Reinforcement

CS786

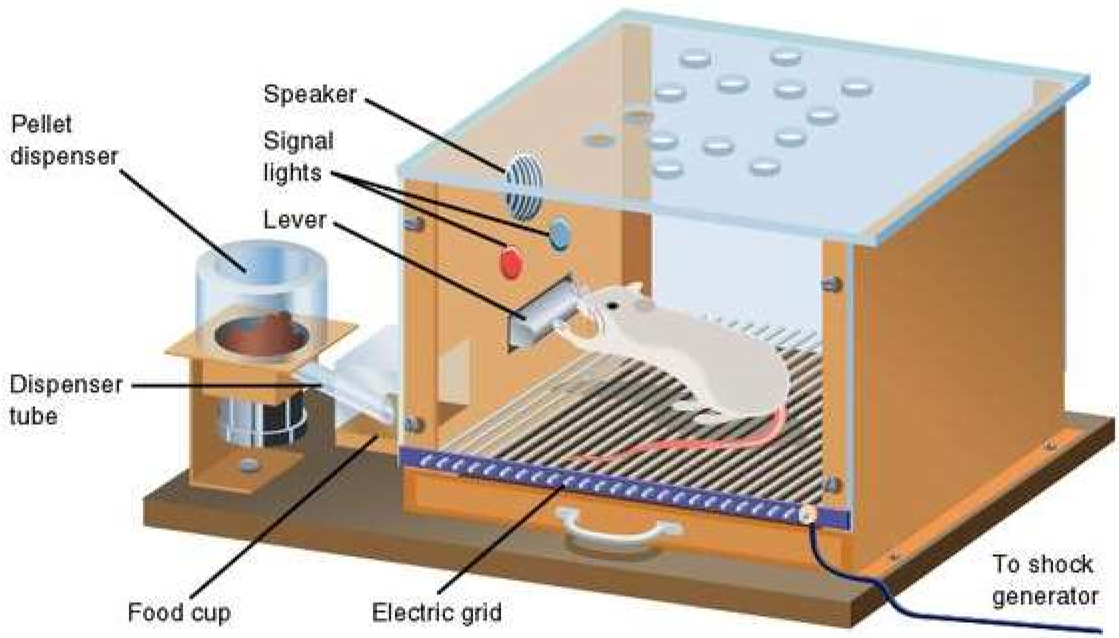
20th August 2024

Association vs reinforcement

- Association: things that occur together in the world, occur together in the mind
  - Tested using classical conditioning
  - Environment acts on the observer
- Reinforcement: actions that are rewarded become desirable in future
  - Tested using operant/instrumental conditioning
  - Observer acts on the environment

Operant conditioning

- Observers act upon the world, and face consequences
  - Consequences can be interpreted as rewards



Modeling classical conditioning

- Most popular approach for years was the Rescorla-Wagner model

$$\Delta V_X^{t+1} = \alpha_X \; \beta(\lambda - V_{tot}),$$

$$V_X^{t+1} = V_X^t + \Delta V_X^{t+1}$$

Some versions replace V_{tot} with V_x; what is the difference?

- Could reproduce a number of empirical observations in classical conditioning experiments

http://users.ipfw.edu/abbott/314/Rescorla2.htm

Can modify to accommodate reward prediction

- Original equation
  - Update size based on *associative strength* available

$$V_X^{n+1} = V_X^n + \alpha(\lambda - V_{tot})$$

- Bush-Mosteller model of reinforcement, for action *a*

$$V_a^{n+1} = V_a^n + \alpha(R^n - V_a^n)$$

The MDP framework

- An MDP is the tuple {S,A,R,P}
  - Set of states (S)
  - Set of actions (A)
  - Possible rewards (R) for each {s,a} combination
  - P(s'|s,a) is the probability of reaching state s' given you took action a while in state s

An example MDP

- States: hungry, taste-deprived, full, happy, unhappy
- Actions: go to hostel mess, delivery from restaurant, make Maggi
- Reward(state, action)
  - R(hungry, mess) = 10
  - R(taste-deprived, mess) = -100
- State transition probability:
- Hungry to full, maggi = 0.4
- Taste-deprived to happy, mess = 0

Solution strategy

- Update value and action policy iteratively

$$AP(s) := \arg\max_a \{\sum_{s'} P(s'|s,a)(R(s',a) + \gamma V(s'))\}$$

$$V(s) := \sum_{s'} P(s'|s, AP(s))(R(s', AP(s)) + \gamma V(s'))$$

https://towardsdatascience.com/getting-started-with-markov-decision-processes-reinforcement-learning-ada7b4572ffb

Solving an MDP

- Solving an MDP is equivalent to finding an action policy AP(s)
  - Tells you what action to take whenever you reach a state s
  - Typical rational solution is to maximize future-discounted expected reward

$$\arg\max_{AP(s_t)} \sum_{t=0}^{\infty} \gamma^t R_{a^t}(s_t)$$

Solution strategy

- Notation:
  - P(s’|s,a) is the probability of moving to s’ from s via action a
  - R(s’,a) is the reward received for reaching state s’ via action a
- Update value and action policy iteratively

$$AP(s) := \arg\max_a \{\sum_{s'} P(s'|s,a)(R(s',a) + \gamma V(s'))\}$$

$$V(s) := \sum_{s'} P(s'|s,AP(s))(R(s',AP(s)) + \gamma V(s'))$$

https://towardsdatascience.com/getting-started-with-markov-decision-processes-reinforcement-learning-ada7b4572ffb

Part of a larger universe of AI models

Control over actions?

