Методы обучения с подкреплением и их применение

Лекция 6

Киселёв Глеб Андреевич
к.т.н., старший преподаватель
ФФМиЕН РУДН
тел.:+79067993329
email: kiselev@isa.ru

AI SOLUTIONS

Институт проблем искусственного интеллекта

ФЕДЕРАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЦЕНТР Информатика и Управление РОССИЙСКОЙ АКАДЕМИИ НАУК

Конечные автоматы (finite state machine)

Формальный язык – множество конечных слов (строк, цепочек) над конечным алфавитом. Пример: языки программирования

Алфавит Σ - непустое конечное множество символов. Например, $\Sigma = \{0,1\}$ – бинарный алфавит, ASCII, Unicode – алфавиты для машинного кода.

Слово (строка, цепочка) w – последовательность символов алфавита, например (a,b,a) = aba
Пустое слово ε – последовательность из 0 символов. Не входит в алфавит.
Формальный язык L – множество слов в алфавите Σ.

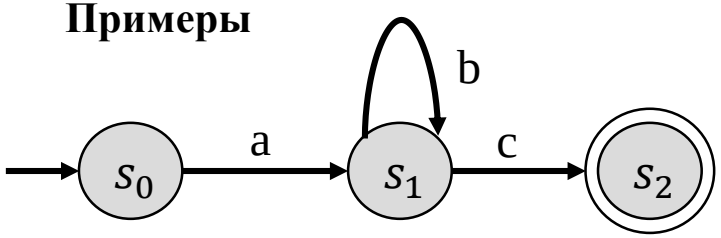
Автомат – математическая модель устройства, имеющая 1 вход, 1 выход и в 1 момент времени находящаяся в 1 состоянии.

Конечный автомат – автомат, число возможных внутренних состояний которого конечно.

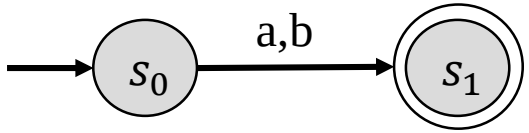
Детерминированный конечный автомат (ДКА) – кортеж из 5 элементов: M= <Q, Σ, δ, q_0, F>
Q – множество состояний, Σ – конечный входной алфавит, δ: $Q \times \Sigma \rightarrow Q$ – функция переходов, q_0 - начальное состояние, F – множество конечных состояний.

Язык L называется *регулярным*, если существует M, который распознает L.

