Learning Bayes-Optimal Representation in Partially Observable Environments via Meta-Reinforcement Learning with Predictive Coding

Po-Chen Kuo
University of Washington
Allen Institute for Neural Dynamics
pckuo@uw.edu

Han Hou
Allen Institute for Neural Dynamics
han.hou@alleninstitute.org

Will Dabney
Google Deepmind
wdabney@deepmind.com

Edgar Y. Walker
University of Washington
eywalker@uw.edu

Abstract

Learning a compact representation summarizing history is essential for decision-making, planning, and generalization in partially observable environments. Memory-based meta-reinforcement learning (RL) has been shown to learn near Bayes-optimal policy under partial observability. However, its learned representations can fail to achieve equivalence to minimally-sufficient, Bayes-optimal belief states, potentially hindering its robustness and generalization. To overcome this challenge, we propose a meta-RL framework for learning an explicit belief representation by incorporating self-supervised predictive modules inspired by predictive coding from neuroscience literature. Our approach outperforms conventional meta-RL by generating more interpretable and task-relevant representations, which better capture the underlying task structure and dynamics. Using state machine simulation, we demonstrate the learned representations are more equivalent to Bayes-optimal states and linked to improved future prediction and policy learning. Our results suggest that self-supervised future prediction is a promising technique for enhancing representation learning in partially observable environments.

1 Introduction

In real-world environments, agents — whether biological or artificial — rarely have complete information about the environmental states crucial for decision-making and planning, as observations are often noisy, non-stationary, and stochastic. This issue is known in reinforcement learning (RL) as partial observability [1], and is one of the major challenges in deploying real-world RL systems [2]. Under partial observability, learning the optimal policy depends on the entire sequence of past observations and actions, called history. An open question in RL under partial observability is how to learn a representation that serves as a compact summary of the history while being as effective as the actual history for future prediction and policy learning.

Partially observable tasks can typically be formalized as partially observable Markov Decision Processes (POMDPs, Fig. 1A) [3], which allow a Bayesian treatment by explicitly maintaining a belief state (the posterior over hidden states), and updating the belief using Bayesian inference [4]. It has been shown that meta-RL, in particular recent advances in memory-based meta-RL, provides powerful and scalable deep learning methods for developing agents that can efficiently learn and adapt under uncertainty [5–10]. Often parametrized with recurrent neural networks (RNNs), memory-based

1^{st} Workshop on NeuroAI @ 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

meta-RL acquires inductive biases from data through training to maximize return on a distribution over tasks. This approach has been shown to derive agents that behave near Bayes-optimally under partial observability, both theoretically and empirically [11, 12]. However, its learned representations are not equivalent to the minimally-sufficient, Bayes-optimal belief states [12], potentially hindering its robustness, generalization capability, and learning of temporally-extended exploration [5, 12].

On the other hand, humans and animals can learn complex predictive models to guide decision-making under uncertainty and rapid adaptation in novel environments [13, 14]. Experimental and theoretical neuroscience studies postulate that the brain generates predictions about incoming observations and utilizes prediction errors to update its internal models [15]. This hypothesized neural mechanism, formalized as predictive coding [16, 17], has been instrumental in understanding diverse cognitive processes, including feature learning in sensory areas [18, 19], motor control in the cerebellum [20], cognitive maps in the hippocampus [21], and value learning in the striatum [22]. Furthermore, predictive objectives have been shown in machine learning to be key auxiliary objectives for representation learning, preventing representation collapse in neural networks [23–25].

Motivated by the modular predictive coding neural circuits and extending on previous works using auxiliary training objectives for meta-RL [9, 10], in this study, we propose an end-to-end memory-based meta-RL framework with self-supervised future predictive modules to learn explicit belief representations in partially observable environments. We focus on evaluating the learned representations and demonstrate that the proposed meta-RL agents with self-supervised predictive modules can learn representations with higher equivalence to Bayes-optimal states compared to conventional meta-RL agents. Specifically, (i) we show the proposed method can learn an interpretable, low-dimensional representation of the history that captures relevant task structures and dynamics, which facilitates policy learning and planning, even for tasks requiring exploration and information seeking; (ii) using state machine simulation [12], we quantitatively demonstrate the proposed method can learn representations more equivalent to Bayes-optimal states than conventional meta-RL methods. Overall, we show that meta-RL with self-supervised predictive modules provides a promising approach for representation learning in partially observable environments.

2 Related work

Previous works have proposed memory-based meta-RL and examined their Bayes-optimality. Black box meta-RL such as RL^2 learns a recurrent policy through hidden states in RNNs conditioned on the entire history (Fig. 1B) [6, 7]. In principle, Bayes-optimal policies can be learned by encoding the belief state in RNN hidden states. These belief representations emerge implicitly as a byproduct of training deep meta-RL policies [5]. Several studies have demonstrated that belief states can be decoded from RNN hidden states [10, 26, 27], and the learned policy converges to the Bayes-optimal one theoretically and empirically in bandit problems [11, 12]. However, comparison of meta-learned representations in RL^2 using state machine simulation suggests their representations are not equivalent to minimally-sufficient Bayes-optimal states, likely due to failures in injectivity and potentially hindering its robustness and generalization [12].

In addition to black-box methods, several works have explored auxillary training objectives and alternative architectures for meta-RL. Zintgraf et al. [9] proposes VariBAD, where a variational autoencoder is used to learn a posterior over task embeddings, providing effective meta-learning models for task inference under partial observability. Akuzawa et al. [10] employs hierarchical state-space models to learn task embedding and belief states separately. Wang et al. [28] assumes full-observability during training and propose decoupled representation learning and belief modeling using a separate random policy to generate training samples. Our proposed approach draws inspiration from this line of studies using auxillary objectives for representation learning.

