Exploring Meta-Learning Techniques in Reinforcement Learning

Johannes Hygrell
Karlsruhe Institute of Technology
Karlsruhe, Germany
johannes.hygrell@student.kit.edu

Karam Daaboul
Karlsruhe Institute of Technology
Karlsruhe, Germany
daaboul@kit.edu

***Abstract*—The field of reinforcement learning has seen steady improvements in recent years but relies on high numbers of training iterations when faced with new tasks. Meta-reinforcement learning attempts to mimic the human ability to leverage knowledge from prior experiences when faced with new tasks, presenting the concept of learning to adapt model parameters quickly. This seminar paper outlines the meta-reinforcement learning objective and several current algorithmic approaches. It also explains using a new benchmark to address the prevailing problem of narrow task distributions in current evaluations.**

I. Introduction

For many years, advances in computational power and training approaches have led to better results in machine learning. A recent and widely known example of improved performance is the success of Google DeepMind's *Agent57*, which in 2020 was the first deep reinforcement learning agent to beat the average human score on every game in the Atari57 benchmark [1].

Traditionally, approaches have focused on learning to perform tasks well from scratch, relying on the availability of large datasets to train agents sufficiently. Comparing this form of learning to how humans perform tasks reveals a significant difference. Humans do not need to (and would not be able to) train for a task thousands of times to perform well. Instead, we leverage knowledge from prior experiences to understand new tasks within a few or no attempts. In many real-world cases, quickly adjusting to new tasks could be a critical requirement. An example use-case could be the use of robots in factories or warehouses. If a robot has been trained to perform tasks like placing objects at certain locations, moving to a specific location, detecting object types, etc., then learning a new task, such as rotating objects, could be accomplished in a few steps due to the experience from the other tasks.

The emerging field of *meta-learning* has seen the development of approaches that train a model to be easily adjustable when faced with new tasks. This concept, also called *learning how to learn* [2], can be utilized in all forms of machine learning. While some approaches were initially designed for supervised learning, meta-learning can also be used to train reinforcement learning agents to adapt to new tasks. Applying the meta-learning concept to reinforcement learning introduces additional considerations, mainly due to the need for exploration. The agent must not only adapt effectively to new task data but also control the data it gathers through its interactions with the environment. Therefore, it is crucial to train the agent to interact with new environments in a way that results in the collection of effective data necessary for adaptation. The scope of this paper is limited to approaches for reinforcement learning problems. After introducing the general problem framing of meta-reinforcement learning and considerations that various approaches need to address in section II, some of the current and well-known algorithms for meta-reinforcement learning are described and compared in section III. In section IV, the problem of narrow task distributions in current approaches and a potential benchmark for improved evaluations is discussed.

II. Preliminaries

A. Reinforcement Learning

Before introducing meta-reinforcement learning, let us first define the relevant components of standard reinforcement learning. The reinforcement learning problem can be defined mathematically using *Markov Decision Processes* (MDP). An MDP $M = (S, A, P, R)$ with a set of states S, set of available actions A, transition function P and reward function R holds information about the interaction between the reinforcement learning agent and its environment. Trajectories τ describe the interaction through a number of time steps, holding information about the visited states, actions taken and received rewards. In the context of reinforcement learning, these MDPs can be seen as different tasks. They fully the describe the problem. Reinforcement learning assumes that the *Markov Property* of an MDP holds. This means that all future states only depend on the current state and not on any further past states in a trajectory. From this assumption, one can derive that an agent only needs to consider the current state (along with expected rewards) to act within the environment. All relevant information from past states have to also be stored in the current state at any time during the agent-environment interaction.

Agents receive information about the current state of the environment and received reward through sensors. There are two main methods of reinforcement learning which differ in how new actions are selected. *Value-based* methods approximate an optimal value function which maps between selectable actions and their expected reward. Actions with higher assigned values

are preferred by the agent. The most common value-based method is *Q-Learning*. Alternatively, *policy-based* methods update the *policy* directly without using a value function. A policy $\pi_\theta(s, a) = p(s|a)$ outputs which action a to take given a perceived state s. We solve an optimization objective to find the parameters θ that maximize the expected reward:

$$J(\theta) = \mathbb{E}_\pi[r(\tau)] \tag{1}$$

with expected reward $r(\tau)$ for trajectory τ. This can be done using a *policy gradient* approach. The parameter values are updated along the derived gradient of the policy:

$$\theta \longleftarrow \theta + \alpha \nabla_\theta J(\theta) \tag{2}$$

with the derived gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi[r(\tau) \nabla_\theta \log \pi(\tau)]. \tag{3}$$

