Seminar Paper: Model-Based Meta-Reinforcement Learning

Ikels; Joel
Karlsruhe Institute of Technology
Institute of Applied Informatics and Formal Description Methods
Karlsruhe, Germany
joel.ikels@student.kit.edu

***Abstract*—In machine learning, meta-learning methods aim for fast adaptability to unknown tasks using prior knowledge. Model-based meta-reinforcement learning enables to combine model-based reinforcement learning with meta reinforcement learning for increased sample efficiency. The goal of this paper is to provide an overview of important meta reinforcement learning ideas in order to focus on introducing algorithmic approaches that excel in the model-based meta reinforcement learning domain.**

I. Introduction

Adaptive behaviour lies in the the very nature of life as we know it. By forming a variety of behaviours, the animal brain enables its host to continuously adapt to environmental changes [25]. Toddlers, for example, can learn how to walk in sand in several moments, whereas robots often struggle to adapt fast and show rigid behaviour encountering a task not seen before. Fast adaption is possible because animals do not learn from scratch and leverage prior knowledge to solve a new task. In machine learning, the domain of meta-learning takes inspiration of this phenomenon by enabling a learning-machine to come up with a hypothesis on how to solve a new task using information from prior hypotheses of similar tasks [24]. The domain of *model-based reinforcement learning* (MBRL) comprises methods that enable an agent to successfully master complex behaviours using a deep neural network as a model of a tasks system dynamics [2]. By combining meta-learning and MBRL, agents can learn how to quickly learn new behaviours even if the environment or the task at hand changes [17], [23], [13], [3]. The goal of this paper is to provide an overview of important *meta reinforcement learning* (MRL) ideas in order to focus on introducing algorithmic approaches that excel in the *model-based meta-reinforcement learning* (MBMRL) domain. In doing so, the concepts of meta-learning and *model-free meta-reinforcement learning* (MFMRL) are introduced in the beginning, followed by three perspectives on MRL - gradient-based, recurrence-based and variational inference-based MRL. Next, a short introduction of MBRL forms a transition to the main part which introduces three algorithmic approaches that cover gradient-based, recurrence-based, and variational inference-based MBMRL.

II. Preliminaries

A. Meta-Learning

In machine learning, transfer-learning means to leverage prior knowledge of learned task data when learning from new task data, assuming that both data sets share many common features [12]. Many transfer-learning techniques successfully fine-tune a pre-trained neural network on large data from new tasks but show limited performance on few-shot learning, where data is limited [22]. Meta-learning, in contrast, aims to learn models that can be quickly adapted to new tasks [24] and can be described as a set of methods that apply a learned prior of common task structure to make generalizeable inference with small amounts of data [11]. Following this learning how to learn paradigm, being able to quickly adapt to new tasks can be viewed in the light of a few-shot learning setting where the goal of meta-learning is to be able to adapt a model f_θ to an unseen task $\mathcal{M}_j$ of a distribution of tasks $p(\mathcal{M})$ with a small amount of K data samples [10]. The meta-learning procedure usually is divided into meta-training with n meta-learning tasks $\mathcal{M}_i$ and meta-testing with y meta-test tasks $\mathcal{M}_j$ both drawn from $p(\mathcal{M})$ without replacement [11]. During meta-training, task data may be split into train and test sets usually representing K data points of a task $\mathcal{D}^{\text{meta-train}} = \{(\mathcal{D}^{\text{tr}}_{i=1}, \mathcal{D}^{\text{ts}}_{i=1}), \dots (\mathcal{D}^{\text{tr}}_{i=n}, \mathcal{D}^{ts}_{i=n})\}$. Meta-testing task data $\mathcal{D}^{\text{meta-test}} = (\mathcal{D}^{\text{meta-test}}_{j=1}, \dots, \mathcal{D}^{meta-test}_{j=y})$ is hold out during meta-training [11]. Meta-training is then performed with $\mathcal{D}^{\text{meta-train}}$ and can be viewed as bi-level learning of model parameters [20]. In the inner-level, an update algorithm Alg with hyperparameters ψ must find task specific parameters ϕ_i by adjusting meta-parameters θ. In the outer-level, θ must be adjusted to minimize the cumulative loss of all ϕ_i across all learning tasks by finding common characteristics of different tasks through meta parameters $\theta^\star$:

$$\overbrace{\theta^\star = \arg\min_\theta \sum_{i=1}^{n} \mathcal{L}_{\mathcal{D}_i \sim \mathcal{M}_i}(\phi_i)}^{\text{outer-level}} \quad \text{where } \underbrace{\phi_i = Alg^{\psi}_{\mathcal{D}_i \sim \mathcal{M}_i}(\theta)}_{\text{inner-level}} \tag{1}$$

Once $\theta^\star$ is found it can be used during meta-testing for quick adaption: $\phi_j = Alg(\theta^\star, \mathcal{D}^{meta-test}_j)$

It should be noted that instead of learning good initial parameters for fast adaption of unseen tasks, multi-task learning aims to receive a model that is trained to be conditioned on all training tasks of a distribution of tasks [4]. However, by thinking of ϕ_i as task specific heads of a multi-task network, it has been shown that the problem formulations of (gradient-based) meta-learning and multi-task learning are equivalent but both methods differ in their employed optimization algorithms [26].

