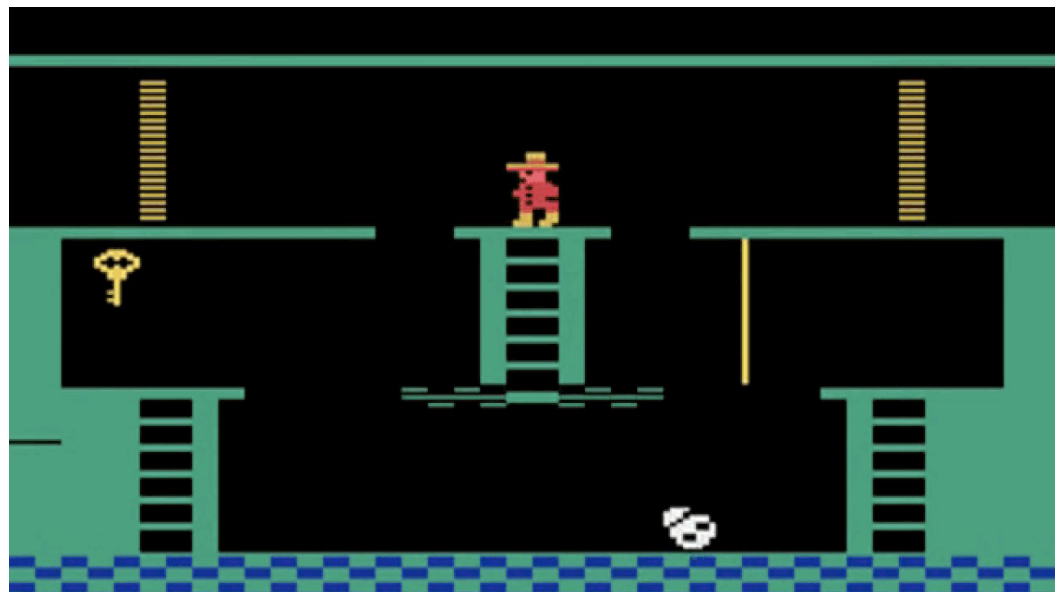


Исследование среды

Аржаев Павел
20 августа 2019

Tinkoff.ru

Задача исследование среды





Exploration и Exploitation

Определения задачи исследования среды:

- Агент должен найти стратегию с высокой наградой, которая, в то же время, может потребовать долгой последовательности действий с низкой наградой.
- Агент должен выбрать между новым поведением и старым поведением с максимальной из исследованных наградой.



Оптимальная стратегия

Возможно ли ее найти?

Как определить оптимальность?

По возрастанию того, насколько сложно может быть явно проанализировать стратегию:

- Многорукий бандит (1-шаговый МП без состояния)
- Многорукий бандит с состоянием
- Небольшой, конечный МП
- Непрерывный, бесконечный МП



Многорукие бандиты

Есть несколько действий, каждое имеет некоторое распределение награды:

$$\mathcal{A} = \{\text{pull}_1, \text{pull}_2, \dots, \text{pull}_n\}$$

$$r(a_n) \sim p(r|a_n)$$

Это распределение параметризовано, мы можем хранить belief об этих параметрах. Можно представить как POMDP со стейтом из этих параметров

$$r(a_i) \sim p_{\theta_i}(r_i) \quad \hat{p}(\theta_1, \dots, \theta_n)$$

Введем понятие regret для измерения оптимальности стратегии:

$$\text{Reg}(T) = TE[r(a^*)] - \sum_{t=1}^T r(a_t)$$

<http://iosband.github.io/2015/07/28/Beat-the-bandit.html>

Если решить этот POMDP, получим оптимальную стратегию исследования.

Однако, есть стратегии значительно проще.

