras "Dimensionality Reduction for Probabilistic Movement Primitives". *IEEE-RAS Humanoid Robots*, pp. 794-800, 2014.

[7] J. Peters, K. Mülling and Y. Altün, "Relative Entropy Policy Search".*Twenty-Fourth National Conf. on Artificial Intelligence*, pp 182-189, 2011.