CSCI 2951-F Final Project: Learning to Reinforcement Learn

By Vanya Cohen, Yury Gitman, Nishanth Prakash, and Liam Walsh

December 2017

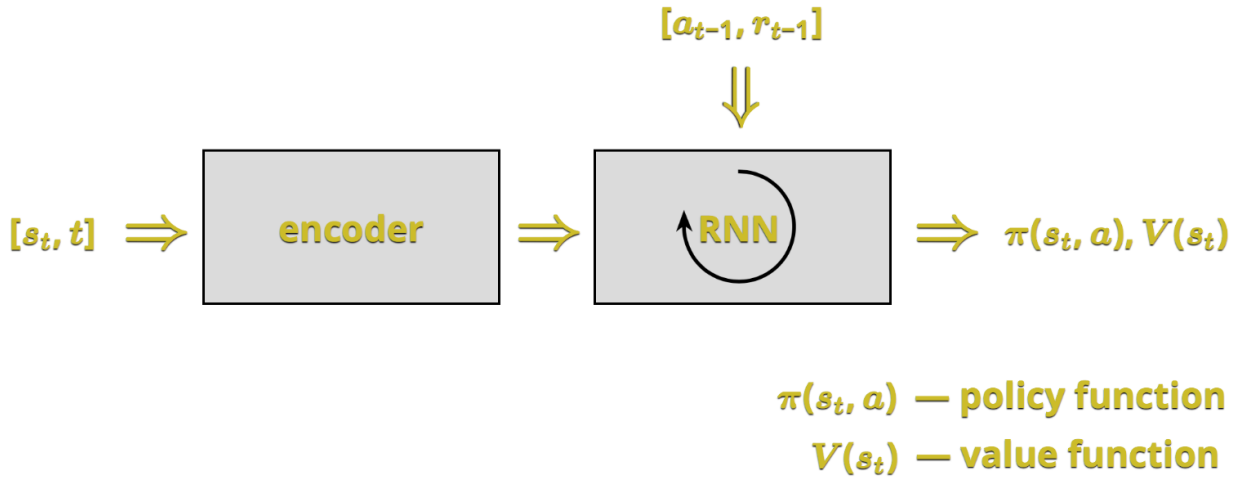
1 Introduction

Learning to Reinforcement Learn [Wang et al. 2016] proposes a framework called deep meta-reinforcement learning in attempting to address the poor performance of reinforcement learning algorithms when trained on small amounts of data and inability efficiently generalize to novel environments.

Typically, reinforcement learning algorithms require large amounts of data to train and perform well, and they generalize poorly across similar tasks, unable to capitalize on past learning to achieve high performance. Better reinforcement learning algorithms would train systems to quickly solve problems that are structurally similar to ones they have seen. For example, consider the task of reaching a goal within a maze. Such an RL agent should solve the maze, regardless of the goal's position, leveraging past learning of the maze's structure.

2 Background

Deep meta-reinforcement learning involves training an RNN using policy gradient methods (A2C) on particular instances of a general task. The RNN captures information related to the structure of the task (the meta) which is then exploited by the algorithm when it encounters on new instances of the task. Overall, deep meta-RL learns about the underlying distribution of specific instances of general tasks, and how to take advantage of that knowledge to choose better actions than a typical, non-meta RL agent would.



At each step of the RNN, the network observes the world state, the time-step, and the previous action and reward. Then it outputs policy and value functions for the given time-step.