The two methods of reinforcement learning can also be combined using an *actor-critic* algorithm. Here, two models are updated in parallel. The actor learns the optimal policy and thereby controls how the agent acts. The critic uses a value function to evaluate and feedback the selected action. Over time, the actor learns to perform better by adjusting based on the received feedback, and the critic learns to evaluate the selected actions better.

B. Meta-Reinforcement Learning

To train an agent for adaption to new tasks, the agent needs to be trained on several different tasks during a *meta-training* phase. The model can then be tested on a previously unseen task during a subsequent *meta-testing* phase. This is effectively a higher level of hold-out sets, where each task also contains a normal training-testing split. A task $\mathcal{M}_i$ drawn from the distribution over tasks $p(\mathcal{M})$ provides samples for training, after which the performance on the test-set of the task guides the improvement of the model, just as in standard reinforcement learning. The sample size is reduced to K *shots* to limit the number of samples available to the agent when learning a new task. Here, only K samples can be drawn both from all meta-training tasks and the final task during meta-testing.

Let us compare the objective of meta-reinforcement learning to standard reinforcement learning to understand the goal of meta-learning and the functionality of the approaches shown in section III. For our standard learning problem, we attempt to optimize a policy π_θ which delivers the highest expected reward during interaction with the environment:

$$\theta^* = \max_\theta \mathbb{E}_{\pi_\theta(\tau)}[R(\tau)], \tag{4}$$

with expected reward $R(\tau)$ for a trajectory τ. When adjusting this to a meta-reinforcement learning approach, we introduce the use of adapted parameters phi:

$$\theta^* = \max_\theta \sum_{i=1}^{n} \mathbb{E}_{\pi_{\phi_i}(\tau)}[R(\tau)], \tag{5}$$
$$\text{where } \phi_i = f_\theta(\mathcal{M}_i),$$

with $\mathcal{M}_i \sim p(\mathcal{M})$ representing the sampled task from our distribution of available tasks. Note how we still optimize the parameter θ, while the optimization objective is calculated based on the adapted parameter ϕ_i. We use a model f_θ to calculate ϕ_i, this is how the initial parameters still affect the objective function. This results in the meta-training of initial parameters θ which enable π_{ϕ_i} to achieve high expected rewards. During meta-testing, an update of ϕ_i based on the newly sampled test task $\mathcal{M}_i$ will result in the updated policy.

III. Notable Approaches

A. Model-Agnostic Meta-Learning (MAML)

Model-Agnostic Meta-Learning (MAML) is adaptable to any model that utilizes gradient descent for training. Its fundamental principle is to optimize initial model parameters so that they can be significantly adjusted by one or a few gradient steps. This objective is designed to be sensitive; thus, even minimal updates can result in considerable parameter changes. This method does not involve learning a distinct update function, which eliminates the need for extra parameters [3].

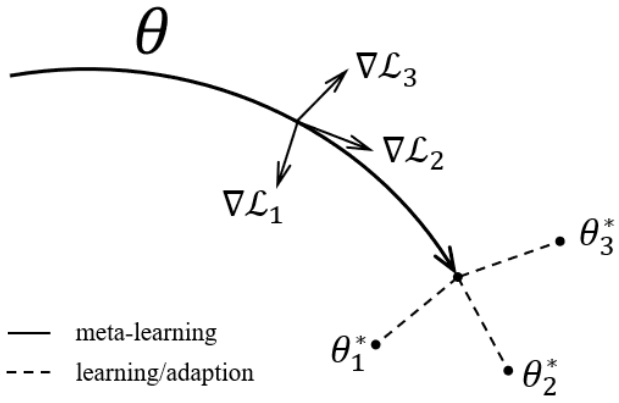

Fig. 1. The MAML algorithm seeks a representation θ that can swiftly adapt to new tasks.

