Recursive History Representations for Unsupervised Reinforcement Learning in Multiple-Environments

Anonymous Authors[1]

Abstract

In recent years, the area of Unsupervised Reinforcement Learning (URL) has gained particular relevance. In this setting, an agent is pre-trained in an environment with reward-free interactions, often through a maximum state entropy objective that drives the agent towards a uniform coverage of the state space. It has been shown that this pre-training phase leads to significant performance improvements in downstream tasks later given to the agent to solve. The multiple-environments version of this setting introduces the problem of controlling the performance trade-offs in the environment class and leads to the following question: Can we build Pareto optimal policies for multiple-environments URL? In this work, we answer this question by proposing a novel non-Markovian policy architecture to be trained with the maximum state entropy objective. This architecture showcases significant empirical advantages when compared to state-of-the-art Markovian agents.

1. Introduction

The unsupervised pre-training of learning models on massive unlabelled datasets has lead to remarkable successes in supervised learning tasks, such as language modelling (e.g., Brown et al., 2020; Alayrac et al., 2022) and image generation (Ramesh et al., 2022). With this inspiration, Unsupervised Reinforcement Learning (URL) (Laskin et al., 2021) brings a similar learning paradigm to sequential decision problems. In this setting, a learning agent first interacts with a reward-free environment to pre-train an exploratory behavior. Then, the pre-trained behavior is exploited to address a wide range of downstream supervised tasks, with the goal of improving the adaptation efficiency w.r.t. learning from scratch (i.e., starting with a randomly initialized behavior).

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

While several approaches have been considered for the unsupervised pre-training phase (e.g., Bellemare et al., 2016; Pathak et al., 2017; Eysenbach et al., 2018), the *Maximum State Entropy* (MSE) objective (Hazan et al., 2019; Mutti & Restelli, 2020) has emerged as a powerful alternative. The MSE objective prescribes the agent to maximize the entropy of the state visitations induced by its behavior. In this way, whatever task the agent has to face in the supervised phase, the goal state will be reached with a significant probability under the exploratory behavior, which allows the agent to quickly adapt to the optimal task-specific behavior. This recipe allows to achieve compelling results when tackling the URL problem in both continuous control (Mutti et al., 2021) and visual domains (Liu & Abbeel, 2021; Seo et al., 2021; Yarats et al., 2021).

Recent works (Mutti et al., 2022b; Parisi et al., 2021) have considered a further generalization of the URL problem, in which the unsupervised pre-training is performed over a class of environments instead of a single one, and the supervised phase challenges the agent with any task defined on any environment within the class. Notably, the multiple-environments formulation is not a trivial extension of the single-environment URL setting: One can trade-off the pre-training performance over the environments in any way, making the problem inherently multi-objective (Mutti et al., 2022b; Hayes et al., 2022). Optimizing for the average performance over the class may lead to a pre-trained behavior that is bad for some of the environments, so that the agent may struggle with the subsequent supervised tasks in those environments. Conversely, directly accounting for the tail of the pre-training objective, as in (Mutti et al., 2022b), could limit the efficiency of the adaptation w.r.t. the optimal environment-specific behavior. The previous considerations lead to wonder whether a unique behavior that is simultaneously maximizing the pre-training objective in all the environments of the class can be actually learned. While finding such a behavior within the class of stationary Markovian strategies seems far-fetched, the analysis in (Mutti et al., 2022a) suggests that a non-Markovian strategy of this kind exists. Unfortunately, computing the optimal non-Markovian strategy for the MSE objective is known to be impractical in general, as the set of all the histories grows exponentially with the episode horizon (Mutti et al., 2022a). In

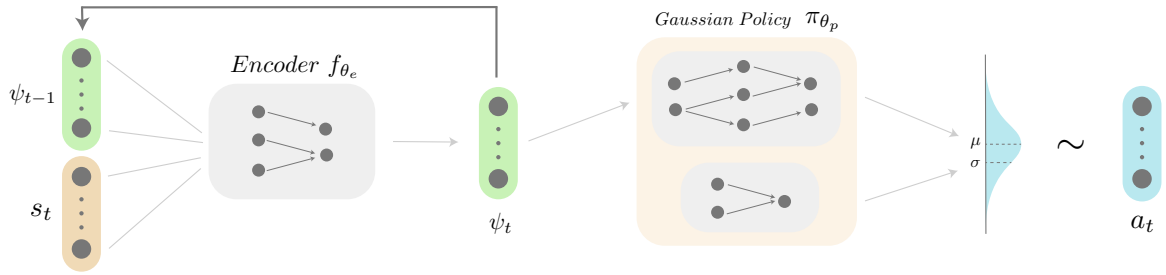


Figure 1. Illustration of the policy architecture learned by MEMENTO. The recursive nature is highlighted with a connection between the output of the history encoder and the input of the history encoder (at the next time step).

