Exploration

reading: Szepesvári 4.2
The exploration challenge

- 3 core challenges of RL: temporal credit assignment, generalization, and exploration
- We’ve studied the first two; how about exploration?
- But what is exploration?
- In other words, if I give you two algorithms, how would you decide which one explores better?
Evaluation metrics for good exploration

• Assume episodic RL for simplicity
• Interaction protocol: For episode $t=1,2,\ldots, T$
  • Learner generates an episode using policy $\pi_t$
  • $\pi_t$ may be chosen according to previous data
• Pure exploration: After $T$ rounds, learner outputs a policy $\hat{\pi}$
• Metric: $J(\pi^*) - J(\hat{\pi})$ (recall that $J(\pi) := \mathbb{E}_{s \sim d_0}[V^\pi(s)]$)
• This is a random variable! We want alg to perform well most of the times, so we often look at the $(1-\delta)$-quantile of this r.v.
• Equivalently: to guarantee that $J(\pi^*) - J(\hat{\pi}) \leq \epsilon$ with probability at least $1-\delta$, how large $T$ needs to be (as a function of $\epsilon, \delta$, and other problem-dependent parameters)
• Such $T$ is called the sample complexity of the algorithm
Evaluation metrics for good exploration

• In pure exploration, we care about getting a good policy at the end of training. So $t \leq T$ is “training”, and after that it’s test phase
• Training/test distinction may not exist in some applications, where we just want to continuously improve the performance online
• This is the exploration-exploitation setting
• Evaluation metric: (cumulative & pseudo) regret $\sum_{t=1}^{T} (J(\pi^*) - J(\pi_t))$
• Also a r.v.; people consider both high-probability regret (i.e., $(1-\delta)$-quantile) and expected regret
• If algorithm keeps improving and get closer and closer to $\nu^*$, we should obtain sublinear regret: $\sum_{t=1}^{T} J(\pi^*) - J(\pi_t) = o(T)$
  • In the limit, the average regret $\frac{1}{T} \sum_{t=1}^{T} (J(\pi^*) - J(\pi_t)) \to 0$, “no-regret”
Comments on sample complexity and regret

• If you care about final performance, measure sample complexity; if you care about continuous improvement, measure regret.

• You might notice #1: we never mention the word “exploration” in these definitions!
  • unless you make explicit assumptions to avoid it, exploration is a natural and inherent part of RL; no need to call out!

• You might notice #2: we wanted to focus solely on the exploration challenge, but end up posing the entire learning problem…
  • empirical methods often perform exploration and learning (e.g., temporal credit assignment) separately
  • however, it turns out that to systematically explore (and get provable guarantees), you often cannot separate exploration and learning—they need to depend on each other.
Uniform exploration in multi-armed bandits

• MAB: finite-horizon MDP with $H = 1$ (i.e., contextual bandits) and a single (deterministic) starting state. Reward is stochastic

• Assume that reward lies in $[0, 1]$. Let the expected reward for action $i$ be $\mu_i$, and the highest one be $\mu^* = \mu_{i^*}$ (so $J(\pi^*) = \mu^*$)

• A simple algorithm for exploration: for $t=1,2,\ldots,T$
  • Choose action No. $(t \bmod |A|)$, and observe random reward
  • Finally, let $\hat{\mu}_i$ be the average over rewards from action $i$
  • Output the action $\hat{i}$ with the highest $\hat{\mu}_i$ (so $J(\hat{\pi}) = \mu_{\hat{i}}$)

• Sample complexity of this algorithm: $O\left(\frac{|A|}{\epsilon^2} \ln \frac{|A|}{\delta}\right)$
  • The $\ln |A|$ factor can be improved by more clever alg
Uniform exploration in Contextual Bandits

- CB: finite-horizon MDP with $H = 1$. Starting state is random, and typically state space is large (i.e., cannot do tabular)
- Need function approximation: consider policy-based methods
- Assume a class of policies $\Pi$
  - Recall parametrized policy in PG; for now we assume $\Pi$ is finite but can be exponentially large (you only want to pay $\log |\Pi|$)
  - No further assumption (e.g., $\pi^* \in \Pi$). Instead of requiring learner to achieve $J(\pi^*)$, only require it to achieve $\max_{\pi \in \Pi} J(\pi)$
- Naive alg: treat each policy as an “meta-action”, $O\left(\frac{|\Pi|}{e^2 \ln \frac{|\Pi|}{\delta}}\right)$
- Can do much better: $O\left(\frac{|A|}{e^2} \ln \frac{|\Pi|}{\delta}\right)$
  - Uniform exploration + importance sampling
  - $|A|$ comes from: importance weight blows up range of variable from $[0, 1]$ to $[0, |A|]$
Exploration and exploitation

- If we have an algorithm that achieves $\frac{C}{c^2}$ sample complexity (C absorbs all the other quantities), can we get a no-regret alg?
- Explore-then-exploit: given $T$ rounds/episodes, explore for $T_1$ rounds, then deploy the learned policy for the rest of rounds
- Regret bound:
  - $\sum_{t=1}^{T_1} (J(\pi^*) - J(\pi_t)) \leq T_1$: assume we get nothing during exploration
  - We spend $T_1$ rounds exploring: back out $\epsilon = \sqrt{\frac{C}{T_1}}$
  - $\sum_{t=T_1+1}^{T} (J(\pi^*) - J(\pi_t)) \leq \sqrt{\frac{C}{T_1}}(T - T_1)$
  - Combine the two and optimize $T_1$: $O(C^{1/3}T^{2/3})$
  - Typically suboptimal when $T$ is large; optimal algorithm scales as $\sqrt{T}$
Exploration and exploitation

