Combining Model-Based and Model-Free Meta-Reinforcement Learning

Jerry Zhenbang Tan
Department of Computer Science
Stanford University
ztan035@stanford.edu

Andrew Dworschak
Department of Computer Science
Stanford University
andrew8k@stanford.edu

Abstract

Meta-reinforcement learning (RL) algorithms seek to efficiently learn by leveraging shared structure between different tasks and environments. RL systems can make decisions in one of two ways. In the model-based approach, a system uses a predictive model of the world to answer questions of the form "what will happen if I do x?" to choose the best action. In the alternative model-free approach, the modeling step is bypassed altogether in favor of learning a control policy directly. These two methods have their associated advantages and disadvantages. Model-free approaches are traditionally shown [Pong et al., 2020] to require a large number of samples to produce an effective policy, hindering their real-world applicability, whereas model-based approaches are more sample efficient but often do not achieve the asymptotic performance [Pong et al., 2020] of model-free approaches due to bias in the learned models. This problem with model-based learning is exacerbated in meta-reinforcement learning settings where the number of samples in the support set may be insufficient for the model to fully explore the system dynamics, especially in settings with either sparse or zero exploration rewards.

In this project propose an algorithm that aims to retain the benefits of both model-based and model-free RL methods. We focus on instruction-based meta-reinforcement learning settings (IMRL) as in Liu et al. [2020]. To our best knowledge, there has been no prior work that combines model-based and model-free approaches in this setting.

We experiment with three algorithms. The first algorithm combines the MAML and DREAM algorithms using a learned gradient embedder to optimize the policy during exploration. The second algorithm jointly minimizes the loss of the dynamics model and the state representation. The third combines planning with both model-based and model-free approaches.

We evaluate our methods on the *distracting bus and map* [Liu et al., 2020] problem, also known as *grid-world with busses*. Using the DREAM algorithm as a baseline, we observe similar or poorer performance with algorithms two and three. With the first gradient embedding algorithm, however, we observe a significant meta-test time improvement when compared to the standard DREAM, though the meta-training performance is similar between the algorithms. We believe that these results demonstrate the merit in combining model-based and model-free approaches.

CS330: Deep Multi-Task and Meta-Learning, Fall 2020, Stanford University, CA.

1 Introduction

Human-level intelligence is able to acquire new skills with limited experience utilizing past knowledge and memories. However, most successful reinforcement learning (RL) agents to-date still require large amounts of data to achieve human-level performance, often orders of magnitude more. The meta-reinforcement learning (meta-RL) problem definition forces the practitioner to address this problem by extracting prior knowledge from a set of tasks that are different than the testing environment. However, the number of tasks required for current algorithms to learn an effective prior still poses a significant barrier to real-world applications of reinforcement learning. One possible explanation for the massive amount of data required is that unlike humans, the architecture of current algorithms prevent them from efficiently integrating knowledge of the environment dynamics into their decisions.

The relative efficiency of model-based RL approaches compared to model-free approaches lends credibility to this hypothesis: model-based approaches obtain a large amount of information from every sample since they can use each sample to learn the system dynamics and not just the most efficient policy. However, this comes at the cost of larger asymptotic bias: when the system dynamics cannot be learned perfectly, as is the case for most complex problems, the output policy can be highly sub-optimal.

Another possible explanation of the sample inefficiency of model-free methods [Mao et al., 2018] is the entangled learning of the dynamics and the task (or in meta-RL settings, the dynamics distribution and the task distribution). For example, while DREAM [Liu et al., 2020] disentangles the objective of exploration and instruction and thus reduces the amount of meta-training data required to train an effective agent, its instruction policy still learns the knowledge representation of the meta-training problems in an entangled way; for the instruction agent to be able to solve a specific environment, the encoder need to produce an useful embedding of the environment that informs the instruction agents about the required information (e.g. transition dynamics), whereas producing a useful embedding requires the agent to know about what is important to solve the environment. This entanglement, which seems to be innate in model-free approaches, may slow down their learning process.