Они асимптотически стремятся к $\text{regret} \sim O(\log(T))$



ОПТИМИЗМ

Будем хранить среднюю награду для действия $\hat{\mu}_a$ |

exploitation: pick $a = \arg \max \hat{\mu}_a$

optimistic estimate: $a = \arg \max \hat{\mu}_a + C\sigma_a$

Идея: пробуем каждое действие, пока не убедимся, что оно не лучшее

Вариант коэффициента (Finite-time Analysis of the Multiarmed Bandit Problem):

$$a = \arg \max \hat{\mu}_a + \sqrt{\frac{2 \ln T}{N(a)}}$$

Другое название - UCB (upper confidence bound)



ОПТИМИЗМ В RL

$$r^+(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \mathcal{B}(N(\mathbf{s})) \quad a = \arg \max \hat{\mu}_a + \sqrt{\frac{2 \ln T}{N(a)}}$$

Вводим модель плотности $p_\theta(\mathbf{s})$

Значение высоко, если состояние близко к уже посещенному

Функцию можно использовать для “подсчета” состояний

В случае дискретного пространства:

$$P(\mathbf{s}) = \frac{N(\mathbf{s})}{n} \quad P'(\mathbf{s}) = \frac{N(\mathbf{s}) + 1}{n + 1}$$

↑
←
←

probability/density
count
total states visited

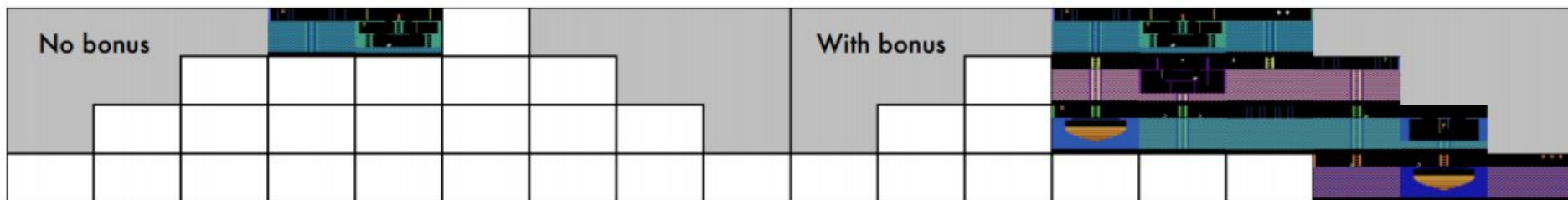
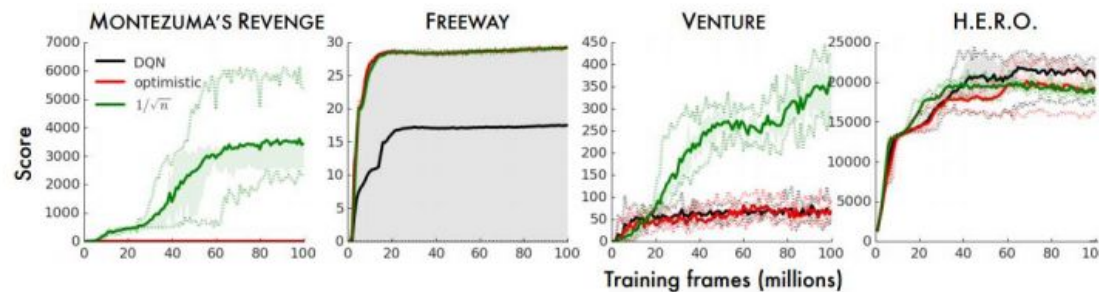
В случае непрерывного пространства

$$p_\theta(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i)}{\hat{n}} \quad p_{\theta'}(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i) + 1}{\hat{n} + 1}$$

$$\hat{N}(\mathbf{s}_i) = \hat{n} p_\theta(\mathbf{s}_i) \quad \hat{n} = \frac{1 - p_{\theta'}(\mathbf{s}_i)}{p_{\theta'}(\mathbf{s}_i) - p_\theta(\mathbf{s}_i)} p_\theta(\mathbf{s}_i)$$



ОПТИМИЗМ В RL



Bellemare et al. "Unifying Count-Based Exploration..."