3 Meta-RL with self-supervised predictive modules for representation learning

To efficiently learn optimal policies in partially observable environments, the critical challenge lies in learning a good representation of belief states from history (Fig. 1A). Black-box meta-RL methods like RL^2 (Fig. 1B) require simultaneously learning state representation and decision-making using only reward signals, typically making the agents struggle to learn good representations

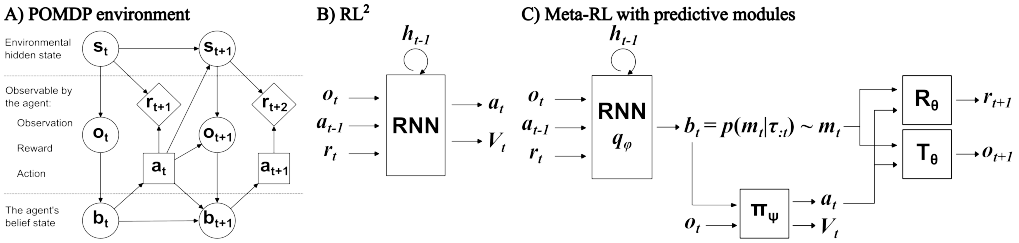

Figure 1: Representation learning under partial observability using meta-RL with predictive modules. A) In POMDPs, a belief state over the environmental hidden state is maintained and used for policy learning. B) Memory-based meta-RL (e.g. RL^2) simultaneously learns representation and policy using only the reward signal. C) The proposed meta-RL with self-supervised predictive modules separates representation learning with predictive modeling and policy learning with reward signal.

and subsequently hindering them from effectively learning optimal policies [5, 12]. To overcome this challenge, we propose an end-to-end memory-based meta-RL framework with self-supervised predictive modules to learn an explicit belief representation. Our motivations are two-fold: (i) neurobiological motivation under predictive coding hypothesis where modular neural circuits perform distinct sensory and reward predictions, and (ii) theoretical motivation that a good representation should summarize the relevant history predictive of the future and reflect uncertainty about the belief.

The proposed framework consists of a variational autoencoder for predictive representation learning and a policy network for decision-making based on the learned representation (Fig. 1C). The predictive modules begin with an RNN encoder q_ϕ, which takes as input the current observation o_t, the current reward r_t, and the previous action a_{t-1}. Unlike RL^2, which outputs policy and value function directly (Fig. 1B), here the RNN outputs a belief state b_t (the posterior) over the latent states m_t conditioned on the history $\tau_{0:t}$. The posterior is trained via reward and observation prediction akin to predictive coding in neuroscience using the reward decoder R_θ and the observation decoder T_θ, respectively, to predict upcoming rewards and observations given the action taken by the policy network. The predictive modules are optimized by maximizing the evidence lower bound (ELBO, see A.1) using the reparametrization trick [29]. This approach learns an explicit probabilistic belief representation over the latent state, which summarizes the history and is predictive of future outcomes. The policy network π_ψ is implemented as a feedforward neural network that receives the belief state b_t as inputs. Since policy learning is decoupled from representation learning, the policy network can be efficiently trained using model-free policy gradient algorithms [30] (see A.3 for details).

The whole model is trained in a self-supervised, end-to-end fashion using standard meta-learning approach. Our framework does not rely on privileged information during training or a separate random policy to generate samples for training the predictive models (e.g., [28]). Note our parameterization is similar in principle to previous works [9, 10, 31], with modifications tailored for POMDPs without assuming stationarity or structures of the hidden states.

4 Experiment

We design the following experiments to answer the question: whether meta-RL with self-supervised predictive modules learns representations with higher equivalence to Bayes-optimal states than conventional meta-RL? Specifically, we adopt the *state machine simulation* analysis used in Mikulik et al. [12] to examine the equivalence of representation and computation between meta-RL agents and Bayes-optimal solution. In essence, two state machines can be considered computationally equivalent if they can *simulate* each other — that is, if for any given states in one machine we can find a mapping onto the other machine such that both their *state transitions* and *outputs* are the same [32]. This analysis examines whether there exists a consistent way of interpreting every state in one machine as a state in the other. While decoding is commonly used to evaluate representation similarity [10, 26, 27], it considers only correlation whereas state machine simulation allows thorough assessment of representation equivalence based on structural and computational relevance.

