Lab 1: Probabilistic Reasoning and Decision Making

INF581: Advanced Machine Learning and Autonomous Agents

Last Updated: January 3, 2023

Completion Instructions: Read this document; complete the file `lab1.py` (where indicated `#TODO`).
Hints: Run this file as a script to visualize the output of your tasks; or move the contents (including taking the contents under `if __name__ == "__main__":`) into a Jupyter Notebook.

Submission Instructions: See `README.md`

Figure 1 shows a 5×5 grid of tiles representing a room (scenario/environment). The task is to design the decision making for a virtual cat (agent) to catch the virtual rat (target). The room is dark, and the rat cannot be seen, however the agent knows the following: The rat moves by exactly 1 tile, either horizontally or vertically (i.e., taxicab-distance) per time step, within the bounds of the scenario, starting at one of the entry points – uniformly at random. It will cause an audible crinkle with probability 0.9 when over certain tiles (green, or orange), and with probability 0 over other tiles; furthermore, it will cause a rustle noise with probability 0.8 over certain tiles (red, or orange), and 0 otherwise. On orange tiles, both noises are caused independently of each other. The cat-agent waits for 5 time steps before making a decision to *pounce* or not on one of the tiles in an attempt to catch the rat.

Let's denote: $y_t \in \{1, \ldots, 25\}$ the position (state) at time step t (one of the 25 tiles); starting at some y_1 (entry tile). And $\mathbf{x}_t \in \{0, 1\}^2$ is the 2-dimensional observation at time t (e.g., $\mathbf{x}_t = [1, 0]$ if there is a crinkle but no rustle, etc). The agent accumulates observations $\mathbf{x}_{1:5} = \mathbf{x}_1, \ldots, \mathbf{x}_5$, then makes the decision/action a to pounce ($a = s$: onto tile s) or not attempt $a = 0$. The agent is rewarded with $r(a, s) = 10$ for catching the rat (when $a = s$; the rat is always captured), -3 for missing it ($a \neq s$), and 0 for a non-attempt ($a = 0$).

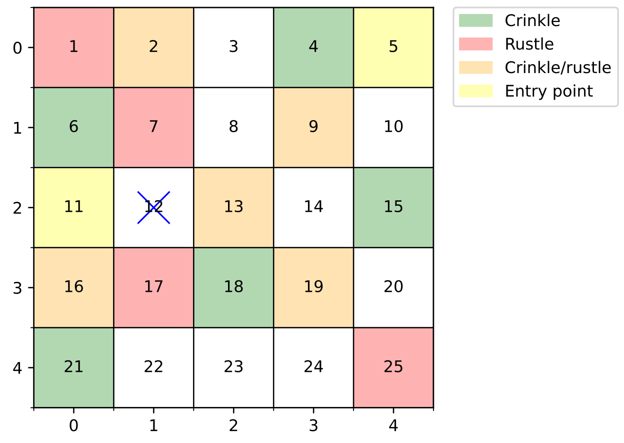


Figure 1: Scenario/environment. The agent (position marked by $\times$; but this information is only needed for the Bonus task) stays still and observes a sequence of noises, e.g., $\mathbf{x}_{1:5} = [[0, 0], [1, 0], [0, 0], [1, 0], [0, 0]]$ and then must decide which tile the target is likely to be on and if it is worth attempting to catch it.

The Tasks:

Given an observation $\mathbf{x}_{1:5}$:

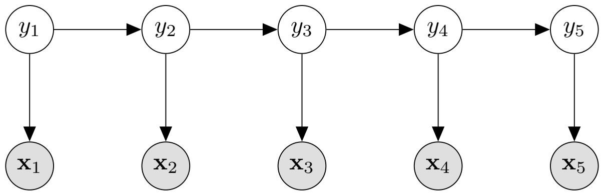
Task 1

Find the probability of all *possible paths* $y_{1:5} = y_1, \dots, y_5$ that the rat may have taken, i.e., obtain

$$P(Y_{1:5}|\mathbf{x}_{1:5})$$

Where to implement this: function `P_Ho` (`H` for pat`H` ($\equiv y_{1:5}$), and `o` for set of `o`bservations ($\equiv \mathbf{x}_{1:5}$)).

