Reinforcement Learning

Intro, Behavior Cloning, CEM

Александр Костин
telegramm: @Ko3tin
LinkedIn: kostinalexander

Критерии оценивания

Домашние задания - 4 штуки:

- 2 простые по 2 балла
- 2 сложные по 4 балла

- Мягкий дедлайн - 1 неделя
- Жесткий дедлайн - 2 недели

Каждый день после мягкого дедлайна снижает оценку на 0.1 \ 0.2 балла для простых и сложных заданий соответственно.

Еще будет **проект-исследование** на 10 баллов.

"Максимум" баллов за домашки - 8 баллов. Все, что выше, засчитывается в сумму за проект

GRADE = $10 * (0.6 * HW/8 + 0.4 * PR/10)$, округляется до целого

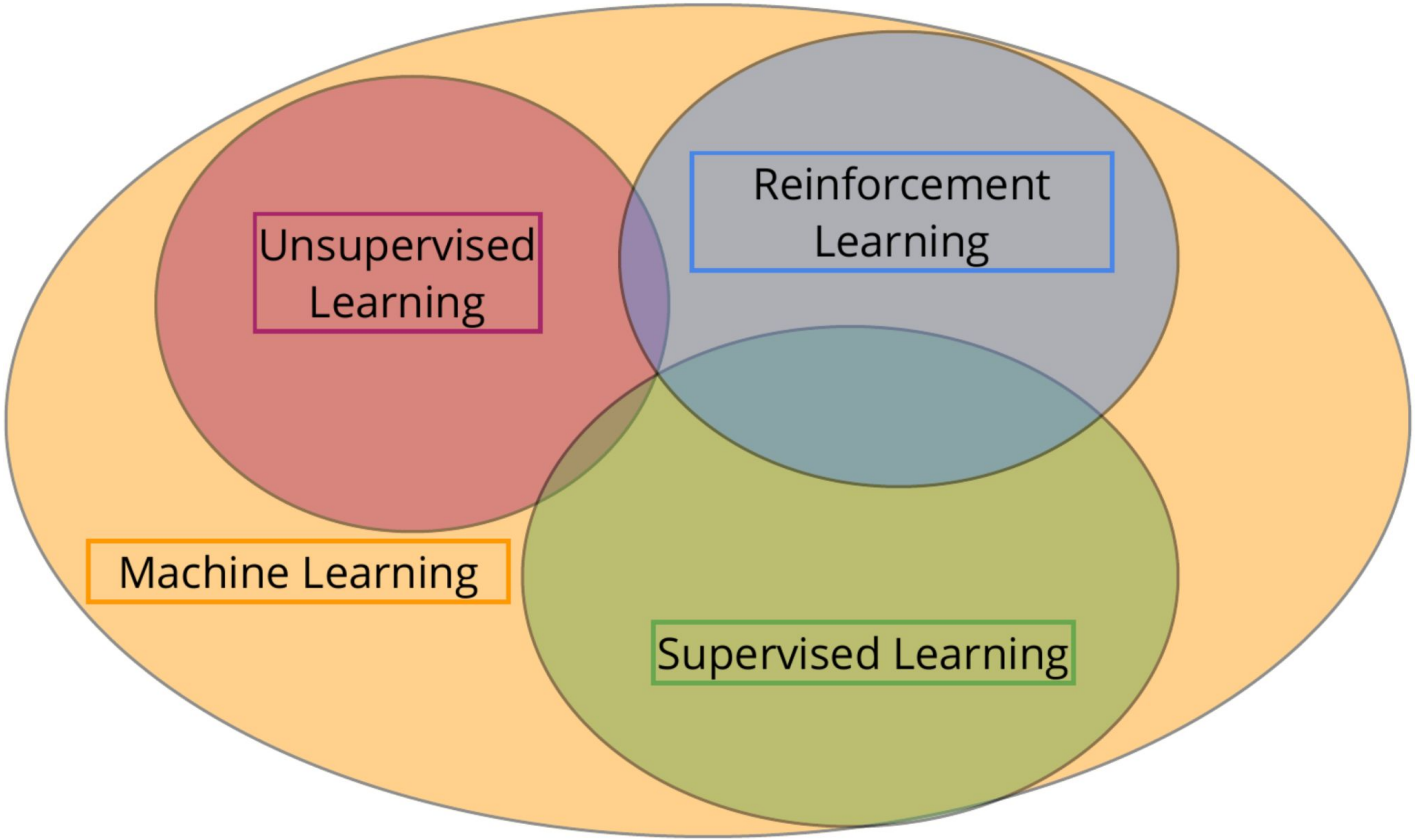
Группа ТГ

https://t.me/+y7CNdMajNiwwNDgy

План курса

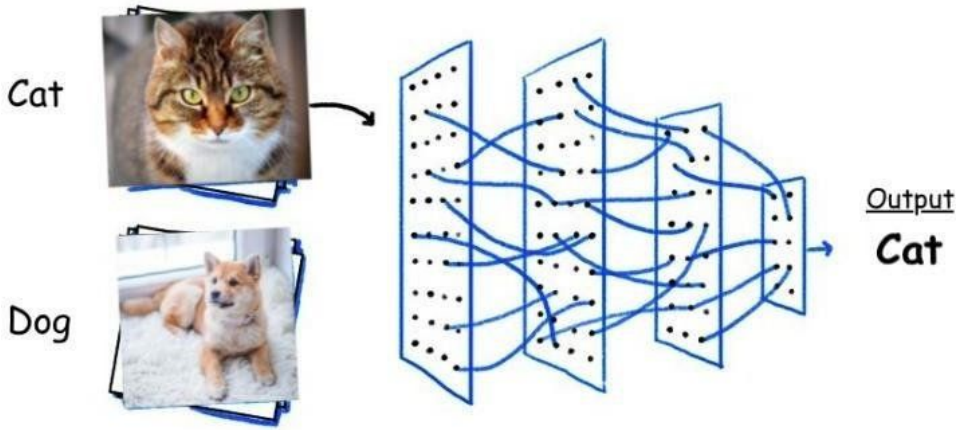
1. Интро RL.Метод кросс-энтропии. (HW1)
2. Бандиты (PR)
3. Уравнение Беллмана и динамическое программирование (HW2)
4. Model free алгоритмы. Табличное обучение
5. Глубокие алгоритмы. Методы на основе ценностей: DQN (HW3)
6. Методы на основе политики
7. Метод Actor-Critic (HW4)
8. Сдача проектов

Recap


Unsupervised Learning
Reinforcement Learning
Machine Learning
Supervised Learning

Обучение с учителем

- Имеется выборка:
  $D = \{(x_i, y_i)\}$
- Учим отображение:
  $f(x_i) = \hat{y}_i$
- Такое что:
  $\hat{y}_i \approx y_i$
- При этом правильные ответы y_i известны



Обучение с учителем

Но что если у нас нет разметки?