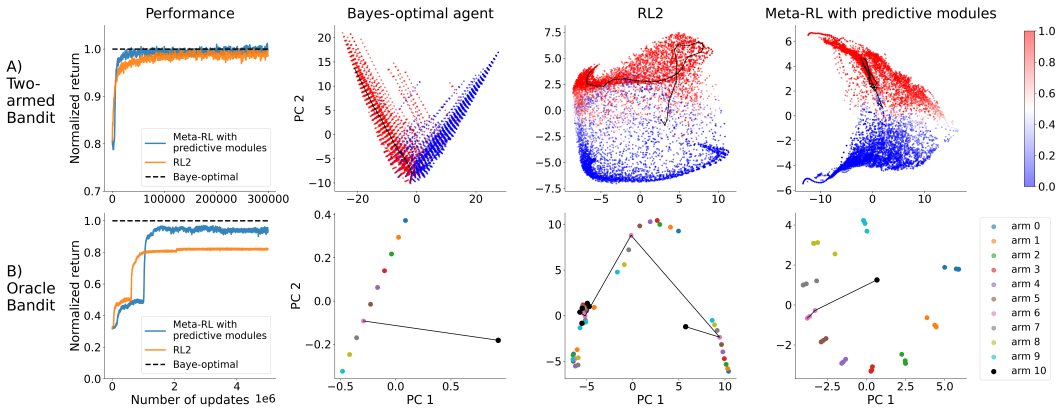

Figure 2: Meta-RL with predictive modules learns interpretable, task-relevant representations, facilitating effective policy learning. A) For the two-armed bandit task, return (left 1, normalized by expected return of the Bayes-optimal solution) of RL^2 (orange) and meta-RL with predictive modules (blue) across training. Bayes-optimal states (left 2), learned representation of RL^2 (left 3), and that of meta-RL with predictive modules (right). Each point is one state in the system, with color indicating the probability of choosing a_2. Black curves show one example trajectory. B) For the oracle bandit task, similar to A, with state space coloring indicating the most likely arm to choose in that state. Note meta-RL with predictive modules learns a representation capturing task structures and dynamics.

State machine simulation In tasks where Bayes-optimal states are analytically tractable, we can compare meta-RL representations with Bayes-optimal belief states using state machine simulation. The goal is to measure the state and output dissimilarities after using the best possible mapping function. If both state and output dissimilarities are low in both mapping directions, then the two representations can be considered equivalent. Briefly, the procedures are as follows (and see A.4 for details): (i) two mapping functions (parametrized as multi-layered perceptrons) are trained to map from meta-learned states to Bayes-optimal states and from Bayes-optimal states to meta-learned states; (ii) The *state dissimilarity* D_s is measured as the mean square error (MSE) of the mapped states against the target states; (iii) The *output dissimilarity* D_o is measured as the difference in return as generated by the output of the original states and the output of the mapped states. Here we compare D_s and D_o of the proposed meta-RL with predictive modules against those of conventional meta-RL (RL^2) to evaluate the quality of their learned representations.

Tasks We consider two exemplar tasks for evaluating representation learning under partial observability. The first is the multi-armed bandit task also considered in Mikulik et al. [12], where the meta-learned representations in RL^2 models are shown not to be equivalent to minimally-sufficient Bayes-optimal states. We will first examine whether our proposed meta-RL with predictive modules can learn Bayes-optimal states in a two-armed bandit task (see A.2).

Black box meta-RL like RL^2 often struggles to learn when exploration and information seeking are required [5]. We hypothesize this is due to ineffective representation learning. To exemplify this point, we consider an *oracle bandit* as the second task: in an 11-arm bandit environment, one of the first ten arms a_{1-10} is the target arm that gives a payout of 5, whereas the rest nine are non-target arms that each gives a payout of 1. The last arm a_{11} is the "oracle" arm whose payout is ≤ 1 and informs the target arm in the form of $\frac{1}{10}$ of the target arm index (e.g. a reward of 0.3 from a_{11} indicates a_3 is the target arm). This is similar to Wang et al. [7] but differs in that here no structured feedback is given, making learning a good representation more challenging and critical (see A.2). A successful policy requires paying an immediate exploration cost to acquire information for long-term gain.

5 Results

5.1 Two-armed bandit

For multi-armed bandits the Bayes-optimal solution can be derived with the Gittins index method [33]. In a two-armed bandit task, we find that while both RL^2 and meta-RL with predictive modules

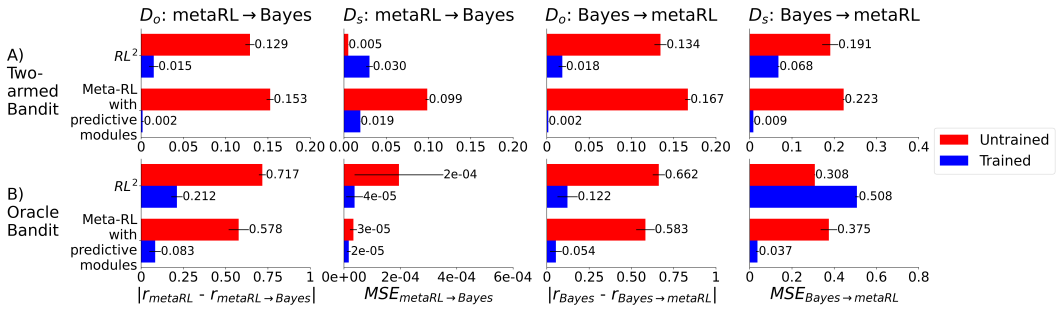

Figure 3: State machine simulation shows meta-RL with predictive modules learns representations with higher equivalence to Bayes-optimal states. A) For the two-armed bandit task, meta-RL with predictive modules attains much lower state transition dissimilarity D_s and output dissimilarity D_o than RL^2 in both mapping directions, indicating higher equivalence to Bayes-optimal state. (Error bar: standard error of the mean across different models.) B) Similar results for the oracle bandit task.

ultimately approach the performance of Bayes-optimal policy (Fig. 2A), meta-RL with predictive modules converges faster. Visualization of states shows meta-RL with predictive modules learns a low-dimensional representation that is structurally more similar to the Bayes-optimal states than RL^2 (Fig. 2A). The qualitative observation is corroborated by the state machine simulation results in Fig. 3A (and Fig. 4 in Appendix). Before training (red), both RL^2 and meta-RL with predictive modules have high state and output dissimilarities, indicating the untrained networks are far from Bayes-optimal (except for D_s of $RL^2 \to$Bayes, which further implies that using decoding alone is not enough for comparing representation, see A.4 for discussion). After training (blue), meta-RL with predictive modules achieves much lower D_s and D_o in both mapping directions, showing that its learned representations are more equivalent to the Bayes-optimal states. In contrast, some dissimilarity measures remain high for RL^2 after training, indicating the learned representations of RL^2 are not equivalent to Bayes-optimal states, similar to findings in Mikulik et al. [12].

