Meta-Reinforcement Learning Robust to Distributional Shift via Model Identification and Experience Relabeling

Russell Mendonca [*1] **Xinyang Geng** [*1] **Chelsea Finn** [2] **Sergey Levine** [1]

1. Introduction

Meta-reinforcement learning (meta-RL) algorithms enable agents to perform new tasks by leveraging experience from previously seen related tasks (Duan et al., 2016; Wang et al., 2016; Finn et al., 2017), by extending the meta-learning framework (Schmidhuber, 1987; Thrun & Pratt, 1998; Naik & Mammone, 1992; Bengio et al., 1991). However, the performance of these methods depends crucially on how close the new tasks are to the meta-training task distribution. Meta-trained agents can adapt quickly to tasks that are similar to those seen during meta-training, but lose much of their benefit when adapting to tasks that are too far away from the meta-training set.

Many meta-RL methods either utilize a variant of model-agnostic meta-learning (MAML) and adapt to new tasks with gradient descent (Finn et al., 2017; Rothfuss et al., 2018; Zintgraf et al., 2018; Rusu et al., 2018; Liu et al., 2019; Gupta et al., 2018; Sung et al., 2017; Houthooft et al., 2018), or use an encoder-based formulation that adapt by encoding experience with recurrent models (Duan et al., 2016; Wang et al., 2016; Fakoor et al., 2020; Stadie et al., 2018), attention mechanisms (Mishra et al., 2017) or variational inference (Rakelly et al., 2019). The latter class of methods generally struggle when adapting to out-of-distribution tasks, because the adaptation procedure is entirely learned and carries no performance guarantees with out-of-distribution inputs (as with any learned model). Methods that utilize gradient-based adaptation have the potential of handling out-of-distribution tasks more effectively, since gradient descent corresponds to a well-defined and consistent learning process that has a guarantee of improvement regardless of the task (Finn & Levine, 2018). However, in the RL setting, these methods (Finn et al., 2017; Rothfuss et al., 2018) utilize on-policy policy gradient methods for meta-training, which require a very large number of samples during meta-training (Rakelly et al., 2019).

[*]Equal contribution [1]University of California, Berkeley [2]Stanford University. Correspondence to: Russell Mendonca <russellm@berkeley.edu>.

Accepted at Workshop on Inductive Biases, Invariances and Generalization, International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

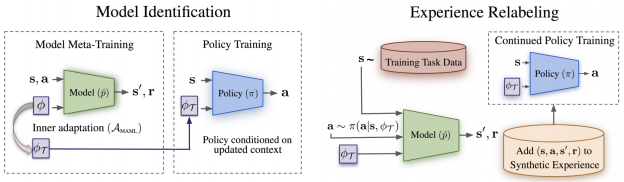

Figure 1. Overview of our approach. The model context variable (ϕ) is adapted using gradient descent, and the adapted context variable ($\phi_{\mathcal{T}}$) is fed to the policy alongside state so the policy can be trained with standard RL (Model Identification). The adapted model is used to relabel the data from other tasks by predicting next state and reward, generating synthetic experience to continue improving the policy (Experience Relabeling).

In this paper, we aim to develop a meta-RL algorithm that can both adapt effectively to out-of-distribution tasks and be meta-trained efficiently via off-policy value-based algorithms. We propose to leverage a simple insight: *dynamics and reward models* can be adapted consistently, using gradient based update rules with off-policy data, even if policies and value functions cannot. These models can then be used to train policies for out-of-distribution tasks without using meta-RL at all, by generating synthetic experience for the new tasks. Based on this observation, we propose **model identification and experience relabeling** (MIER), a meta-RL algorithm that makes use of two independent novel concepts: **model identification** and **experience relabeling**. Model identification refers to the process of identifying a particular task from a distribution of tasks, which requires determining its transition dynamics and reward function. We use a gradient-based *supervised* meta-learning method to learn a dynamics and reward model and a (latent) *model context variable* such that the model quickly adapts to new tasks after a few steps of gradient descent on the context variable. The context variable must contain sufficient information about the task to accurately predict dynamics and rewards. The policy can then be conditioned on this context (Schaul et al., 2015; Kaelbling, 1993) and therefore does *not* need to be meta-trained or adapted. Hence it can be learned with any standard RL algorithm, avoiding the complexity of meta-reinforcement learning. We illustrate the model identification process in the left part of Figure 1.

