Fraunhofer IIS

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG TECHNISCHE FAKULTÄT

Exploration in Deep RL

Christopher Mutschler

Agenda

- Motivation, Problem Definition & Multi-Armed Bandits
- Classic Exploration Strategies
  - Epsilon Greedy
  - (Bayesian) Upper Confidence Bounds
  - Thomson Sampling
- **Exploration in Deep RL**
  - Count-based Exploration: Density Models, Hashing
  - Prediction-based Exploration:
    - Forward Dynamics
    - Random Networks
    - Physical Properties
  - Memory-based Exploration:
    - Episodic Memory
    - Direct Exploration
- Summary and Outlook

Key Exploration Problems in Deep RL

1. The “Hard-Exploration” Problem

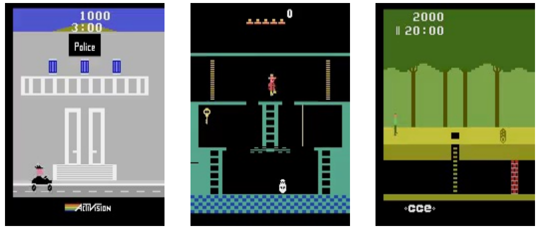
- Exploration in environments with very sparse or even *deceptive* rewards
- Random exploration is prone to failure as it will rarely find successful states or obtain meaningful feedback from the environment
- Montezuma’s Revenge is one of such examples

2. The Noisy-TV Problem

- Initially proposed by Burda et al.[1]: An agent seeks for novelty in the environment and finds a TV with uncontrollable & unpredictable output (e.g., Gaussian noise)
  → this will attract the agent’s attention forever!
- The agent obtains new rewards from the noisy TV consistently but fails to make any meaningful progress and becomes a “couch potato”

Agent in maze with TV

Agent in maze without TV



problems with sticky actions

Images taken from https://openai.com/blog/reinforcement-learning-with-prediction-based-rewards/

[1] Yuri Burda et al.: Exploration by Random Network Distillation. ICLR 2019.

Exploration in Deep RL

- But is it all so much different from what we studied with bandits?
- Recap: classes of exploration methods:
  - Optimistic exploration:
    - A new state is always a good state
    - We must estimate the state visitation frequencies or novelty
    - Typically realized by means of exploration bonuses
  - Thompson sampling style algorithms:
    - Learn distribution over Q-functions or policies
    - Sample and act according to sample
  - Information gain style algorithms
    - We reason about information gain from visiting new states
  - Entropy-loss & Noise-based algorithms:
    - Implicit exploration through induction of noise

We talked about this

Not our focus here

Exploration in Deep RL

- Let us revisit Upper Confidence Bounds:

$$a_t^{UCB} = \arg\max_{a \in \mathcal{A}} Q(a) + \underbrace{c \cdot \sqrt{\frac{2 \log t}{N_t(a)}}}_{\text{Exploration Bonus}}$$

- We can make use of several exploration bonus functions
  (don't worry about all the elements, most important is that it decreases with $N(a)$!)
- Open question: how can we make use of such methods in an MDP?
  - Idea: Count-based Exploration:
    - use $N(s, a)$ or $N(s)$, and
    - add an *exploration bonus!*

Intrinsic Rewards as Exploration Bonuses

- Instead of $r(s,a)$ we provide $r^+(s,a) = r(s,a) + \mathcal{B}\big(N(s)\big)$

decreases with $N(s)$

- We can give this to any model-free agent!
- A general formulation looks like this:

$$r_t = r_t^e + \beta \cdot r_t^i$$

  - β is a hyperparameter that adjusts the balance between exploitation and exploration
  - r_t^e is called the extrinsic reward form the environment at time t
  - r_t^i is called the intrinsic reward, i.e., the exploration bonus at time t

- The intrinsic reward is/can be inspired intrinsic motivation[1] and we can transfer those findings to RL too:
  1. Discovery of novel states
  2. Improvement of the agent's knowledge about the environment

[1] *Pierre-Yves Oudeyer and Frederic Kaplan: How can we define intrinsic motivation? 8th Intl. Conf. Epigenetic Robotics*

Agenda

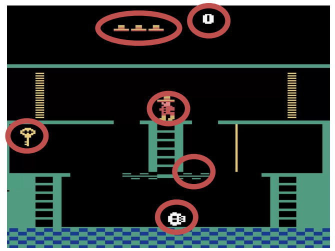
- Motivation, Problem Definition & Multi-Armed Bandits
- Classic Exploration Strategies
  - Epsilon Greedy
  - (Bayesian) Upper Confidence Bounds
  - Thomson Sampling
- **Exploration in Deep RL**
  - **Count-based Exploration: Density Models, Hashing**
  - Prediction-based Exploration:
    - Forward Dynamics
    - Random Networks
    - Physical Properties
  - Memory-based Exploration:
    - Episodic Memory
    - Direct Exploration
- Summary and Outlook

