Contextual 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 Contextual Learning

We begin by defining a state value function that takes into account the sensitivity to certain reward ranges, as in [REF-Tano] we write

$$V^h(s_t) = \mathbf{E}^\pi\big[\sum_{\tau}^{\infty}\gamma^\tau f^h(r_{t+\tau})\big] \tag{1}$$

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 the agent is in at time t is related contextually with states that have a similar representation?

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 by weighting the surrounding weights with a function $\Gamma^\epsilon(s_t)$

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

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

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

and the loss function for each RPE function is given by

$$\mathcal{L}^h = ||V^h(s_t) - \hat{V}^h(s_t)||^2 \tag{4}$$

which if we take the following expansion

$$V^h(s_t) = \mathbf{E}^\pi\big[f^h(r_t) + \gamma\sum_\tau \gamma^\tau f^h(r_{t+1+\tau})\big] = f^h(r_t) + V^h(s_{t+1}) \tag{5}$$

allows us to write the following objective function

$$\mathcal{L}^h = ||f^h(r_t) + \gamma V^h(s_{t+1}) - \hat{V}^h(s_t)||^2. \tag{6}$$

which implies new definition for the RPE

$$\delta_t^h = f^h(r_t) + \gamma V^h(s_{t+1}) - \hat{V}^h(s_t). \tag{7}$$

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

$$\frac{\partial\mathcal{L}^h}{\partial w_j^h} = 2\delta_t^h\frac{\partial\delta_t^h}{\partial w_j^h} \tag{8}$$

where in calculating the latter partial derivative we need to take into account that only the approximated value function can be differentiated in respect to $\mathbf{w}$ so

$$\frac{\partial\delta_t^h}{\partial w_j^h} = -\frac{\partial\hat{V}^h(s_t)}{\partial w_j^h} = -\sum_i \Gamma^\epsilon(s_t)_i\phi_i \tag{9}$$

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

$$w_j \leftarrow w_j - \alpha^h 2\delta_t^h\sum_i \Gamma^\epsilon(s_t)_i\phi_i \tag{10}$$

4.1 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

$$A^h(s,a) \leftarrow A^h(s,a) + \alpha\delta_t^h \tag{11}$$

where we sum all channels

$$A^T(s,a) = \sum_h \theta_h A^h(s,a) \tag{12}$$

and define our policy through a softmax over the total

$$\pi(a|s,\theta) = \frac{e^{\sum_h A^T(s,a)}}{\sum_a A^T(s,a)} \tag{13}$$

5 Experiments

6 Conclusion