this paper, we propose a recurrent architecture (see Figure 1) that exploits function approximation to compress the history of interactions into a compact representation. Especially, we present a methodology (Section 3), MEmory-based Maximum ENTropy Optimization (MEMENTO), adapted from the MEPOL algorithm (Mutti et al., 2021), to simultaneously optimize the behavior and the history representation following the same MSE objective over the class of environments. Finally, we provide an illustrative experimental evaluation (Section 4) that shows how a unique Pareto dominant behavior over the class of environments can be learned through the proposed methodology.

2. Notation

An episodic *Controlled Markov Process* (CMP) is defined by a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, \mu, T)$, where $\mathcal{S}$ is a continuous state space, $\mathcal{A}$ is a continuous action space, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$[1] is the transition model, such that $P(s'|a, s)$ denotes the probability of reaching state $s' \in \mathcal{S}$ when taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, and $\mu : \Delta(\mathcal{S})$ is the initial state distribution, and T is the horizon of an episode.

A *policy* encodes the behavior of an agent interacting with a CMP. A non-Markovian policy $\pi \in \Pi_{\text{NM}}$ is defined through a function $\pi : \mathcal{H}_T \to \Delta(\mathcal{A})$ that maps any history $h \in \mathcal{H}_T$ to a probability distribution over actions, where $\mathcal{H}_T$ denotes the set of all the histories up to length T. A Markovian policy $\pi \in \Pi_{\text{M}}$ instead maps the last state of the history to a probability over actions $\pi : \mathcal{S} \to \Delta(\mathcal{A})$. Clearly, the set of non-Markovian policies is more general $\Pi_{\text{M}} \subset \Pi_{\text{NM}}$.

Running a policy π over a CMP induces a distribution over the states of the latter, which is is called the *marginal state distribution* and it is defined as $d^\pi(s) := \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{P}(s_t = s)$. The MSE objective for unsupervised pre-training asks the agent to find a policy maximizing the entropy of this distribution (Hazan et al., 2019; Mutti et al., 2021), i.e.,[2]

$$\max_{\pi \in \Pi} H(d^\pi) := -\mathop{\mathbb{E}}_{s \sim d^\pi}[\log d^\pi(s)]. \tag{1}$$

[1]The symbol $\Delta(\mathcal{X})$ denotes the simplex over the space $\mathcal{X}$.
[2]The set of policies Π denotes either Π_{M} or Π_{NM}.

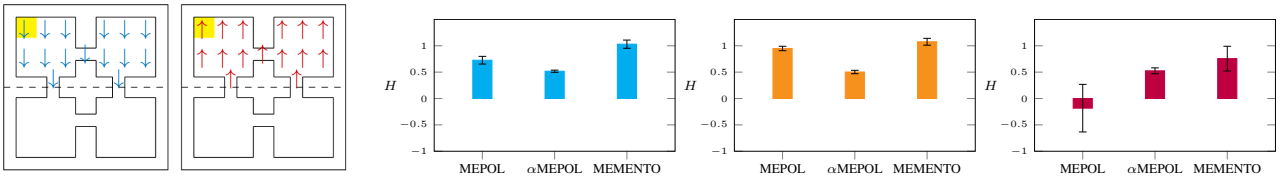
3. Architecture

Learning an optimal non-Markovian policy has been shown in (Mutti et al., 2022a) to be a computationally hard problem. To face this negative complexity result, we introduce an architecture that exploits function approximation in order to learn a compact representation of the history. To achieve this behaviour, we propose an architecture with parameters $\boldsymbol{\Theta} = (\boldsymbol{\theta_e}, \boldsymbol{\theta_p})$, and composed of two modules: a history encoder parametrized by $\boldsymbol{\theta_e}$, and a subsequent diagonal Gaussian policy parametrized by $\boldsymbol{\theta_p}$, as illustrated in Fig.1.