Рекомендация музыки

Имеем:

- Есть много разных пользователей
- Есть много разной музыки

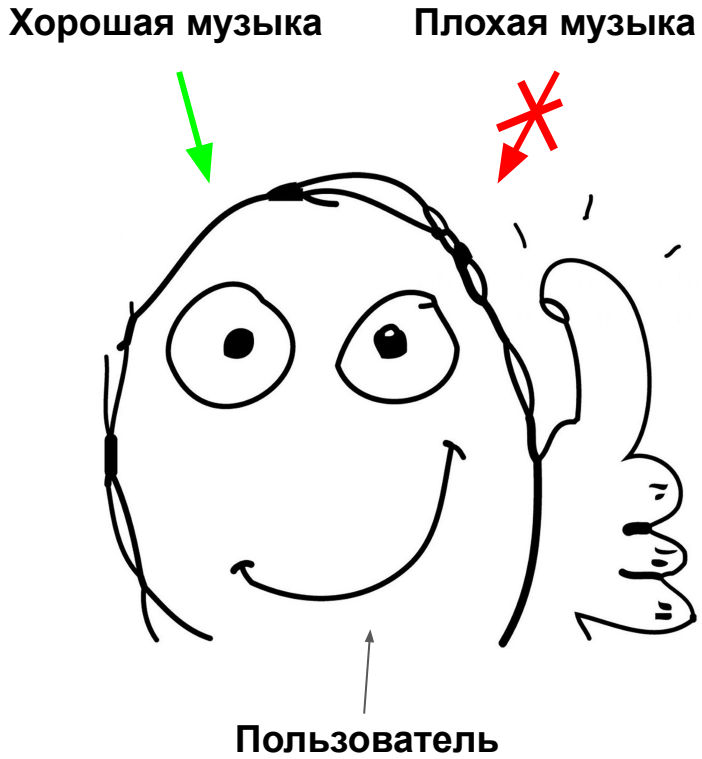
Хотим:

- Рекомендовать музыку
- Пользователи продолжали пользоваться сервисом

Рекомендация музыки

Классический подход:

1. Взять какое-то эвристическое решение
2. Выкатить в прод
3. Собрать датасет
4. Обучить модель
5. Вернуться на шаг 2



Научим робота ходить

Имеем:

- У робота есть лидар
- Двигатели, управляющие ногами

Хотим:

- Робот добежал до финиша
- Быстро

Научим робота ходить

Классический подход:

1. Взять какое-то эвристическое решение
2. Выкатить в прод
3. Собрать датасет
4. Обучить модель
5. Вернуться на шаг 2

Проблемы

1. Как исследовать пространство возможных действий?
2. Как искать лучшую стратегию?

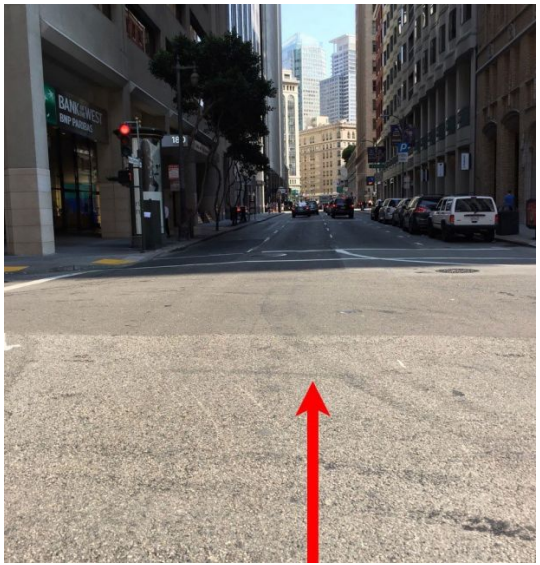
Behavior Cloning

Что, если попросить эксперта сказать, какие действия хорошие? А затем применить обучение с учителем.

На примере self-driving cars (беспилотный автомобиль):

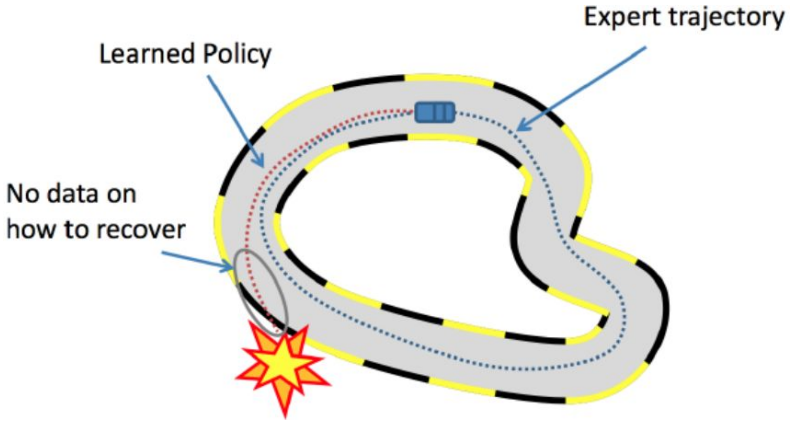
1. Берем хорошего водителя
2. Отправляем ездить по маршруту
3. Записываем видео его траекторий (s_i)
4. Сохраняем историю его действий (a_i)
5. Обучаем ML модель π:

$$a_i \approx \pi(s_i)$$



Behavior Cloning

Неизбежно, наш агент заедет туда, где эксперт никогда не был. Агент не знает, как себя там вести.



Проблема Distributional Shift:
Наши наблюдения меняются
с изменением стратегии

DAGGER

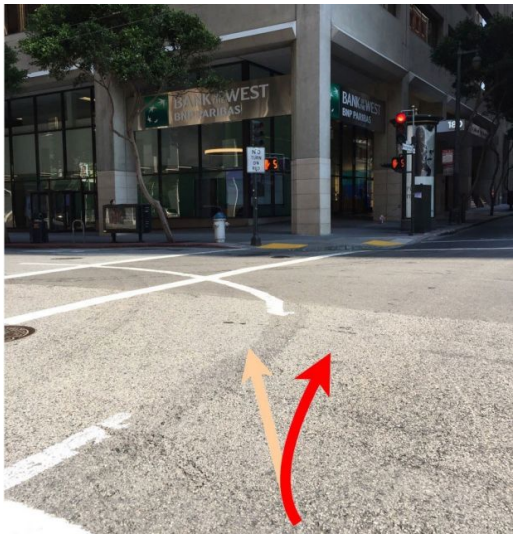
Можно просить эксперта возвращать агента на путь истинный.

На примере self-driving cars (беспилотный автомобиль):

- Отправляем водителя ездить по маршруту
- Записываем его траектории s_i и действия a_i
- Обучаем ML модель π: $a_i \approx \pi(s_i)$

Делаем, пока не удовлетворены:

- ❖ Пускаем агента в город, собираем траектории si
- ❖ Просим эксперта дать правильные действия a для собранных траекторий i si



DAGGER

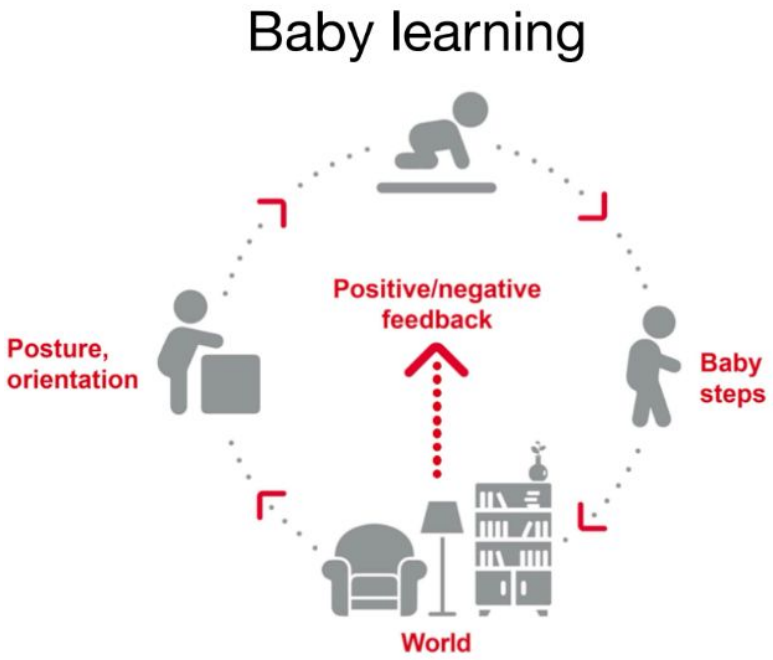
Плюсы:

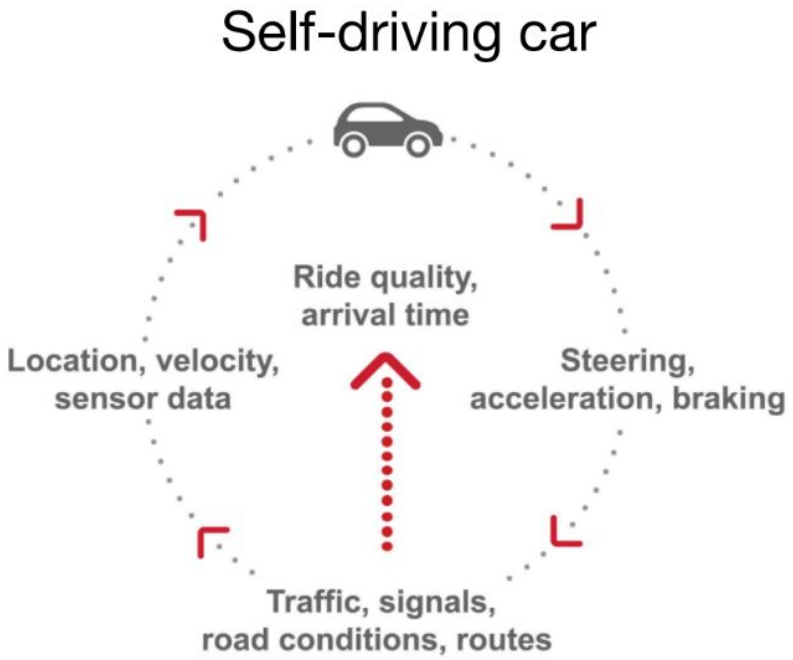
- Очень прост
- Иногда хорошо работает

Минусы:

- Эксперт нужен в режиме онлайн
- Агент не будет лучше эксперта
- Не всегда эксперт вообще знает, что делать!