Hints: Use Figure 2 to factorize $P(y_{1:5}, \mathbf{x}_{1:5})$. Note that $P(\boldsymbol{X}_t|y_t)$, $P(Y_t|y_{t-1})$, and $P(Y_1)$ are already implemented, as `P_Xy`, `P_Yy`, and `P_Y`. Note that most paths will have probability 0. Note that $P(y_{1:5}, \mathbf{x}_{1:5}) \propto P(y_{1:5}|\mathbf{x}_{1:5})$ and $\frac{1}{Z}P(y_{1:5}, \mathbf{x}_{1:5}) = P(y_{1:5}|\mathbf{x}_{1:5})$ for appropriate Z.



$$\begin{tikzcd}
y_1 \arrow[r] \arrow[d] & y_2 \arrow[r] \arrow[d] & y_3 \arrow[r] \arrow[d] & y_4 \arrow[r] \arrow[d] & y_5 \arrow[d] \\
\mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \mathbf{x}_4 & \mathbf{x}_5
\end{tikzcd}$$

Figure 2: Bayesian Network representing $P(y_{1:5}, \mathbf{x}_{1:5})$.

Task 2

Obtain the marginal probability distribution on the final state, with which we can estimate the *most likely position* of the rat at time $t = 5$, and its associated probability, i.e.,

$$\hat{y}_5 = \arg\max_{y_5} P(y_5|\mathbf{x}_{1:5})$$
$$P(\hat{y}_5|\mathbf{x}_{1:5}) = \max_{y_5} P(y_5|\mathbf{x}_{1:5})$$

Where to implement this: `P_Yo` (`Y` $\equiv Y_5$) should provide the distribution.

Hints: Marginalize out $Y_1, \dots, Y_4$ from the result that you obtained in Task 1.

Task 3

Calculate the expected reward for each action, $Q(a) = \mathbb{E}[r(Y_5, a)|\mathbf{x}_{1:5}] \mid \forall a$ and use this to decide on the best action to take, i.e.,

$$a^* = \arg\max_{a \in \{0,1,\dots,25\}} \mathbb{E}[(r(Y_5, a|\mathbf{x}_{1:5}))]$$

Where to implement this: In function `Q_A` (the values), and function `act` (for deciding the action).

Hints: The expectation models uncertainty regarding the true (unknown) state Y_5. Fig. 3 (left) shows an influence diagram. Fig. 3 (right) shows the outcome under an optimal action with a maximal reward. The reward function is described above (and already implemented for you in `env.rwd`).

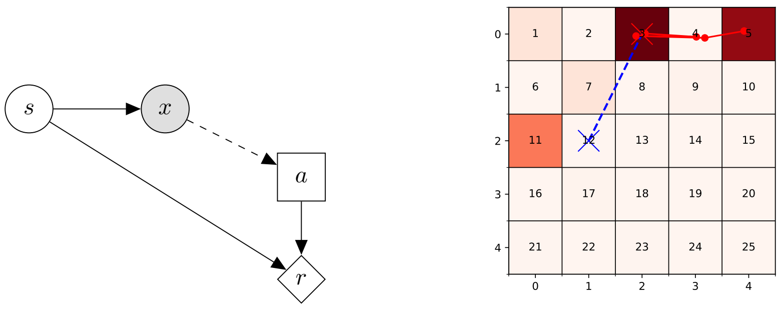


Figure 3: Influence diagram for decision making (left), and an example outcome (right) where darker shades = more confident (more expected reward); final (true) target position y_5 is denoted by $\times$.

Bonus Tasks

1. Consider a stochastic reward proportional to distance: the catch rate is `max(0.95 - dist*0.05,0)` where `dist` is the distance pounced. You would need to change the reward function to test this.
2. Consider environments of different dimensions and dynamics, and how it affects performance as well as the computational complexity of your implementation.

If done correctly, neither of these tasks should interfere with the performance of your agent on the original `Environment`.