5.2 Oracle bandit

Knowledge of the task structure can be used to derive the Bayes-optimal state and policy for the oracle bandit task (see A.2). After meta-training, meta-RL with predictive modules can learn the Bayes-optimal policy, paying an immediate cost to sample the oracle arm and utilize the information for long-term gain (Fig. 2B). In contrast, RL^2 converges to a suboptimal policy where it learns to sample the oracle arm upfront, but the use of such information is less consistent (Fig. 2B). Visualization of representation shows that meta-RL with predictive modules learns an interpretable, low-dimensional representation capturing task structures and dynamics, whereas RL^2 fails to learn an interpretable representation (Fig. 2B), which explains its suboptimal behavior. Further, state machine simulation in Fig. 3B (and Fig. 5 in Appendix) shows that after training (blue), meta-RL with predictive modules attains much lower state transition and output dissimilarities in both mapping directions, demonstrating its learned representation are more equivalent to the Bayes-optimal states. In contrast, dissimilarity measures remain high for RL^2 after training, indicating that RL^2 fails to learn an effective representation when the task requires exploration and information seeking.

6 Discussion and conclusions

In this study we showed meta-RL augmented with neurally-inspired self-supervised predictive coding modules can effectively learn good representations in partially observable environments, facilitating policy learning. We find that in bandit tasks where black-box meta-RL method was previously shown incapable of learning Bayes-optimal states, the proposed meta-RL with predictive modules can learn representation more equivalent to Bayes-optimal states as measured with state machine simulation. In the oracle bandit task where exploration and information seeking are required, we show that our approach can learn an interpretable, low-dimensional representation that successfully captures relevant task structure and dynamics. In contrast, Black-box meta-RL fails to learn a compact and informative representation, which in turn hinders learning the optimal policy.

Scope and limitations In this work we only considered tasks with known/ derivable Bayes-optimal solutions to aid evaluation. Additional work is needed to validate the results in tasks with larger observation and action spaces and involving non-stationarity. In addition, we have only considered next-step future prediction. One future direction would be to incorporate different time scales with temporal abstractions (e.g. options). Finally, representation is important for generalization. We have not considered how representations may affect the ability for out-of-distribution (OOD) generalization, or how to learn representations that facilitates OOD generalization.

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[2] Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

[3] Anthony R Cassandra, Leslie Pack Kaelbling, and Michael L Littman. Acting optimally in partially observable stochastic domains. In *Aaai*, volume 94, pages 1023–1028, 1994.

[4] Jayakumar Subramanian, Amit Sinha, Raihan Seraj, and Aditya Mahajan. Approximate information state for approximate planning and reinforcement learning in partially observed systems. *Journal of Machine Learning Research*, 23(12):1–83, 2022.

[5] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

[6] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[7] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[8] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for pomdps. In *International conference on machine learning*, pages 2117–2126. PMLR, 2018.

[9] Luisa Zintgraf, Sebastian Schulze, Cong Lu, Leo Feng, Maximilian Igl, Kyriacos Shiarlis, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: Variational bayes-adaptive deep rl via meta-learning. *Journal of Machine Learning Research*, 22(289):1–39, 2021.

[10] Kei Akuzawa, Yusuke Iwasawa, and Yutaka Matsuo. Estimating disentangled belief about hidden state and hidden task for meta-reinforcement learning. *PMLR*, pages 73–86, 2021.

[11] Pedro A Ortega, Jane X Wang, Mark Rowland, Tim Genewein, Zeb Kurth-Nelson, Razvan Pascanu, Nicolas Heess, Joel Veness, Alex Pritzel, Pablo Sprechmann, et al. Meta-learning of sequential strategies. *arXiv preprint arXiv:1905.03030*, 2019.

[12] Vladimir Mikulik, Grégoire Delétang, Tom McGrath, Tim Genewein, Miljan Martic, Shane Legg, and Pedro Ortega. Meta-trained agents implement bayes-optimal agents. *Advances in neural information processing systems*, 33:18691–18703, 2020.

[13] Andy Clark. Whatever next? predictive brains, situated agents, and the future of cognitive science. *Behavioral and brain sciences*, 36(3):181–204, 2013.

[14] Georg B Keller and Thomas D Mrsic-Flogel. Predictive processing: a canonical cortical computation. *Neuron*, 100(2):424–435, 2018.

[15] Wolfram Schultz and Anthony Dickinson. Neuronal coding of prediction errors. *Annual review of neuroscience*, 23(1):473–500, 2000.

[16] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87, 1999.

[17] Karl Friston. A theory of cortical responses. *Philosophical transactions of the Royal Society B: Biological sciences*, 360(1456):815–836, 2005.

[18] Shohei Furutachi, Alexis D Franklin, Andreea M Aldea, Thomas D Mrsic-Flogel, and Sonja B Hofer. Cooperative thalamocortical circuit mechanism for sensory prediction errors. *Nature*, pages 1–9, 2024.

