Neural Networks and Biological Modeling

Professor Wulfram Gerstner
Laboratory of Computational Neuroscience
Assistant: Felipe Gerhard (felipe.gerhard@epfl.ch)

Mini-Project: Reinforcement Learning

The goal of the mini-project is to implement a reinforcement learning paradigm using a rate-based neuron model. The setup is similar to the grid-world that you got familiar with in problem set # 6: A rat is running around inside an arena and learns the position of a goal area where it gets a reward. After implementing the basic set-up in step 1 and studying the performance as a function of some of the network parameters (step 2), you will implement a psychological paradigm called *reversal learning* (step 3). Here, the position of the reward is abruptly changed after a number of trials and the rat has to learn the new reward location. Finally, you study the effect of *extinction* or "forgetting". This relates the SARSA algorithm on the neural network level to phenomena known in classical psychology.

Step 1: Implement the neural network model.

First, you should implement the environment and the neural network that controls the movement of the rat in the arena. Consider the following specifications:

- The rat is moving around in a rectangular arena with unit area. The position $s(t) = (s_x(t), s_y(t))$ of the (point-like) rat is encoded in the activity of a population of place cells. Let there be 25x25 place cells for which the activity of the jth cell is given by $r_j(s) = \exp(-\frac{(x_j-s_x)^2+(y_j-s_y)^2}{2\sigma^2})$ where the centers of the place cells are arranged on a 25x25 grid (with grid distance 1/24). Set $\sigma = 0.05$.
- The output layer of the neural network consists of eight neurons (*action units*). The activity of the a-th output neuron represents the Q-value of moving in the direction $2\pi a/8$, given the current state. Each output neuron is connected to all neurons in the input layer with connection weights w_{aj}. The activity of the output neuron is $Q(s,a) = \sum_j w_{aj} r_j(s)$.
- When the rat is in state s, its next action is a step of length $l = 0.03$ in the direction of $a^* = \arg\max_a Q(s,a)$ with probability $1-\epsilon$ or a random movement in one of the eight directions with probability ϵ (this strategy is called *epsilon-greedy*). For the beginning, set $\epsilon = 0.5$.
- When the rat hits the wall (its position after a movement exceeds the unit area), move it back inside the arena and assign a reward of size -2.
- A hidden goal (goal A) is defined as a circle of radius 0.1 around the centre with coordinates $(0.8, 0.2)$. When the rat enters the goal area, it receives a reward of +10 and the trial is stopped. At the beginning of a trial, the rat starts at position $(0.1, 0.5)$.
- In each time step, the weights w_{aj} are updated according to the SARSA algorithm with learning rate $\eta = 0.005$, reward discount rate $\gamma = 0.95$ and eligibility trace decay rate $\lambda = 0.95$. For the SARSA algorithm, use the formulas for the synaptic update rule and eligibility trace that were given in the lecture (lecture 6, see your notes or the slides on the moodle).

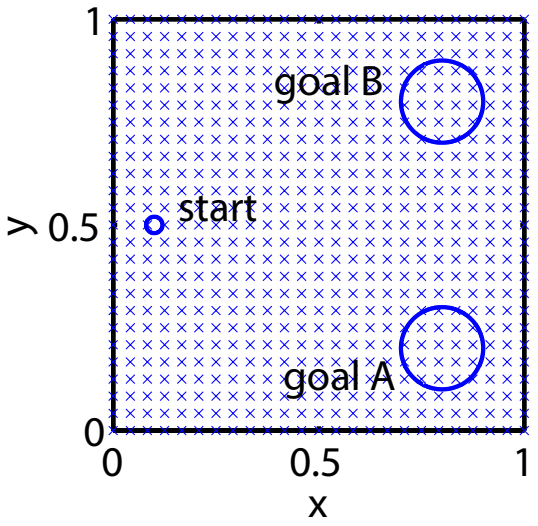

Figure 1: Geometry of the experiment (in step 3): At the beginning of each trial, the rat starts at position $(0.1, 0.5)$ (small blue circle). There are two different goal areas (big blue circles). In each trial, only one of them is triggering the reward. The blue crosses represent the centers of the place fields.

Step 2: Analyze the neural network model.

- **Learning curve**: Simulate at least 10 independent rats that run 50 trials each. If within a trial, the rat does not find the goal within N_{max} steps, abort the trial. Depending on your implementation, choose e. g. $N_{max} = 10000$. Plot the learning curve, i. e. the number of time steps it takes in each trial until the goal is reached ("latency"), averaged over all rats. If your implementation is correct, the curve should decrease and reach a plateau after a certain number of trials. How close is the performance compared to an optimal action policy?
- **Integrated reward**: Instead of looking at the time it takes to hit the target area, you can also look at the total reward that was received on each trial. Is the result consistent with the latency curve?
- **Exploration-exploitation**: The parameter ϵ controls the balance between exploration and exploitation (why?). See how different values of $0 \leq \epsilon \leq 1$ change the performance. For the comparison, you could plot several learning curves for different values of ϵ or use e. g. the average latency in the last 10 trials as a performance measure. Let ϵ decrease from a large value at the beginning to a small value at the end of training.
- **Navigation map** (optional): Visualize the $Q(a, s)$ map for different stages of learning (e. g. plot the preferred direction at each place field center using arrows or a color-code).

(turn the page!)

Step 3: Study the effects of reversal learning.

Consider the situation for which the position of the goal area is shifted after a certain number of trials. The rat then has to learn the new position of the reward. This paradigm is known in the psychology literature as *reversal learning*. Often, it is observed that learning to reach the second goal location is slower than for the first one, due to an initial persistency to move to the first goal. Here, you will study whether the SARSA algorithm reproduces this behavior.

- **Implement reversal learning**: Add a second goal area (goal B), with radius 0.1 and center coordinates $(0.8, 0.8)$. For the first 25 trials, a reward is given if the rat hits goal A. From trial 26 on, the reward is given only in the goal area B. After another 25 trials, the situation is reversed again and goal A is rewarded. Switch again after 25 trials. Study the learning curve and integrated reward curve. Compare the learning times: Does it take the rat longer to learn to move to goal B than to goal A? What about the subsequent reversals? Does the rat "remember" the goal locations? Interpret your results.
- **Extinction**: Add another sequence of trials in which none of the goal areas is rewarded. End each trial after a fixed time and record the number of time steps that were spent inside each of the two goal areas. Do you observe *extinction*, i. e. does the rat visit both places equally after some trials? What is the time course of this effect?
- **Merits of punishment** (optional): Change the set-up such that when the rat enters the currently unrewarded goal area, it gets a reward of size -5. Does this help the rat to switch to the other target at the reversal points?

Step 4: Summarize your findings.

Write a concise report that presents the results of your model analysis. Marks will be based on the clarity of the presentation and on the thoroughness of the analysis.

(A final note on the implementation: You can use any programming language that you like; however support can only be given for MATLAB and Python. In the case of MATLAB and Python, make sure you write the update equations of the SARSA algorithm in matrix-vector form, i. e. avoid *for-loops! If you have questions or want to make an appointment for a personal discussion, write me an email (felipe.gerhard@epfl.ch). Have fun!)*