B. Model-free Meta-Reinforcement Learning

In reinforcement learning (RL), a task can be described as a Markov Decision Process (MDP) $\mathcal{M} = \{S, A, p\left(s_{t=1}\right), p\left(s_{t+1} \mid s_t, a_t\right), r, H\}$ with a set of states S, a set of actions A, a reward function $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, an initial state distribution $p(s_1)$, a transition probability distribution $p\left(s_{t+1} \mid s_t, a_t\right)$, and a discrete-time finite or continuous-time infinite horizon H. RL aims to find a policy $\pi^\star$ that solves the MDP mapping from states s_t to actions a_t while receiving the highest possible reward. In a model-free RL setting, this is achieved by iteratively improving the policy based on task data $\mathcal{D} = \{(s_1, a_1, r_1 \dots s_H), \dots\}$ of previous policy rollouts. Under the assumption that meta-training tasks M_i and meta-testing tasks M_j are drawn from the same distribution of tasks $p(M)$, MFMRL aims to find a meta policy $\pi_{\theta^\star}$ that performs well on average on unseen meta-testing tasks and thus can be adapted quickly to new tasks [9]. Referring to the meta-learning goal in Equation 1, $\pi_{\theta^\star}$ is represented by a neural network $f_{\theta^\star}$ that embeds common characteristics of different tasks and enables quick adaption to new tasks [10]. During meta-training in MFMRL, Alg finds a set of task specific parameters ϕ_i for several task specific policies π_{ϕ_i}. The negative expected reward of this policy being applied to $\mathcal{M}_i$ is the loss $\mathcal{L}_{\mathcal{D}_i}(\phi_i)$:

$$\mathcal{L}_{\mathcal{D}_i \sim \mathcal{M}_i}(\phi_i) = -E_{s_t, a_t \sim \pi_{\phi_i}}\left[\sum_{t=1}^{H} R_i\left(s_t, a_t\right)\right] \tag{2}$$

C. Gradient-based Meta-Reinforcement Learning

"Gradient-based meta-learning methods leverage gradient descent to learn the commonalities among various tasks" [16, p. 1]. One such method introduced by Finn et al. [10] is *Model-Agnostic Meta-Learning* (MAML). The key Idea of MAML is to tune a models initial parameters such that the model has maximal performance on a new task. Here, bi-level meta learning is achieved by bi-level optimization. A models task specific optimization in the inner-level (O1) and a task agnostic meta optimization in the outer-level (O2). Instantiated for RL, MAML uses policy gradients of a neural network model f_θ. Before meta-training a batch of training tasks $\mathcal{D}^{\text{meta-train}}$ is sampled. Each training iteration i covers one task $\mathcal{M}_i$. During meta-training, task data in the form of $\mathcal{K}$ trajectories $\mathcal{D}_i^{tr} = \{(s_1, a_1, r_1 \dots s_H), \dots \mathcal{K}\}$ is sampled per iteration with roll-outs from f_θ. In O1, task specific parameters ϕ_i are generated by adapting f_θ with gradient descent as Alg. This is done based on the gradient of sampled training trajectories from rollouts with f_θ and a step size $\alpha = \psi$ which is usually a fixed hyperparamter but can also be meta learned:

$$\phi_{i_{\text{MAML}}} = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{D}_i^{\text{tr}}}(\theta) \tag{3}$$

Then, testing trajectories $\mathcal{D}_i^{ts}$ are sampled with $\phi_{i_{\text{MAML}}}$ and the loss $\mathcal{L}_{\mathcal{D}_i^{\text{ts}}}(\phi_i)$ is computed. After differentiating $\phi_{i_{\text{MAML}}}$ through the optimization process of O1, parameters θ are being adjusted via stochastic gradient descent and a meta step-size β in O2 to reduce the cumulative loss of all task specific functions across all newly sampled task data:

$$\theta \longleftarrow \theta - \beta \nabla_\theta \sum_{i=1}^{n} \mathcal{L}_{\mathcal{D}_i^{\text{ts}}}(\phi_{i_{\text{MAML}}}) \tag{4}$$