Обучение с подкреплением



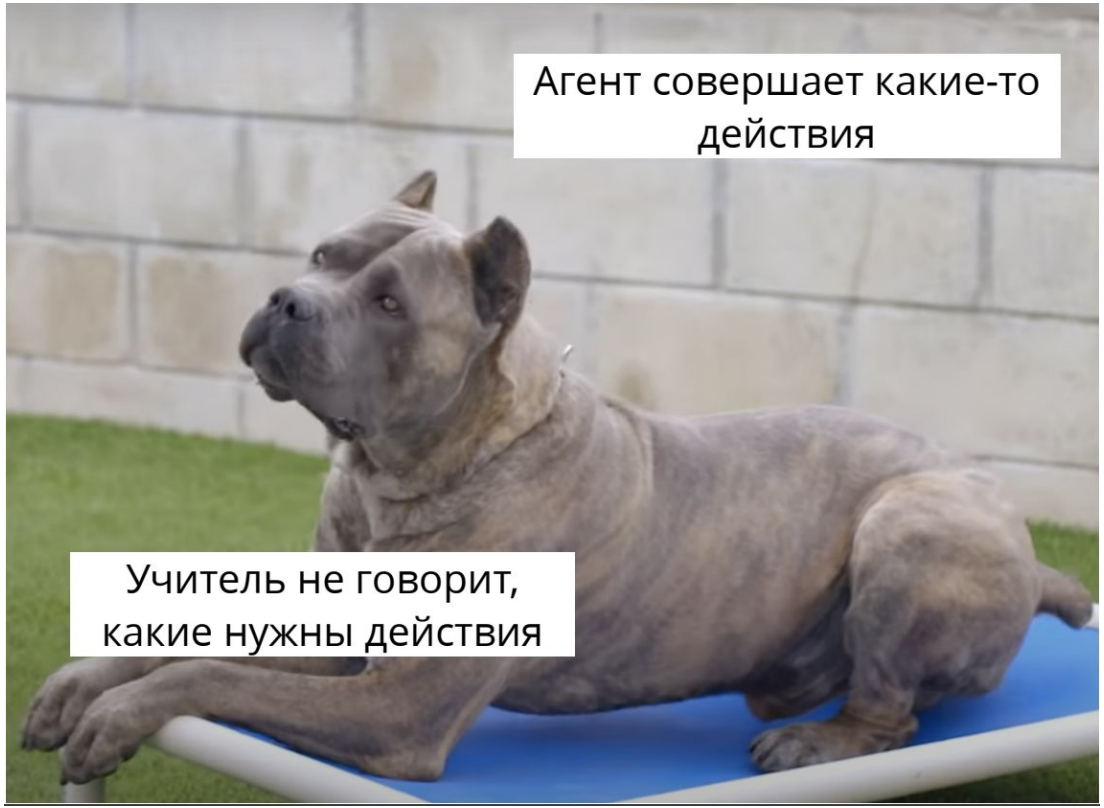


Награда за достижение целей

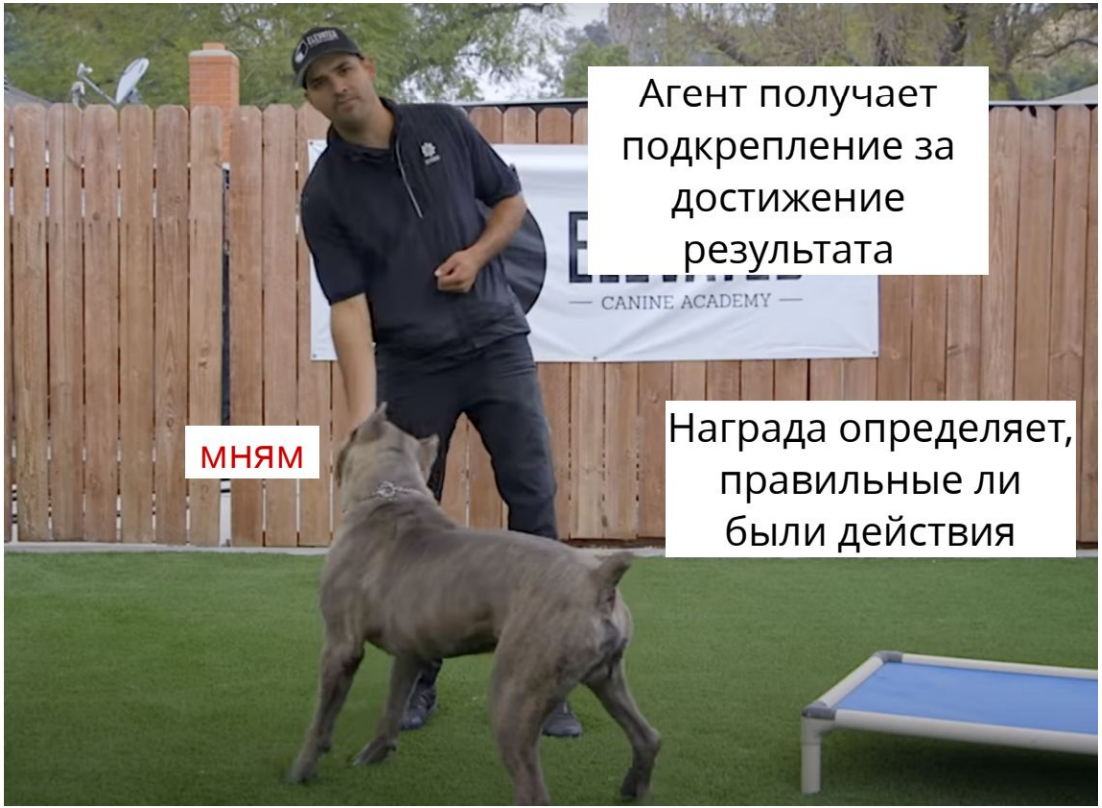
Награда за достижение целей

Награда за достижение целей

Награда за достижение целей



Награда за достижение целей



Примеры ППР (Decision Process)

Робототехника

Наблюдения:

- Изображения с видеокамеры
- Данные с датчиков

Действия:

- Усилие подаваемое на сочленения робота

Цели:

- Движение вперед
- Решение составных задач (перенос предметов)
- …

Примеры ППР (Decision Process)

Шахматы (или другие настольные игры)

Наблюдения:

- Расстановка фигур на доске

Действия:

- Выбор фигуры и хода ей

Цели:

- Победа
- Или, хотя бы, ничья

Примеры ППР (Decision Process)

Автоматизированная торговля на бирже

Наблюдения:

- История изменения стоимости акций
- …

Действия:

- Покупка и продажа акций

Цели:

- Максимизация прибыли

Обучение с подкреплением

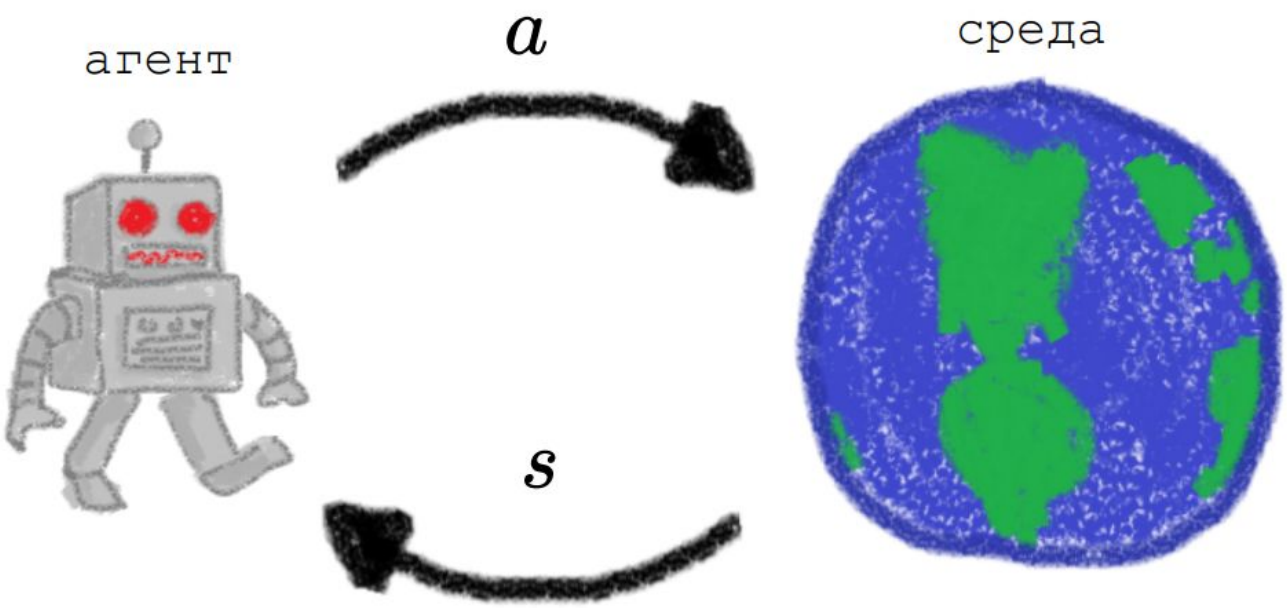
Политика агента:

a = π(a|s)

a ~ π(a|s)

Цель агента:

$\mathbb{E}_{p(\tau|\pi)}(\sum_{t=0}^{T} r_t) \to \max_{\pi}$



Обучение с подкреплением

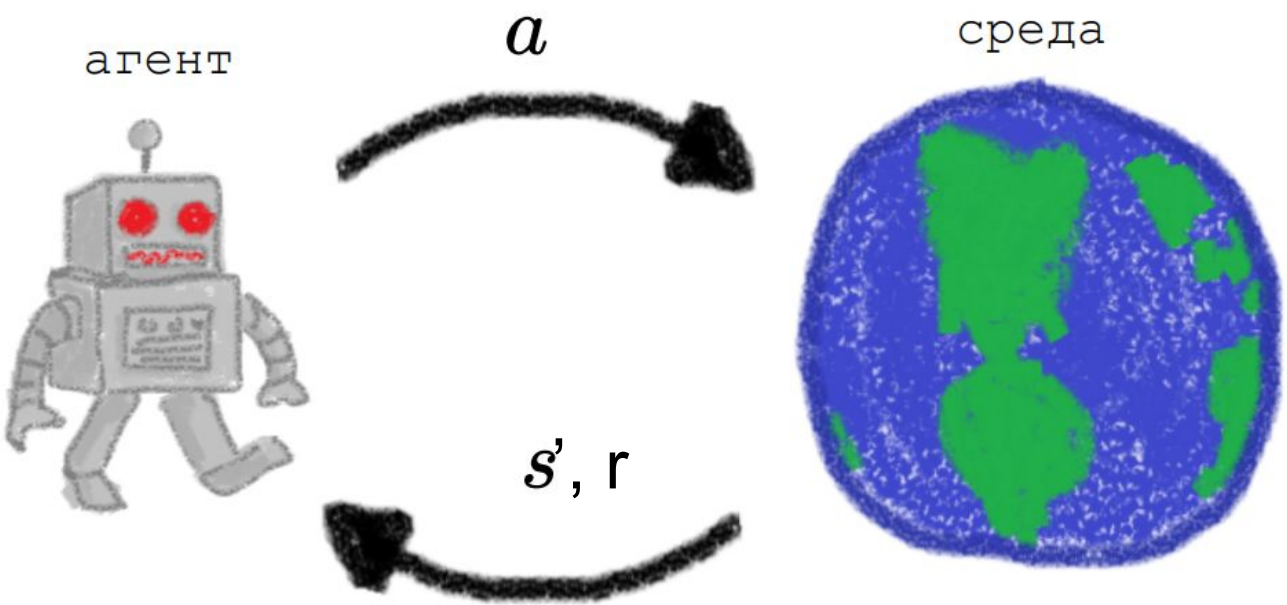
Политика агента:

a = π(a|s)

a ~ π(a|s)

Цель агента:

$\mathbb{E}_{p(\tau|\pi)}(\sum_{t=0}^{T} r_t) \rightarrow \max_{\pi}$



Выбор награды

Какой выбор награды лучше?

