Neural Networks 84 (2016) 17–27

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

From free energy to expected energy: Improving energy-based value function approximation in reinforcement learning

Stefan Elfwing[a,b,*], Eiji Uchibe[a,b], Kenji Doya[b]

[a] *Department of Brain Robot Interface, ATR Computational Neuroscience Laboratories, 2-2-2 Hikaridai, Seikacho, Soraku-gun, Kyoto 619-0288, Japan*

[b] *Okinawa Institute of Science and Technology Graduate University, 1919-1 Tancha, Onna-son, Okinawa 904-0495, Japan*

ARTICLE INFO

Article history:
Received 17 July 2015
Received in revised form 22 April 2016
Accepted 28 July 2016
Available online 26 August 2016

Keywords:
Reinforcement learning
Restricted Boltzmann machine
Expected energy
Function approximation
SZ-Tetris

ABSTRACT

Free-energy based reinforcement learning (FERL) was proposed for learning in high-dimensional state and action spaces. However, the FERL method does only really work well with binary, or close to binary, state input, where the number of active states is fewer than the number of non-active states. In the FERL method, the value function is approximated by the negative free energy of a restricted Boltzmann machine (RBM). In our earlier study, we demonstrated that the performance and the robustness of the FERL method can be improved by scaling the free energy by a constant that is related to the size of network. In this study, we propose that RBM function approximation can be further improved by approximating the value function by the negative expected energy (EERL), instead of the negative free energy, as well as being able to handle continuous state input. We validate our proposed method by demonstrating that EERL: (1) outperforms FERL, as well as standard neural network and linear function approximation, for three versions of a gridworld task with high-dimensional image state input; (2) achieves new state-of-the-art results in stochastic SZ-Tetris in both model-free and model-based learning settings; and (3) significantly outperforms FERL and standard neural network function approximation for a robot navigation task with raw and noisy RGB images as state input and a large number of actions.

1. Introduction

Sallans and Hinton (2004) proposed free-energy based reinforcement learning (hereafter, *FERL*) to handle high-dimensional state and action spaces. In the FERL method, the value function is approximated by the negative free energy, F, of a restricted Boltzmann machine (RBM) (Freund & Haussler, 1992; Hinton, 2002; Smolensky, 1986): $Q = -F = -\langle E \rangle + H$ for action-value based learning, where $\langle E \rangle$ is the expected energy and H is the entropy of the network. A considerable limitation of the FERL method is that it only works well with binary, or close to binary, state input. Furthermore, it is known that RBMs, traditionally, are not invariant to different state representations and require that the number of active states (values close to one) is much fewer than the number of non-active states (values close to zero) to work well.

We have earlier demonstrated (Elfwing, Uchibe, & Doya, 2013) that the robustness and the learning performance of FERL can be improved by scaling the free energy by a constant scaling factor, Z (i.e., $Q = -F/Z$), that is related to the size of the network. The purpose of this study is to show that expected energy based RBM function approximation (hereafter, *EERL*: $Q = -\langle E \rangle$) can achieve competitive learning performance, not only in tasks with binary state input and fewer active than non-active states, but also in tasks with continuous state input and in tasks with more active than non-active states. In the latter cases, we introduce a simple normalization by removing the mean of a state vector from each of its elements to improve the learning performance even further.

To validate our proposed method, we first use three versions of a gridworld task where the state input consists of (1) grayscale images of handwritten digits from the MNIST data set (LeCun, Bottou, Bengio, & Haffner, 1998); (2) inverted MNIST images; and (3) RGB images of the different objects from the CIFAR-10 data set (Krizhevsky, 2009). The purpose of the first version of the task is to test the learning performance of our proposed method for a task setting that is traditionally considered well-suited for RBMs: i.e., close to binary state input with much fewer active than non-active states. The purpose of the other two versions of the task is the opposite, i.e., a task with more active than non-active states, and a task with continuous state input. We then use the stochastic SZ-Tetris benchmark (Szita & Szepesvári, 2010) to validate the performance of EERL, in both model-free (Sarsa(λ)) and model-based