When adapting to out-of-distribution tasks at meta-test time,

the adapted context variable may itself be out of distribution, and the context-conditioned policy might perform poorly. However, since MIER adapts the model with gradient descent, we can continue to improve the model using more gradient steps. To continue improving the policy, we leverage all data collected from other tasks during meta-training, by using the learned model to *relabel* the next state and reward on every previously seen transition, obtaining synthetic data to continue training the policy. We call this process, shown in the right part of Figure 1, **experience relabeling**. This enables MIER to adapt to tasks outside of the meta-training distribution, outperforming prior meta-reinforcement learning methods in this setting.

2. Meta Training with Model Identification

We further discuss how we can reformulate the meta-RL problem into a model identification problem, where we train a fast-adapting model to rapidly identify the transition dynamics and reward function for a new task. The supervised meta-learning setting consists of a distribution of tasks $\rho(\mathcal{T})$ where each task $\mathcal{T}$ is a pair of input and output random variables $(X_\mathcal{T}, Y_\mathcal{T})$. Given a small dataset $\mathcal{D}^{(\mathcal{T})}_{adapt}$ sampled from a specific task $\mathcal{T}$, the objective is to build a model that performs well on the evaluation data $\mathcal{D}^{(\mathcal{T})}_{eval}$ sampled from the same task. Model agnostic meta-learning (Finn et al., 2017) is an approach which solves the supervised meta-learning problem by optimizing the loss of the model after few steps of gradient descent on data from the new task.

Unlike the standard supervised MAML formulation, we condition our model on a latent context vector, and we only change the context vector when adapting to new tasks. Since all task-specific information is thus encapsulated in the context vector, conditioning the policy on this context should provide it with sufficient information to solve the task. This architecture is illustrated in the left part of Figure 1. We denote the model as $\hat{p}(\mathbf{s}', \mathbf{r}|\mathbf{s}, \mathbf{a}; \theta, \phi)$, where θ is the neural network parameters and ϕ is the latent context vector that is passed in as input to the network. One step of gradient adaptation, which we call $\mathcal{A}_{\text{MAML}}\left(\theta, \phi, \mathcal{D}^{(\mathcal{T})}_{adapt}\right)$, can be written as follows:

$$\phi_\mathcal{T} = \phi - \alpha \nabla_\phi E_{(\mathbf{s},\mathbf{a},\mathbf{s}',\mathbf{r})\sim\mathcal{D}^{(\mathcal{T})}_{adapt}} [-\log \hat{p}\left(\mathbf{s}', \mathbf{r}|\mathbf{s}, \mathbf{a}; \theta, \phi\right)]$$

We use the log likelihood as our objective for the probabilistic model. We then evaluate the model using the adapted context vector $\phi_\mathcal{T}$, and minimize its loss on the evaluation dataset to learn the model. Specifically, we minimize the model meta-loss function $J_{\hat{p}}(\theta, \phi, \mathcal{D}^{(\mathcal{T})}_{adapt}, \mathcal{D}^{(\mathcal{T})}_{eval})$ to obtain the optimal parameter θ and context vector initialization ϕ:

$$\arg\min_{\theta,\phi} J_{\hat{p}}\left(\theta, \phi, \mathcal{D}^{(\mathcal{T})}_{adapt}, \mathcal{D}^{(\mathcal{T})}_{eval}\right)$$

$$= \arg\min_{\theta,\phi} E_{(\mathbf{s},\mathbf{a},\mathbf{s}',\mathbf{r})\sim\mathcal{D}^{(\mathcal{T})}_{eval}} [-\log \hat{p}\left(\mathbf{s}', \mathbf{r}|\mathbf{s}, \mathbf{a}; \theta, \phi_\mathcal{T}\right)]$$

Given the latent context variable from the adapted model $\phi_\mathcal{T}$, the meta-RL problem can be effectively reduced to a standard RL problem, as the task specific information has been encoded in the context variable. We can therefore apply any standard model-free RL algorithm to obtain a policy, as long as we condition the policy on the latent context variable (Schaul et al., 2015; Kaelbling, 1993). In our implementation, we utilize the soft actor-critic (SAC) algorithm (Haarnoja et al., 2018), though any efficient model-free RL method could be used.