REPTILE by Nichol et al. [19] is a first order implementation of MAML and updates parameters towards a solution that is close to each tasks manifold of optimal solutions. In contrast to MAML, task specific gradients do not need to be differentiated through the optimization process and train and test data do not have to be sampled in the inner and outer step. This makes REPTILE more computationally efficient with similar performance. During meta-training one task $\mathcal{M}_i$ is sampled from $p(M)$ without replacement for each iteration. Then, task data in the form of $\mathcal{K}$ trajectories $\mathcal{D}_i = \{(s_1, a_1, r_1 \dots s_H), \dots \mathcal{K}\}$ is sampled with roll-outs from f_θ. In O1 task specific parameters ϕ_i are generated by adapting f_θ with $k > 1$ steps of stochastic gradient descent:

$$\phi_{i_{\text{REPTILE}}} = \theta - \nabla_\theta \mathcal{L}_{\mathcal{D}_i}(\theta) \tag{5}$$

In O2 the parameters θ are then being adjusted to minimize the euclidean distance between θ and ϕ_i with step-size α:

$$\theta \leftarrow \theta + \alpha\left(\phi_{i_{\text{REPTILE}}} - \theta\right) \tag{6}$$

D. Recurrence-based Meta-Reinforcement Learning

The gradient-based MRL methods introduced in the previous section, are so called optimization-based methods [11]. Recurrence-based MRL comprises so called black-box methods, solving the meta learning objective by using Recurrent Neural Networks (RNN) as a policy [11]. In this regard, the works of Duan et al. [9] and Andrychowicz et al. [1] use a RNN that iteratively learns meta weights $\theta^\star$ with a hidden state h that is maintained between K episodes of interactions, $\mathcal{D}_i = \{(s_1, a_1, r_1 \dots x_H), \dots \mathcal{K}\}$, with a task. Across episodes and for each of n different tasks, h is updated with parameters ψ and produces task specific functions by conditioning the output of the RNN generated by θ (i.e. inner-level learning):

$$\phi_{i_{RNN}} = RNN^{\psi}_{\mathcal{D}_i}(\theta) \tag{7}$$

After the last episode of a task has terminated and a new task is drawn, the hidden state is reset and the meta weights are updated:

$$\theta \longleftarrow \theta - \beta \nabla_\theta \sum_{i=1}^{n} \mathcal{L}_{\mathcal{D}_i}(\phi_{i_{RNN}}) \tag{8}$$

In that way, by iterating across different tasks, "good" initial parameters are found and the meta policy is procedurally generated (i.e. outer-level learning).

E. Variational-Inference-based Meta-Reinforcement Learning

The algorithm *probabilistic embedding for actor-critic reinforcement learning* (PEARL) by Rakelly et al. [21] views MRL in the context of solving partially observed markov decision processes (POMDP) where the transition and reward functions of different tasks are unknown but can be estimated by taking actions in the environment. This is methodized by separating task inference (inner-level) and policy adaption (outer-level) by learning a latent vector z that encodes relevant task information a policy $\pi_\theta(a_t \mid s_t, z)$ receives as auxiliary input to be task specific. To do this, a prior over z is placed, $z_{1:n} \sim p(z_{1:n}) = \mathcal{N}(0, I)$, assuming z exhibits gaussian behavior across n meta-learning tasks. Before meta-training, a batch of n tasks M_i is sampled from $p(M)$ without replacement. Then, iteratively for each task, data $\mathcal{D}_i = \{(s_1, a_1, r_1 \dots x_H), \dots \mathcal{K}\}$ of K rollouts from the current π_θ is sampled and stored in an experience replay buffer representing $\mathcal{D}^{\text{meta-train}}$. By sampling task data from the replay buffer, the posterior $p(z|\mathcal{M}_i)$ is inferred via an probabilistic encoder network q_ϕ which is parametrized by parameters ϕ :

$$\phi_{i_{\text{INF}}} = q_{\phi_i}(z \mid \mathcal{D}_i) \approx p(z \mid \mathcal{M}_i) \tag{9}$$

Addressing the inner and outer level of our meta-learning objective 1, soft actor critic (SAC) optimizes parameters θ and ϕ jointly by maximizing the evidence lower bound (ELBO) which maximizes the expected reward of $\pi_\theta(a_t|s_t, z)$ while forcing the inference network q_{ϕ_i} to stay close to the prior of z across tasks.

F. Model-based Reinforcement Learning

In general, a drawback of model-free RL approaches is that they require large amounts of data (i.e. MDP interaction) to learn a useful policy. This makes such approaches impractical for real world applications where, in comparison, data is usually scarce for it can be acquired much slower with physical systems in real environments than in simulated environments [8], [14], [6], [7].

