Sub-optimality as directed exploration in reinforcement learning as probabilistic inference

Reinforcement learning, when cast as inference, can be formulated as optimization of a lower bound on the marginal probability of optimality nuisance variables similar to [9]. The evidence lower bound (ELBO) formulation that gives rise to maximum entropy policy gradients [5] minimizes a Kullback Leibler (KL) divergence between the state-conditional action policy distribution and an auxiliary distribution constructed by implicitly conditioning on observing that actions are "optimal." For this project I will develop a policy learning algorithm based on the same reinforcement learning as inference formalism discussed above under the opposite KL divergence. More specifically, we propose learning to directly map from observed optimality values to the conditional posterior distribution over actions given optimality (and state), to define an algorithm that trains policies using an explicit and controllable ability to sample and learn from sub-optimal action choices. As a point of comparison between these two objectives, consider the classic bound utilized by maximum entropy policy gradients, and the one proposed by us in Equations 1 and 2.

$$J_{ELBO}(\phi) = E_{\tau \sim q_\phi(\tau|O=1)} \left[r(\tau) - \log q_\phi(\tau|O=1) \right] \tag{1}$$

$$J_{KL(p||q)}(\phi) = E_{O \sim p_\theta^{sub}(O)} \left[E_{\tau|O \sim q_\phi(\tau|O)} \left[\frac{\log q_\phi(\tau|O) p(\tau, O)}{p_\theta^{sub}(O) q_\phi(\tau|O)} \right] \right] \tag{2}$$

Motivation

This choice is in part motivated by literature from VAEs, more specifically: [2], [12], [10], [11], [7] and finally [8]. These works found that in training the inference networks of VAEs, the Re-weighted Wake-Sleep (RWS) objective tended to perform substantially better then the classic variational inference objective. Additionally, this objective allows us to perform a unique implementation of scheduled sub-optimal action, and in doing so avoid forced sampling of uniform random actions. this means that we explicitly pick actions not sampled by the current policy under optimality (an example of this phenomena is displayed below in Figure 1). In doing this we more intelligently explore the state space by making actions that are defined to by unlikely under our optimal distribution, but likely under sub optimality. In this way the agent can continuously explore new spaces while still learning from its actions in a more directed fashion then random diffusion.

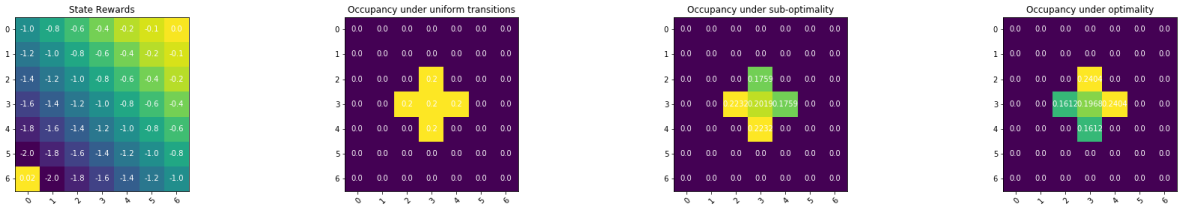

Figure 1: The heatmap on the left represents the reward, where white is the largest reward, and black the smallest. The middle heat map is occupancy after a sub optimal action starting from the middle state. The right heat map represents occupancy under a uniform random policy starting from the middle state.

Model free implementation with trust region updates and a replay buffer

For my research proficiency exam, I would like to investigate this objective and variants of it. These variants include most of the established methods given by the policy gradient literature, including trust region policy optimization (TRPO) [14], proximal policy optimization (PPO) [15], and a prioritized replay buffer [13]. I would then like to compare this objective against the corresponding policy gradient objective to see how it performs, and whether it can improve upon the sample efficiency and stability of established approaches. In order to test these approaches, my proposed model will be implemented for a host of Openai gym tasks that will include some combination of problems in the Algorithms, Atari, Classic control, MuJoCo, Roboschool, Robotics, and Toy text domains. The majority of this task has already been completed, by myself and Prof. Wood, and we need only implement this methodology on a larger range of tasks from the subset described above, therefore this task should only take me roughly one month to complete (as I will want to have solved a sufficient number of gym tasks for the analysis to be meaningful).