Both components are parameterized using neural networks. In this architecture, we represent the current history h_t through an embedding ψ_t that is learned in a recursive and online fashion, so that the learned representation is approximately optimal with respect to the entropy objective of the policy. At each time step t, the previous embedding ψ_{t-1} and the current state s_t are encoded into a new history embedding ψ_t according to the following recursive equation:

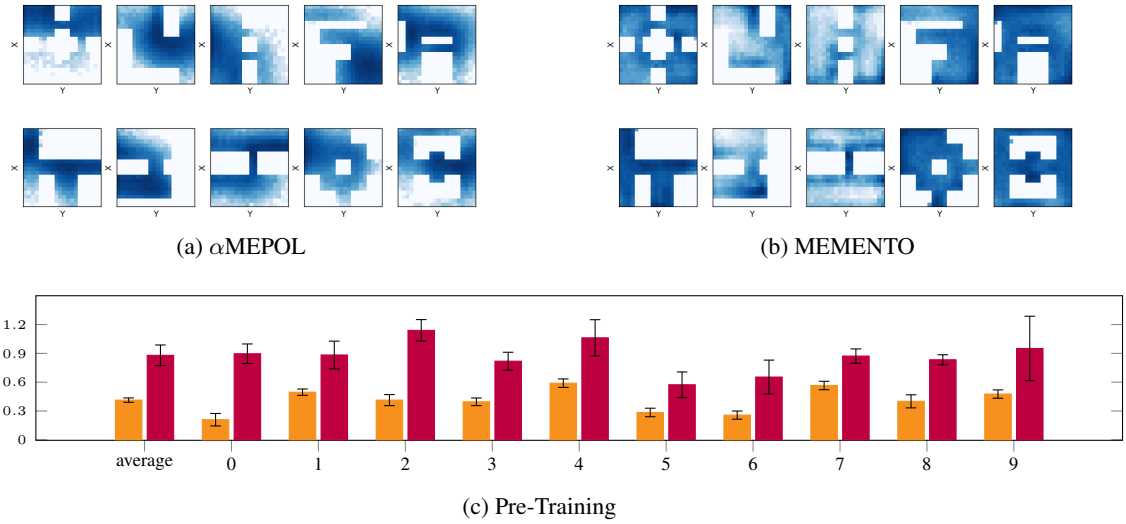
$$\begin{cases} \psi_0 = \mathbf{0} \\ \psi_t = f_{\boldsymbol{\theta_e}}(s_t, \psi_{t-1}). \end{cases}$$

where $f_{\boldsymbol{\theta_e}}$ is a multilayer perceptron parametrized by $\boldsymbol{\theta_e}$. The history embedding ψ_t, at each time step, is used by the consecutive Gaussian policy $\pi_{\boldsymbol{\theta_p}}$ instead of the full history. The Guassian policy is a function of the current embedding ψ_t and returns the parameters μ and σ of a Gaussian distribution from which the action a_t is sampled. While μ is given by a non-linear transformation of the history embedding ψ_t, σ is learned independently of the current embedding, as shown is Fig.1. We refer with MEmory-based Maximum ENTropy Optimization (MEMENTO), to the method that combines the architecture introduced with a model-free policy-gradient algorithm, namely MEPOL (Mutti et al., 2021). This algorithm collects a set of trajectories to compute a k-NN estimator (Singh et al., 2003) of the entropy of the state distribution induced by the policy. The policy gradient is computed to maximize this estimate, which is used as a proxy for the exploration effectiveness of the policy. Appendix A reports the pseudocode of the algorithm.



(a) GridWorld with Slope (b) $p_{\mathcal{M}} = [0.8, 0.2]$ (c) $p_{\mathcal{M}} = [1, 0]$ (d) $p_{\mathcal{M}} = [0, 1]$

Figure 2. Pre-training performance achieved by MEPOL, αMEPOL (α = 0.2), and MEMENTO in the GridWorld with Slope domain, which is depicted in (a). The polices are trained on the class distribution (b) and tested on (b, c, d). We provide 95% c.i. over 16 runs.