Algorithm 1 Model Identification Meta-Training

Input: task distribution $\rho(\mathcal{T})$, training steps N, learning rate α
Output: policy parameter ψ, model parameter θ, model context ϕ

Randomly initialize ψ, θ, ϕ
Initialize multitask replay buffer $\mathcal{R}(\mathcal{T}) \leftarrow \emptyset$
while *θ, ϕ, ψ not converged* **do**
  Sample task $\mathcal{T} \sim \rho(\mathcal{T})$
  Collect $\mathcal{D}^{(\mathcal{T})}_{adapt}$ using π_ψ and ϕ
  Compute $\phi_\mathcal{T} = \mathcal{A}_{\text{MAML}}(\theta, \phi, \mathcal{D}^{(\mathcal{T})}_{adapt})$
  Collect $\mathcal{D}^{(\mathcal{T})}_{eval}$ using π and $\phi_\mathcal{T}$
  $\mathcal{R}(\mathcal{T}) \leftarrow \mathcal{R}(\mathcal{T}) \cup \mathcal{D}^{(\mathcal{T})}_{adapt} \cup \mathcal{D}^{(\mathcal{T})}_{eval}$
  for i = 1 *to* N **do**
    Sample task $\mathcal{T} \sim \mathcal{R}$
    Sample $\mathcal{D}^{(\mathcal{T})}_{adapt}, \mathcal{D}^{(\mathcal{T})}_{eval} \sim \mathcal{R}(\mathcal{T})$
    Update $\theta \leftarrow \theta - \alpha \nabla_\theta J_{\hat{p}}(\theta, \phi, \mathcal{D}^{(\mathcal{T})}_{adapt}, \mathcal{D}^{(\mathcal{T})}_{eval})$
    Update $\psi \leftarrow \psi - \alpha \nabla_\psi J_\pi(\psi, \mathcal{D}^{(\mathcal{T})}_{eval}, \phi_\mathcal{T})$
  end
end

3. Improving Out-of-Distribution Performance by Experience Relabeling

At meta-test time, when our method must adapt to a new unseen task $\mathcal{T}$, it will first sample a small batch of data and obtain the latent context $\phi_\mathcal{T}$ by running the gradient descent adaptation process on the context variable, using the model identification process introduced in the previous section. While our model identification method is already a complete meta-RL algorithm, it has no guarantees of consistency. That is, it might not be able to adapt to out-of-distribution tasks, even with large amounts of data: although the gradient descent adaptation process for the model is consistent and will continue to improve, the context variable $\phi_\mathcal{T}$ produced by this adaptation may still be out-of-distribution for the policy when adapting to an out-of-distribution task. However, with an improved model, we can continue to train the policy with standard off-policy RL, by generating synthetic data using the model. In practice we adapt the model for as many gradient steps as necessary, and then use this model to generate synthetic transitions using states from all previously

seen meta-training tasks, with new successor states and rewards. We call this process experience relabeling. Since the model is adapted via gradient descent, it is guaranteed to eventually converge to a local optimum for any new task, even a task that is outside the meta-training distribution. We illustrate the experience relabeling process in the right part of Figure 1, and provide pseudo-code in Algorithm 2.

When using data generated from a learned model to train a policy, the model's predicted trajectory often diverges from the real dynamics after a large number of time steps, due to accumulated error (Janner et al., 2019). We can mitigate this issue in the meta-RL case by leveraging all of the data from other tasks that was available during meta-training. Although the new task is previously unseen, the other training tasks share the same state space and action space, and so we can leverage the large set of diverse transitions collected from these tasks. Using the adapted model and policy, we can *relabel* these transitions, denoted $(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{r})$, by sampling new actions with our adapted policy, and by sampling next states and rewards from the adapted model. The relabeling process can be written as: $\mathbf{Relabel}(\mathcal{D}, \theta, \phi_{\mathcal{T}}) = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{r}) | \mathbf{s} \in \mathcal{D}; \mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s}, \phi_{\mathcal{T}}), (\mathbf{s}', \mathbf{r}) \sim \hat{p}(\mathbf{s}', \mathbf{r}|\mathbf{s}, \mathbf{a}; \theta, \phi_{\mathcal{T}})\}$.