Can we achieve an even higher sample efficiency by combining it with a model-based approach, even when the model is not necessarily accurate? In this project, we explored how to leverage the rich information in state transitions to learn very efficiently, while still attaining asymptotic performance that exceeds that of direct model-based RL method.

2 Related work

Several methods [Pong et al., 2020, Chebotar et al., 2017], has been proposed to study how to bridge model-free and model-based approaches, achieving a higher efficiency as model-based approaches are while maintaining the performance of model-free approach asymptotically. However, to our best knowledge, there has not been any work specifically targeted at the IRML setting, where purely model-based approaches typically does not perform well.

Mao et al. [2018] identifies that in RL, the entanglement of knowledge of transition dynamics makes transfer learning difficult. In the meta-RL setting, this translates to a more stringent need of shared structures between different problems.

Nagabandi et al. [2019] uses meta-learning to train a dynamics model prior that is able to rapidly adapted to a new environment given recent samples, by using MAML, [Finn et al., 2017] which explicitly trains the parameters of the model such that a small number of gradient steps with a small amount of training data from a new task will produce good generalization performance on that task. On the perspective Bayesian statistics [Grant et al., 2018], the meta-parameters correspond to a priori of the task distribution and the parameters after the inner-loop adaptation correspond to the task specific parameters (posterior). An alternative interpretation is that the inner-loop gradients provides rich and succinct information about the task. In an online reinforcement learning setting, the inner-loop gradients may correspond to how the environment changes. This view inspires our gradient-based embedder approach.

While MAML allows rapid adaptation, and using model predictive control (MPC) to extract a policy from the dynamics model might compensates for model inaccuracy to some extent, when the dynamics or the reward is too complex (i.e. presence of distracting factors) for the algorithm

to learn an accurate model, such as IMRL where exploration reveals no information about rewards, the resulting policy can still be arbitrarily sub-optimal, especially compared to that of asymptotic performances of model-free methods [Pong et al., 2020].

Liu et al. [2020] proposed an approach to disentangle the objectives of the exploration and the instruction agent. This is achieved by extracting from the problem ID, which is a one-hot vector, an embedding that contains useful information to solve the task. The exploration agent's goal is then to recover this embedding. While this approach boost performance in instruction-based settings, it requires a large number of samples to train an effective policy due to the use of model-free policy. In addition, the problem IDs contains no information about the dynamics, but the useful information in the embedding to be extracted must be learnt by the agents. This does not utilize the information in the transition tuples about the problem.

Cappart et al. [2020] proposed an approach to combine model-based and model-free methods by learning a shared representation on the state-embedding. The dynamics model acts as an regularizer on the state embedder to produce a more interpretable latent state embedding of the observation space. It was empirically shown to achieve a performance better than both individual approaches and excels in transfer learning and multi-task learning settings (where a single model can solve all the tasks). We wish to evaluate its performance in the IMRL settings, and explore its compatibility with MAML.

3 Methods

3.1 Gradient Extraction

MAML enables adaptation to a new task by applying a small number of gradient steps on a small number of samples. From the perspective of task distributions, the inner-loop gradients contains rich information about how the task differs from the prior (the meta-parameters). We used this insight to propose a feature extractor/classifier on the extracted inner-loop gradients to infer relevant dynamics information, in addition to the encoder/decoder.

During meta-training, we adapt the model on the exploration episode and minimizes its loss on predicting the execution episodes. This forces the model to learn from the exploration transitions the necessary information to execute the task, since learning irrelevant information (e.g. distracting dynamics) does not contribute to minimizing this meta-training test loss.

However, one underlying concern of this approach is that, finding a good interpreter of the gradients might be as difficult as learning the necessary information about the dynamics. Conversely, if we are able to learn a good gradient embedder, then the algorithm might generalize better to unseen environments/tasks by the virtue of the inductive bias from the adaptation.

Potentially, we could replace both the encoder and the decoder completely with the gradient embedder. However, in practice, this introduces instability, due to the new chicken-and-egg issue discovered in the experiment section.