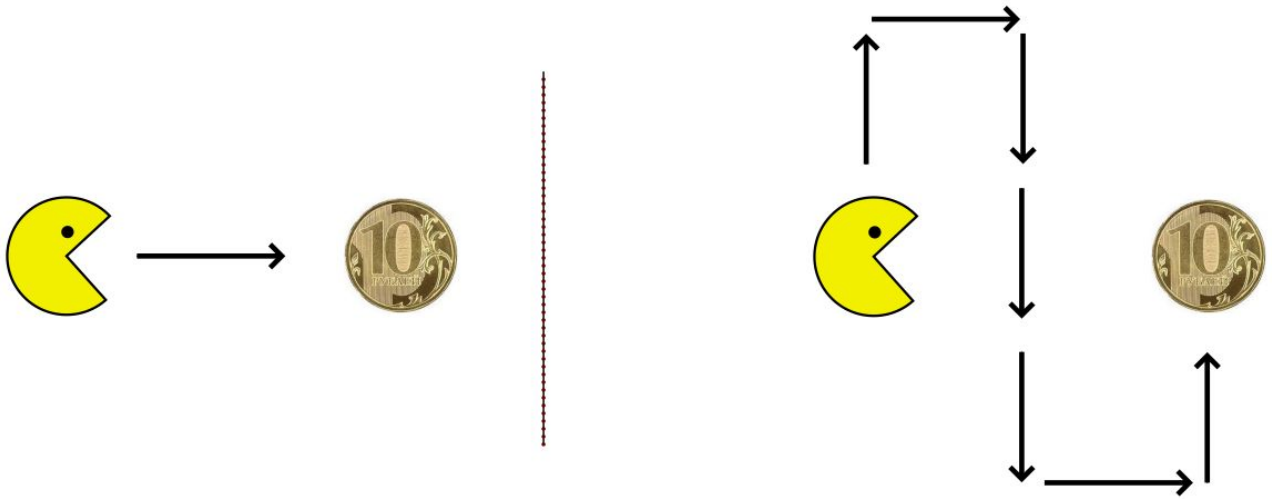
Вариант 1:

- победа: +1
- проигрыш: -1
- ничья: 0

Вариант 2:

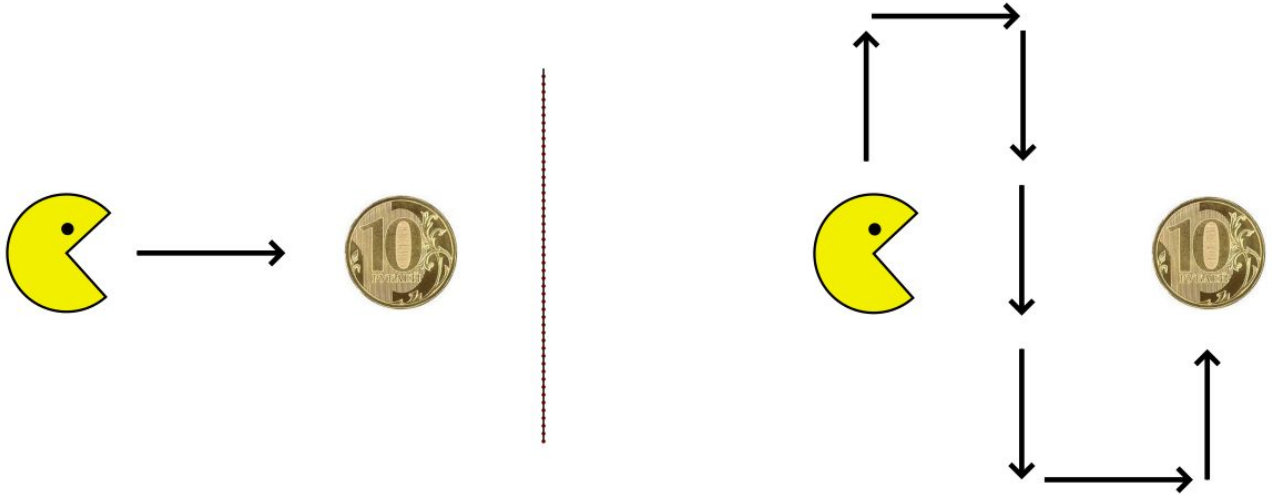
- победа: +1
- проигрыш: -1
- ничья: 0
- взятие фигуры: +1
- потеря фигуры: -1

Дисконтирование награды



Как заставить агента прийти к награде быстро?

Дисконтирование награды



Будем уменьшать награду каждый шаг:

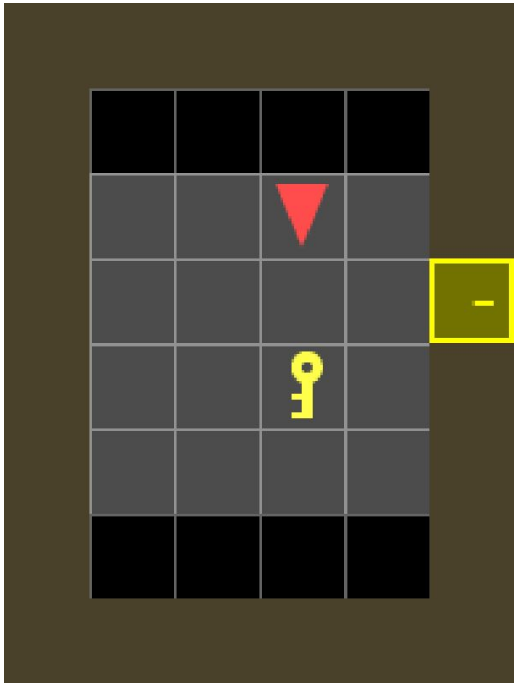
$$\mathbb{E}_{p(\tau|\pi)}\left(\sum_{t=0}^{T} r_t\right) \to \max_{\pi} \longrightarrow \mathbb{E}_{\pi}\left(\sum_{t=0}^{T} \gamma^t r_t\right) \to \max_{\pi}$$

$$\gamma \in [0,1]$$

Марковское свойство

Нужно ли агенту помнить свою историю?

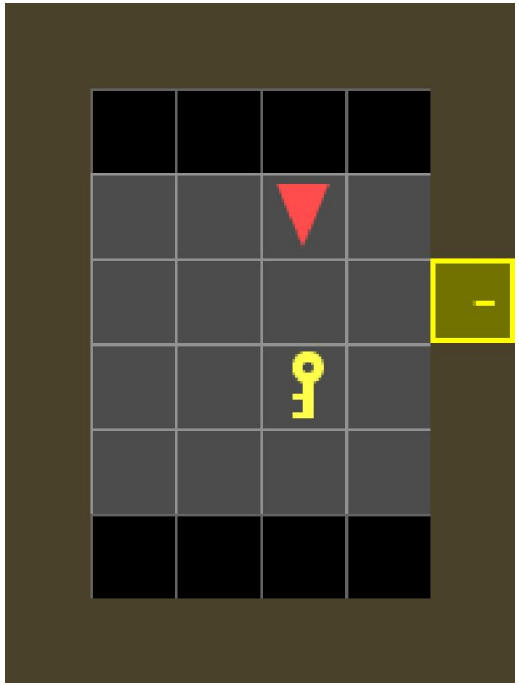
1. координаты агента
2. полная картинка лабиринта
3. координаты агента + есть ли у него ключ



Марковское свойство

Нужно ли агенту помнить свою историю?