* Corresponding author at: Okinawa Institute of Science and Technology Graduate University, 1919-1 Tancha, Onna-son, Okinawa 904-0495, Japan.
E-mail addresses: elfwing@atr.jp (S. Elfwing), uchibe@atr.jp (E. Uchibe), doya@oist.jp (K. Doya).

http://dx.doi.org/10.1016/j.neunet.2016.07.013

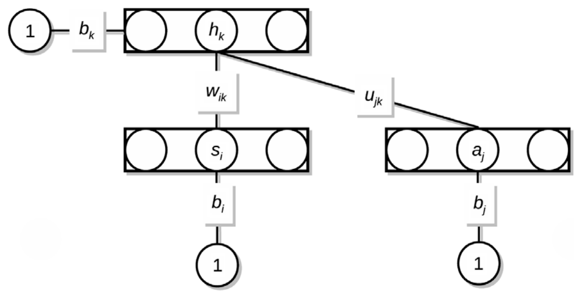

Fig. 1. RBM architecture for action-value based reinforcement learning. In the case of state-value based reinforcement learning, the action nodes, a_j, action biases, b_j, and the action weights, u_{jk}, are not included.

(TD(λ)) learning settings, for a task that is, in general, considered difficult for reinforcement learning algorithms. Finally, we use a robot visual navigation task with raw and noisy RGB camera images as state input. The goal of the task is to navigate to one of two goal areas. The correct goal can be inferred from the color of the upper part of four landmarks, which is randomly changed in each episode. In the robot navigation task, we investigate how well EERL can handle a large number of actions (pairs of velocities of the left and right wheels) by testing settings with 9, 25, and 100 possible actions.

Apart from the pioneering work by Sallans and Hinton (2004) and our earlier studies (Elfwing, Otsuka, Uchibe, & Doya, 2010; Elfwing et al., 2013), there have been few studies using RBMs as function approximation in reinforcement learning. Heess, Silver, and Teh (2012) proposed two energy-based actor-critic policy gradient algorithms and demonstrated that they were more robust and more effective than standard FERL in several high dimensional tasks. Otsuka, Yoshimoto, and Doya (2010) extended the FERL method to handle partially observable Markov decision processes, by incorporating a recurrent neural network that learns a memory representation that is sufficient for predicting future observations and rewards. In our recent work (Elfwing, Uchibe, & Doya, 2015), we demonstrated in the classification domain that the expected energy based RBM method significantly outperforms the free energy based RBM method.

2. Method

2.1. TD(λ) and Sarsa(λ)

The reinforcement learning (Sutton & Barto, 1998) methods that we propose in this study are based on the state-value function learning algorithm TD(λ) (Sutton, 1988) and the action-value function learning algorithm Sarsa(λ) (Rummery & Niranjan, 1994; Sutton, 1996). TD(λ) learns an estimate of the state-value function, V^π, and Sarsa(λ) learns an estimate of the action-value function, Q^π, while the agent follows policy π. If the approximated value functions, $V_t \approx V^\pi$ and $Q_t \approx Q^\pi$, are parameterized by the parameter vector $\boldsymbol{\theta}_t$, then the gradient-descent update of the parameters is computed by

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\delta_t\boldsymbol{e}_t, \tag{1}$$

where TD-error, δ_t, is

$$\delta_t = r_t + \gamma V_t(s_{t+1}) - V_t(s_t), \tag{2}$$

for TD(λ) and

$$\delta_t = r_t + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t), \tag{3}$$

for Sarsa(λ). The eligibility trace vector, $\boldsymbol{e}_t$, is

$$\boldsymbol{e}_t = \gamma\lambda\boldsymbol{e}_{t-1} + \nabla_{\boldsymbol{\theta}_t} V_t(s_t), \qquad \boldsymbol{e}_0 = \boldsymbol{0}, \tag{4}$$

for TD(λ) and

$$\boldsymbol{e}_t = \gamma\lambda\boldsymbol{e}_{t-1} + \nabla_{\boldsymbol{\theta}_t} Q_t(s_t, a_t), \qquad \boldsymbol{e}_0 = \boldsymbol{0} \tag{5}$$

for Sarsa(λ). Here, s_t is the state at time t, a_t is the action selected at time t, r_t is the reward for taking action a_t in state s_t, α is the learning rate, γ is the discount factor of future rewards, λ is the trace-decay rate, and $\nabla_{\boldsymbol{\theta}_t} V_t$ and $\nabla_{\boldsymbol{\theta}_t} Q_t$ are the vectors of partial derivatives of the function approximators with respect to each component of $\boldsymbol{\theta}_t$.