3.2 Shared Representations

In DREAM, the state embedders are learnt as a part of the policies. This dependency on the policies not only makes it difficult to interpret the states, but might also limit its generalization to unseen states, since the learnt embeddings are likely to be unstructured [Cappart et al., 2020]. Minimizing the loss of the dynamics model jointly with the loss of the state embedder results in an humanly interpretable latent space such that linear interpolation between different states are remains meaningful [Cappart et al., 2020], and thus can be seen as a form of regularization to improve performance on unseen states. This is also very important to bridge model-free and model-based approaches, since model inaccuracy might result in predicting a next state that is unseen by the Q-agents.

Since minimizing the losses jointly can result in the state embedder mapping all states to a constant, we additionally introduce an disambiguation loss:

$$L_{d1} = exp(-C||e(s_1) - e(s_2)||_2)$$

Where e is the state embedder and s_1 and s_2 are randomly sampled states from the buffer.

3.3 Shared Representations + Value Prediction (CRAR)

With the shared state representation, we are able to use both the dynamics model and the Q-policy to infer the optimal action at each time step in a way similar to Oh et al. [2017]. The estimated return is recursively defined as:

$$\hat{Q}^d(s,a) = \begin{cases} Q(s,a) \text{ if } d = 0 \\ r + \gamma \operatorname{argmax}_a Q^{d-1}(s',a) \end{cases}$$

Where d is the depth of estimation and r is the (estimated), and the Q-networks will be trained identically to that in DREAM. Model will learn to predict the instruction transition tuples, conditioned on the problem dynamics information (given by either the encoder or the decoder). Then we select the action with the highest $\hat{Q}$ value.

Since the model only plans for a short horizon, the model bias affects this approach much less than purely model-based approaches [Cappart et al., 2020], thus is able to achieve a similar asymptotic performance to model-free methods.

4 Experiments

To evaluate our approaches, we used the *distracting bus and map* grid world environment used in Liu et al. [2020]. It is a IMRL problem where exploration is reward-free, and the dynamics contain distracting information that is not relevant to the task but makes it difficult to learn an accurate purely model-based approach. We trained all algorithms with 20000 runs.

4.1 Gradient Embedder

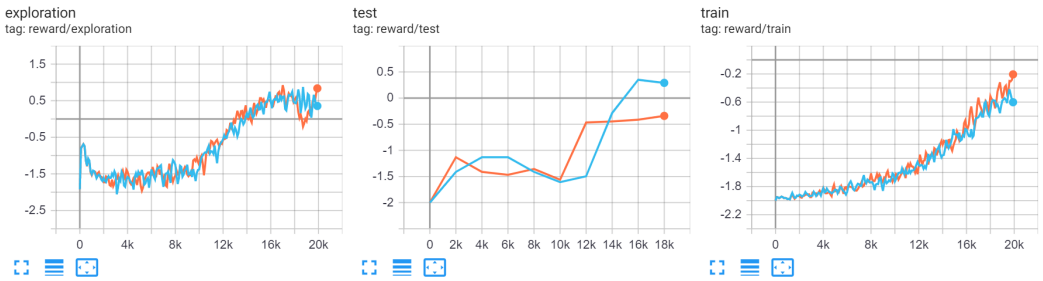


Figure 1: Rewards of DREAM (orange) versus our gradient embedder approach (blue)

To our surprise, the meta-training rewards of our gradient-based approach is similar to that of the original DREAM; nonetheless, the meta-testing performance is consistently improved by a significant margin at the later stages (Figure 1) of the training process, after when the exploration policy starts to learn. After a closer inspection, we hypothesized that our method to train the dynamics model is hindered by a chicken-and-egg problem. Since we adapt on the exploration transitions and minimize loss on the execution transition tuples, learning a meaningful dynamics model would require both of the execution and exploration policy to be to some extend capable of completing their objectives. If the exploration policy is ineffective, the transition tuples might reveal few information about the necessary dynamics to complete the execution objective and thus adapting on the exploration transitions might not produce informative gradients.

On the other hand, if the execution policy is ineffective (e.g. the execution policy never takes the bus/only performs actions that are predictable without knowledge of the dynamics), then the dynamics model might no longer need to adapt to the exploration transitions, resulting in the meta-memorization problem [Yin et al., 2020], and thus the gradients produced are not informative. To address this issue in future, we might incorporate prioritized experience replay [Schaul et al., 2016] and focus on learning only the relevant transitions, so that the requirement of good exploration and execution policies can be relaxed.