MBRL methods sample ground truth data $\mathcal{D}_i = \{(s_0, a_0, s_1), (s_1, a_1, s_2), \dots\}$ from a specific task $\mathcal{M}_i$ and use this data to train a dynamics model $p_\theta(s_{t+1} \mid s_t, a_t)$ that estimates the underlying dynamics of the data to approximate which state follows which action. This is done by optimizing the weights θ to maximize the log-likelihood of the observed data:

$$\begin{aligned} \theta^* &= \underset{\theta}{\operatorname{argmax}} p(\mathcal{D}_i \mid \theta) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{(s_t, a_t, s_{t+1}) \in D_i} \log p_\theta(s_{t+1} \mid s_t, a_t) \end{aligned} \tag{10}$$

The learned dynamics model is then utilized to optimize a sequence of actions (e.g. with model predictive control) or to optimize a policy, making MBRL much more sample efficient than model-free RL [8], [18].

III. Model-based Meta-Reinforcement Learning

Even though MBRL methods show improved sample efficiency, the amount of training data needed to reach "good" performance scales exponentially with the dimensionality of the input state-action space of the dynamics model [5]. Additionally, data scarcity is even more challenging when a system has to adapt online while executing a task [13]. A robot, for example, might encounter sudden changes of system dynamics (e.g. damaged joints) or changes in environmental dynamics (e.g. new terrain conditions) that require fast online adaption in seconds or at worst minutes [13]. Increasing the data efficiency, MBMRL aims to combine MRL with MBRL and let an agent quickly adapt to dynamic changes by coming up with a new solution leveraging knowledge about prior dynamics [23], [17], [15], [3], [27]. It should be noted that each dynamic resembles a MDP and therefore a RL task $\mathcal{M}_i$. Since in real world applications new dynamics can appear at any time, a new task could appear at any time. Hence, a task can be understood as an arbitrary trajectory segment of K timesteps under certain dynamics and a meta-learner is trained to adapt to a distribution of these temporal fragments based on M recent observations [17], [13].

IV. Notable Approaches

A. Learning To Adapt In Dynamic, Real World Environments

An approach that is applicable for gradient and recurrence based MRL methods was presented in the work of Nagabandi et al. [17] targeting online adaption of a robotic system encountering different system dynamics in real world environments. As mentioned above, a task is considered to be a setting of dynamics (e.g. damaged joint, steep slope) where M previous timesteps provide information about the current task setting. During meta-training, $\mathcal{D}^{\text{meta-train}}$ task data consists of trajectory segments of past M observations $\mathcal{D}_i^{tr} = \{(s_1, a_1, r_1 \dots s_M)\}$ and trajectory segments of the next K timesteps $\mathcal{D}_i^{ts} = \{(s_1, a_1, r_1 \dots s_K)\}$. During meta-testing (i.e. online adaption) $\mathcal{D}^{\text{meta-test}}$ task data consists of trajectory segments of past M observations $\mathcal{D}_j = \{(s_1, a_1, r_1 \dots x_M)\}$. Relating to the meta-learning objective defined in Equation 1, the loss of a dynamics model adjusted to be task specific is the negative log-likelihood of the K next timesteps under the model:

$$\mathcal{L}_{\mathcal{D}_i \sim \mathcal{M}_i}(\phi_i) = -\frac{1}{K} \sum_{t=1}^{t=K} \log p_{\phi_i}(s_{t+1} \mid s_t, a_t) \tag{11}$$

Algorithm 1 shows the meta-learning procedure where the dynamics model is trained on all task data collected so far with new task data being sampled frequently using Algorithm 2 with randomly initialized parameters. In the inner level (Algorithm 1, Line 9) Alg updates a dynamics models parameters θ based on past timesteps to perform good in the outer level (Algorithm 1, Line 12) on nearby future timesteps. Instantiated for gradient-based meta RL methods, Alg represents gradient descent with the learning rate ψ whereas instantiated for recurrence-based methods Alg represents a RNN with

parameters ψ responsible for updating its hidden state. During meta-testing, Algorithm 2 is used to evaluate the performance of the meta learners ability to adapt online. This is achieved step-wise by selecting actions planned with model predictive control (MPC) or model predictive path integral (see Williams et al. [28]), whilst the planning horizon H is smaller than the adaption horizon K used in Equation 11.

Algorithm 1 Model-based Meta-Reinforcement Learning (Meta-training)

Require: Distribution $p(\mathcal{M})$ over tasks
Require: Learning rate $\beta \in \mathbb{R}^+$
Require: Number of sampled tasks n
Require: Empty Dataset $\mathcal{D}^{\text{meta-train}} = \{\}$
Require: Task sampling frequency $n_S \in \mathbb{Z}^+$
1: Randomly initialize θ
2: **for** $i = 1, ...$ **do**
3: **if** $i \bmod n_S = 0$ **then**
4: Sample $\mathcal{M}_i \sim p(\mathcal{M})$
5: Collect $\mathcal{D}_i$ using Algorithm 2
6: $\mathcal{D}^{\text{meta-train}} \leftarrow \mathcal{D}^{\text{meta-train}} \cup \{\mathcal{D}_i\}$
7: **end if**
8: **for** $i = 1 \dots n$ **do**
9: $\phi_i \leftarrow Alg^{\psi}_{D_i^{tr}}(\theta)$
10: $\mathcal{L}_i \leftarrow \mathcal{L}_{D_i^{ts}}(\phi_i)$
11: **end for**
12: $\theta \leftarrow \theta - \beta \nabla_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i$
13: $\psi \leftarrow \psi - \eta \nabla_\psi \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i$
14: **end for**
15: **return** (θ, ψ) as ($\theta^\star$, $\psi^\star$)