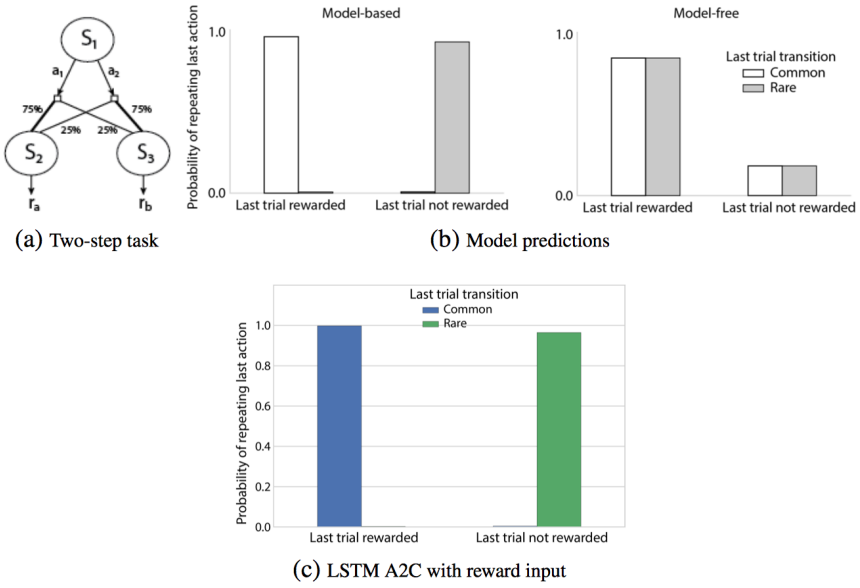
3 Experiment

3.1 The Two-step MDP: what it is and why it is important

The paper discusses multiple experiments to examine how effectively the meta-learning system captures task structure: the invariances across a distribution of similar tasks. They turn to the "Two-Step MDP" historically developed to understand the differences between model-free learning where the agent learns the values of actions in states, and model-based learning where the agent learns an internal model of the environment and evaluates the value of actions at the time of decision-making through look-ahead planning.

Parameter	Exps. 1 & 2	Exp. 3	Exp. 4	Exp. 5	Exp. 6
No. threads	1	1	1	1	32
No. LSTMs	1	1	1	1	2
No. hiddens	48	48	48	48	256/64
Steps unrolled	100	5	150	20	100
β_e	*annealed*	*annealed*	*annealed*	0.05	0.001
β_v	0.05	0.05	0.05	0.05	0.4
Learning rate	*tuned*	0.001	0.001	*tuned*	*tuned*
Discount factor	*tuned*	0.8	0.8	*tuned*	*tuned*
Input	a, r, t	a, r, t	a, r, t	a, r, t, x	a, r, x
Observation	n/a	n/a	n/a	1-hot	RGB (84x84)
No. trials/episode	100	5	150	10	10
Episode length	100	5	150	20	$\leq$3600

Figure 1: Hyperparams (see Exp. 5)



(a) Two-step task (b) Model predictions

(c) LSTM A2C with reward input

Figure 2: Paper results, MDP diagram

The aim of this experiment was to show how meta-RL gives rise to model-based behavior, despite the use of model-free learning to train the system.

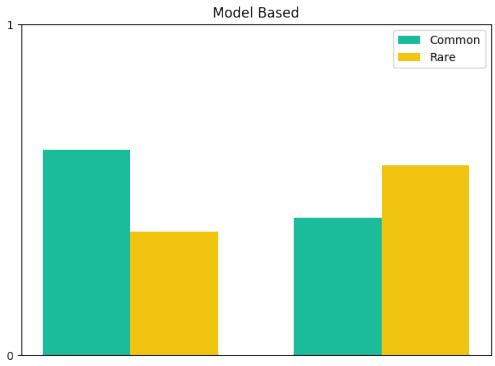
Figure 1(a), describes the two-step MDP. One second-stage state yielded a reward of 1.0 with probability 0.9 (and otherwise zero); the other yielded the same reward with probability 0.1. The identity of the higher-valued state was assigned randomly for each episode. Thus, the expected values for the two first-stage actions were either $r_a = 0.9$ and $r_b = 0.1$, or $r_a = 0.1$ and $r_b = 0.9$. The meta-RL is expected to learn the invariant transition structures across these two possible MDPs (which differ only in where the reward is placed).

The paper focuses on "stay-probabilities" that is, the probability with which a first-stage action is selected at trial $t + 1$ following a second-stage reward at trial t, as a function of whether trial t involved a common transition (e.g. action a_1 at state S_1 led to S_2) or rare transition (action a_2 at state S_1 led to S_3). With model-free control, first-stage actions will tend to be repeated if followed by reward, regardless of transition type, and such actions will tend not to be repeated (choice switch) if followed by non-reward. In contrast, with model-based control the action chosen is an outcome of an interaction between the reward and transition type, reflecting a more goal-directed strategy, which takes the transition structure into account.