(a) αMEPOL (b) MEMENTO

(c) Pre-Training

Figure 3. (a, b) Heatmaps of the state visitations obtained by αMEPOL and MEMENTO for all the configurations of the MultiGrid domain. (c) Pre-training performance achieved by αMEPOL (left bar) and MEMENTO (right bar). We provide 95% c.i. over 16 runs.

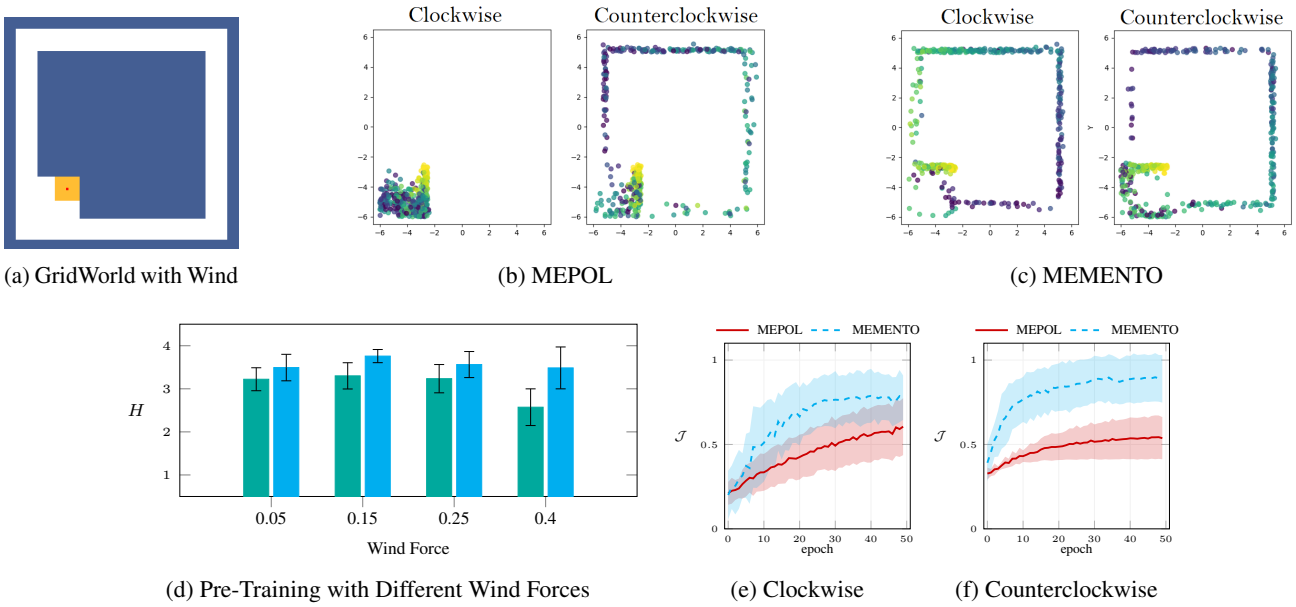
4. Experiments

We now provide an empirical comparison of the pre-training performance, computed with a KNN estimate (Singh et al., 2003) of Eq. (1), that can be achieved by a non-Markovian policy trained with MEMENTO in contrast to the Markovian policies trained with MEPOL (Mutti et al., 2021) and αMEPOL (Mutti et al., 2022b). Especially, we show that

4.1) The non-Markovian policy Pareto dominates the Markovian alternatives over a class of two illustrative continuous gridworld domains;
4.2) Similar considerations extends to larger classes of environments, such as a set of ten continuous gridworlds;
4.3) The non-Markovian policy can efficiently deal with the identification-exploitation dilemma.

4.1. The Pareto Dominance of Non-Markovian Policies

We consider a class $\mathcal{M}$ of two configurations of a continuous GridWorld domain with 2D states and actions, which was used in (Mutti et al., 2022b) as *GridWorld with Slope*. The two configurations share the state space, composed of four rooms connected by narrow hallways, and the action space, such that any action defines a (bounded) increment along the two coordinates representing the agent's movement. The difference between the two environments lies in the transition dynamics. As shown in Figure 2a, the environments have slopes of opposed direction in the upper rooms, which alter the agent's movement in different ways. The yellow area represents the initial position while the arrows represent the slope's direction. Similarly as in (Mutti et al., 2022b), we consider an unbalanced distribution $p_{\mathcal{M}} = [0.8, 0.2]$ over the class of environments. In Figure 2, we compare the performance of optimal policies obtained by running MEPOL (Mutti et al., 2021), αMEPOL (Mutti et al., 2022b) ($\alpha = 0.2$), and MEMENTO for 500 epochs. We show the advantage of MEMENTO in both the average performance and in single environments. We can notice that a policy learned with MEMENTO outperforms the one learned by αMEPOL in the gridworld with upward slope (red arrows), which is rarely visited. The introduction of non-Markovianity allows to reach better performance in the worst case using the average entropy objective and not a specific risk-averse objective like in αMEPOL.