Оптимизм: ХЭШ

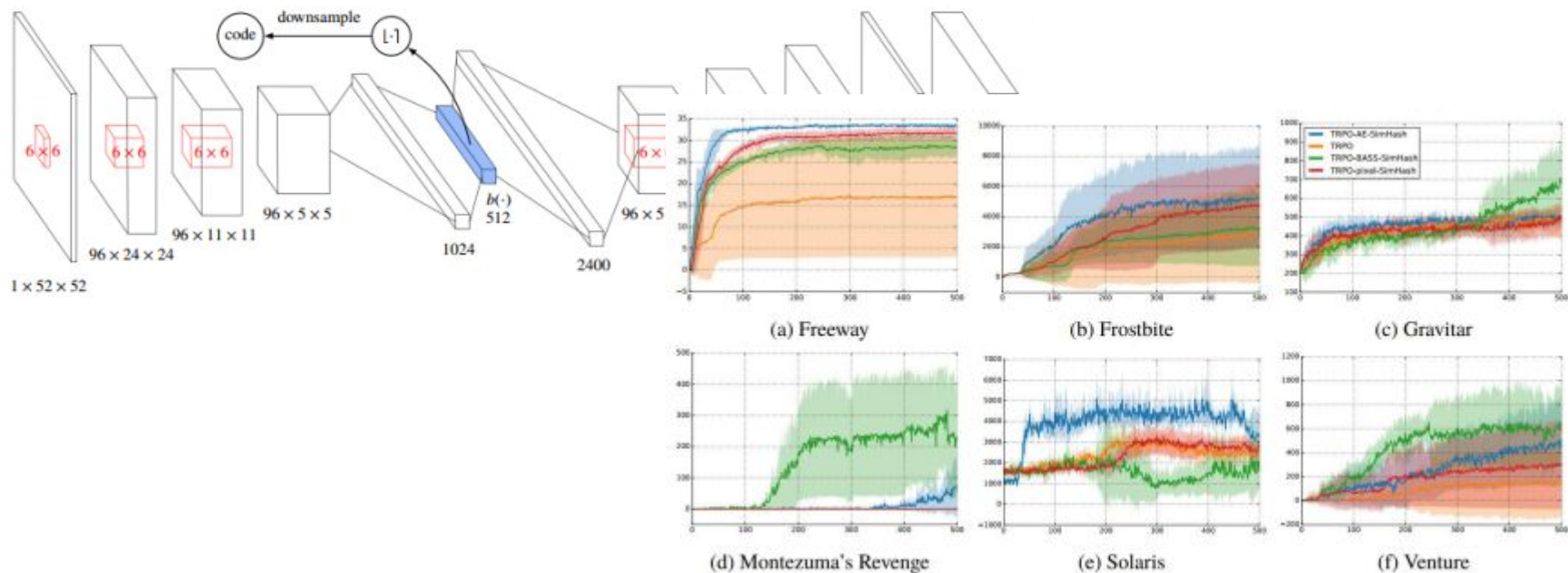
Можно считать посещения состояния, но в другом пространстве

Идея: сжать s с помощью хэш-функции $\phi(s)$, считать $N(\phi(s))$

Стейты с одинаковыми хэшами должны быть близки

Один из вариантов получения такой функции - вариационный автоэнкодер

(Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning)





Оптимизм: модель плотности

Идея: научиться определять, что состояние сильно отличается от ранее посещенных

Можно научить классификатор сравнивать все предыдущие состояния и новое и определять, сильно ли оно отличается.

Вводим два множества:

позитивное: множество, состоящее только из нового состояния

негативное: все предыдущие состояния

Обучим на них классификатор

$D(s)$ - вероятность, что классификатор отнесет новое состояние к позитивному

Говоря о негативном множестве, оптимальный $D(s)$ не равен 1

$$D_s^*(s) = \frac{1}{1 + p(s)}$$

$$p_\theta(s) = \frac{1 - D_s(s)}{D_s(s)}$$



Сэмплирование Томсона

Храним belief о
распределении параметров
наград:

$$\hat{p}(\theta_1, \dots, \theta_n)$$

В model-free RL можно
использовать Q-функцию

Общий алгоритм:

idea: sample $\theta_1, \dots, \theta_n \sim \hat{p}(\theta_1, \dots, \theta_n)$
pretend the model $\theta_1, \dots, \theta_n$ is correct
take the optimal action
update the model

Как представить распределение Q-функций?