Algorithm 2 Data Collection (Meta-training) and Online Model Adaptation (Meta-testing)

Require: $\theta^\star, \psi^\star$ {Meta-learned parameters}
Require: controller()
Require: r {Reward function}
Require: H {Planning horizon}
1: **if** Meta-training=True **then**
2: $\mathcal{D}_{\text{x}} = \mathcal{D}_i$ {Meta-training task data}
3: **else**
4: $\mathcal{D}_{\text{x}} = \mathcal{D}_{\text{j}}$ {Meta-testing task data}
5: **end if**
6: $\mathcal{D}_{\text{x}} \leftarrow \emptyset$
7: **for** each timestep t **do**
8: $\phi_{\text{x}} \leftarrow Alg^{\psi^\star \theta^\star}_{\mathcal{D}_{\text{x}}}$
9: $a \leftarrow$ controller(ϕ_{x}, r, H)
10: Execute a, add result to $\mathcal{D}_{\text{x}}$
11: **end for**
12: Return rollout $\mathcal{D}_{\text{x}}$

B. Gradient-based MBMRL using Embeddings

Kaushik et al. [13] point out that in a MRL setup where situations do not posses strong global similarity, finding a single set of initial parameters is often not sufficient to learn quickly. One potential solution would be to find several initial sets of model parameters during meta-training and, when encountering a new task, use the most suitable one so that an agent (e.g. robot) can adapt through several gradient steps. Their work *Fast Adaptation through Meta-Learning Embeddings* (FAMLE) approaches this solution by extending a dynamical models input with an additional input h, which is a d-dimensional vector describing a situation c_i. The values of h are initially unknown but learned during meta-training, resulting in a situational embedding h_i for each situation. A situation c_i is considered as any circumstance that perturbs the dynamics of as system (e.g. an malfunctioning robot leg). In other words and as discussed in Section III, each situation is a task $c_i = \mathcal{M}_{\text{i}}$ of unknown dynamics of a system sampled from a distribution of unknown dynamics $p(c) = p(\mathcal{M})$ of that system. By meta-training model parameters θ and embeddings h jointly, several initial sets of model parameters are found, each conditioned on a situation represented by h_i resulting in a situation conditioned dynamical model $p_\theta\left(s_{t+1} \mid s_t, a_t, h\right)$. If an unseen situation c_j appears, its similarity to prior situations is measured. The most-likely situational embedding h_{likely} is then used to further initialize the model parameters and therefore enables fast adaption. The meta-learning process of FAMLE is described in Algorithm 3. Prior to meta-training, n situations are sampled inside a simulation from $p(c) = p(M)$ resulting in a set of meta-training tasks $\mathbb{C}$. For each c_i training data $\mathcal{D}_i = \{(s_t, a_t, s_{t+1}) \mid t = 1, \dots, N\}$ is sampled by a simulated robot randomly taking N actions. Then, to be learned situational embeddings $\mathbb{H} = \{h_i \mid i = 1 \dots n\}$ corresponding to each situation are initialized. During meta-training, relating to the meta-learning goal defined in Equation 1, initial model parameters $\theta^\star$and n situational embeddings $\mathbb{H}^\star$ are found that minimize the loss for any situation c sampled out of $c_{1:n}$:

$$\begin{gathered}\theta^\star, \mathbb{H}^\star = \arg\min_{\theta, h_i} \textstyle\sum_{i=1}^{n} \mathcal{L}_{\mathcal{D}_i}(\phi_i, h_i) \\ \text{where } \phi_i, h_i = Alg^{\psi}_{\mathcal{D}_i}(\theta, h)\end{gathered} \tag{12}$$

The loss of a situation-conditioned dynamical model for a specific situation $c_i \sim \mathcal{D}_i$ be as follows:

$$\mathcal{L}_{\mathcal{D}_i}(\phi_i, h_i) = -\frac{1}{K} \sum_{t=1}^{t=K} \log p_{\phi_i}(s_{t+1} \mid s_t, a_t, h_i) \tag{13}$$

Similar to REPTILE, bi-level optimization is achieved by making a gradient-based, task specific update of θ and h in the inner level with a fixed ψ (Algorithm 3, Line 9):

$$\begin{aligned}\phi_i &= \theta - \nabla_\theta \mathcal{L}_{\mathcal{D}_i}(\theta, h) \\ h_i' &= h - \nabla_h \mathcal{L}_{\mathcal{D}_i}(\theta, h)\end{aligned} \tag{14}$$

and simultaneously updating θ and h towards their task specific counterparts in the outer level (Algorithm 3, Line 10-11):

$$\begin{aligned}\theta &\leftarrow \theta - \alpha(\phi_i - \theta) \\ h_i &\leftarrow h_i - \beta(h_i' - h_i)\end{aligned} \tag{15}$$