4.2 Shared Representations

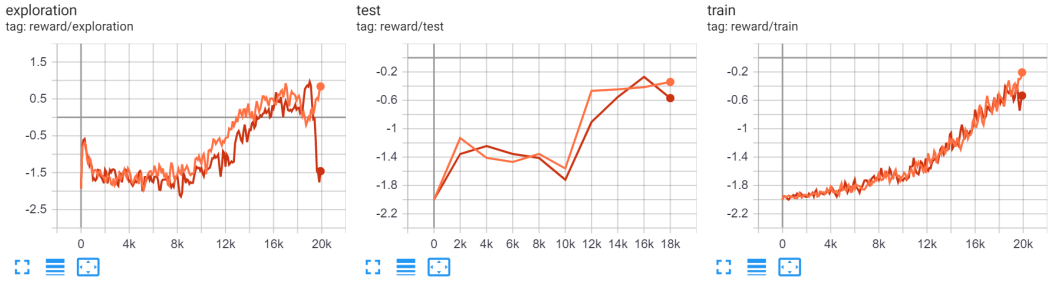


Figure 2: Rewards of DREAM (orange) versus shared state representations (red)

We can see from Figure 2 that using a shared state representation does not help improving the performance. In this simulator environment, the underlying state space is already low dimensional, and the states are fully observable, and all states during the meta-test time can be seen during meta-training. Therefore, it is expected that no performance gain will be achieve, and to evaluate the benefits of using a shared representation under a meta-training context, we might need to experiment on simulators that has higher dimensional inputs, such as raw images.

4.3 Shared Representations + Value Prediction (CRAR)

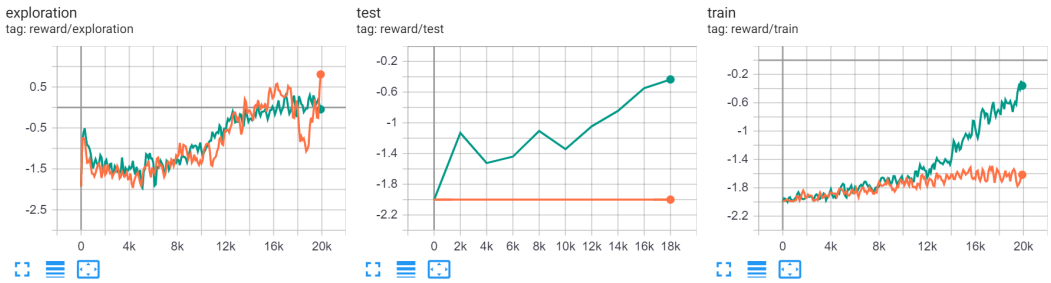
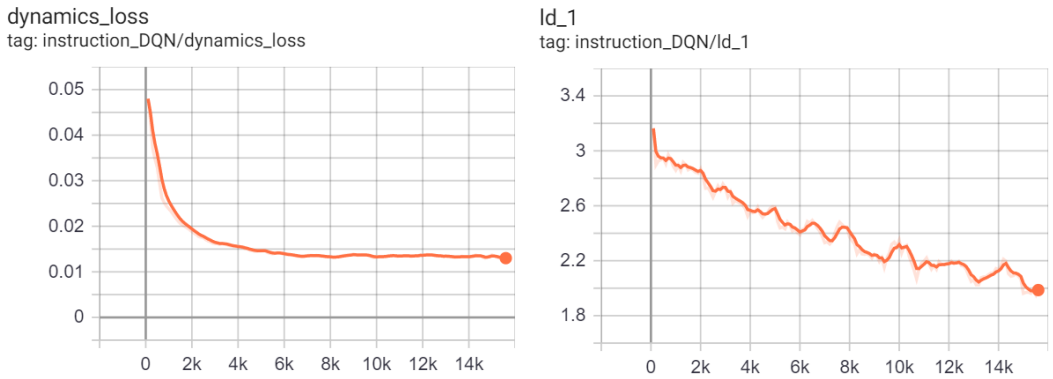


Figure 3: Rewards of DREAM (green) versus modified CRAR (orange)



(a) Transition Dynamics Model Loss

(b) LD1 Loss

Figure 4: Modified CRAR Dynamics Model Loss

Surprisingly, this approach actually worsens the performance. The dynamics model struggles to accurately infer the transition dynamics as shown in Figure 4, and the hybrid planning is not robust to this inaccuracy.