For MAML, the optimization objective is formulated as:

$$\max_\theta \sum_{\mathcal{M}_i \sim p(\mathcal{M})} \mathcal{L}_{\mathcal{M}_i}(f_{\phi_i}), \tag{6}$$

where, in the context of reinforcement learning, the objective function $\mathcal{L}_{\mathcal{M}_i}$ is defined as:

$$\mathcal{L}_{\mathcal{M}_i}(f_{\phi_i}) = \mathbb{E}_{x_t, a_t \sim f_\phi, q_{\mathcal{M}_i}} \left[\sum_{t=1}^{H} R_i(x_t, a_t) \right], \tag{7}$$

representing the expected reward over H time steps for a task $\mathcal{M}_i$, with its specific transition distribution $q_{\mathcal{M}_i}$ and reward function R_i.

The inner update of the model parameters θ using the MAML algorithm is performed by:

$$\theta_i' = \theta + \alpha \nabla_\theta \mathcal{L}_{\mathcal{M}_i}(f_\theta), \tag{8}$$

where α denotes the step size of the inner gradient update. The global parameters θ are then updated using the adapted parameters θ_i':

$$\theta \longleftarrow \theta + \beta \nabla_\theta \sum_{\mathcal{M}_i \sim p(\mathcal{M})} \mathcal{L}_{\mathcal{M}_i}(f_{\theta_i'}), \tag{9}$$

with β representing the meta step size. This structure ensures that θ is optimized to provide an effective starting point for new tasks, leveraging updates from multiple sampled tasks, as illustrated in fig. 1.

B. Model-Agnostic Exploration with Structured Noise (MAESN)

Model-Agnostic Exploration with Structured Noise (MAESN) enhances Model-Agnostic Meta-Learning (MAML) by introducing structured exploration mechanisms, crucial for tasks with sparse rewards where traditional random noise strategies are inadequate [4]. MAESN integrates a latent exploration space that generates time-correlated noise, enabling consistent exploration strategies throughout an episode, thus adapting to new tasks using policy gradient methods alongside meta-learning.

The exploration strategy leverages latent variables y, sampled from a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ with meta-learned parameters μ and σ. These variables condition the policy $\pi_\theta(a|s, y)$, facilitating a coherent exploration approach throughout the episode by drawing y once per episode.

The objective in MAESN is adapted from MAML to incorporate the effects of structured noise:

$$\mathcal{J}_{\mathcal{M}_i}(f_{\phi_i,\omega_i}) = \mathbb{E}_{\substack{x_t,a_t \sim f_{\phi,\omega}, q_{\mathcal{M}_i} \\ \omega \sim \mathcal{N}(\mu,\sigma)}} \left[\sum_{t=1}^{H} R_i(x_t, a_t) \right]. \tag{10}$$

During meta-training, both the policy parameters θ and the variational parameters μ_i, σ_i are optimized for each sampled task τ_i. The optimization employs Trust Region Policy Optimization (TRPO) to maximize the expected reward and minimize the KL divergence from a standard normal distribution:

$$\max_{\theta,\mu_i,\sigma_i} \sum_{\mathcal{M}_i \sim p(\mathcal{M})} \mathcal{J}_{\mathcal{M}_i}(f_{\theta_i',y_i'}) - \sum_{\mathcal{M}_i \sim p(\mathcal{M})} D_{KL}(\mathcal{N}(\mu_i', \sigma_i') || \mathcal{N}(0, I)), \tag{11}$$

with $y_i' \sim \mathcal{N}(\mu_i', \sigma_i')$.

Inner loop updates for μ_i' and σ_i' are given by:

$$\mu_i' = \mu_i + \alpha_\mu \nabla_{\mu_i} \mathcal{J}_{\mathcal{M}_i}(f_{\theta_i,y_i}), \tag{12}$$

$$\sigma_i' = \sigma_i + \alpha_\sigma \nabla_{\sigma_i} \mathcal{J}_{\mathcal{M}_i}(f_{\theta_i,y_i}), \tag{13}$$

using step sizes α_μ and α_σ. Not updating the initial policy parameters θ_i' has shown to enhance performance by focusing solely on effective utilization of the latent exploration space.

The KL-divergence minimization ensures that initializing (μ, σ) to the prior values during meta-testing yields effective exploration outcomes.