To collect observations during online adaptation (i.e. real world interaction), MPC uses the dynamics model to predict

into the future and selects the action sequence τ^* with highest predicted reward:

$$\tau^* = \arg\max_{\tau_i} \sum_{t=0}^{H-1} r\left(s_t, a_t, f_{\theta^*}\left(s_t, a_t, h^*\right)\right) \tag{16}$$

As discussed in Section III, Algorithm 4 also adapts the dynamics model based on M recent observations while making the assumption that a new situation (i.e. task) is taking place after every K control steps. First, based on M recent observations $\mathcal{D}_j = \{(s_t, a_t, s_{t+1}) \mid t = 1, \ldots, M\}$ the most likely situational embedding h_{Likely} is defined:

$$h_{\text{Likely}} = \arg\max_{h \in \mathbb{H}_\star} \mathbb{E}_{\mathcal{D}_j}\left[\log p_{\theta^\star}\left(s_{t+1} \mid s_t, a_t, h\right)\right] \tag{17}$$

Next, the dynamical model is updated online by simultaneously updating h_{Likely} and $\theta^\star$ taking k gradient steps:

$$\begin{aligned} \theta &\leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{D}_|}\left(\theta, h_{\text{likely}}\right) \\ h_j &\leftarrow h_{likely} - \beta \nabla_h \mathcal{L}_{\mathcal{D}_|}\left(\theta, h_{\text{likely}}\right) \end{aligned} \tag{18}$$

Algorithm 3 Model-Based Meta-Reinforcement Learning with FAMLE (Meta-training)

Require: Distribution $p(\mathcal{M}) = p(c)$ over tasks
Require: Learning rate $\alpha \in \mathbb{R}^+$
Require: Learning rate $\beta \in \mathbb{R}^+$
Require: Number of sampled tasks n
Require: Empty Dataset $\mathcal{D}^{\text{meta-train}} = \{\}$
Require: $Alg_c^\psi(\cdot,\cdot)$ k steps of SGD update rule
1: **for** $i = 1 \ldots n$ **do**
2: $c_i \sim p(c)$ {Sample a situation}
3: $\mathbb{C} \leftarrow c_i$ {Save the situation}
4: $\mathcal{D}_i = \{(s_t, a_t, s_{t+1}) | t = 1, \ldots, N\}$ {Simulate and collect data}
5: $\mathcal{D}^{\text{meta-train}} \leftarrow \mathcal{D}^{\text{meta-train}} \cup \{\mathcal{D}_\text{i}\}$ {Save situational data}
6: **end for**
7: **for** $m = 0, 1, \ldots$ **do**
8: $\mathcal{D}_i \sim \mathcal{D}^{\text{meta-train}}$ {Sample situational data}
9: $\phi_i, h_i' = Alg_i^\psi(\theta, h_i)$ {Perform SGD for k steps}
10: $\theta \leftarrow \theta + \alpha(\phi_i - \theta)$ {Move θ towards ϕ_i}
11: $h_i \leftarrow h_i + \beta(h_i' - h_i)$ {Move h_i towards h_i'}
12: **end for**
13: **return** (θ, h) as ($\theta^\star$, $h^\star$)

Algorithm 4 Online Model Adaptation with FAMLE (Meta-testing)

Require: $\theta^\star, \mathbb{H}^\star$ {Meta-learned parameters and embeddings}
Require: $\mathcal{D}_M = \phi$ {Empty set for M recent observations}
Require: MPC()
Require: r {Reward function}
1: $\mathcal{D} \leftarrow \emptyset$
2: **while** not $Solved$ **do**
3: h_{Likely} = most likely $h_i \in \mathbb{H}^\star$ given $\mathcal{D}_j$ and $\theta^\star$
4: $\theta^*, h^* = k$ steps SGD from $\theta^\star, h_{Likely}$ using $\mathcal{D}_j$
5: Apply optimal action $a = MPC(\theta^*, h^*, r)$
6: $\mathcal{D}_j \leftarrow \mathcal{D}_j \cup \{(s_t, a_t, s_{t+1})\}$ {Insert observation}
7: **if** $size(\mathcal{D}_j) > M$ **then** Remove oldest from $\mathcal{D}_j$
8: **end while**
9: **return**