[19] Chengxu Zhuang, Siming Yan, Aran Nayebi, Martin Schrimpf, Michael C Frank, James J DiCarlo, and Daniel LK Yamins. Unsupervised neural network models of the ventral visual stream. *Proceedings of the National Academy of Sciences*, 118(3):e2014196118, 2021.

[20] Daniel M Wolpert, R Chris Miall, and Mitsuo Kawato. Internal models in the cerebellum. *Trends in cognitive sciences*, 2(9):338–347, 1998.

[21] J O'Keefe. The hippocampus as a cognitive map, 1978.

[22] Wolfram Schultz, Peter Dayan, and P Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

[23] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

[24] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[25] Ching Fang and Kimberly L Stachenfeld. Predictive auxiliary objectives in deep rl mimic learning in the brain. *arXiv preprint arXiv:2310.06089*, 2023.

[26] Gaspard Lambrechts, Adrien Bolland, and Damien Ernst. Recurrent networks, hidden states and beliefs in partially observable environments. *arXiv preprint arXiv:2208.03520*, 2022.

[27] Jay A Hennig, Sandra A Romero Pinto, Takahiro Yamaguchi, Scott W Linderman, Naoshige Uchida, and Samuel J Gershman. Emergence of belief-like representations through reinforcement learning. *PLOS Computational Biology*, 19(9):e1011067, 2023.

[28] Andrew Wang, Andrew C Li, Toryn Q Klassen, Rodrigo Toro Icarte, and Sheila A McIlraith. Learning belief representations for partially observable deep rl. In *International Conference on Machine Learning*, pages 35970–35988. PMLR, 2023.

[29] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[30] Volodymyr Mnih. Asynchronous methods for deep reinforcement learning. *arXiv preprint arXiv:1602.01783*, 2016.

[31] Dongqi Han, Kenji Doya, and Jun Tani. Variational recurrent models for solving partially observable control tasks. *arXiv preprint arXiv:1912.10703*, 2019.

[32] Antoine Girard and George J Pappas. Approximate bisimulations for nonlinear dynamical systems. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 684–689. IEEE, 2005.

[33] John C Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 41(2):148–164, 1979.

[34] Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail`, 2018.

A Appendix / supplemental material

A.1 Evidence lower bound

At a given time step t, the learning objective of the RNN encoder q_ϕ is to maximize

$$\mathbb{E}_{\rho(\tau_{0:t})}[\log p_\theta(o_{t+1}, r_{t+1}|a_{0:t})] \tag{1}$$

where $\rho(\tau_{0:t})$ is the distribution of trajectories incurred by the policy π_ψ. Eq. 1 is intractable, and we can instead optimize a tractable evidence lower bound (ELBO), computed as:

$$\begin{aligned}\mathcal{L} = \mathbb{E}_{\rho(\tau_{0:t})}\Bigg[\mathbb{E}_{\prod_{t=0}^{T-1} q_\phi(m_t|\tau_{0:t})} \sum_{t=0}^{T-1} \Big\{ &[\log p_\theta(o_{t+1}|m_t, a_t) + \log p_\theta(r_{t+1}|m_t, a_t)] \\ &- D_{KL}[q_\phi(m_t|\tau_{0:t})||p_\theta(m_t)]\Big\}\Bigg]\end{aligned} \tag{2}$$

The terms $\mathbb{E}_q[\log p_\theta(o_{t+1}|m_t, a_t) + \log p_\theta(r_{t+1}|m_t, a_t)]$ are the (predictive) reconstruction loss, and the term $D_{KL}[q_\phi(m_t|\tau_{0:t})||p_\theta(m_t)]$ is the Kullback-Leibler (KL) divergence between the variational posterior q_ϕ and the prior over the latent variables $p_\theta(m_t)$. To approximate Bayesian filtering for the belief update, the prior is set to the previous posterior $q_\phi(m_{t-1}|\tau_{0:t-1})$ with the initial prior $q_\phi(m_0|\tau_0) = \mathcal{N}(0, I)$. To optimize the ELBO (Eq. 2), the expectation is approximated with Monte Carlo smapling as in standard variational autoencoder (VAE) [29].

A.2 Task details

Two-armed bandit task We consider a Beta-Bernoulli two-armed bandit task with an episode length of 40 time steps. At the beginning of each episode, θ_1 and θ_2, the reward probability baises for the two arms, are independently drawn from a fixed Beta distribution, $p(\theta) = Beta(1, 1)$. θ_1 and θ_2 are then used to define the Bernoulli reward distributions for arm $a \in \{1, 2\}$, respectively. The reward biases θ_a are hidden from the agent. At each time step, an agent choose an arm $a \sim \pi$ sampled from its policy, and receives a binary reward sampled from $Ber(\theta_a)$, i.e. $r_t \sim p(r|\theta_a) = Ber(\theta_a)$. The discounted cumulative return is computed with a discount factor $\gamma = 0.95$. For multi-armed bandit tasks, Bayes-optimal policy can be derived using the Gittins index method [33], and the minimally-sufficient Bayes-optimal state is to keep track of the count of total pulls and the count of rewarded pulls for each arm. Therefore for the two-armed bandit task, the Bayes-optimal states are 4-dimensional: $(n_{a_1}, n_{r_{a1}}, n_{a_2}, n_{r_{a2}})$, where n_a and n_{r_a} denote the count of total pulls and the count of rewarded pulls for arm a, respectively.