Constrained stochastic policies in continuous control tasks with normalizing flows

The next task that I would like to implement, is constrained normalizing flows for continuous control tasks. In many continuous control tasks, the associated policy is clipped so as to be between some specific interval. Historically, the easiest remedy to this

was to parameterize a continuous distribution defined over a finite interval (such as a beta distribution) and then simply stretch or contract this distribution to fit the specified interval [3]. Another popular solution was to use distributions with support over $\mathbf{R}^n$, and simply map samples outside of the specified interval to within the interval. In the first case, issues with optimization, as well as sampling can occur, due to certain parameterizations of the beta distribution (i.e. under certain α, and β). In the second case we get unwanted clustering near the boundary of the support, thus poorly defined behavior of the policy. By using a more expressive, parameterized constrained policy, we can create agents with well specified behavior in constrained continuous control tasks. This agent model will be tested using each of the objectives described above on a subset of continuous control tasks, and compared to both the clipping approach, as well as the beta distribution approach. I have yet to work with normalizing flows, however this should just amount to replacing the current agent model within the context of the larger objective, and should again constitute roughly a month of work.

A model based reinforcement learning approach following full re-weighted wake-sleep updates

The final task that I would like to investigate is a model based approach under the RWS objective. By incorporating a model of the world, we can can reduce the number of interactions with the word required for the agent to solve the system. Additionally, this generative model fits perfectly into the original RWS update given in [1]. This principle was briefly shown in [4] and more recently in [6] to drastically reduce the sample complexity in the learning of a host of high dimensional reinforcement learning tasks. Finally, because the basic algorithm (RWS) is built to train a generative model alongside the inference model, it would be easy algorithmically to incorporate this objective. This complete RWS training procedure might yield a novel training objective with respect model based reinforcement learning. In order to test this objective, we will consider a model based approach, as well as a model free approach and look at convergence rate per samples of the environment to see if any improvement exists. This result will again be tested on a subset of Openai gym tasks. This final task will likely be the most difficult, as some environments will not conform to a model based approach, to avoid this confusion, I will only consider a subset of tasks that are either relatively simple, or have been proven to work under a model based approach.

Time line

Task	time frame	description
Literature Review	01/09/19 - 14/09/19	This literature review will be completed in order to make sure I am up to date on any and all papers that are associated with the subject material.
Model free baselines	14/09/19 - 07/10/19	This stage will represent the bulk of the overall implementation time as I will have to put together PPO, TRPO, A3C, DDPG, and DQN, for some subset of tasks.
Importance weighting scheme implementation and task selection	07/10/19 - 21/10/19	I will need to determine a set of tasks to apply the proposed training objective to that represent a wide range of reinforcement learning problems, and that will highlight both the strengths and weaknesses of the objective.
Add trust-region updates and prioritized replay buffer (objective + baselines)	21/10/19 - 07/11/19	Based upon the preliminary results that I have already implemented, these two additions will be necessary in order to make the objective tractable for large scale problems.
Add constrained RL policies with normalizing flows	07/11/19 - 01/12/19	Implementation of this should only require modifying the existing agent policy model to include a constrained normalizing flow for the continuous tasks, which should be quick.
Model based RL implementation baseline	01/12/19 - 14/12/19	I will put together the basic model based approaches currently available for a subset of gym tasks that were shown in previous papers.
Model based RL implementation custom objective	14/12/19 - 07/01/20	At this stage I will set up the full re-weighted wake sleep update for my objective, and compare it in sample complexity to the model free approach, as well as the other model based approaches that I have implemented previously.
Collect Results / Organize story	07/01/20 - 14/01/20	At this stage I will look through the results of each of these investigations and formulate an overarching narrative for the research examination.
Write Paper	14/01/20 - 01/02/20	This is the stage where I will write and revise the research examination paper so that it is ready to be given to my committee.

References

[1] Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep, 2014.

[2] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[3] Po-Wei Chou, Daniel Maturana, and Sebastian Scherer. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 834–843. JMLR. org, 2017.

[4] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[5] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[6] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model-based reinforcement learning for atari, 2019.

[7] Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. Auto-encoding sequential monte carlo. *arXiv preprint arXiv:1705.10306*, 2017.

[8] Tuan Anh Le, Adam R Kosiorek, N Siddharth, Yee Whye Teh, and Frank Wood. Revisiting reweighted wake-sleep. *arXiv preprint arXiv:1805.10469*, 2018.

[9] Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

[10] Chris J Maddison, John Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Teh. Filtering variational objectives. In *Advances in Neural Information Processing Systems*, pages 6573–6583, 2017.

[11] Christian A Naesseth, Scott W Linderman, Rajesh Ranganath, and David M Blei. Variational sequential monte carlo. *arXiv preprint arXiv:1705.11140*, 2017.

[12] Tom Rainforth, Adam R Kosiorek, Tuan Anh Le, Chris J Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter variational bounds are not necessarily better. *arXiv preprint arXiv:1802.04537*, 2018.

[13] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2015.

[14] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[15] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.