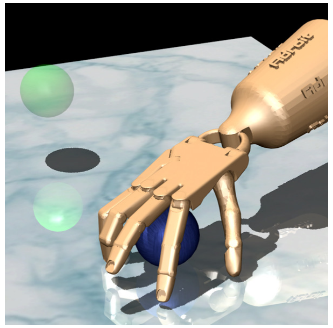
Count-based Exploration

- We want novel states to surprise the agent
  → we need a metric that measures how novel a state appears to us
- Intuitive idea: count how many times we see a particular state & apply the bonus accordingly
  → Let $N_n(s)$ be the empirical count of visits of a state s in the sequence $s_{1:n}$.

- But wait, as we deal with **high-dimensional** or **continuous state spaces**, we still have 2 problems:
  1. Many states we will never see at all
  2. Many states we will never see again

  → So, what is a "count" after all?
  → Counting will become somehow "useless"…





- Side-note: we need a non-zero count for most states, even if we haven't seen them before

- But some states are more similar than others!
  → This might be useful to exploit!

Count-based Exploration: Density Models

- Idea: Density Models
  1. Fit a density model $p(s;\theta)$ to approximate the frequency of visits
  2. Derive a pseudo count from the model

true density at time step T:

$$p(s) = \frac{N(s)}{n}$$

$\longrightarrow$

true density at time step T+1 after observing s:

$$p'(s) = \frac{N(s)+1}{n+1}$$

- Can we get $p(s;\theta)$ and $p(s;\theta')$ to satisfy the above?

Marc G. Bellemare et al.: Unifying Count-Based Exploration and Intrinsic Motivation. NIPS 2016.

Count-based Exploration: Density Models

- Idea: Density Models
  1. Fit a density model $p(s;\theta)$ to approximate the frequency of visits
  2. Derive a pseudo count from the model

true density at time step T:

$$p(s;\theta) = \frac{\hat{N}(s)}{\hat{n}}$$

true density at time step T+1 after observing s:

$$p(s;\theta') = \frac{\hat{N}(s)+1}{\hat{n}+1}$$

$\rightarrow \hat{N}(s) = \hat{n}p(s;\theta)$

$\rightarrow \hat{n} = \frac{\hat{N}(s)+1-p(s;\theta')}{p(s;\theta')}$

- Can we get $p(s;\theta)$ and $p(s;\theta')$ to satisfy the above?
- Sure:
  1. Fit model $p(s;\theta)$ to all states $\mathcal{D}$ seen so far
  2. Take a step T and observe s_T
  3. Fit new model $p(s;\theta')$ to $\mathcal{D} \cup s_T$
  4. Use $p(s;\theta)$ and $p(s;\theta')$ to estimate $\boxed{\hat{N}(s)}$
  5. Set $r_i^+ = r_i + \mathcal{B}\left(\hat{N}(s)\right)$
  6. Repeat.

$\rightarrow \hat{N}(s) = \frac{p(s;\theta)[1-p(s;\theta')]}{p(s;\theta')-p(s;\theta)}$

but how?

→ solve linear system

Marc G. Bellemare et al.: Unifying Count-Based Exploration and Intrinsic Motivation. NIPS 2016.