Meta-testing adapts μ and σ to new tasks using gradient updates on the policy gradient objective:

$$\max_{\mu,\sigma} \mathcal{J}_{\mathcal{M}_i}(f_{\theta,y}). \tag{14}$$

This adaptation is performed using a gradient calculation that propagates through the sampling operation $y \sim \mathcal{N}(\mu, \sigma)$:

$$\nabla_{\mu,\sigma} \eta = \mathbb{E}_{\substack{x_t,a_t \sim f_{\theta,y}, q_{\mathcal{M}_i} \\ y \sim \mathcal{N}(\mu,\sigma)}} \left[\nabla_{\mu,\sigma} \log p_{\mu,\sigma}(y) \sum_{t=1}^{H} R_i(x_t, a_t) \right]. \tag{15}$$

Experiments demonstrate that MAESN significantly enhances exploration effectiveness and learning speed in environments with sparse or temporally extended rewards compared to both MAML and traditional learning-from-scratch approaches.

C. Probabilistic Embeddings for Actor-Critic Reinforcement Learning (PEARL)

Probabilistic Embeddings for Actor-Critic Reinforcement Learning (PEARL) innovatively addresses the challenges of on-policy meta-learning methods, like MAML, by adopting a more sample-efficient off-policy learning framework. On-policy learning depends on data generated from the current policy's interactions, requiring fresh data for each learning iteration, which is inherently inefficient. Off-policy learning, however, allows the agent to learn from an extensive cache of past interactions, thereby enhancing *sample efficiency*.

PEARL separates task inference from policy adaptation, employing on-policy methods for the former and off-policy methods for the latter. This bifurcation allows for a broader and more efficient use of historical data in policy training, avoiding the typical data mismatch problems by using recent data that closely matches the current policy distribution.

Central to PEARL's architecture is the latent context variable z, inferred from historical interactions and used to dynamically adapt the policy $\pi_\theta(a|s, z)$ to the specifics of the inferred task. This task inference leverages an inference network $q_\phi(z|c)$, which builds a probabilistic model from past state transitions up to a point N in task τ, denoted by $c = c^\tau_{1:N}$. This network parameterizes the posterior distribution $p(z|c)$, minimizing overfitting and irrelevant dependencies by focusing solely on crucial information.

PEARL's framework can be likened to a Partially Observable Markov Decision Process (POMDP), where instead of maintaining a belief about the current observable state, PEARL constructs a belief over the latent task variables. This is a significant shift as it treats the environment as an MDP sampled from a distribution of MDPs, with the agent acting based on the optimal policy given the inferred z. The latent variable Z is critical for enabling the agent to perform under conditions of uncertainty about the task, with posterior sampling from z guiding decision-making:

$$q_\phi(z|c_{1:N}) \propto \prod_{n=1}^{N} \Psi_\phi(z|c_n), \tag{16}$$

where $\Psi_\phi(z|c_n) = \mathcal{N}(f_\phi^\mu(c_n), f_\phi^\sigma(c_n))$, describing Gaussian factors for each transition, predicting mean μ and variance σ.

Integration with the Soft Actor-Critic (SAC) algorithm, which is known for maximizing reward and entropy, enhances PEARL's ability to navigate the balance between exploration and exploitation efficiently. The training loop updates the parameters ϕ, θ_π, and θ_Q for the inference network, actor, and critic functions, respectively:

$$\phi \longleftarrow \phi - \alpha_1 \nabla_\phi \sum_i (\mathcal{L}^i_{critic} + \mathcal{L}^i_{KL}), \tag{17}$$

$$\theta_\pi \longleftarrow \theta_\pi - \alpha_2 \nabla_\theta \sum_i \mathcal{L}^i_{actor}, \tag{18}$$

$$\theta_Q \longleftarrow \theta_Q - \alpha_3 \nabla_\theta \sum_i \mathcal{L}^i_{critic}. \tag{19}$$

where $\mathcal{L}^i_{KL} = \beta D_{KL}(q_\phi(z|c^i) \| p(z))$ is used to control the divergence between the learned and prior distributions of the latent variables $p(z)$, ensuring that the inference remains grounded and does not deviate excessively from prior knowledge.