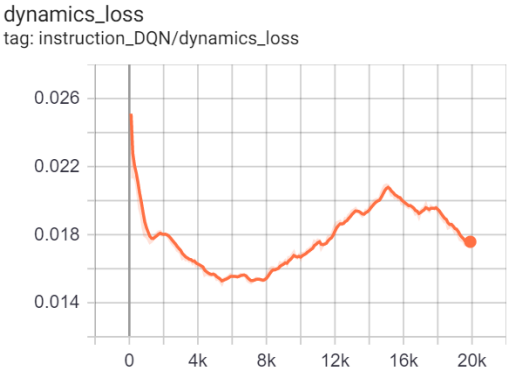


Figure 5: Modified CRAR Dynamics Model Loss (when only using shared representations). As shown in the loss plot, the dynamics model achieves a good loss not because the model is accurate, but because the a lot of the execution transition tuples do not require knowledge of the current dynamics (e.g. constantly choose to ride bus on a grid without bus) during the early phrase and the model. As as result, the model can achieve a small loss without learning the key dynamics required to solve the task.

In addition, we found that the model almost always produces the same sequence of actions (when not adhering to the epsilon-greedy strategy), resulting in staying in the same states, which might be the cause of the slower learning. One possible reason is that the dynamics model is accurate on the seen transition tuples, but has limited knowledge of subsequent transitions, and the initialized Q-values are inaccurate during earlier stages of the training. The low loss seems to be a form of 'confirmation bias'. To address this problem, we mix some exploration transition tuples into the trajectories when training the dynamics model.

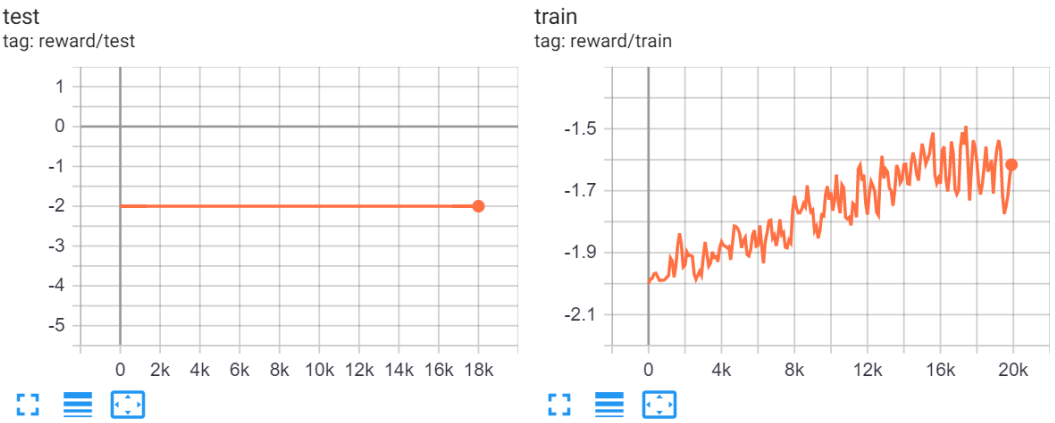
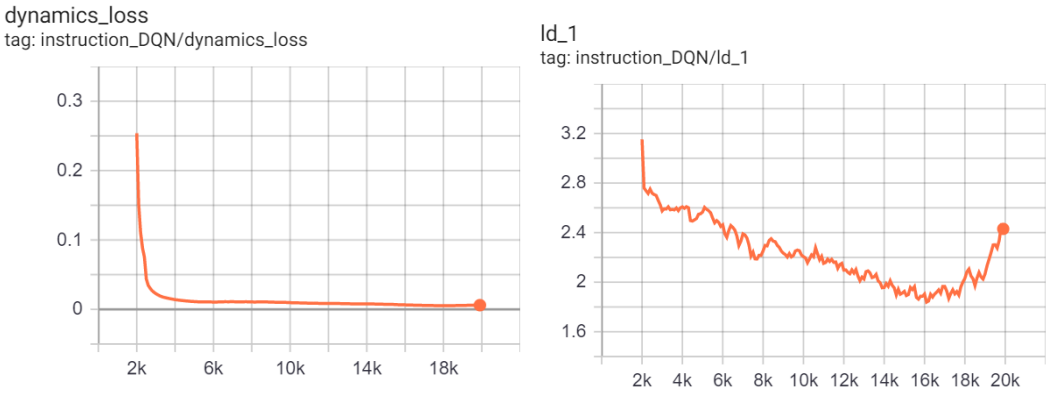