(a) GridWorld with Wind (b) MEPOL (c) MEMENTO

(d) Pre-Training with Different Wind Forces (e) Clockwise (f) Counterclockwise

Figure 4. Experiments in the GridWorld with Wind domain, which is depicted in (a). (b, c) Point-maps of the state visitations induced by MEPOL and MEMENTO in the *Clockwise Wind* configuration (on the left) and *Counterclockwise Wind* configuration (on the right). (d) Pre-training performance achieved by MEPOL (left bars) and MEMENTO (right bars) for different wind forces (95% c.i. over 16 runs). (e, f) Supervised fine-tuning performance achieved by TRPO (Schulman et al., 2015) with corresponding initializations.

4.2. Scaling to Larger Classes of Environments

To provide evidence of the capacity to scale to larger classes of environments we employed the setting used in (Mutti et al., 2022b), which is called *MultiGrid*. We consider a class of ten different gridworld configurations sampled with equal probability. We compare the performance of optimal policies obtained by running for 500 epochs αMEPOL (Mutti et al., 2022b) ($\alpha = 0.1$) and MEMENTO.[3] In Figure 3c, we report the pre-training performance in all the configurations. We can see that also with an increased number of environments MEMENTO is able to obtain a superior performance overall. To further visualize the advantages of our architecture, we provide in Figure 3a and 3b the heatmaps of the state visitations obtained from 100 trajectories with the policies learned by αMEPOL and MEMENTO. We can notice how the non-Markovian policy reaches a better coverage of the state space.

4.3. The Identification-Exploitation Dilemma

To highlight the benefits of the identification of the environments in a class, we created another set of gridworlds. A graphical representation of the state space is presented in Figure 4a, while the action space is the same as in the previously seen environment. We consider two configurations sampled with equal probability, both with a wind that modifies the agent's actions except in the bottom-left room. One configuration has clockwise wind and the other one has counterclockwise wind. The hardness of this environment can be controlled by modifying the strength of the wind. Up to a certain value of wind, the agent can enter the corridor also with a headwind, allowing it to explore the corridor in one direction without a considerable loss in performance.

In Figure 4b and Figure 4c, we show a batch of trajectories generated using the policy learned by MEPOL and by MEMENTO. We can see that a Markovian policy, once the wrong entrance is picked, is not able to recover from the mistake, wandering around the same entrance. A non-Markovian policy can instead identify the environment from what happens at that entrance and in case of a mistake, it can correct itself and move towards the other entrance. In Figures 4e, 4f), we show results obtained by fine-tuning using TRPO (Schulman et al., 2015) the policies pre-trained with MEPOL and MEMENTO. We can see how the advantage in entropy reflects into a faster fine-tuning process, in the counterclockwise configuration in particular this difference is more evident, with MEMENTO reaching in less than 10 epochs the same return reached by MEPOL in 50 epochs. In particular, we repeated the training process with different wind forces and in Figure 4d we can see how a Markovian policy can perform quite well up to a certain wind strength, while a non-Markovian policy can identify the environment, adapt to it, and reach consistent entropy results.

[3] We avoid comparing with MEPOL in this experiment, as it is dominated by αMEPOL in this setting (Mutti et al., 2022b).

References

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

Hayes, C. F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L. M., Dazeley, R., Heintz, F., et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):1–59, 2022.

Hazan, E., Kakade, S., Singh, K., and Van Soest, A. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pp. 2681–2691. PMLR, 2019.

Laskin, M., Yarats, D., Liu, H., Lee, K., Zhan, A., Lu, K., Cang, C., Pinto, L., and Abbeel, P. Urlb: Unsupervised reinforcement learning benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Liu, H. and Abbeel, P. Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34, 2021.

