MetaCURE: Meta Reinforcement Learning with Empowerment-Driven Exploration

Jin Zhang [*1] **Jianhao Wang** [*1] **Hao Hu** [1] **Tong Chen** [1] **Yingfeng Chen** [2] **Changjie Fan** [2] **Chongjie Zhang** [1]

A. Proofs for Section 3.1

Proof.

$$
\begin{aligned}
&\mathcal{J}^{\pi_e}(C_{:H}, \mathcal{K}) \\
&= I^{\pi_e}(C_{:H}; \mathcal{K}) \\
&= \mathbb{E}_{(c_{:H},\kappa)\sim(C_{:H},\mathcal{K})}\left[\log \frac{p^{\pi_e}(c_{:H}|\kappa)}{p^{\pi_e}(c_{:H})}\right] && (1)\\
&= \mathbb{E}_{(c_{:H},\kappa)\sim(C_{:H},\mathcal{K})}\left[\sum_{t=0}^{H-1} \log \frac{p^{\pi_e}(c_t|c_{:t},\kappa)}{p^{\pi_e}(c_t|c_{:t})}\right] && (2)\\
&= \mathbb{E}_{(c_{:H},\kappa)\sim(C_{:H},\mathcal{K})}\left[\sum_{t=0}^{H-1} \log \frac{p^{\pi_e}(a_t, r_t, s_{t+1}|c_{:t},\kappa)}{p^{\pi_e}(a_t, r_t, s_{t+1}|c_{:t})}\right] && (3)\\
&= \mathbb{E}_{(c_{:H},\kappa)\sim(C_{:H},\mathcal{K})}\left[\sum_{t=0}^{H-1} \log \frac{p^{\pi_e}(a_t|c_{:t},\kappa)p(r_t, s_{t+1}|\kappa, c_{:t}, a_t)}{p^{\pi_e}(a_t|c_{:t})p(r_t, s_{t+1}|c_{:t}, a_t)}\right] && (4)\\
&= \mathbb{E}_{(c_{:H},\kappa)\sim(C_{:H},\mathcal{K})}\left[\sum_{t=0}^{H-1} \log \frac{p(r_t, s_{t+1}|\kappa, s_t, a_t)}{p(r_t, s_{t+1}|c_{:t}, a_t)}\right]. && (5)
\end{aligned}
$$

Eq. (3) holds as $c_t = (s_t, a_t, r_t, s_{t+1})$ and s_t is contained in $c_{:t}$.

B. Additional Implementation Details

Encoder structure: the context encoder should be able to effectively extract task information from experience sequences. In practice, we design a context encoder using temporal convolution and soft attention, similar to SNAIL (Mishra et al., 2018).

Other details: MetaCURE is implemented with PyTorch (Paszke et al., 2019). Generally it takes about 12-40h to converge on the MuJoCo task sets, and can be further accelerated with parallel sampling. Hyper-parameters are selected by a simple grid search.

C. Environment and Hyper-parameter Settings

In this subsection, we provide detailed settings of reward functions and hyper-parameters, which are shown in Table 1 and 2. All tasks obtain sparse reward functions, providing zero rewards if the agent is outside the range of goals. The agent is additionally penalized with control costs.

[*]Equal contribution [1]Institute for Interdisciplinary Information Sciences, Tsinghua University, China [2]Fuxi AI Lab, NetEase, China. Correspondence to: Jin Zhang <jin-zhan20@mails.tsinghua.edu.cn>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

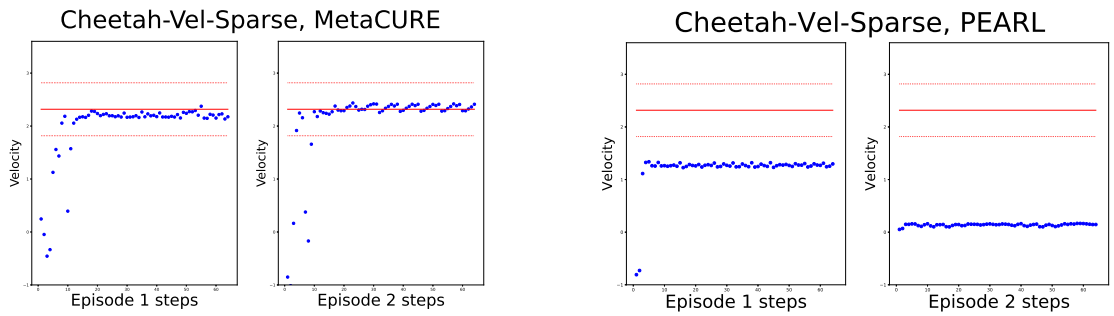


Figure 1. Visualization of MetaCURE and PEARL on Cheetah-Vel-Sparse. While MetaCURE efficiently explores possible goal velocities in the first episode and exploits in the second episode, PEARL fails to effectively explore.

In the Meta-World tasks, the agent gets non-zero rewards only if it "successes" in the task, which is given as a binary signal by the environment.