Algorithm 2 Experience Relabeling Adaptation

Input: test task $\hat{\mathcal{T}}$, multitask replay buffer $\mathcal{R}(\mathcal{T})$, Adaptation steps for context N_{fast}, Training steps for policy N_p, Training steps for model N_m
Output: policy parameter ψ

Collect $\mathcal{D}^{(\hat{\mathcal{T}})}_{adapt}$ from $\hat{\mathcal{T}}$ using π_ψ and ϕ
for i = 1 *to* N_{fast} **do**
  Update $\phi_{\mathcal{T}}$ using $\mathcal{A}_{\text{MAML}}$
end
while ψ *not converged* **do**
  for i = 1 *to* N_p **do**
    Sample $\mathcal{T} \sim \mathcal{R}$ and $\mathcal{D}^{(\mathcal{T})} \sim \mathcal{R}(\mathcal{T})$
    $\hat{\mathcal{D}}^{(\hat{\mathcal{T}})} \leftarrow \mathbf{Relabel}(\mathcal{D}^{(\mathcal{T})}, \theta, \phi_{\hat{\mathcal{T}}})$
    Train policy $\psi \leftarrow \psi - \alpha \nabla_\psi J_\pi(\psi, \hat{\mathcal{D}}^{(\hat{\mathcal{T}})}, \phi_{\mathcal{T}})$
  end
end

4. Experimental Evaluation

We aim to answer the following questions in our experiments: **(1)** Can MIER meta-train efficiently on standard meta-RL benchmarks, with meta-training sample efficiency that is competitive with state-of-the-art methods? **(2)** How does MIER compare to prior meta-learning approaches for extrapolation to meta-test tasks with out-of-distribution (a) reward functions and (b) dynamics? **(3)** How important is experience relabeling in leveraging the model to train effective policies for out-of-distribution tasks?

4.1. Meta-Training on Meta-RL Benchmarks

We first evaluate MIER on standard meta-RL benchmarks, which were used in prior work (Finn et al., 2017; Rakelly et al., 2019; Fakoor et al., 2020). Results are shown in Figure 2. We compare to PEARL (Rakelly et al., 2019), which uses an off-policy encoder-based method, but without consistent adaptation, meta Q-learning (MQL) (Fakoor et al., 2020), which also uses an encoder, MAML (Finn et al., 2017) and PRoMP (Rothfuss et al., 2018), which use MAML-based adaptation with on-policy policy gradients, and RL2 (Duan et al., 2016), which uses an on-policy algorithm with an encoder. We plot the meta-test performance after adaptation (on **in-distribution** tasks) against the number of meta-training samples, averaged across 3 random seeds. On these standard tasks, we run a variant of our full method which we call MIER-wR (MIER without **experience relabeling**), which achieves performance that is comparable to or better than the best prior methods, indicating that our **model identification** method provides a viable meta-learning strategy that compares favorably to state-of-the-art methods.

4.2. Adaptation to Out-of-Distribution Tasks

Next, we compare the performance of our full method (MIER), and MIER without experience relabeling (MIER-wR), to prior meta-learning methods for adaptation to out-of-distribution tasks, both on tasks with varying reward functions and tasks with varying dynamics. All algorithms are meta-trained with the same number of samples (2.5M for Ant Negated Joints, and 1.5M for all other domains) before evaluation.

Extrapolation over reward functions: To evaluate extrapolation to out-of-distribution rewards, we first test on the half cheetah velocity extrapolation environments introduced by Fakoor et al. (2020).[1] Half-Cheetah-Vel-Medium has training tasks where the cheetah is required to run at target speeds ranging from 0 to 2.5 m/s, while Half-Cheetah-Hard has training tasks where the target speeds are sampled from 0 to 1.5 m/s, as depicted in Figure 4(a). In both settings, the test set has target speeds sampled from 2.5 to 3 m/s. In Figure 3, we see that our method matches MQL on the easier Half-Cheetah-Vel-Medium environment, and outperforms all prior methods including MQL on the Half-Cheetah-Vel-Hard setting, where extrapolation is more difficult. Furthermore we see that experience relabeling improves performance for our method for both settings.