1. координаты агента
2. полная картинка лабиринта
3. координаты агента + есть ли у него ключ

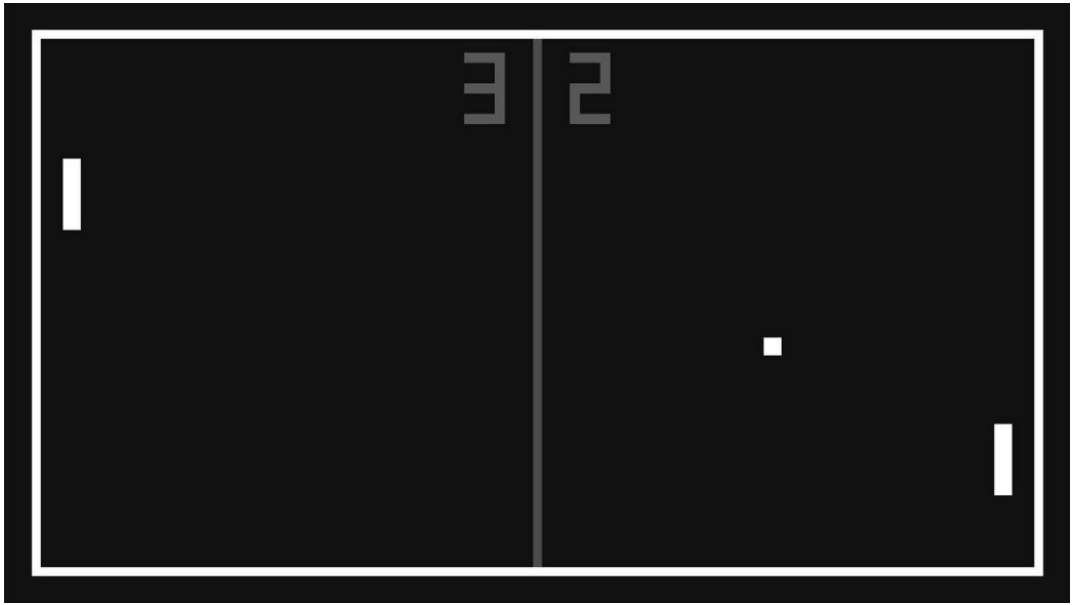


2 и 3 не требуют хранения истории

Markov property:

$$p(R_t, S_{t+1} \mid S_t, A_t, R_{t-1}, S_{t-1}, A_{t-1}, \ldots) = p(R_t, S_{t+1} \mid S_t, A_t)$$

Пример немарковости



Куда движется шарик?

Задача Reinforcement Learning

$s \sim S$ - состояния (дискретные \ непрерывные)

$a \sim A$ - действия (дискретные \ непрерывные)

$p(s_{t+1} | s_t, a_t)$ - динамика переходов в среде (марковская)

$p(s_0)$ - распределение над начальными состояниями

$r(s, a)$ - награда за действие a в состоянии s

$\pi(a|s)$ - политика агента

$p(\tau|\pi) = p(s_0)\prod_{t=0}^{T} \pi(a_t|s_t)p(s_{t+1}|a_t,s_t)$ - политика агента

где $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T, a_T)$ - траектория агента

В общем случае заранее не известно

Оптимизируем для получения награды

Метод кросс-энтропии

Вообще, мы бы хотели максимизировать по π среднюю кумулятивную дисконтированную награду:

$$J(\pi) = \mathbb{E}_{p(\tau|\pi)} \sum_{t=0}^{T} \gamma^t r(s_t, a_t)$$

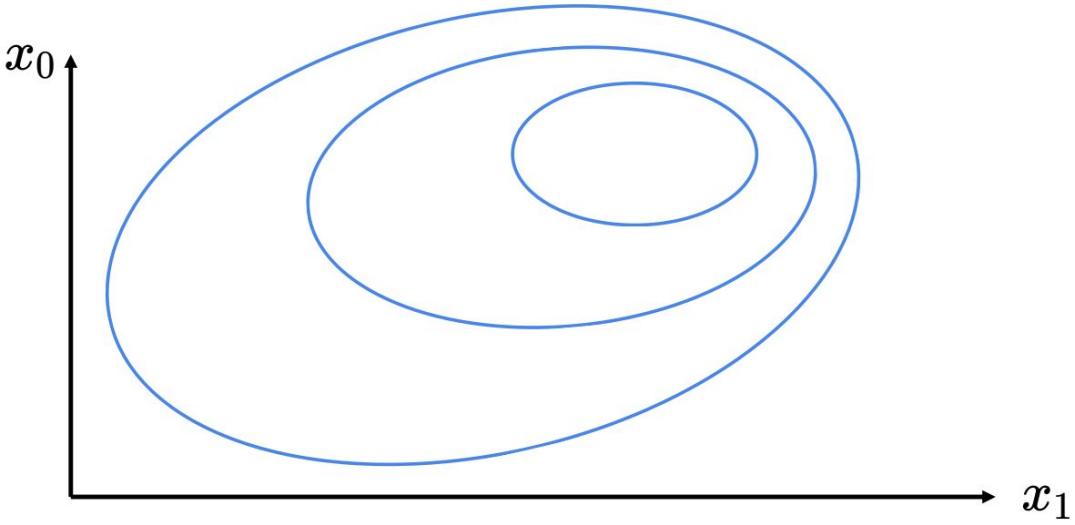
Пока абстрагируемся!