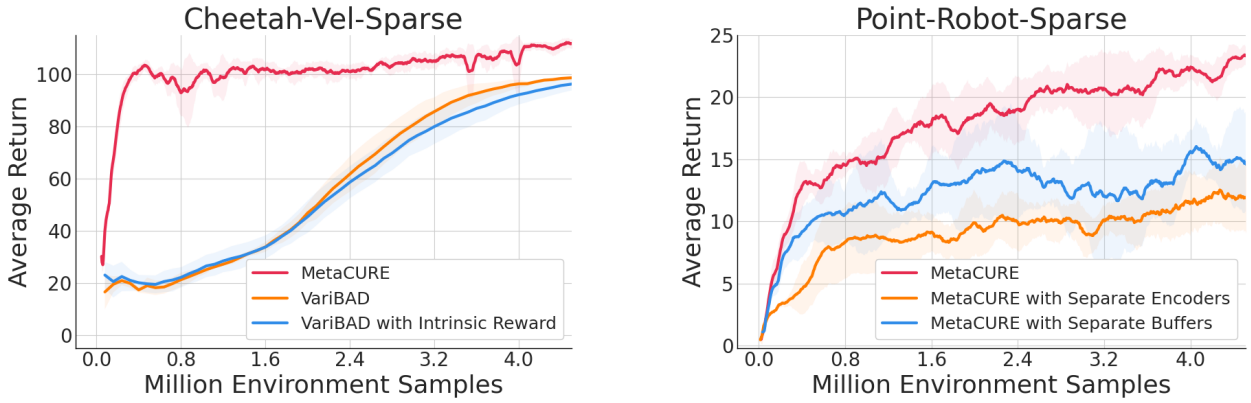
D. Visualizations

We show additional visualization results on Cheetah-Vel-Sparse, as shown in Figure 1.

E. Ablation Studies

E.1. Ablation Study: Hyper-parameters

We test MetaCURE with different hyper-parameters on Cheetah-Vel-Sparse. As shown in Table 3, MetaCURE is generally robust to the choice of hyper-parameters.



(a) (b)

Figure 2. (a) A VariBAD variant that uses our intrinsic reward does not improve in performance. VariBAD does not separate exploration and exploitation, and fails to achieve satisfactory performance with our intrinsic reward. (b) Ablation study on MetaCURE's knowledge and data sharing. Sharing the task inference component and the replay buffer greatly improves MetaCURE's learning efficiency.

E.2. Ablation Study: Baseline with Intrinsic Reward

We test a variant of VariBAD (Zintgraf et al., 2019) that uses $r_{exploration}$ for training its policy. As shown in Figure 2(a), directly combining our intrinsic reward with VariBAD does not lead to satisfactory performance, and still underperforms MetaCURE. This is because that VariBAD does not separate exploration and exploitation policies, and does not support learning intrinsically motivated exploration behaviors and unbiased exploitation behaviors together.

Table 1. Adaptation length and goal settings for environments used for evaluation. Goals are uniformly distributed in *Goal range* and non-zero informative reward is provided only when the distance between the agent's position/speed and the goal is smaller than *Goal radius*. As for Meta-World Reach and Meta-World Reach-Wall, the goal range and goal radius follow the settings in the original paper (Yu et al., 2020).

Environment	# of adaptation episodes	Max steps per episode	Goal type	Goal range	Goal radius
Cheetah-Vel-Sparse	2	64	Velocity	[0,3]	0.5
Walker-Vel-Sparse	2	64	Velocity	[0,2]	0.5
Reacher-Goal-Sparse	2	64	Position	Semicircle with radius 0.25	0.09
Point-Robot-Sparse	4	32	Position	Semicircle with radius 1	0.3
Walker-Rand-Params	4	64	Velocity	1.5	0.5
Hopper-Rand-Params	4	64	Velocity	1.5	0.5
Meta-World Reach	4	150	Position	/	/
Meta-World Reach-Wall	4	150	Position	/	/

Table 2. Hyperparameter settings for MetaCURE in different environments.

Environment	Latent size	β	λ	Batch size	Learning rate
Cheetah-Vel-Sparse	5	0.1	5	64	3e-4
Walker-Vel-Sparse	5	0.1	5	64	3e-4
Reacher-Goal-Sparse	5	1	1	64	3e-4
Point-Robot-Sparse	5	1	0.3	96	3e-4
Walker-Rand-Params	5	1	5	256	3e-4
Hopper-Rand-Params	5	1	5	256	3e-4
Meta-World Reach	5	1	0.3	512	1e-4
Meta-World Reach-Wall	5	1	0.3	512	1e-4

Table 3. MetaCURE's hyper-parameter ablation studies on the Cheetah-Vel-Sparse task set.

Learning rate	β	λ	Performance
3e-4	1e-1	5	112.1$\pm$3.4
3e-4	3e-1	5	109.1$\pm$5.7
3e-4	1e-1	2	110.9$\pm$6.4
3e-4	3e-1	2	108.2$\pm$4.2
1e-4	1e-1	5	111.5$\pm$4.4

E.3. Ablation Study: Knowledge and Data Sharing

To show the effect of sharing task encoder and buffer, we test two variants of MetaCURE on Point-Robot-Sparse: one uses separate encoders for the exploration and exploitation policy, and the other one uses separate buffers for training the policies, as shown in Figure 2(b). Both variants suffer from a great decrease in learning efficiency, as data and knowledge are not utilized effectively.

References

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pp. 1094–1100. PMLR, 2020.

Zintgraf, L., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. In *International Conference on Learning Representations*, 2019.