We also evaluate reward function extrapolation for an Ant that needs to move in different directions, with the training

[1] Since we do not have access to the code used by Fakoor et al. (2020), quantitative results for the easier cheetah tasks are taken from their paper, but we cannot evaluate MQL on other more challenging tasks.

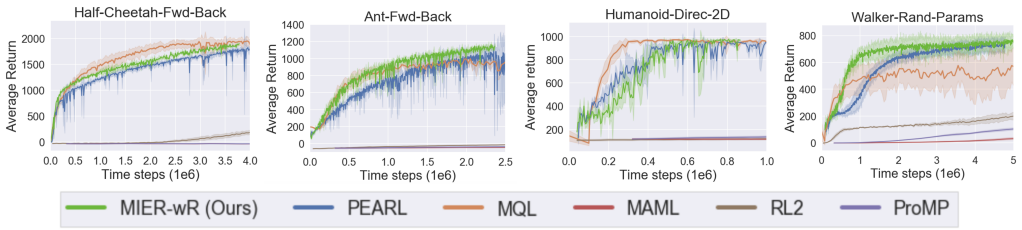

Figure 2. Performance on standard meta-RL benchmarks. Return is evaluated over the course of the *meta-training* process on meta-test tasks that are **in-distribution**.

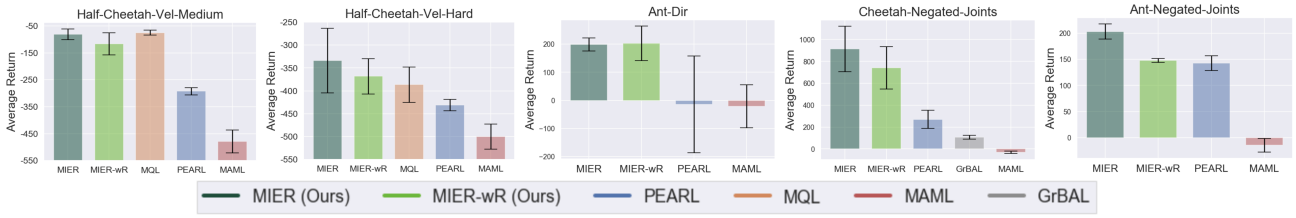

Figure 3. Performance on out-of-distribution tasks. All algorithms are meta-trained with the same amount of data, and then evaluated on out-of-distribution tasks. Cheetah-Velocity and Ant-Direction environments have varying reward functions, while Cheetah-Negated-Joints and Ant-Negated-Joints have different dynamics.

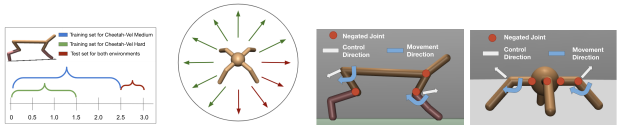

Figure 4. Illustration of out-of-distribution adaptation tasks: (a) Cheetah-Velocity Medium (target velocity training set in blue, test set in red) and Cheetah-Velocity Hard (target velocity training set in green, test set in red), (b) Ant Direction (target direction training tasks in green, test tasks in red), (c) Cheetah Negated Joints and (d) Ant Negated Joints.

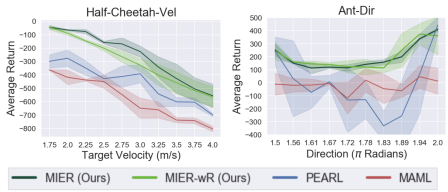

Figure 5. Performance evaluated on validation tasks of varying difficulty. For Cheetah Velocity, the training distribution consists of target speeds from 0 to 1.5 m/s, and so tasks become harder left to right along the x axis. Ant Direction consists of training tasks ranging from 0 to 1.5 π radians, so the hardest tasks are in the middle.

set comprising directions sampled from 3 quarters of a circle, and the test set containing tasks from the last quadrant, as shown in Figure 4(b). We see in Figure 3 that our method outperforms prior algorithms by a large margin in this setting. We provide a more fine-grained analysis of adaptation performance on different tasks in the test set in Figure 5. We see that while the performance of all methods degrades as validation tasks get farther away from the training distribution, MIER and MIER-wR perform consistently better than MAML and PEARL.