Oracle bandit task The oracle bandit task is designed to exemplify an environment where a successful policy requires paying an immediate exploration cost to acquire information to improve long-term return. In an 11-arm bandit environment with an episode length of 6 time steps, one of the first ten arms a_{1-10} is selected uniformly randomly as the target arm a^*, which will give a payout of 5 upon choosing. The other nine arms out of a_{1-10} are non-target arms, each of which will give a payout of 1 upon choosing. The last arm, a_{11}, is the oracle arm whose payout informs the index of the target arm in a form of $1/10$ of the target arm a^*, i.e. $r(a_{11}) = 0.1 * a^*$ (e.g. a reward of 0.3 from a_{11} indicates that a_3 is the target arm). The discounted cumulative return is computed with a discount factor $\gamma = 0.95$. Note this setting is similar to Wang et al. [7] but differs in that in their set up this oracle information was provided using a one-hot encoding format, but in our formulation no other feedback than the reward itself is given, which requires learning a good representation of the history. With knowledge of the task structure, the Bayes-optimal policy is to always pull the oracle arm at the beginning, and continue pulling the target arm as informed by the reward information from the oracle till the end of the episode. Under the Bayes-optimal policy, the minimally sufficient statistic is to keep track of $(p_{a_{11}}, r_{a_{11}})$, where $p_{a_{11}} \in \{0, 1\}$ is binary denoting whether the oracle arm has already been pulled, and $r_{a_{11}}$ denotes the reward amount from the oracle if pulled previously.

A.3 Agent details

RL^2 Our implementation of RL^2 for black-box memory-based meta-RL follows previous literature [6, 7]. Here we use RNNs with 256 hidden units and hyperbolic tangent activation functions. As shown in Fig. 1B, the input includes the current observation o_t, the previous action in one-hot format a_{t-1}, and the associated reward in scalar format r_t. Output of the network is composed of a scalar value baseline V_t and a vector of logits for each action a_t. Actions are then sampled from a softmax distribution defined by the logits. The network is trained end-to-end with the Advantage Actor Critic algorithm [30] (Parts of the implementation code are based on [34], under MIT license). The gradient of the objective function is given by:

$$\begin{aligned}\nabla\mathcal{L}_{A2C} &= \nabla\mathcal{L}_{\pi} + \nabla\mathcal{L}_{V} + \nabla\mathcal{L}_{entropy} \\ &= \frac{\partial \log \pi(a_t|\tau_{:t};\psi)}{\partial\psi}\delta_t(\tau_{:t};\psi_V) + \beta_V\delta_t(\tau_{:t};\psi_V)\frac{\partial V}{\partial\psi_V} + \beta_e\frac{\partial H(\pi(a_t|\tau_{:t};\psi))}{\partial\psi}\end{aligned} \tag{3}$$

where

$$\begin{aligned}\delta_t(\tau_{:t};\psi_V) &= R_t - V(\tau_{:t};\psi_V) \\ R_t &= \sum_{i=0}^{k-1}\gamma^i r_{t+1} + \gamma^k V(\tau_{:t+k};\psi_V)\end{aligned} \tag{4}$$

defines the n-step temporal difference error advantage function δ_t, the discounted n-step bootstrapped return R_t with discount factor γ, k the number of remaining time steps in the current episode, and V the value function parametrized by ψ_V. The neural network policy is denoted as π and parametrized by ψ, and H_π is the entropy of the policy. Finally, β_V and β_e are hyperparemeters for controlling the relative weighting of value estimation loss and entropy regularization. For the two-armed bandit task we use $\beta_V \in \{0.01, 0.05\}$ and $\beta_e \in \{0.03, 0.05\}$, and for the oracle bandit task we use $\beta_V \in \{0.01, 0.05\}$ and $\beta_e \in \{0.3, 0.5\}$. The parameters are trained via backpropagation through time using the Adam Optimizer with a learning rate of 5e-5.

Meta-RL with predictive modules The self-supervised predictive modules are formulated as a VAE. For encoder q_ϕ we use RNNs with 256 hidden units and hyperbolic tangent activation functions. Latent dimension m_t is set to be 4 for the two-armed bandit task and 8 for the oracle bandit task. Therefore the output dimension of the encoder RNN is twice the above latent dimension for estimating both the mean and the variance of the latents, as standard in VAE. The decoders R_θ and T_θ are multi-layered perceptrons (MLPs) with one hidden layer of 32 units and ReLU activation functions. As shown in Fig. 1C, the encoder-decoder framework takes as input the current observation o_t, the previous action in one-hot format a_{t-1}, and the associated scalar reward r_t, and is set up to make prediction of the upcoming observations o_{t+1} and rewards r_{t+1}, conditioned on the trajectory as incurred by the policy network π_ψ described below. The entire VAE is trained to maximize the ELBO (Eq. 2) as derived in A.1. A coefficient of 0.01 is used for the relative contribution of the KL-term for the VAE training objective. We use the Adam Optimizer with a learning rate of 7e-5 to train the VAE using backpropagation through time.

The policy network π_ψ is parametrized as a MLP with one hidden layer of 32 units and hyperbolic tangent activation functions. Similar to the previous paragraph on RL^2, the policy network is trained with the Advantage Actor Critic algorithm to optimize the same loss function as described in Eq. 3. We use the same choice of β_V and β_e as before. An Adam Optimizer with a learning rate of 5e-5 is used to optimize the policy network. Note although the policy loss depends on the parameters of the encoder q_ϕ, we do not backpropagate the policy loss gradient through the encoder as the goal of the encoder is to learn a belief over the latent states such that the belief alone should be a sufficient representation for policy learning (similar to Zintgraf et al. [9]). Parts of the implementation code are based on [9] (under MIT license).

