

UBC MLRG (Winter 2017): Reinforcement Learning

Why Reinforcement Learning?



<https://www.youtube.com/watch?v=Ih8EfvOzBOY>

<https://www.youtube.com/watch?v=SH3bADiB7uQ>

<https://www.youtube.com/watch?v=nUQsRPJ1dYw>

Building up to Reinforcement Learning

- Reinforcement learning (RL) is **very general/difficult**:
 - It includes many other machine learning problems as special cases.
- Good introductory book:
 - “Introduction to Reinforcement Learning” by Sutton & Barto.
- Other names for reinforcement learning:
 - Approximate dynamic programming.
 - Neurodynamic programming.
 - Control theory.
- To build up to RL, let’s start with supervised learning:
 - Introduce notation, and discuss ways RL is harder.

Supervised Learning

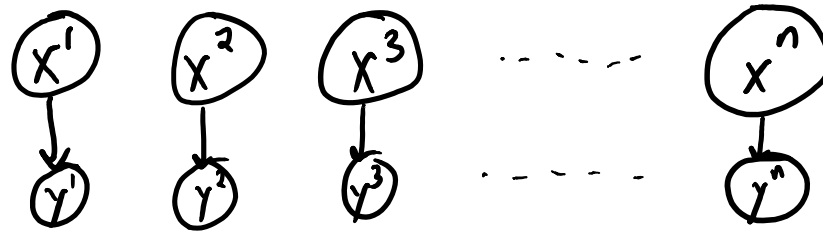
- **Supervised learning** notation:
 - We have input **features** x^t .
 - There are possible **outputs** y^t .
 - We have a **loss function** $L(x^t, y^t)$.
 - E.g., loss of 0 if you classify correctly and loss of 1 if you classify incorrectly.
- **Reinforcement learning** notation:
 - The features are referred to as **states** s^t .
 - The outputs are referred to as **actions** a^t .
 - The (negative) loss function is called the **reward** $r(s^t, a^t)$.
 - E.g., reward of 0 if you classify correctly and reward of -1 if you classify incorrectly.

Supervised Learning

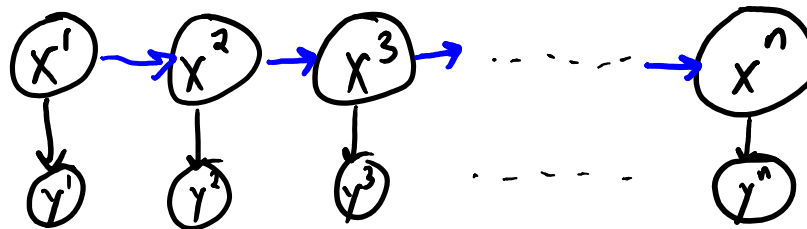
- **Supervised learning** training phase:
 - We have ‘n’ training examples, we can do whatever we want with them.
 - The output of training is a **classifier**: maps from x^t to y^t .
 - This is called a **policy** in RL: policies map from s^t to a^t .
- Goal: classifier minimizes loss \Leftrightarrow policy maximizes reward
- Some models give **score for each label**:
 - For example, softmax gives probability of each y^t given x^t .
 - This is a **Q function**: $Q(s^t, a^t)$ is “value” of action a^t in state s^t .
 - Given a policy, we can define the **value function** $V(s^t)$ as “value” given policy (which may be deterministic or stochastic).

State-Space Models

- In standard setup, the x^t are **IID samples**:



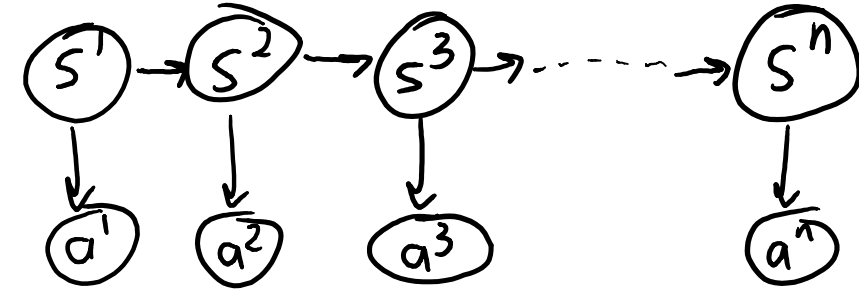
- In state-space models, the x^t come from a **Markov chain**:



- Value of x^t depends on the value of x^{t-1} .
- We obtain IID samples in the special case of no dependencies.
- Learning in this full-observed DAG is pretty similar.

Markov Decision Processes

- State-space model in RL notation



- In Markov decision processes (MDPs), s^t also depends on a^{t-1} .