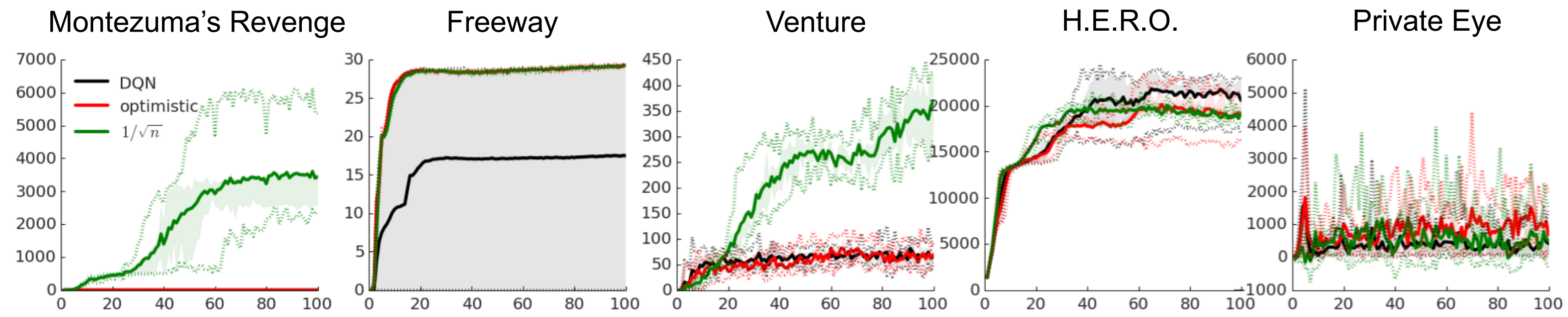
Count-based Exploration: Density Models

- **Open issue #1: What bonus $\mathcal{B}(\hat{N}(s))$ could we choose?**
  - Upper Confidence Bounds: $\mathcal{B}(\hat{N}(s)) = \sqrt{\frac{2\log t}{\hat{N}(s)}}$
  - MBIE-EB[1,2]: $\mathcal{B}(\hat{N}(s)) = \sqrt{\frac{1}{\hat{N}(s)}}$
  - BEB[3]: $\mathcal{B}(\hat{N}(s)) = \frac{1}{1+\hat{N}(s)}$

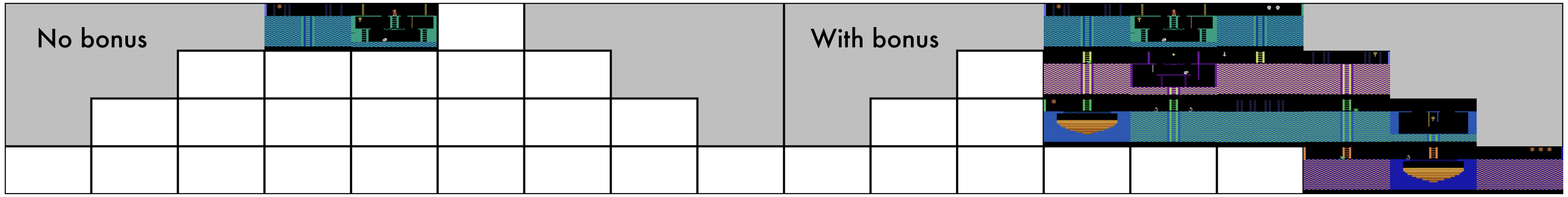
- **Open issue #2: What density model could we choose?**
  - Note: we only need rough densities (no need for accuracy or normalization, and we do also not need to sample from it (such in GANs or VAEs!) – it only needs to get up for states that have higher density
  - Context Switching Trees (CTS)[2,4]
  - PixelCNN[5,6]
  - Gaussian Mixture Models (GMM)[7]
  - …

[1] *Strehl & Littman: An analysis of model-based Interval Estimation for Markov Decision Processes. 2008.*
[2] *Marc G. Bellemare et al.: Unifying Count-Based Exploration and Intrinsic Motivation. NIPS 2016.*
[3] *Kolter & Ng: Near-Bayesian Exploration in Polynomial Time. ICML 2009.*
[4] *Marc G. Bellemare et al.: Skip Context Tree Switching. ICML 2014.*
[5] *Georg Ostrovski et al.: Count-Based Exploration with Neural Density Models. ICML 2017.*
[6] *Arron van den Oord et al.: Conditional Image Generation with PixelCNN Decoders. NIPS 2016.*
[7] *Zhao & Tresp: Curiosity-Driven Experience Prioritization via Density Estimation. NIPS Deep RL Workshop. 2018.*

Count-based Exploration: Density Models



Average training score with and without exploration bonus or optimistic initialization in 5 Atari 2600 games. Shaded areas denote inter-quartile range, dotted lines show min/max scores



"Known world" of a DQN agent trained for 50 million frames with (right) and without (left) count-based exploration bonuses, in Montezuma's Revenge.

Marc G. Bellemare et al.: Unifying Count-Based Exploration and Intrinsic Motivation. NIPS 2016.

Counting after Hashing

- Alternative idea:
  - Map high-dimensional states into a k-bit hash code via $\phi(s)$ and count $N(\phi(s))$ instead of $N(s)$

  → Shorter codes = more hash collisions

  → Similar states = similar hashes?

- **Locality-Sensitive Hashing (LSH)**[1]
  - Hashing scheme that preserves the distancing information between data points
    → close vectors obtain similar hashes, distant vectors have different hashes
  - SimHash[2] uses the angular distance to measure similarity:

$$\phi(s) = \mathrm{sgn}(Ag(s)) \in \{-1,1\}^k, \text{ where}$$

    - $A \in \mathbb{R}^{k \times D}$ is a matrix with each entry drawn from $\mathcal{N}(0,1)$, and
    - $g \colon \mathcal{S} \to \mathbb{R}^D$ is an optional preprocessing function
  - Larger k's lead to higher granularity and fewer collisions.

