Deep V-Learning for Pool-Based Active Learning

March 7, 2022

1 Introduction

Even though pool-based Active Learning (P-AL) is the more popular setting for Active Learning, it creates significant challenges when reinforcement learning is applied to it. The most prominent one is that both Q-Learning and Policy Gradient Methods require a fixed action space. In the P-AL setting the action space has the same size as the pool of unlabeled images to choose from (or a subsample thereof). Even for simple datasets P-AL methods require a samplesize of >100 to work effectively, creating a large action space for the reinforcement learning agent.

1.1 Contribution

- First V-Learning Approach for P-AL

- Variable sample sizes to fit every dataset

- (Dataset and model agnostic state space) *Probably not the first*

2 Related Work

3 Background

3.1 V-Learning

V-Learning ([1] p.119) is closely related to Q-Learning. It estimates the value of states rather that the value of all actions given a state (Q-Learning [1] p.131) This requires the environment to provide possible future states $s_{t+1} \in S_{t+1}$ given any state s_t. A policy π_v typically chooses the most promising future state $argmax\, V(s_{t+1})$

Temporal difference (TD) learning for a neural network parametrized by θ gives us a nearly identical update formula compared to Q-Learning

$$\theta \leftarrow \theta + \eta \left(r_t + \gamma \hat{V}_\theta(s_{t+1}) \right) \tag{1}$$

3.2 Active Learning as a V-Learning MDP

We change the usual MDP formulation from predecessing literature, which, based on a presented sample of unlabeled samples of size k, defines a state space $S := \mathbb{R}^{k \times f}$ with features f and consequently an action space of $A := \{1, \dots, k\}$.

We instead define the state space to be 1-dimensional and use the batch dimension b for a variable sample size.

$$S := \mathbb{R}^{3+|Z|} := [\text{BvsSB, Entropy, F1, Z}] \tag{2}$$

V-Learning does not defines an action space, since the actions are implicit, as defined by the policy π_v.

This formulation poses a problem when storing transitions. The starting state s_t inuitively will be choosen datapoint s_{t,a_t}, but since we don't want to store the full follow-up state $s_{t+1} \in \mathbb{R}^{b \times 3+|Z|}$ we use the average over the batch-dimension. This serves as a proxy of the impact of the implicit action a_t on the follow-up state s_{t+1}.

This results in a stored transition $\phi_t := \{s_{t,a_t}, r_t, \bar{s}_{t+1}, d_t\}$ where $\bar{s}_{t+1}$ is the average of the follow-up state and d_t indicates wether a terminal state was reached.

4 Methodology

4.1 Model and Reinforcement Learning

We use a MLP with a single output neuron $f_\theta : \mathbb{R}^{3+|Z|} \to \mathbb{R}$ as agent network. As stated before, the sample of presented datapoints uses the batch dimension for sample size, resulting in b point-wise predictions of the agent.

For fitting the agent we use a target network f_{θ^-} [3], an n-step return ([1] p. 142) with $n = 3$ and priorizited experience replay [2]

References

[1] Andrew G. Barto Richard S. Sutton. *Reinforcement Learning: An Introduction.* MIT Press, Massachusetts, 2 edition, 2020.

[2] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[3] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.