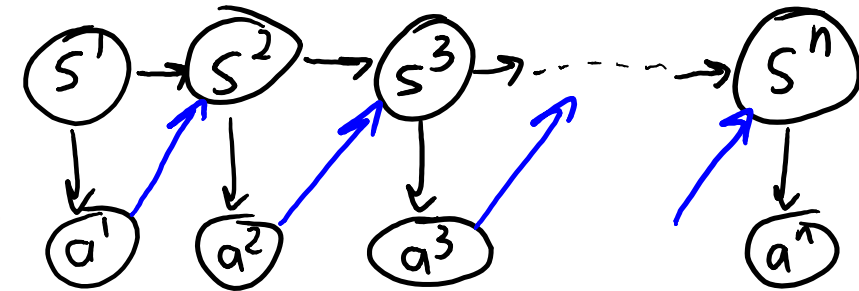
- The action affects the value of the next state.

- Here we need planning:

- Choose actions that will lead to future states with high reward.

- In MDPs we assume we have the “model”:

- Know all rewards $r(s^t, a^t)$ and transition probabilities $p(s^t | s^{t-1}, a^{t-1})$.



- Give “model”, we can find optimal values/policy by dynamic programming:

- [Value iteration and policy iteration](#) (next week).

Reinforcement Learning

- Reinforcement learning is MDPs when we don't know the "model".
 - All we can do is take actions and observe states/rewards that result.
- We need to simultaneously solve three problems:
 - We need to solve a supervised learning problem, $r(s^t, a^t)$.
 - We need to discover dynamics of a state-space model, $p(s^t | s^{t-1}, a^{t-1})$.
 - We need to plan an MDP policy maximizing long-term reward, $s^t \rightarrow a^t$.
- All while working with simulations.
- Unfortunately, this combination gives a few more challenges...

Active Learning

- Let's go back to the **basic supervised learning** setting:
 - Features s^t are just IID samples.
- **Active learning** considers the following variation:
 - The training **examples are unlabeled**.
 - The learner can **query the user to label** a training example s^t .
 - Goal is to do well with a **fixed budget** of queries.
- The fixed budget means we can't visit all features/states.
 - Here we need **exploration**: which states do we visit to learn the most?

Online Learning and Bandit Feedback

- In **online learning** there is **no separate training/testing** phase:
 - We receive a sequence of features/states s^t .
 - We have to choose prediction/action a^t on each example as it arrives.
 - Our “score” is the average loss/reward over time.
 - Here we need to **predict well as we go** (not at the end).
 - You **pay a penalty for trying bad actions** as you are learning.
- A common variation is with **bandit feedback**:
 - We **only observe the reward function $r(s^t, a^t)$ for actions a^t that we choose.**
 - Here we have an **exploration vs. exploitation trade-off**:
 - Should we explore by picking an a^t we don't know much about?
 - Should we exploit by picking an a^t that gives high reward?

Causal Learning

- Causal learning:
 - Observational prediction:
 - Do people who take Cold-FX have shorter colds?
 - Causal prediction:
 - Does taking Cold-FX cause you to have shorter colds?
 - Counter-factual prediction:
 - You didn't take Cold-FX and had long cold, would taking it have made it shorter?
- Here we need to learn effects of actions.
 - Including predicting effects of new actions.
- We may not control the actions: off-policy learning.
 - Actions are often randomized, but still want to find best actions.

Reinforcement Learning

- Reinforcement learning is MDPs when we don't know the "model".
 - All we can do is take actions and observe states/rewards that result.
- We need to consider:
 - Modeling how (s^t, a^t) combinations affects reward (supervised learning)
 - Learning how (s^t, a^t) affects s^{t+1} (state-space models, causality).
 - Planning for long-term reward (MDPs).
 - Exploring space of states and actions (active learning, bandit feedback).
- Two common frameworks:
 - Monte Carlo methods collects a lot of simulations to turn it into an MDP.
 - Temporal-difference learning considers online prediction as you go.
 - Need to consider exploration vs. exploitation, penalties for trying bad actions.

Related Problems

- **Inverse reinforcement learning**, apprenticeship learning, etc.:
 - Learning from an expert **without an explicit reward function**.
- **Hidden state-space models**:
 - The actual state is hidden, and x^t is just an observation based on the state.
 - Hidden Markov models, Kalman filters, LQR control.
- **Partially-observed MDPs (POMDPs)**:
 - MDPs and reinforcement with hidden state-space model.
 - Hard even when you know the “model”.

Schedule

Date	Topic	Presenter
Jan 10	Motivation/Overview	Mark
Jan 17	MDPs (policy iteration, value iteration)	Nasim
Jan 24	Monte Carlo (estimators, on-policy/off-policy learning)	Julie
Jan 31	TD learning, eligibility traces (least squares TD)	Raunak
Feb 7	Sarsa, Q-learning	Jennifer
Feb 14	Functional approximation, TD-Gammon, ATARI	Michael
Feb 21	Planning, temporal abstraction	Ricky
Feb 28	Policy gradient, Monte-Carlo tree-search, AlphaGo	Stephen
Mar 7	Optimal control, flying helicopters	Issam
Mar 14	POMDPs part 1	Sharan
Mar 21	POMDPs part 2	Jason
Mar 28	Value-Iteration Networks	Julieta
April 4	RL in Practice	Glen