Extending Policy Shaping to Continuous State Spaces

Thomas Wei[1], **Taylor A. Kessler Faulkner**[2], **Andrea L. Thomaz**[1]
[1]The University of Texas at Austin, Electrical and Computer Engineering
[2]The University of Texas at Austin, Computer Science
2501 Speedway, EER 7.826, Austin, TX 78712
thomasw219@gmail.com, +1-214-632-0600

Abstract

Policy Shaping (Griffith et al. 2013), is a Human-in-the-loop Reinforcement Learning (HRL) algorithm. We extend this work to continuous states with our algorithm, Deep Policy Shaping (DPS). DPS uses a feedback neural network that learns the optimality of actions from noisy feedback combined with an RL algorithm. In simulation, we find that DPS outperforms or matches baselines averaged over multiple hyperparameter settings and varying feedback correctness.

Introduction

Many HRL methods have made the leap to large state spaces (Warnell et al. 2018; Xiao et al. 2020; Arakawa et al. 2018; Arumugam et al. 2019). However, many of these methods use human feedback without a reward function, use the feedback to shape the reward, or do not consider noisy feedback. We introduce Deep Policy Shaping (DPS), which combines information from a feedback-generalizing neural network with an off-policy RL algorithm (Figure 1). To account for noisy feedback, we use techniques from deep learning with noisy labels, specifically modifying the loss function to reflect feedback confidence (Mnih and Hinton 2012). To improve the quality of our feedback generalization's uncertainty estimates, we use deep ensembles, which use a shared base to reduce computation and reuse signal, as used in another HRL algorithm, FRESH (Xiao et al. 2020). We compare DPS to baselines, varying the proportion of correct feedback. As DPS is designed to work with people, optimizing hyperparameter settings prior to learning may be impossible. We find that DPS outperforms or matches baselines on average over multiple hyperparameter settings.

Background

Some prior work in deep HRL uses human input as the only reward (Warnell et al. 2018; Arumugam et al. 2019), or uses feedback to shape the reward function (Xiao et al. 2020). Differing from both, DPS interprets feedback as input on the policy. There is prior work on interpreting the feedback as policy information, for example the original Policy Shaping method (Griffith et al. 2013). A method of integrating TAMER feedback with reward (Knox and Stone 2010) was extended to deep learning with DQN-TAMER (Arakawa et al. 2018). Unlike DQN-TAMER, DPS approximates the probability of optimal actions from feedback, allowing noisy feedback. Also, DPS multiplies the distributions from its feedback network with the RL algorithm output, applicable to any RL method that generates action distributions.

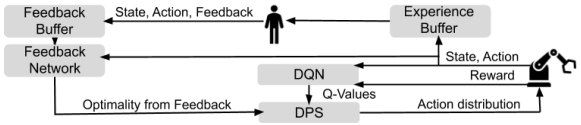

Figure 1: Deep Policy Shaping Pipeline.

Method

The original Policy Shaping interprets feedback as a statement on the optimality of an action in a state (Griffith et al. 2013). This interpretation allows the derivation of a closed-form expression for the probability of the optimality of an action based on feedback. Computing the probability of optimality requires feedback on that state action pair, otherwise the expression reduces back to the uniform prior. However, in continuous and high dimensional state spaces, many states that the agent sees will likely have not received feedback. In DPS, we use a neural network to approximate the conditional probability of an action's optimality in a state given feedback: $P(a \text{ is optimal} | s, a, D)$. The state is given as input to the neural network and each action has its own corresponding sigmoid head. To account for *a priori* information on the consistency of feedback with the true optimality of a state-action pair, we model the ground truth optimality as a latent random variable and derive the likelihood that our estimate of the probability of optimality produced the feedback received (Mnih and Hinton 2012):

$$L(y|h) = yC^h(1-C)^{1-h} + (1-y)C^{1-h}(1-C)^h \quad (1)$$

where h is feedback, C is the consistency of the feedback with the true optimality, and y is the probability that the action is optimal. As with binary cross entropy loss, we can take the negative of the log-likelihood as our loss function.

Deep Policy Shaping is meant to shape the exploration of any off-policy RL algorithm. For all the experiments we use a Deep Q Network (DQN) for RL with Boltzmann exploration. To avoid adverse effects of overfitting, we reduce

P	DPS	TAMER	DQN	DPS vs DQN	TAMER vs DQN	DPS vs TAMER	ANOVA
1	**15763951.1**	14200672.6	13547898.6	$\mathbf{p < 0.005}$	$p = 0.074244$	$\mathbf{p < 0.005}$	$\mathbf{p < 0.001}$
0.75	**14139224.6**	13625897.1	13547898.6	$p = 0.068316$	$p = 0.899995$	$p = 0.129681$	$p = 0.054321$
0.6	13334046.4	12713258.9	**13547898.6**	$p = 0.697596$	$\mathbf{p < 0.01}$	$p = 0.064097$	$\mathbf{p < 0.01}$

Table 1: The mean AUC at indicated levels of feedback correctness, with p-values. TAMER refers to DQN-TAMER.

the effect of feedback over time by clamping the outputs of the distribution to within the range $[0.5 - \beta, 0.5 + \beta]$ where $\beta(t) = 0.5 * \alpha^t$ and α is a hyperparameter from $(0, 1)$ which controls the rate of reduction. Like discrete Policy Shaping, DPS combines the distribution over actions created by the RL algorithm with the estimate of the optimality distribution by multiplying them together: $\pi \propto \pi_R \times \pi_F$.