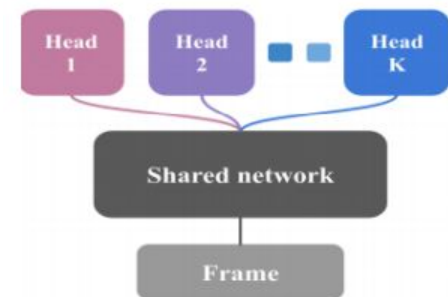
Используем bootstrap ansamble:

- Ресемплить датас \mathcal{D} N раз, получа $\mathcal{D}_1, \dots, \mathcal{D}_N$
- Обучить на каждом полученном датасете f_{θ_i} модель
- При сэмплировании выбрать одну из обученных моделей

Проблема: обучать N тяжелых нейросетей дорого

Решений: разделить лишь некоторые из слоев:

(Osband et al. "Deep Exploration via Bootstrapped DQN")





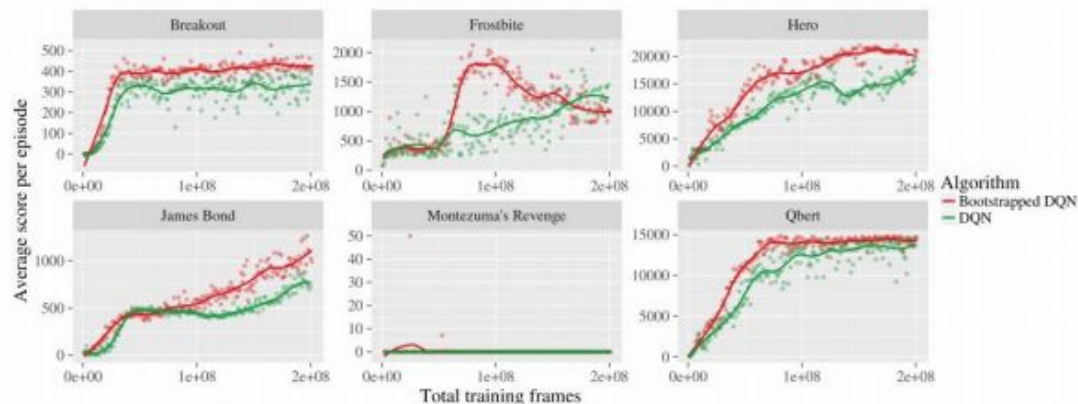
Сэмплирование Томсона

Eps-greedy выбирает либо exploitation, либо действие, независящее от предыдущего

В некоторых задачах это может привести к неразрешимым проблемам

При сэмплировании Томпсона, выбирая случайные Q-функции, однажды мы выберем стратегию, умеющую всплывать (несколько раз плыть вверх)

После того, как одна стратегия научиться всплывать - это распространится на всех





Прирост информации

Байесовский дизайн экспериментов:

Исследуем неизвестную z

Энтропия оценки $\mathcal{H}(\hat{p}(z))$

Энтропия оценки z после наблюдения $\mathcal{H}(\hat{p}(z)|y)$

$$\text{IG}(z, y) = E_y[\mathcal{H}(\hat{p}(z)) - \mathcal{H}(\hat{p}(z)|y)]$$

$$\text{IG}(z, y|a) = E_y[\mathcal{H}(\hat{p}(z)) - \mathcal{H}(\hat{p}(z)|y)|a]$$

Алгоритм для бандитов (Learning to Optimize via Information-Directed Sampling)

$y = r_a$, $z = \theta_a$ (parameters of model $p(r_a)$)

$g(a) = \text{IG}(\theta_a, r_a|a)$ – information gain of a

$\Delta(a) = E[r(a^*) - r(a)]$ – expected suboptimality of a

choose a according to $\arg \min_a \frac{\Delta(a)^2}{g(a)}$