Model training The above meta-RL models (RL^2 and ours, meta-RL with predictive modules) are trained on internal GPU clusters (NVIDIA GeForce RTX 4090), which takes about 1G GPU memory and about 10-12 hours for training per model.

A.4 State machine simulation

Following the procedures of state machine simulation introduced in Mikulik et al. [12], we consider whether a meta-RL system (RL^2 or our proposed approach, meta-RL with predictive modules) can both *simulate* and *be simulated by*, a Bayes-optimal agent for a given task.

To evaluate how well a state machine M *simulates* another machine N, a function ϕ is first learned to map the states S_N in N to into the state space S_M of M. As enumeration over all possible trajectories are not practical if not possible, quality of a simulation is measured along trajectories sampled from some reference distribution. Given trajectories from a reference distribution, quality of the simulation is then measured by (i) the *state-transition dissimilarity* D_s, measured as the mean-squared error (MSE) between the embedded states $\phi(S_N)$ and the target states S_M, and (ii) the *output dissimilarity* D_o, measured as the difference in the expected return generated from the states S_N using the machine N and those generated from the states $\phi(S_N)$ using the machine M. If both the state-transition and output dissimilarities D_s and D_o are low/ negligible, then we establish that M simulates N. If both M simulates N and N simulates M, then we can say M and N are computationally equivalent, and their states are equivalent.

In practice, the mapping function ϕ is implemented as an MLP with ReLU activations and three hidden layers of 64, 128, and 64 units, respectively. The MLP is trained with the Adam Optimizer with learning rate 0.001 and batch size 64. The training set is consisted of 500 trajectories, and the results reported are from another test set of 300 trajectories. Compared with Mikulik et al. [12], where the reference distribution is generated by the meta-RL agent, here we modify the procedure by generating the reference distribution with the Bayes-optimal agent, as this provides an even more stringent condition where the simulation is evaluated in the regime of Bayes-optimal solutions.

When evaluating how well a Bayes-optimal agent *simulates* a meta-RL system, we first train an MLP mapping meta-RL states into Bayes-optimal states by minimizing the MSE between the mapped states and the target Bayes-optimal states. After training, quality of the simulation is measured on a test set by evaluating the state-transition dissimilarity (D_s: metaRL$\rightarrow$Bayes, as denoted in Fig. 3) and the output dissimilarity (D_o: metaRL$\rightarrow$Bayes as denoted in Fig. 3). If both D_s: metaRL$\rightarrow$Bayes and D_o: metaRL$\rightarrow$Bayes are low after training, then the Bayes-optimal agent simulates the meta-RL agent.

On the other hand, to evaluate how well a Bayes-optimal agent *is simulated by* a meta-RL one, an MLP is trained to map Bayes-optimal states into meta-RL states, and the state-transition dissimilarity is denoted D_s: Bayes$\rightarrow$metaRL and the output dissimilarity denoted D_o: Bayes$\rightarrow$metaRL in Fig. 3. If both D_s: Bayes$\rightarrow$metaRL and D_o: Bayes$\rightarrow$metaRL are low after training, then the Bayes-optimal agent is simulated by the meta-RL agent.

If all the above four dissimilarity measures are low/ negligible, then we can say that the meta-RL agent and the Bayes-optimal agent are computationally equivalent and their representations are equivalent. Our results in Fig. 3 show that after training the proposed meta-RL with predictive modules can attain much lower dissimilarities in all four measures than conventional RL^2.

Note before training the state transition dissimilarity D_s can be low, similar to what Mikulik et al. [12] reported and discussed — untrained RNN may maintain a verbose representation of the history by embedding each trajectory to a unique hidden state, which can be subsequently mapped to Bayes-optimal minimally sufficient statistic with low error using expressive enough functions, like the MLPs used in the above procedure. This observation also implies using decoding alone may not provide a thorough assessment of representation equivalence, and we need to consider their structural and computational relevance when comparing representations.