• When the exploration algorithm (during $T_1$) is uniform, the full algorithm is sometimes called “epoch greedy”
• Has similar properties to epsilon-greedy
Exploration and exploitation

• Example of a popular algorithm for regret minimization: UCB1 (Auer et al’02)

• At any round $t$,
  • let $n_t(a)$ be the number of times we’ve chosen $a$ so far
  • let $r_t(a)$ be the empirical average of rewards from $a$
  
  Define $U_t(a) := r_t(a) + \mathcal{R} \sqrt{\frac{2 \log t}{n_t(a)}}$, where $\mathcal{R}$ is the range of reward

  • Choose the action greedily w.r.t. $U_t(\cdot)$

• UCB stands for “Upper confidence bound”: can show that $\mu_a \leq U_t(a)$ for all $t$ simultaneously with high probability

• Bonus term (2nd term) drops if action is taken more ($n_t(a) \uparrow$)

• Never stop exploring any action ($\log t \uparrow$)

• Main principle for exploration: optimism in face of uncertainty
Exploration in MDPs

• So far we’ve focused on exploration in (multi-arm or contextual) bandits.
• In bandits, we’ve seen that taking uniformly random actions can be quite effective
• How about MDPs?
Random exploration can be inefficient

visited in $2^{-H}$ fraction of all trajectories

Freeway (one of the Atari games)

“Freeway + RL”: https://youtu.be/44CilPmlimQ
Exploration in MDPs

• The construction is called “combination lock”
  • Ultimate killer examples for most heuristic exploration strategies
  • e.g., epsilon greedy, softmax, policy gradient, …
• Why difficult?
  • Consider searching over a complete tree, where only one leaf is rewarding (marked red)
  • Obvious lower bound: you need to try (almost) all the paths
• A variant of comb lock
  • If the exploration strategy does not leverage state identity, no way to distinguish between comb lock vs exp tree
  • fun fact: they are bisimilar
Exploration in MDPs: Deterministic case

- If the MDP is fully deterministic, how can we explore efficiently?
- Exploration = visit each state-action pair reachable from initial state once
- Goal: in each episode, visit a new state-action pair. This way we are done in |SxA| episodes.
  - argument adapted from Szepesvári Sec 4.2.3
- Observation 1: there always exists some states visited in previous episodes that have unexplored actions
- Observation 2: from previous data, we know how to get to those states!
- Algorithm: build a partial MDP over visited states. choose any state with unexplored actions, get to that state by planning in the partial MDP, then take the action.
- Deterministic transition + stochastic reward: visit each (s,a) enough times such that reward estimation is accurate enough
Exploration in MDPs: Extending to stochastic case

• Optimism-based interpretation of the previous algorithm:
  • In round (episode) $t$, define the following MDP $M_t$
  • For $(s, a)$ visited before, transition & reward is the same as in $M$
  • Otherwise: transition to a special chain of “heaven” states (which don’t exist in $M$) that gives maximum reward $R_{\text{max}}$ each time step before termination
  • Explore by using the optimal policy of $M_t$
  • Optimism: imagine the best for unexplored state-action pairs; mathematically, we have $\forall \pi, J_M(\pi) \leq J_{M_t}(\pi)$

• Extend the idea to stochastic MDPs: R-max [Brafman & Tennenholtz’02]
  • Define $M_t$ similarly: if $(s, a)$ has been visited sufficient number of times, use the empirical estimation of transition and reward in $M_t$; otherwise it transitions to the “heaven” states
  • Explore by using the optimal policy of $M_t$
Exploration in large MDPs

- Literature on exploration in tabular MDPs with polynomial sample complexity is sometimes referred to as PAC-MDP
  - typically, $\text{poly}(|S|, |A|, H)$ (ignoring PAC parameters $\epsilon, \delta$)
- Why don’t we use PAC-MDP algorithms in practice?
  - $|S|$ is too large
  - PAC algorithm strongly rely on state identity
  - i.e., You tell whether a state is novel by comparing it with previously visited states
  - Identity is meaningless in large problems: you may never see the same state twice!
- PAC-RL for function approximation?
  - Assume we are given value-function class $F$ to model $Q^*$
  - Goal: $\text{poly}(\log|F|, |A|, H)$ sample complexity
  - There are hardness results showing that this is impossible [Krishnamurthy et al’16; Jiang et al’17; Du et al’19]
Exploration in large MDPs

• Implication of hardness of exploration with function approximation
  • Cannot efficiently explore in unstructured environments even with the help of good function approximation
  • Need to consider structured environments
• What kind of structures enable sample-efficient exploration in RL?
Zoo of RL Exploration

Finite MDPs [Kearns & Singh’98] (small #states)
Metric space [Kakade et al’03]
Abstraction [Li’09] (small #abstract states)
LQR control [Ibrahimi et al’12] (small #variables)

[P(x'|x,a) = x]

MDPs w/ low-rank transition matrix [Barreto et al’11] (small matrix rank)

POMDPs w/ rich observation and reactive value function (small #hidden-states)
Deterministics + [Krishnamurthy et al’16]

Same setup in PSRs [Littman et al’02] (small system dim.)

All these settings yield low Bellman rank
Unified algorithm, polynomial guarantee

New