HUMBOLDT-UNIVERSITÄT ZU BERLIN
BERNSTEIN CENTRE FOR COMPUTATIONAL NEUROSCIENCE

Bernstein Center for Computational Neuroscience
Humboldt-Universität zu Berlin
Philippstr. 13 House 6

Phone: 030/2093-9110
Fax: 030/2093-6771
webpage: http://www.bccn-berlin.de/

Models of Neural Systems I, WS 2008/09
Computer Practical 5

Reinforcement Learning

Today we implement a variant of a reinforcement learning algorithm. In general, reinforcement learning lies between supervised and unsupervised learning. In supervised learning, the learning algorithm receives feedback in terms of the correct learning patterns; it is exactly told what its response should be. In contrast, in reinforcement learning the feedback is rather general; it is told whether a response was 'good' or 'bad'. We implement a variant called temporal-difference (TD) learning, for which there is good evidence that the brain uses this learning algorithm.

Exercises

1. **Temporal Difference (TD) algorithm**
   The TD algorithm learns to predict the total future rewards $P(t)$ on the basis of sensory stimuli. Its main component is so-called reward prediction error $\delta(t)$, which evaluates the difference between the actual and predicted future reward. The goal of the algorithm is to minimise the error by means of modification of the weights $\mathbf{w}(t)$ relating the current state $\mathbf{s}(t)$ to the future reward:

   $$P(t) = \mathbf{w}(t)\mathbf{s}(t) \tag{1}$$

   It can be shown that the problem can be solved with the following weights' update rule:

   $$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\delta(t)\mathbf{s}(t-1), \tag{2}$$

   where η is a learning rate.

   The TD algorithm can be implemented by performing the following tasks in each consecutive discrete time step:

   (a) Update a state vector $\mathbf{s}(t)$ so that its components represent (non)-occurrence (encoded by 1s and 0s) of stimulus in the past time steps with the first component representing the current time step and the following components going back in past.

(b) Calculate the predicted future reward according to Equation 1.

(c) Estimate the prediction error in each time step with the following expression: $\delta(t) = P(t) - P(t-1) + r(t)$, where $r(t)$ is the reward at the current time step.

(d) Update the weights according to TD update rule (Equation 2) and go to the next time step.

(e) Repeat (a)-(d) with $t = t + 1$. Note that (as for other learning paradigms) the number of iterations needed for learning depends on the learning task and the parameters.

2. **Classical conditioning**

We now simulate a simple classical conditioning experiment in which a reward is associated with an arbitrary stimulus occurring earlier (such as food with the ringing of a bell in a Pavlov experiment).

Imagine a monkey in front of a TV screen. From time to time the screen shows different symbols and in some cases these symbols are followed by a reward (e.g. orange juice for the monkey).

(a) We define a trial as a sequence of several 15 discrete time steps t. At a time step $t = 5$ a stimulus occurs (a symbol appears on the TV screen).

Hint: Implement a state vector which has as many components as there are time steps in the trial.

(b) Set a reward value $r(t)$ so that it is always zero, except at time step $t = 10$ where it is set to 1.

(c) Initialise the weights with a null vector. Run the TD algorithm several trials each time using the same stimuli and rewards.

Note: The weights are NOT re-initialised after each trial.

(d) Plot the prediction error $\delta(t)$, the $\mathbf{w}(t)$, and the reward prediction $P(t)$ as a function of trials. Explain the obtained results. How do you interpret the variables? What has been learned by the algorithm?

(e) Compare the prediction error with the activity of dopamine neurons shown Figure 2 in the paper from Schultz, W. (1998). Predictive reward signal of dopamine neurons. J Neurophysiol 80(1):1-27.

3. (Optional) **Extending the paradigm**

(a) Extend the paradigm so that there is not only one type of stimulus, but in total 4 types of stimuli (still only one stimulus per trial). Each stimulus type needs its own state and weight vector representation (so instead of a e.g. 1×15 array, your vectors should become 4×15 arrays). Each stimulus type should have an individual reward probability; one of them is followed by a reward at time step 15 with 100%, one with 75% and one with 50% and one with 25%. Plot again the prediction error for each stimulus type as a function of trials and compare the weight vectors after learning.

Compare your results with Figure 3B and 4 in this paper: Morris et al. (2005). Coincident but Distinct Messages of Midbrain Dopamine and Striatal Tonically Active Neurons. Neuron 43(1), 133-143.

(b) Modify the weight change into: $\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\delta(t)\mathbf{e}(t)$. $\mathbf{e}(t)$ is the so-called eligibility trace and contains a representation of past states. It is updated after the weight changes in each time step by $\mathbf{e}(t) = \lambda\mathbf{e}(t-1)+\mathbf{s}(t)$. The parameter λ determines for how long past states stay represented. For $\lambda = 0$ the eligibility traces always contains only the most recent state which is equivalent to the basic implementation above. How do nonzero values for λ change learning?

CONTACT

ROBERT SCHMIDT (ITB, R. 2316) PHONE: 2093-8926 EMAIL: R.SCHMIDT@BIOLOGIE.HU-BERLIN.DE

BARTOSZ TELENCZUK (ITB, R. 1309) PHONE: 2093-8838 EMAIL: B.TELENCZUK@BIOLOGIE.HU-BERLIN.DE