A.5 Additional results

A.5.1 State machine simulation: two-armed bandit

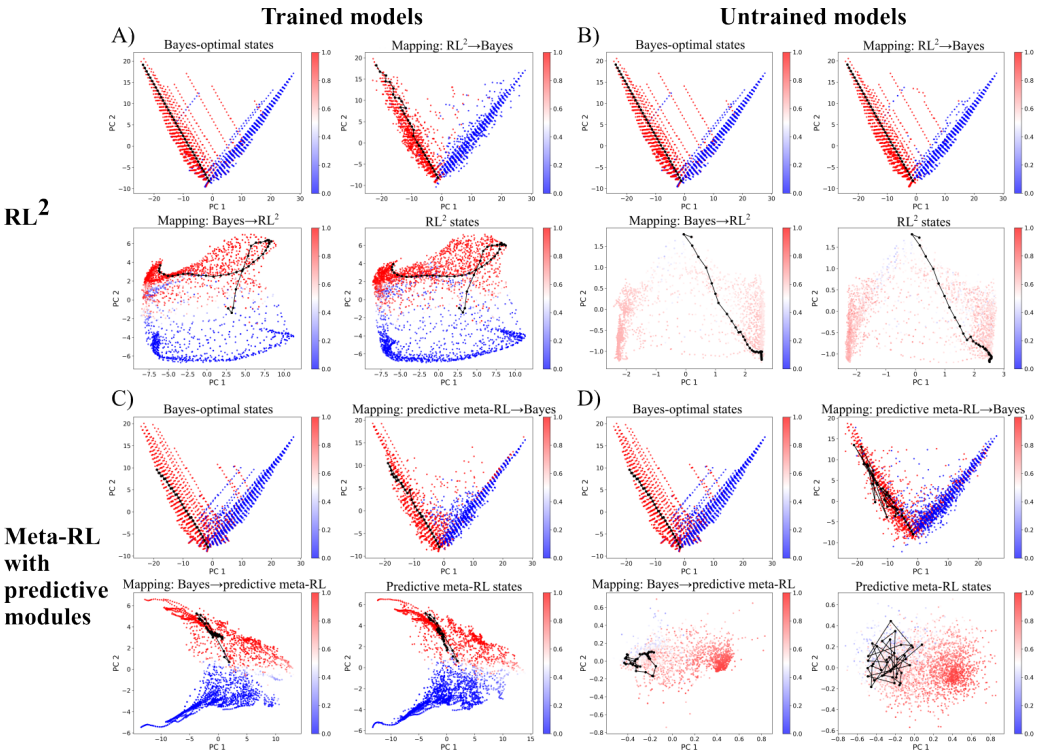

Figure 4: State machine simulation on the two-armed bandit task. For each meta-RL system of A) trained RL^2, B) untrained RL^2, C) trained meta-RL with predictive modules, and D) untrained meta-RL with predictive modules, we evaluate how well the meta-RL system in question can *simulate* and *be simulated by* the Bayes-optimal solution. For each subplot, visualization of state mapping are presented in clockwise order: (i) top left, Bayes-optimal states, (ii) top right, mapping meta-RL states into Bayes states, (iii) mapping Bayes-optimal states into meta-RL states, and (iv) meta-RL states. Note even for untrained meta-RL systems, it is possible to map their states into the Bayes states with low error, which indicates the untrained RNNs may maintain a verbose representation of the history and a thorough assessment of representation equivalence should consider not just correlation but also structural and computational relevance.

A.5.2 State machine simulation: oracle bandit

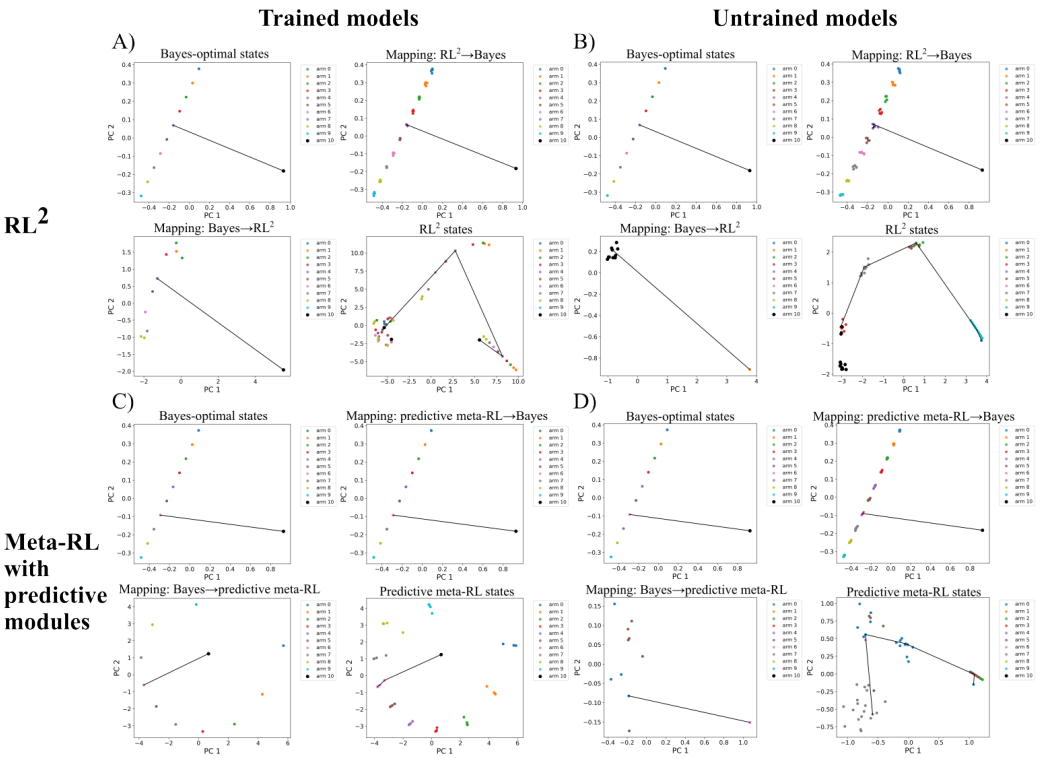

Figure 5: State machine simulation on the oracle bandit task. For each meta-RL system of A) trained RL^2, B) untrained RL^2, C) trained meta-RL with predictive modules, and D) untrained meta-RL with predictive modules, we evaluate how well the meta-RL system in question can *simulate* and *be simulated by* the Bayes-optimal solution. For each subplot, visualization of state mapping are presented in clockwise order: (i) top left, Bayes-optimal states, (ii) top right, mapping meta-RL states into Bayes states, (iii) mapping Bayes-optimal states into meta-RL states, and (iv) meta-RL states. Note only the proposed approach, trained models of meta-RL with predictive modules, are able to learn an interpretable representation capturing the underlying task structure with one dot for the oracle arm and ten other clusters for the ten possible target arms that are geometrically arranged according to their payout differences. All other models, including the trained models of RL^2, fail to learn such compact representations.