During meta-testing, PEARL adapts to new tasks by iteratively refining the inferred context z, using it to adjust the policy for optimal task-specific performance. Experimental comparisons reveal that PEARL not only outperforms MAML in terms of sample efficiency but also shows significant improvements over MAESN. PEARL's structured approach to off-policy learning and task-specific adaptation allows it to start adapting to unseen tasks more quickly and achieve higher performance levels with fewer trajectories. This is particularly evident in environments with sparse rewards, where PEARL's ability to effectively utilize historical data and rapidly adjust to new information proves advantageous.

IV. Outlook

A. Experimental observation and benchmarks

While the presented approaches show promising results for the potential and practicability of meta-learning, there remain problems which might not be apparent due to the chosen evaluation for the approaches: All methods are trained on and evaluated using data with only minor parametric environmental differences between tasks. Limiting variations to parametric differences enables the algorithm to more easily place the newly seen task related to training tasks, thereby simplifying the problem.

Sampling from a narrow task distribution simplifies the learning challenge but fails to mirror real-world demands. The agent must utilize knowledge of non-parametric differences, such as unique muscle movements, to effectively benefit from meta-learning in practical scenarios. To address the current training and evaluation problem, a new benchmark containing a set of more distinct meta-learning tasks has been created [5]. The *Meta-World* benchmark consists of 50 tasks with unique movement objectives. Its largest set-up is designed to hold out 5 tasks while meta-training on the remaining 45.

Notably, the benchmark tasks show both parametric and non-parametric variability, as both are essential for a functional training distribution. If an agent is meant to learn how to adapt to unseen tasks that differ *non-parametrically*, the model needs to learn to look and account for those differences. This is only possible if the sampled meta-training tasks display non-parametric variation. The data set still requires *parametric variability* within the 50 tasks too, however, due to two reasons. First, if only one parametric profile would exist for every non-parametrically distinct task in the task distribution, the agent might simply assign the detected environmental parameter values to experienced tasks. Being able to use this assignment to distinguish between tasks instead of learning to watch for non-parametric differences could interfere with the intended learning algorithm and nullify the goal of the benchmark. Second, the concept of meta-learning assumes a shared structure within the task distribution, as this is needed for efficient adaption to new tasks [5]. A better task-overlap and shared structure are achieved by forcing the agent to generalize and learn about varying-parameter profiles within each task set-up.

As expected, experimental results show that current methods such as MAML struggle with the broader task distribution presented by the benchmark. Future approaches could use this evaluation tool to train for and determine the generalization efficiency of their algorithms.

B. Conclusion

Meta-learning, including the sub-field of meta-reinforcement learning, is a relatively new addition to machine learning. Successful algorithm developments have enabled machines to learn rapidly from prior training experiences in recent years. These algorithms demonstrate the feasibility of meta-learning, showcasing a range of objectives and techniques. They are employed not only to expedite policy parameter updates and enhance policy adaptation speed but also to swiftly update models of data distributions and improve exploration during meta-testing. Although MAML and MAESN employ similar strategies for parameter updates, PEARL introduces a distinct method that significantly enhances training efficiency.

The field continues to face challenges, such as narrow task distributions, yet substantial progress and expansion have occurred briefly. Future enhancements might reduce data and learning requirements during meta-testing or foster better experience transfer across varied task environments. These improvements are crucial for deploying meta-reinforcement learning in more dynamic, critical real-world applications.

References

[1] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell, "Agent57: Outperforming the Atari Human Benchmark," *arXiv*, 3 2020. [Online]. Available: http://arxiv.org/abs/2003.13350

[2] C. B. Finn, "Learning to Learn with Gradients," Ph.D. dissertation, University of California, Berkeley, 2018.

[3] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," *34th International Conference on Machine Learning, ICML 2017*, vol. 3, pp. 1856–1868, 3 2017. [Online]. Available: http://arxiv.org/abs/1703.03400

[4] A. Gupta, R. Mendonca, Y. Liu, P. Abbeel, and S. Levine, "Meta-Reinforcement Learning of Structured Exploration Strategies," *Advances in Neural Information Processing Systems*, vol. 2018-December, pp. 5302–5311, 2 2018. [Online]. Available: http://arxiv.org/abs/1802.07245

[5] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning," Stanford University, UC Berkeley, Columbia University, University of Souther California, Robotics at Google, Tech. Rep., 2019. [Online]. Available: https://github.com/rlworkgroup/metaworld