We use OpenAI and Mujoco's Reacher-v2 environment (Brockman et al. 2016; Todorov, Erez, and Tassa 2012). We discretize the actions to idle, center joint clockwise and counter clockwise, and elbow joint clockwise and counter clockwise. We use a sparse reward, giving the robot rewards of 1 when it is within 0.015 of the goal and zero otherwise. We compare to DQN-TAMER (Arakawa et al. 2018) and a DQN. The oracle, a pretrained DQN, gives binary feedback: positive to optimal actions and negative otherwise.

We compare the average performance of each algorithm over 100 hyperparameter settings sampled randomly from a distribution as in practice, HRL algorithms would likely see little to no hyperparameter optimization due to the significant cost of obtaining human feedback. We repeat over varied correctness of feedback. We apply noise to all states and actions uniformly. Each feedback is flipped with some probability $1 - P$, $P \in [0.6, 0.75, 1.0]$, and the consistency hyperparameter is set $C = P$. There are α_h and α_q hyperparameters for DQN-TAMER that control how the influence of human feedback decays over the course of the learning process. DQN-TAMER does not define $\alpha_h(t)$ and $\alpha_q(t)$ in regards to feedback accuracy, so we do not adjust for P. Future work will determine the best distribution for α and definition of $\alpha_h(t)$ and $\alpha_q(t)$ for each P. However, this would involve hyperparameter search over a much larger space.

Results and Conclusion

All results are shown in Table 1, with significance calculated using a one way ANOVA with post-hoc Tukey HSD test. With $P = 1$, DPS significantly outperforms DQN-TAMER by 11% in area under the learning curve (AUC). Both DPS and DQN-TAMER outperform the DQN alone, DPS significantly so by 16%. With the feedback correctness reduced to $P = 0.75$, DPS insignificantly outperforms both DQN-TAMER and the DQN. With $P = 0.6$, DQN-TAMER performs significantly worse than the DQN by 6%, and DPS has a non-significantly higher AUC than DQN-TAMER. These results suggest that DPS matches or outperforms DQN-TAMER over many hyperparameter settings. This result is useful when hyperparameters cannot be tuned.

Future work could include adding active learning to DPS, which may reduce the total amount of feedback needed. In addition, since in practice the true consistency of feedback is unavailable, future work will test the performance of DPS when the C parameter does not match P.In this work, we introduce DPS and show that it outperforms or matches a DQN and DQN-TAMER over various settings of feedback correctness when averaged over multiple hyperparameter settings, which are difficult to tune prior to training.

Acknowledgements

This work is supported by ONR awards N000141612835 and N000141612785, NSF awards 1564080 and 1724157, and NSF-GRFP Grant DGE-1610403.

References

Arakawa, R.; Kobayashi, S.; Unno, Y.; Tsuboi, Y.; and Maeda, S.-i. 2018. Dqn-tamer: Human-in-the-loop reinforcement learning with intractable feedback. arXiv preprint arXiv:1810.11748 .

Arumugam, D.; Lee, J. K.; Saskin, S.; and Littman, M. L. 2019. Deep Reinforcement Learning from Policy-Dependent Human Feedback. CoRR abs/1902.04257.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym.

Griffith, S.; Subramanian, K.; Scholz, J.; Isbell, C. L.; and Thomaz, A. L. 2013. Policy Shaping: Integrating Human Feedback with Reinforcement Learning. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., Advances in Neural Information Processing Systems 26, 2625–2633. Curran Associates, Inc.

Knox, W. B.; and Stone, P. 2010. Combining manual feedback with subsequent MDP reward signals for reinforcement learning. In AAMAS.

Mnih, V.; and Hinton, G. 2012. Learning to Label Aerial Images from Noisy Data. In Proceedings of the 29th Annual International Conference on Machine Learning (ICML 2012).

Todorov, E.; Erez, T.; and Tassa, Y. 2012. MuJoCo: A physics engine for model-based control. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 5026–5033.

Warnell, G.; Waytowich, N. R.; Lawhern, V.; and Stone, P. 2018. Deep TAMER: Interactive Agent Shaping in High-Dimensional State Spaces. In AAAI.

Xiao, B.; Lu, Q.; Ramasubramanian, B.; Clark, A.; Bushnell, L.; and Poovendran, R. 2020. FRESH: Interactive Reward Shaping in High-Dimensional State Spaces using Human Feedback.