Figure 6: Modified CRAR Rewards (Mixing Exploration Transitions)

The dynamics model now achieves almost zero loss, but its performance still does not increase. The disambiguation loss is especially concerning; an increasing disambiguation loss might imply that the dynamics model is mapping different states to the same point. We hypothesized that either the model is not sufficiently deep to be able to represent all the transition dynamics conditioned on dynamics embeddings, or that the shared representation fails to produce a state embedding that is sufficiently robust to model error, such that the estimated Q-value obtain by the hybrid planning is vastly different from the true one.



(a) Transition Dynamics Model Loss (mixing transition tuples)

(b) LD1 Loss

Figure 7: Modified CRAR Dynamics Model Loss (Mixing Exploration Transitions)

5 Conclusions and future work

In this project, we have attempt to improve the meta-training efficiency of DREAM by combining model-based and model-free approaches. Our gradient-based classifier shows a promising improvement over the original DREAM approach. While bridging model-free and model-based approaches remains challenging, our project has shred light onto some of the critical issues to be addressed, including the 'reintroduced' chicken-and-egg problem when training the dynamics model.

For future work, theoretically we might achieve better results with the gradients embedder if the inner-loop gradients are more algorithmically stable [Liu et al., 2017], which might allow easier classification/interpretation of the environment dynamics. This might be achieved by using MAML++ [Antoniou et al., 2019]. In addition, using a feed-forward network approach might lose the structural information in the inner-loop gradients of the dynamics model. Rather than using an MLP, we might instead extract dynamics information with a graph neural network, such as applying graph convolution. Last but not least, we would like to experiment on different environments and repeat more training results to get a more accurate evaluation of our methods.

For the CRAR approach, we need to instead train a dynamics model on the raw states, and measure if it is the state representation or the planning process that resulted in the loss of performance.

References

Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml, 2019.

Quentin Cappart, Thierry Moisan, Louis-Martin Rousseau, Isabeau Prémont-Schwarz, and Andre Cire. Combining reinforcement learning and constraint programming for combinatorial optimization, 2020.

Yevgen Chebotar, Karol Hausman, Marvin Zhang, Gaurav Sukhatme, Stefan Schaal, and Sergey Levine. Combining model-based and model-free updates for trajectory-centric reinforcement learning, 2017.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes, 2018.

Evan Zheran Liu, Aditi Raghunathan, Percy Liang, and Chelsea Finn. Explore then execute: Adapting without rewards via factorized meta-reinforcement learning, 2020.

Tongliang Liu, Gábor Lugosi, Gergely Neu, and Dacheng Tao. Algorithmic stability and hypothesis complexity, 2017.

Jiayuan Mao, Honghua Dong, and Joseph J. Lim. Universal agent for disentangling environments and tasks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=B1mvVm-C-`.

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, 2019.

Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network, 2017.

Vitchyr Pong, Shixiang Gu, Murtaza Dalal, and Sergey Levine. Temporal difference models: Model-free deep rl for model-based control, 2020.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2016.

Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization, 2020.

Appendix

Our code is available at here (CRAR modified) and here (gradient-embedder)