Extrapolation over dynamics: To study adaptation to out-of-distribution dynamics, we constructed variants of the HalfCheetah and Ant environments where we randomly negate the control of randomly selected groups of joints as shown in Figures 4(c) and 4(d). During meta-training, we never negate the last joint, such that we can construct out-of-distribution tasks by negating this last joint, together with a randomly chosen subset of the others. For the HalfCheetah, we negate 3 joints at a time from among the first 5 during meta-training, and always negate the 6th joint (together with a random subset of 2 of the other 5) for testing, such that there are 10 meta-training tasks and 10 out-of-distribution evaluation tasks. For the Ant, we negate 4 joints from among the first 7 during meta-training, and always negate the 8th (together with a random subset of 3 of the other 7) for evaluation, resulting in 35 meta-training tasks and 35 evaluation tasks, out of which we randomly select 15.

In addition to PEARL and MAML, we compare against GrBAL (Nagabandi et al., 2018), a model based meta-RL method. We note that we could not evaluate GrBAL on the reward extrapolation tasks, since it requires the analytic reward function to be known during planning, but we can compare to this method under varying dynamics. From Figure 3, we see that performance on Cheetah-Negated-Joints with just context adaptation (MIER-wR) is substantially better than PEARL and MAML and GrBAL, and there is further improvement by using the model for relabeling (MIER). On the more challenging Ant-Negated-Joints environment, MIER-wR shows similar performance to PEARL, and leveraging the model for relabeling again leads to better performance for MIER.

References

Bengio, Y., Bengio, S., and Cloutier, J. Learning a synaptic learning rule. In *IJCNN-91-Seattle International Joint Conference on Neural Networks*, volume ii, pp. 969 vol.2–, 1991.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Fakoor, R., Chaudhari, P., Soatto, S., and Smola, A. J. Meta-q-learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJeD3CEFPH.

Finn, C. and Levine, S. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HyjC5yWCW.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Gupta, A., Mendonca, R., Liu, Y., Abbeel, P., and Levine, S. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems*, pp. 5302–5311, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Houthooft, R., Chen, R. Y., Isola, P., Stadie, B. C., Wolski, F., Ho, J., and Abbeel, P. Evolved policy gradients. *arXiv preprint arXiv:1802.04821*, 2018.

Janner, M., Fu, J., Zhang, M., and Levine, S. When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*, 2019.

Kaelbling, L. P. Learning to achieve goals. In *IJCAI*, 1993.

Liu, H., Socher, R., and Xiong, C. Taming maml: Efficient unbiased meta-reinforcement learning. In *International Conference on Machine Learning*, pp. 4061–4071, 2019.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

Naik, D. K. and Mammone, R. Meta-neural networks that learn by learning. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, volume 1, pp. 437–442. IEEE, 1992.

Rakelly, K., Zhou, A., Quillen, D., Finn, C., and Levine, S. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv preprint arXiv:1903.08254*, 2019.

Rothfuss, J., Lee, D., Clavera, I., Asfour, T., and Abbeel, P. Promp: Proximal meta-policy search. *arXiv preprint arXiv:1810.06784*, 2018.

Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., and Hadsell, R. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1312–1320. JMLR.org, 2015.

Schmidhuber, J. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

Stadie, B. C., Yang, G., Houthooft, R., Chen, X., Duan, Y., Wu, Y., Abbeel, P., and Sutskever, I. Some considerations on learning to explore via meta-reinforcement learning. *arXiv preprint arXiv:1803.01118*, 2018.

Sung, F., Zhang, L., Xiang, T., Hospedales, T., and Yang, Y. Learning to learn: Meta-critic networks for sample efficient learning. *arXiv preprint arXiv:1706.09529*, 2017.

Thrun, S. and Pratt, L. Learning to learn: Introduction and overview. In *Learning to learn*, pp. 3–17. Springer, 1998.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. M. Learning to reinforcement learn. *ArXiv*, abs/1611.05763, 2016.

Zintgraf, L. M., Shiarlis, K., Kurin, V., Hofmann, K., and Whiteson, S. Caml: Fast context adaptation via meta-learning. *arXiv preprint arXiv:1810.03642*, 2018.