C. Variational Inference based MBMRL

The work *Model-Based Meta-Reinforcement Learning for Flight with Suspended Payloads* by Belkhale et al. [3] introduces a meta-learning approach that enables a quadcopter to adapt online to various payloads physical properties (e.g. mass, tether length) using variational inference. Intuitively each payload causes different system dynamics and therefore defines a task to be learned. Since it is unlikely to accurately model such dynamics by hand and not realistic to know every payloads properties values beforehand, the meta learning goal is the rapid adaption to unknown payloads without prior knowledge of the payload's physical properties. That's why a probabilistic encoder network q_ϕ finds a task specific latent vector z_i which is then fed into a dynamics network p_θ as an auxiliary network. Using z, the dynamics network learns to model the factors of variation that affect the payload's dynamics and are not present in the state s. During meta-training, initial model parameters $\theta^\star$ and n task specific z_i must be learnt that maximize the probability for correct model predictions regarding each training task:

$$\theta^\star = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \sum_{(s_t, a_t, s_{t+1}) \in \mathcal{D}_\text{i}} \log p_\theta\left(s_{t+1} \mid s_t, a_t, z_i\right) \tag{19}$$

In order to approximate the computationally intractable posterior $p_\theta(z_{1:n}|\mathcal{D}^{\text{meta-train}})$, a variational posterior following a gaussian prior over z is inferred with q_ϕ:

$$\phi_i = q_{\phi_i}\left(z_i\right) = \mathcal{N}\left(\mu_i = 0, \Sigma_i = 1\right) \approx p_\theta(z_i|\mathcal{D}_\text{i}) \tag{20}$$

The evidence lower bound (ELBO) below serves as an approximation of $\log p(\mathcal{D}^{\text{meta-train}} \mid \theta)$, since $p_\theta(z_{1:n}|\mathcal{D}^{\text{meta-train}})$ remains unknown:

$$\begin{aligned} \log p(\mathcal{D}^{\text{meta-train}} \mid \theta) &\geq \text{ELBO}\left(\mathcal{D}^{\text{meta-train}} \mid \theta, \phi_{1:n}\right) \\ &\doteq \textstyle\sum_{i=1}^{n} \mathbb{E}_{z_i \sim q_{\phi_i}} \sum_{(s_t, a_t, s_{t+1}) \in \mathcal{D}_\text{i}} \log p_\theta(s_{t+1} \mid s_t, a_t, z_i) \\ &\quad - \text{KL}(q_{\phi_i}(z_i) \| p(z_i)) \end{aligned} \tag{21}$$

Algorithm 5 Model-Based Meta-Reinforcement Learning using Variational Inference

// Meta-training
Train p_{θ^*} given $\mathcal{D}^{\text{meta-train}}$ and Equation 22
// Meta-testing
Initialize variational parameters: $\phi^* \leftarrow \{\mu^{\text{meta-test}} = 0, \Sigma^{\text{meta-test}} = 1\}$
for each timestep t **do**
    Solve optimal action a_t given p_{θ^*}, q_{ϕ^*}, and MPC
    $\mathcal{D}^{\text{meta-test}} \leftarrow \mathcal{D}^{\text{meta-test}} \cup \{s_t, a_t, s_{t+1}\}$
    Infer variational parameters ϕ^* given $\mathcal{D}^{\text{meta-test}}$ and Equation 23
end for

Therefore, by maximizing ELBO, $\phi_{1:n}$ and θ are optimized jointly according to the meta-learning goal defined in Equation 1:

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \max_{\phi_{1:n}} \text{ELBO}\left(\mathcal{D}^{\text{meta-train}} \mid \theta, \phi_{1:n}\right) \tag{22}$$

Algorithm 5 shows the meta-training and the meta-testing process. During meta-testing (i.e. online adaption), θ^* represents the prior knowledge which is used to choose actions via MPC. Fast adaption to unseen tasks is then achieved by maximizing ELBO for task specific parameters with a fixed θ^*:

$$\phi^* = \underset{\phi}{\operatorname{argmax}} \text{ELBO}\left(\mathcal{D}^{\text{meta-test}} \mid \theta^*, \phi_{1:y}\right) \tag{23}$$

V. Conclusion

Model-based meta-reinforcement learning is a promising domain that slowly gains traction in the machine learning and robotics community. Experimentation has shown that MBMRL methods can learn with up to 1000 times less data compared to model free RL approaches, while reaching similar performance [17]. Therefore, the data efficiency of these methods complements MBRL perfectly and may hold exiting opportunities for RL applications in the near future.

In particular the underlying principles of FAMLE and variational inference based MBMRL seem to be promising directions since they enable very fast online adaption to unknown tasks. However, because the functionality of variational inference based MBMRL is very similar to the functionality of variational autoencoders, resulting algorithms may also generate blurry data (i.e. next state prediction) due to the chosen prior of the variational posterior. This could be a great drawback in dynamically detailed, novel situations that require "well informed" model-based decisions. In the manner of FAMLE, on the contrary, situational embeddings could be learnt hierarchically with a (temporal) convolutional neural network (CNN) and be fed into a CNN dynamics model as auxiliary input. This could enable recognition of more complex, detailed situations and modeling of complex situation dynamics, whereupon an agent may by able to achieve more complex goals.