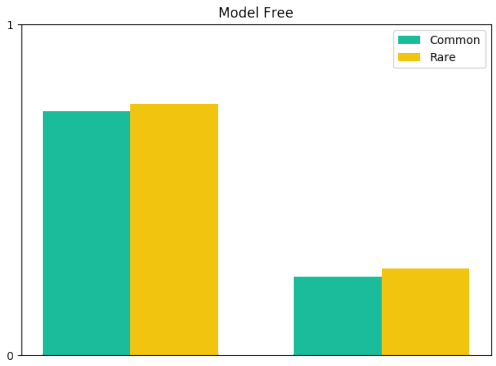
The paper shows that the stay-probability analysis performed on the meta-RL agent's choices show a pattern similar to that of model-based control.

3.2 Result Replication

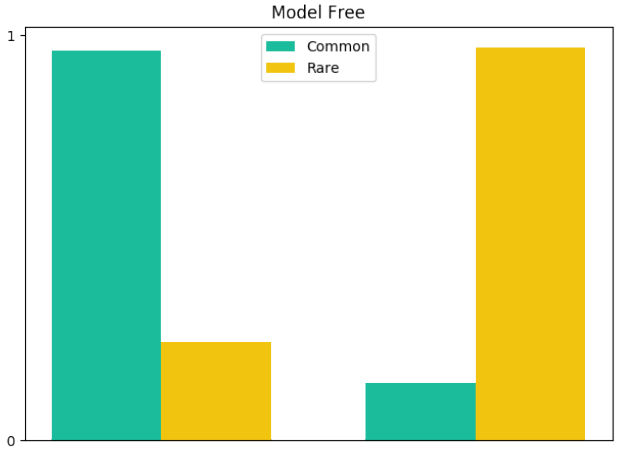
Figure 3 shows the replication results obtained from our experiments. We see that the overall behaviours of the three agents are similar to those seen in the original paper, even though the exact numbers differ (due to differences in hyperparameters used). The paper doesn't describe how exactly the stay-probabilities are calculated



(a) Model-based Agent Behaviour



(b) Model-free Agent Behaviour



(c) LSTM A2C Agent Behaviour

Figure 3: Model-based vs. Model-free Agent Behaviours vs. LSTM A2C

(learning rate and exploration strategies influence these statistics). Our replication results are closer to the results presented in the other neuroscience papers (Daw et al., 2005, Kool et al., 2016) upon which the [Wang et al. 2016] experiment was based.

4 Conclusion

The original paper claims an LSTM can encode optimal learning algorithm for the space of Markov Decision Processes, and that it is possible to train such an LSTM with standard reinforcement learning techniques (in particular A2C).

The two-step MDP experiment we implemented for this technical report supports the claim by showing that deep meta-learning can learn the common transition structure for every MDP in the target space. This is shown by how similar the chart of the meta agent is to a standard model-based RL approach. In this problem, a model-based RL agent would learn the underlying transition probabilities, and thus, after one run through a particular MDP, would be able to recognize where the reward actually is. If the agent took an action that dropped them in a state through the rare transition (that is, only 25 % of getting there based on one particular action), whatever reward it received would instantly inform its next action, as it understands how actions affect what state the agent ends up in. On the other hand, a model-free RL agent would not be able to as easily identify the reward location since it has no understanding of the underlying transition structure. If it takes an action, and gets a reward, it will repeat that action, even if the action it took placed it in a state that is rare to get to.

While replicating the experiment we found that proposed framework is not robust to changes of hyper-parameters, i.e. number of steps over MDP unrolled for every step of gradient descent, learning rate, discount factor, value and entropy loss factors and random seed. For now this conclusion is more of an empirical nature, and it might be interesting for the future research to carefully examine robustness of the framework to each of the hyper-parameters over multiple experiments. It also worth mentioning that we were not able to observe positive effect of entropy loss which is intended to make optimization more robust.

We suggest for the future research training the framework across natural environments (e.g. different Atari

games) which don't have such explicitly common structures like in the presented experiments, but which at the same time contain common relations in physics and our perception of the real world.

5 References:

1. Nathaniel D Daw, Samuel J Gershman, Ben Seymour, Peter Dayan, and Raymond J Dolan. Model-based influences on humans' choices and striatal prediction errors. Neuron, 69(6):1204–1215, 2011.

2. Wouter Kool, Fiery A Cushman, and Samuel J Gershman. When does model-based control pay off? PLoS Comput Biol, 12(8):e1005090, 2016.

3. Wang, Jane X., et al. Learning to reinforcement learn. arXiv preprint arXiv:1611.05763 2016.