States observable?

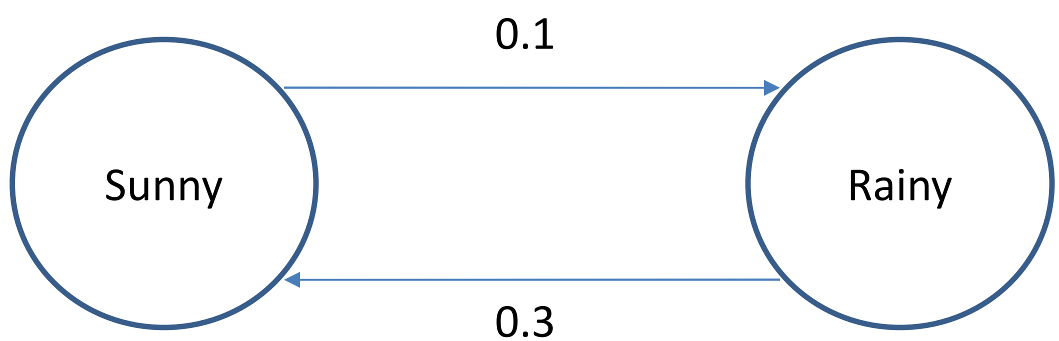
	No	Yes
No	HMM	POMDP
Yes	Markov chain	MDP

Modeling human decisions?

- States are seldom nicely conceptualized in the real world
- Where do rewards come from?
- Storing transition probabilities is hard
- Do people really look ahead into the infinite time horizon?

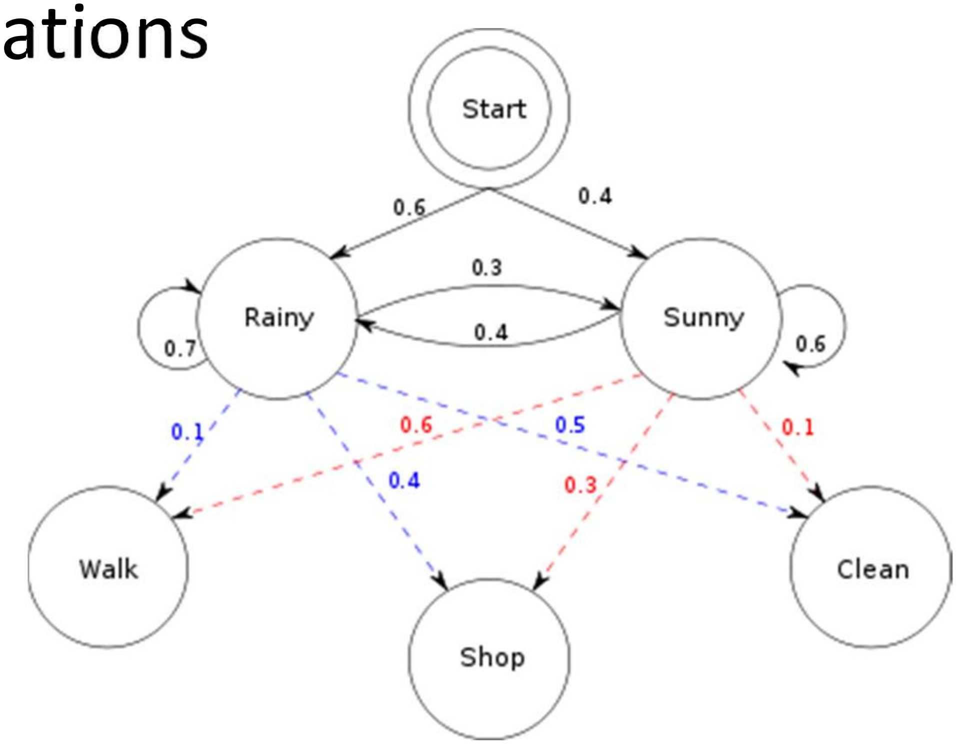
Markov chain

- Goal in solving a Markov chain
  - Identify the stationary distribution



HMM

- HMM goal → estimate latent state transition and emission probability from sequence of observations



MDP → RL

- In MDP, {S,A,R,P} are known
- In RL, R and P are not known to begin with
- They are *learned* from experience
- Optimal policy is updated sequentially to account for increased information about rewards and transition probabilities
- Model-based RL
  - Learns transition probabilities P as well as optimal policy
- Model-free RL
  - Learns only optimal policy, not the transition probabilities P