Feature Learning with Reward Sensitive Representations

Gonçalo Guiomar **Tiago Costa** **Margarida Sousa**

Abstract

...

1 Introduction

Basic points to put here:

- Exploration of high-dimensional state-spaces is costly
- Having a compact representation is necessary to efficiently transverse environments
- Reward signals in nature are not symmetric in frequency of occurrence: positive rewards are sparse and localized and negative rewards are diffused
- These signals are also contextual; animals seem to bootstrap policies of new environments on what has been learned in other environments
- Besides, the brain processes positive and negative reward information through separate pathways
- How can we combine this into a more efficient RL algorithm?
- We'll attempt to define a reward and context sensitive algorithm that learns policies over a set of environments
- We combine graph representation learning techniques with reward sensitivity functions in order to speed up learning across different MDPs

2 Background

3 Learning the State Embeddings

4 Reward Sensitive Representations

4.1 Context-dependent State-Value Functions

As agents experience environments, the probability of observing novel states is conditioned on the underlying Markov Decision Process (MDP). The observation of certain states is thus contextual to the observation of past and future states.

We begin by defining a state value function that takes into account the proximity of the surrounding states as proxy for calculating the values; in the usual Function Approximation formalist [REF-Sutton] the state-value function is given as a linear sum over feature vectors i.e.

$$V(s_t) = \sum_i w_i \phi_i(s_t) \tag{1}$$

where ϕ_i is the i-th component of the feature vector corresponding to state s_t.

Although feature vectors allow for the compression of state relevant information in a way that allows for very high-dimensional state-spaces to be explored, we lose a bit of a notion of context for each state; to which extent is the state that I'm in related with states that have a similar representation? and furthermore, to which extent should the information gathered in surrounding states affect the representation of the current state.

To explore this idea we define a novel value function as dependent on the dot product of the state embeddings in a surrounding ball of size ϵ given by a weight metric function $\Gamma^{\epsilon}(s_t)$ so that the state value function is now given by

$$V^h(s_t) = \sum_i \Gamma^{\epsilon}(s_t)_i \langle \phi_i, \mathbf{w}^h \rangle \tag{2}$$

with $\langle \phi_i, \mathbf{w}^h \rangle$ the inner product between the weights $\mathbf{w}$ and the embeddings of each state ϕ_i

$$\Gamma^{\epsilon}(s_t)_i = \begin{cases} 0 & \text{if } ||\phi(s_t) - \phi_i|| > \epsilon \\ \frac{1}{1+\beta_V ||\phi(s_t)-\phi_i||}, & \text{otherwise} \end{cases} \tag{3}$$

We'll be trying to minimize the Reward Prediction Error (RPE) with rewards filtered by a function $f^h(\delta_t)$ i.e.

$$\delta_t^h = f^h(r_t) + \gamma V^h(s_{t+1}) - V^h(s_t) \tag{4}$$

and the loss function for each RPE function is given by

$$\mathcal{L}^h = ||\delta_t^h||^2 \tag{5}$$

We can learn the weights $\mathbf{w}$ by the usual gradient descent rule

$$\frac{\partial \mathcal{L}^{\langle}}{\partial w_j^h} = 2\delta_t^h \frac{\partial \delta_t^h}{\partial w_j^h} \tag{6}$$

where the latter partial derivative term can be expanded thus

$$\frac{\partial \delta_t}{\partial w_j^h} = \gamma \frac{\partial V^h(s_{t+1})}{\partial w_j^h} - \frac{\partial V^h(s_t)}{\partial w_j^h} = \sum_i \left(\gamma \Gamma^{\epsilon}(s_{t+1})_i \phi_i - \Gamma^{\epsilon}(s_t)_i \phi_i \right) \tag{7}$$

which will give us a final update rule for each weight w_j

$$w_j \leftarrow w_j + \alpha^h 2\delta_t^h \sum_i \left(\gamma \Gamma^{\epsilon}(s_{t+1})_i \phi_i - \Gamma^{\epsilon}(s_t)_i \phi_i \right) \tag{8}$$

4.2 Policy

As a first approach we can try to model the policy with an action preference formalism that takes the RPEs of each channel h and feeds it into a corresponding preference

$$A^h(s,a) \leftarrow A^h(s,a) + \alpha \delta_t^h \tag{9}$$

where we sum all channels

$$A^T(s,a) = \sum_h \theta_h A^h(s,a) \tag{10}$$

and define our policy through a softmax over the total

$$\pi(a|s,\theta) = \frac{e^{\sum_h A^T(s,a)}}{\sum_a A^T(s,a)} \tag{11}$$

5 Experiments

6 Conclusion