Mutti, M. and Restelli, M. An intrinsically-motivated approach for learning highly exploring and fast mixing policies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Mutti, M., Pratissoli, L., and Restelli, M. Task-agnostic exploration via policy gradient of a non-parametric state entropy estimate. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Mutti, M., De Santi, R., and Restelli, M. The importance of non-markovianity in maximum state entropy exploration. *arXiv preprint arXiv:2202.03060*, 2022a.

Mutti, M., Mancassola, M., and Restelli, M. Unsupervised reinforcement learning in multiple environments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022b.

Parisi, S., Dean, V., Pathak, D., and Gupta, A. Interesting object, curious agent: Learning task-agnostic exploration. *Advances in Neural Information Processing Systems*, 34, 2021.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

Seo, Y., Chen, L., Shin, J., Lee, H., Abbeel, P., and Lee, K. State entropy maximization with random encoders for efficient exploration. In *International Conference on Machine Learning*, pp. 9443–9454. PMLR, 2021.

Singh, H., Misra, N., Hnizdo, V., Fedorowicz, A., and Demchuk, E. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23 (3-4):301–321, 2003.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pp. 11920–11931. PMLR, 2021.

A. Algorithm

In the following pseudocode off_{iters} stands for the number of off-policy iterations performed in each epoch.

Algorithm 1 MEMENTO

Input: Horizon T, number of batches N, batch-size B, trust-region threshold δ, learning rate α, nearest neighbors k, max off-policy iterations max_{iters}
initialize θ
for $epoch = 0, 1, 2, ...$ until convergence **do**
  draw N batches of B trajectories of length T
  $\theta^{'} \leftarrow \theta, off_{iters} \leftarrow 0$
  while $D_{KL}(\hat{d}_T(\theta)||\hat{d}_T(\theta^{'})) \leq \delta \;\; \& \;\; off_{iters} \leq max_{iters}$ **do**
    $\theta^{'} = \theta^{'} + \alpha \nabla_{\theta'} \hat{H}_k(\hat{d}_T(\theta^{'})|\hat{d}_T(\theta))$
    $off_{iters} \leftarrow off_{iters} + 1$
  end while
  $\theta \leftarrow \theta^{'}$
end for
Output: diagonal Gaussian policy $\pi_{\boldsymbol{\theta_p}}$, history encoder $f_{\boldsymbol{\theta_e}}$

B. Experimental details

Here we specify the experimental details of the methods run within the experimental section, namely MEMENTO, MEPOL, and αMEPOL.

B.1. MEMENTO

Parameters	**Slope**	**Corridor**	**MultiGrid**
Number of epochs	500	500	500
Samples collected for each	200	100	200
Horizon(T)	400	400	400
Batch Size	5	1	5
KL threshold (δ)	15	0.1	15
Learning rate	10^{-4}	$4 * 10^{-5}$	10^{-5}
Max off policy iterations	30	30	30
Number of neightbors (k)	30	30	30
recursive encoder layer sizes	(300,300)	(300,300)	(300,300)
history vector length	100	100	100
Policy hidden layers sizes	(300,300)	(300,300)	(300,300)
History encoder output act. function	ReLU	ReLU	ReLU
Policy and History encoder act. function	ReLU	ReLU	ReLU
Number of seeds	8	8	4

Table 1. Hyperparameter MEMENTO

B.2. MEPOL

Parameters	Slope	Corridor
Number of epochs	500	500
Samples collected for each	200	100
Horizon(T)	400	400
Batch Size	5	1
KL threshold (δ)	15	0.1
Learning rate	10^{-5}	$4 * 10^{-5}$
Max off policy iterations	30	30
Number of neightbors (k)	30	30
Policy hidden layers sizes	(300,300)	(300,300)
Policy hidden layers act. function	ReLU	ReLU
Number of seeds	8	8

Table 2. Hyperparameter MEPOL

B.3. αMEPOL

Parameters	Slope	MultiGrid
Number of epochs	500	500
Samples collected for each	200	200
Horizon(T)	400	400
Batch Size	5	5
KL threshold (δ)	15	15
Learning rate	10^{-5}	10^{-5}
Max off policy iterations	30	30
Number of neightbors (k)	30	30
α	0.35	0.1
Policy hidden layers sizes	(300,300)	(300,300)
Policy hidden layers act. function	ReLU	ReLU
Number of seeds	8	4

Table 3. Hyperparameter αMEPOL