Nonetheless, to the best of the authors knowledge, the field of MRL hasn't developed any real world applications yet. Still, the underlying principles of fast adaptability in real world environments and (re-)usability of prior knowledge may be one key component in developing highly autonomous artificial intelligence systems running with decreasing human supervision. Therefore, MBMRL algorithms may have to prove their value in future RL research but hold great invention potential, especially when it comes to RL applications that demand (quick) learning and reacting informed by prior knowledge.

References

[1] Marcin Andrychowicz, Misha Denil, Sergio Gómez Colmenarejo, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3988–3996, 2016.

[2] Christopher G. Atkeson and Juan Carlos Santamaria. Comparison of direct and model-based reinforcement learning. *Proceedings - IEEE International Conference on Robotics and Automation*, 4:3557–3564, 1997.

[3] Suneel Belkhale, Rachel Li, Gregory Kahn, Rowan McAllister, Roberto Calandra, and Sergey Levine. Model-Based Meta-Reinforcement Learning for Flight with Suspended Payloads. *IEEE Robotics and Automation Letters*, 6(2):1471–1478, 2021.

[4] Rich Caruana, Lorien Pratt, and Sebastian Thrun. Multitask Learning *. 28:41–75, 1997.

[5] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, Freek Stulp, Sylvain Calinon, and Jean-Baptiste Mouret. A survey on policy search algorithms for learning robot controllers in a handful of trials. *IEEE Transactions on Robotics*, 36(2):328–347, jul 2018.

[6] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. *Advances in Neural Information Processing Systems*, 2018-December:4754–4765, may 2018.

[7] Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-Based Reinforcement Learning via Meta-Policy Optimization. sep 2018.

[8] Marc Peter Deisenroth and Carl Edward Rasmussen. PILCO: A model-based and data-efficient approach to policy search. *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 465–472, 2011.

[9] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL2: Fast Reinforcement Learning via Slow Reinforcement Learning. nov 2016.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *34th International Conference on Machine Learning, ICML 2017*, 3:1856–1868, 2017.

[11] Chelsea B Finn. Meta Learning Dissertation. pages 1–3, 6–8, 2018.

[12] Courville Aaron Goodfellow lan, Bengio Yoshua. Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville - Google Books, 2016.

[13] Rituraj Kaushik, Timothée Anne, and Jean-Baptiste Mouret. Fast Online Adaptation in Robotics through Meta-Learning Embeddings of Simulated Priors. *IEEE International Conference on Intelligent Robots and Systems*, pages 5269–5276, mar 2020.

[14] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-Ensemble Trust-Region Policy Optimization. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, feb 2018.

[15] Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware Dynamics Model for Generalization in Model-Based Reinforcement Learning. *37th International Conference on Machine Learning, ICML 2020*, PartF168147-8:5713–5722, may 2020.

[16] Yoonho Lee and Seungjin Choi. Gradient-based meta-learning with learned layerwise metric and subspace. *35th International Conference on Machine Learning, ICML 2018*, 7:4574–4586, 2018.

[17] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv*, pages 1–17, 2018.

[18] Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing, and Sergey Levine. Neural Network Dynamics for Model-Based Deep Reinforcement Learning with Model-Free Fine-Tuning. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 7579–7586, aug 2017.

[19] Alex Nichol, Joshua Achiam, and John Schulman. On First-Order Meta-Learning Algorithms. Technical report, 2018.

[20] Aravind Rajeswaran, Chelsea Finn, Sham Kakade, and Sergey Levine. Meta-learning with implicit gradients. *arXiv*, 2019.

[21] Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:9291–9301, mar 2019.

[22] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. Technical report, 2017.

[23] Steindór Sæmundsson, Katja Hofmann, and Marc Peter Deisenroth. Meta Reinforcement Learning with Latent Variable Gaussian Processes. *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, 2:642–652, mar 2018.

[24] Jurgen Schmidhuber. Evolutionary principles in self-referential learning. on learning now to learn: The meta-meta-meta...-hook. Diploma thesis, Technische Universitat Munchen, Germany, 14 May 1987.

[25] Peter Sterling and Simon Laughlin. *Principles of neural design*. MIT press, 2015.

[26] Haoxiang Wang, Han Zhao, and Bo Li. Bridging Multi-Task Learning and Meta-Learning: Towards Efficient Training and Effective Adaptation. page 139, 2021.

[27] Qi Wang and Herke van Hoof. Model-based Meta Reinforcement Learning using Graph Structured Surrogate Models. feb 2021.

[28] Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M Rehg, Byron Boots, and Evangelos A Theodorou. Information Theoretic MPC for Model-Based Reinforcement Learning.