Пусть есть некоторая функция f(x) : **X** $\longrightarrow$ **R**

- Хотим ее максимизировать по x
- Но не умеем считать производную по x

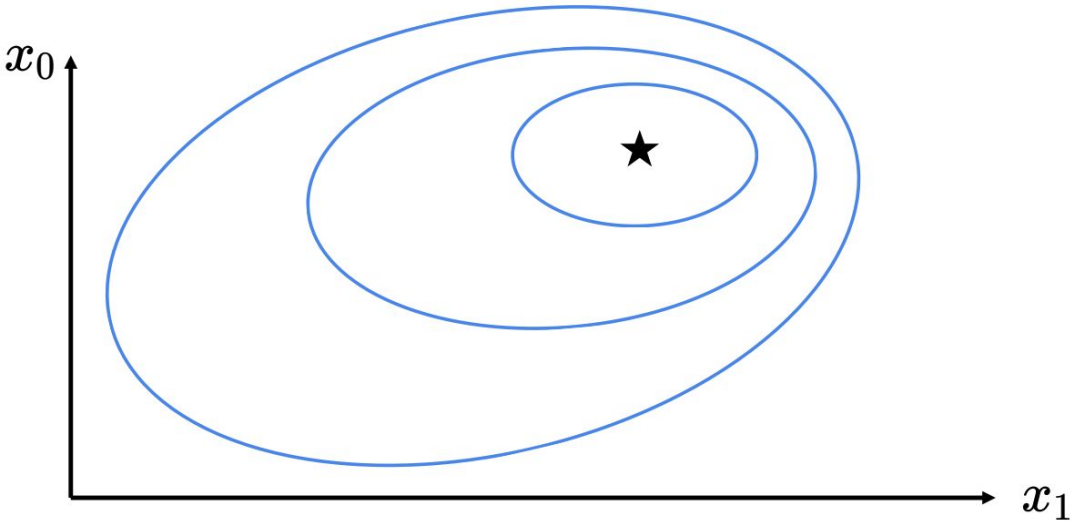
Метод кросс-энтропии

Идея: вместо оптимального x^* найдем "оптимальное" распределение

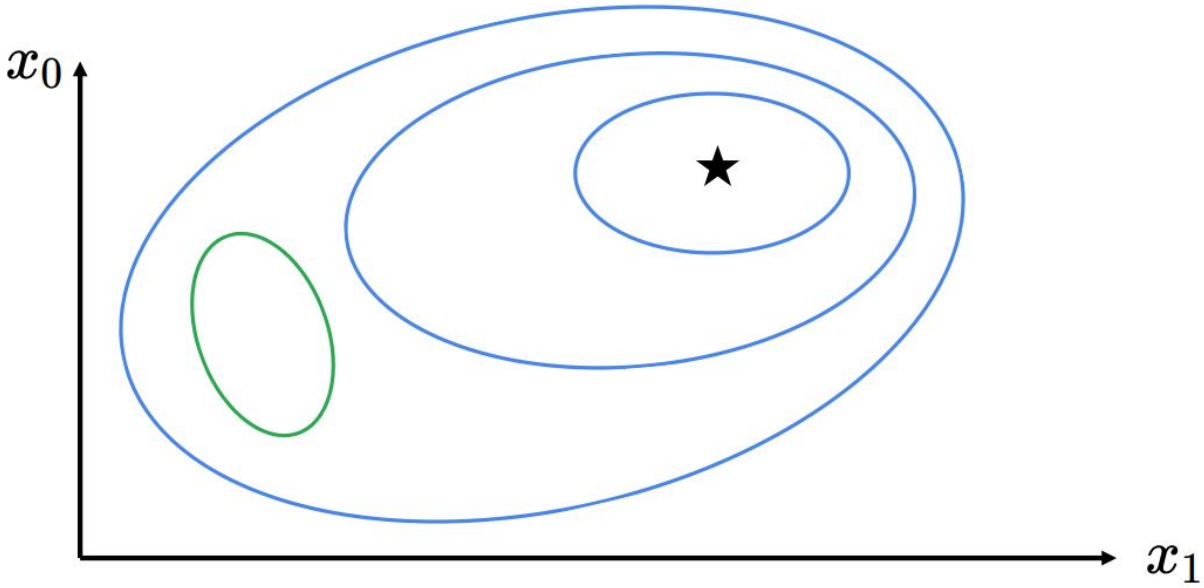


Метод кросс-энтропии

Идея: вместо оптимального x^* найдем "оптимальное" распределение

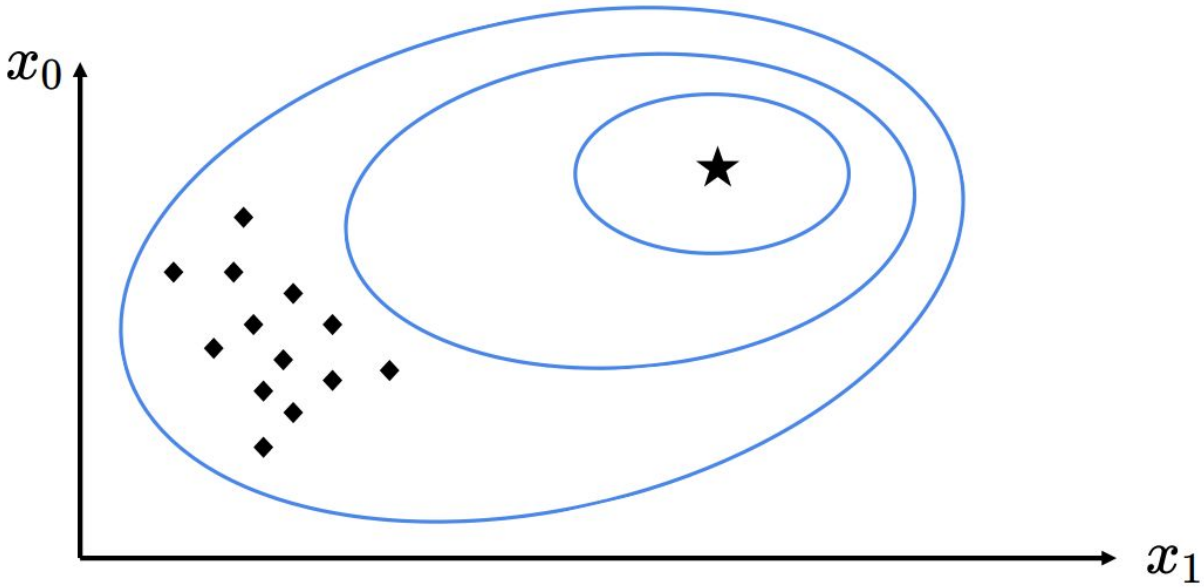


Метод кросс-энтропии



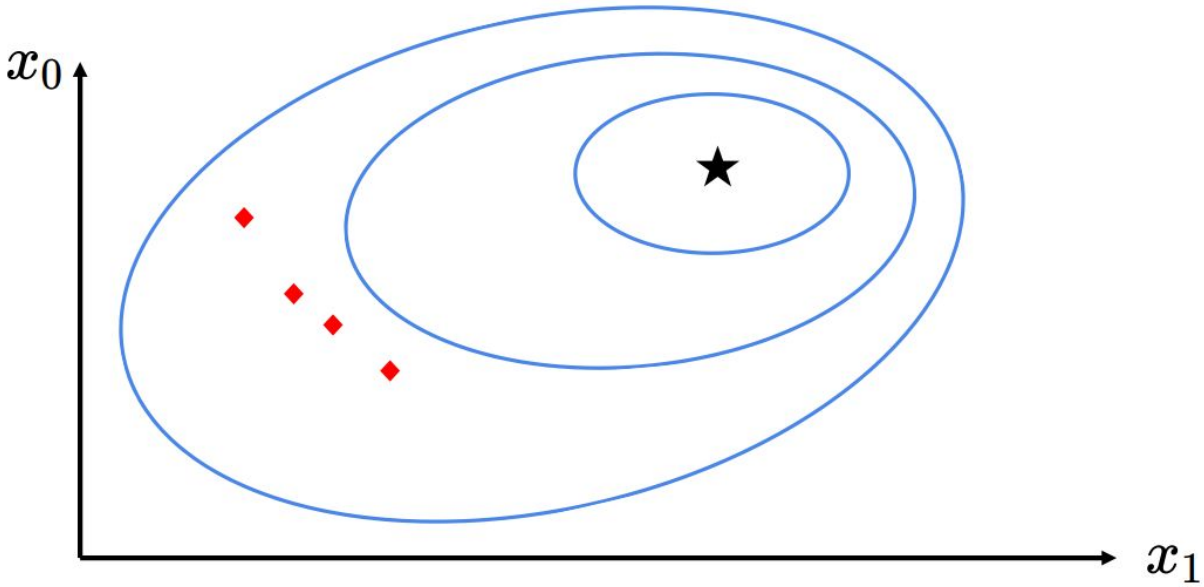
1) Инициализируем распределение q^0

Метод кросс-энтропии



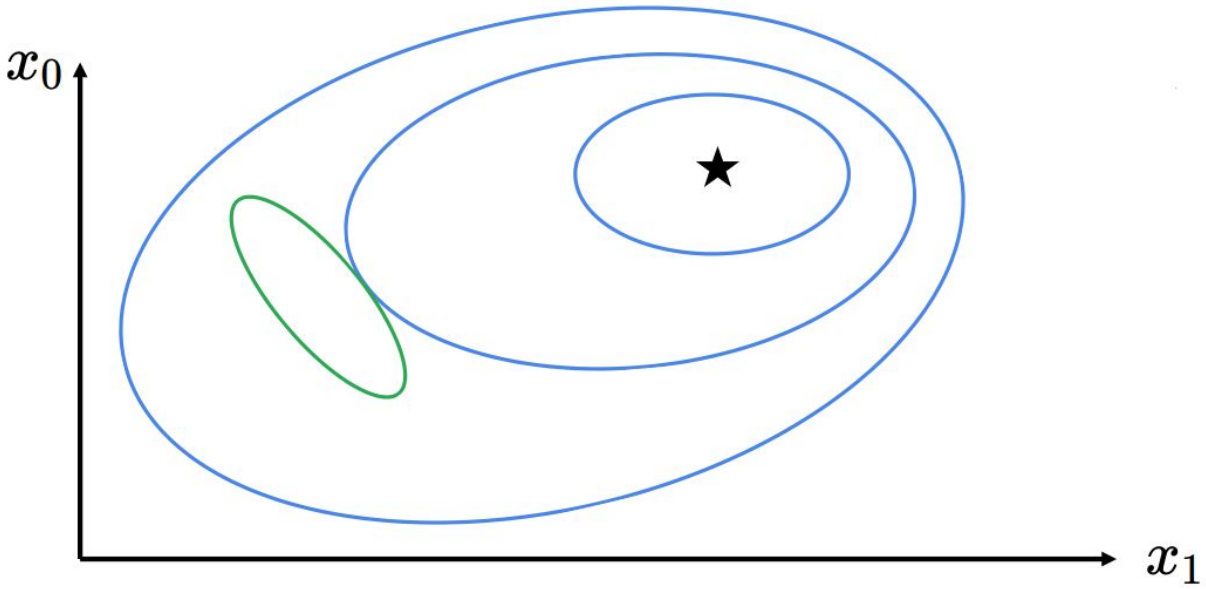
1) Инициализируем распределение q^0
2) Сэмплим $x_i \sim q^0$

Метод кросс-энтропии



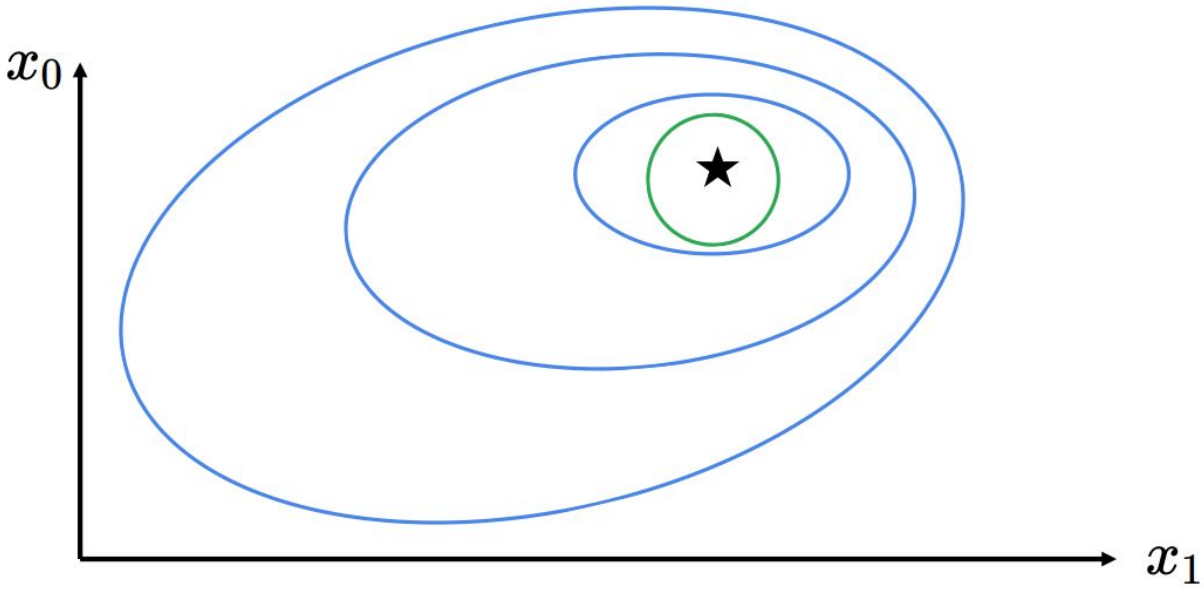
1) Инициализируем распределение q^0
2) Сэмплим $x_i \sim q^0$
3) Выбираем M x_i с наибольшим f - элиты

Метод кросс-энтропии



1) Инициализируем распределение q^0
2) Сэмплим $x_i \sim q^0$
3) Выбираем M x_i с наибольшим f - элиты
4) Подстраиваем q^1 под элиты

Метод кросс-энтропии



1) Инициализируем распределение q^0
2) Сэмплим $x_i \sim q^0$
3) Выбираем M x_i с наибольшим f - элиты