[1] Haoran Tang et al.: Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. NIPS 2017.
[2] Moses Charikar: Similarity Estimation Techniques from Rounding Algorithms. STOC 2002.

Counting after Hashing

- Alternative idea:
  - Map high-dimensional states into a k-bit hash code via $\phi(s)$ and count $N(\phi(s))$ instead of $N(s)$

  → Shorter codes = more hash collisions

  → Similar states = similar hashes?

- **Locality-Sensitive Hashing (LSH)**[1]

Algorithm 1: Count-based exploration through static hashing, using SimHash

1 Define state preprocessor $g : \mathcal{S} \to \mathbb{R}^D$
2 (In case of SimHash) Initialize $A \in \mathbb{R}^{k \times D}$ with entries drawn i.i.d. from the standard Gaussian distribution $\mathcal{N}(0, 1)$
3 Initialize a hash table with values $n(\cdot) \equiv 0$
4 **for** each iteration j **do**
5 Collect a set of state-action samples $\{(s_m, a_m)\}_{m=0}^{M}$ with policy π
6 Compute hash codes through any LSH method, e.g., for SimHash, $\phi(s_m) = \mathrm{sgn}(Ag(s_m))$
7 Update the hash table counts $\forall m : 0 \le m \le M$ as $n(\phi(s_m)) \leftarrow n(\phi(s_m)) + 1$
8 Update the policy π using rewards $\left\{ r(s_m, a_m) + \frac{\beta}{\sqrt{n(\phi(s_m))}} \right\}_{m=0}^{M}$ with any RL algorithm

[1] *Haoran Tang et al.: Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. NIPS 2017.*

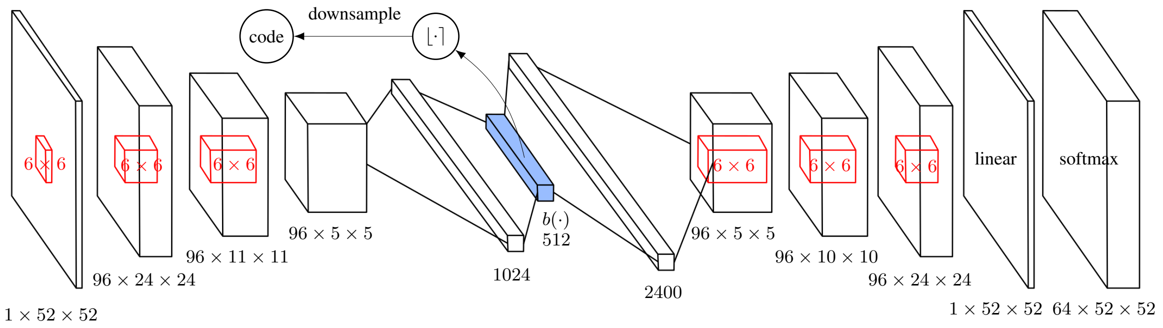
Counting after Hashing

- Alternative idea:
  - Map high-dimensional states into a k-bit hash code via $\phi(s)$ and count $N(\phi(s))$ instead of $N(s)$

  → Shorter codes = more hash collisions

  → Similar states = similar hashes?

- **Learning Hash-Codes**[1]
  - SimHash works poorly on high-dimensional input with complex structure (such as images) as measuring the similarity on pixel-level fails to capture semantic similarity
  - Idea: learn a compression using an autoencoder
  - A special dense layer uses k sigmoid functions in the latent space to generate a binary activation map



[1] *Haoran Tang et al.: Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. NIPS 2017.*

Counting after Hashing

- Alternative idea:
  - Map high-dimensional states into a k-bit hash code via $\phi(s)$ and count $N(\phi(s))$ instead of $N(s)$

  → Shorter codes = more hash collisions

  → Similar states = similar hashes?

- **Learning Hash-Codes[1]**
  - SimHash works poorly on high-dimensional input with complex structure (such as images) as measuring the similarity on pixel-level fails to capture semantic similarity
  - Idea: learn a compression using an autoencoder
  - A special dense layer uses k sigmoid functions in the latent space to generate a binary activation map

$$L\left(\{s_n\}_{n=1}^{N}\right) = -\frac{1}{N}\sum_{n=1}^{N}\Big[\underbrace{\log p(s_n)}_{\text{reconstruction loss}} \underbrace{- \tfrac{\lambda}{K}\textstyle\sum_{i=1}^{D}\min\left\{\left(1-b_i(s_n)\right)^2, b_i(s_n)^2\right\}}_{\text{Sigmoid activation being closer to binary}}\Big],$$

[1] *Haoran Tang et al.: Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. NIPS 2017.*