2.2. Free energy value function approximation

The use of a RBM as a function approximator for reinforcement learning was proposed by Sallans and Hinton (2004). A RBM is a bidirectional neural network (see Fig. 1) which in the FERL method consists of binary state nodes, $\boldsymbol{s}$, binary hidden nodes, $\boldsymbol{h}$, and, in the case of action-value function learning, binary action nodes $\boldsymbol{a}$. The ith state node, s_i, is connected to hidden node h_k by the weight w_{ik}, and the jth action node, a_j, is connected to hidden node h_k by the weight u_{jk}. In addition, the state nodes, the hidden nodes and the action nodes are all connected to a constant bias input with a value of 1, with connection weights b_i, b_k, and b_j, respectively. The action vector $\boldsymbol{a}$ has an "one-out-of-J" representation and functions as a fixed input to the network for each action. Let $\boldsymbol{a}_j$ denote the vector for action j, where a_j is equal to one and the rest of the action nodes are equal to zero.

For state-value function learning, the energy, E, of the RBM for state vector $\boldsymbol{s}$ is given by

$$E(\boldsymbol{s}, \boldsymbol{h}) = -\sum_{k=1}^{K}\sum_{i=1}^{I} s_i w_{ik} h_k - \sum_{i=1}^{I} b_i s_i - \sum_{k=1}^{K} b_k h_k, \tag{6}$$

and for action-value function learning, the energy, E, of the RBM for state vector $\boldsymbol{s}$ and action vector $\boldsymbol{a}_j$ is given by

$$\begin{aligned} E(\boldsymbol{s}, \boldsymbol{a}_j, \boldsymbol{h}) &= -\sum_{k=1}^{K} h_k \left[\sum_{i=1}^{I} s_i w_{ik} + \sum_{j^*=1}^{J} a_{j^*} u_{j^*k}\right] \\ &\quad - \sum_{i=1}^{I} b_i s_i - \sum_{j^*=1}^{J} b_{j^*} a_{j^*} - \sum_{k=1}^{K} b_k h_k \\ &= -\sum_{k=1}^{K} h_k \left[\sum_{i=1}^{I} s_i w_{ik} + u_{jk}\right] \\ &\quad - \sum_{i=1}^{I} b_i s_i - b_j - \sum_{k=1}^{K} b_k h_k. \end{aligned} \tag{7}$$

Here, I is the number of state nodes, K is the number of hidden nodes, and J is the number of actions. The free energy, F, can be computed as the sum of the expected energy, $\langle E \rangle$, and the negative entropy, H, where the expectations are taken with respect to the posterior distribution of the hidden values ($P(\boldsymbol{h}|\boldsymbol{s})$ and $P(\boldsymbol{h}|\boldsymbol{s}, \boldsymbol{a}_j)$). The expected hidden activation (i.e., the probability that the hidden value is equal to one) of hidden node k is given by

$$\langle h_k \rangle = P(h_k = 1|\boldsymbol{s}) = \sigma\left(\sum_{i=1}^{I} s_i w_{ik} + b_k\right) = \sigma(x_k) \tag{8}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{9}$$

for state-value function learning and

$$\begin{aligned} \langle h_{jk} \rangle &= P(h_k = 1|\boldsymbol{s}, \boldsymbol{a}_j) = \sigma\left(\sum_{i=1}^{I} s_i w_{ik} + u_{jk} + b_k\right) \\ &= \sigma(x_{jk}), \end{aligned} \tag{10}$$