4) Подстраиваем q^1 под элиты
5) Повторяем с шага 2 до сходимости

Метод кросс-энтропии

Чтобы подстроить q под элиты, минимизируем KL-дивергенцию:

$$KL(p_{data}||q) = \int p_{data}(x) \log \frac{p_{data}(x)}{q(x)} dx$$

$$\min_q KL(p_{data}||q) = \min_q \left[-\mathbb{E}_{x \sim p_{data}} \log q(x)\right]$$

На k-й итерации:

$$q^{k+1} = \arg\min_q \left[-\sum_{x \in \mathcal{M}^k} \log q(x)\right]$$

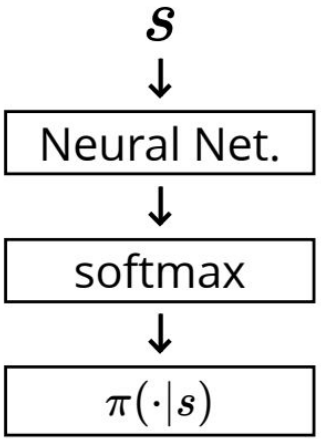
где $\mathcal{M}^k$ - элиты, собранные на прошлой итерации

Метод кросс-энтропии

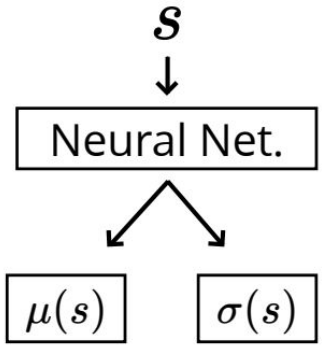
Как применить для оптимизации политики?

Будем максимизировать J(π_θ)по параметрам θ.

Где π_θ - нейронная сеть, параметризующая распределение.



дискретные действия



непрерывные действия

Метод кросс-энтропии

Алгоритм для RL:

- **ввод:** α - процент сохраняемых элит
- инициализируем π_θ
- **повторять**
  - пускаем π_θ собирать траектории **T**
  - вычисляем перцентиль δ = percentile(**T**, p = α)
  - выбираем элиты M = filter_elites(**T** , δ)
  - частично обучаем π_θ прогнозировать a_i по s_i, где $(a_i, s_i) \in M$

Репо курса

https://github.com/Fw9wef/hse_rl_course