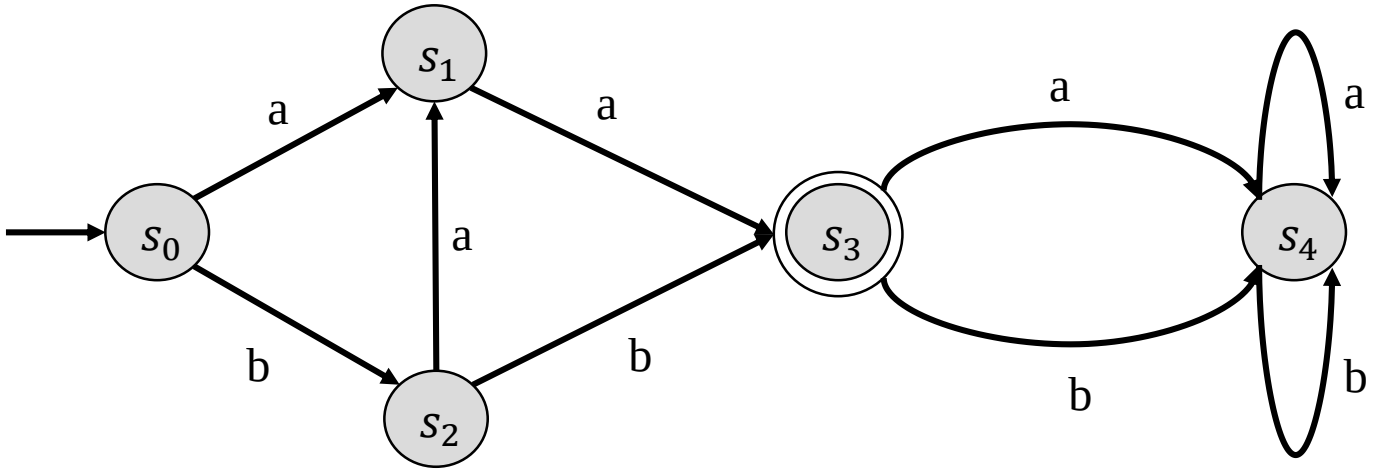
Примеры



Автомат распознает слова вида ab^*c



Автомат распознает язык со словами a или b



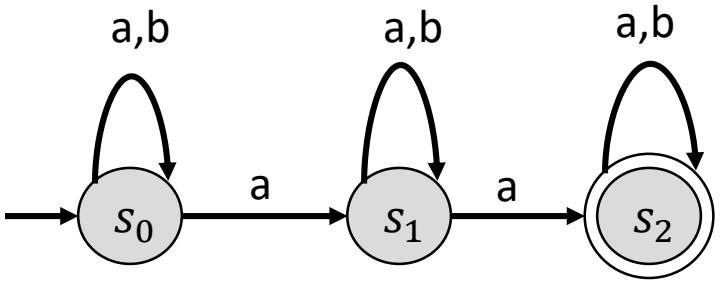
Автомат распознает слова aa, baa, bb, но не слова bbb, abab, abb. s_4 - состояние ловушка. Формально все слова автомата можно описать:
ab^*a, bab^*a или bb.

Недетерминированные конечные автоматы

Недетерминированный конечный автомат (НКА) – кортеж из 5 элементов: A= <Q, Σ, E, Q_0, F>

Q – множество состояний, Σ – конечный входной алфавит, E: Q × Σ × Q – функция (отношение) переходов, Q_0 - множество начальных состояний, F – множество конечных состояний.

Основное отличие – из одного и того же состояния может быть несколько равноценных переходов по одной и той же букве в разные состояния.



Автомат распознает язык состояний из слов с не менее, чем двумя символами «a» или язык $\{a, b\}^* a \{a, b\}^* a \{a, b\}^*$.

Вероятностные автоматы

Вероятностный автомат – кортеж из 5 элементов: M = <S, $F_r, M, \mu(a^-, f|a, s), \pi_0$>
S – конечное множество входных сигналов; F – конечное множество выходных сигналов (действий) автомата; M – конечное или счётное множество состояний автомата; $\mu(a^-, f|a, s)$ - условная вероятность перехода автомата из состояния «a» при входном сигнале (действии) «s» в состояние «a^-» при выходном сигнале (действии) «f»; π_0 - стартовое распределение вероятностей состояний.

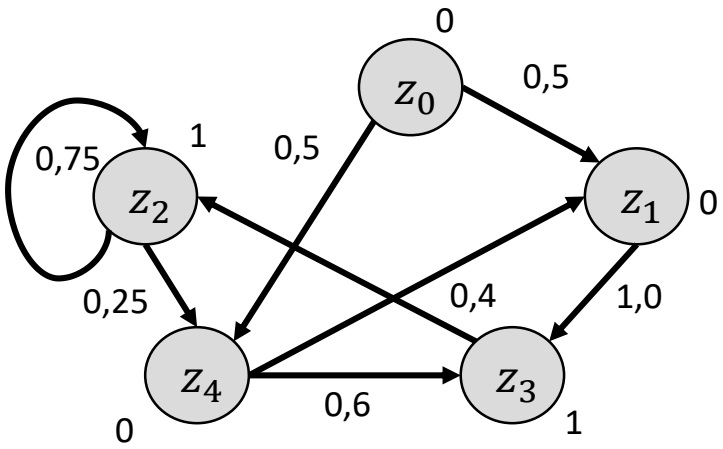
Автомат Мура определяется выражением M и предположением, что при $\mu(a^-|a, s) \neq 0$, выполнено $\mu(a^-, f|a, s) = \mu(f|a^-)$

Пусть s$\in \{-1, +1\}$. Автомат функционирует в стационарной случайной среде, C($p_1, p_2, \dots, p_r$), если его действие f_i, произведенное в момент времени t (t=0,1,2,…) влечёт появление на входе автомата в момент времени t+1 сигнала s = -1 (штрафа) с вероятностью p_i и сигнала s = +1 (выигрыша) с вероятностью $q_i = 1 - p_i$, где i = 1, 2, …, r. Т.е. при одинаковой последовательности входных сигналов, поступающих при использовании *разных* действий, автомат должен вести себя *одинаково*.

Пример

Пусть задан τ – детерминированный вероятностный автомат. τ = [0,0,1,1,0], c=(c_0, c_1, c_2, c_3, c_4)

$$P = \begin{bmatrix} 0 & 0{,}5 & 0 & 0 & 0{,}5 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0{,}75 & 0 & 0{,}25 \\ 0 & 0 & 0{,}4 & 0 & 0{,}6 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$



Требуется найти суммарные финальные вероятности пребывания этого автомата в z_2 и z_3

$$\begin{cases} c_1 = c_4 \\ c_2 = 0{,}75c_2 + 0{,}4c_3 \\ c_3 = c_1 \\ c_4 = 0{,}25c_2 + 0{,}6c_3 \\ c_1 + c_2 + c_3 + c_4 = 1 \end{cases} \quad \begin{cases} c_1 = \frac{5}{23} \\ c_2 = \frac{8}{23} \\ c_3 = \frac{5}{23} \\ c_4 = \frac{5}{23} \end{cases} \quad \Rightarrow \quad c_2 + c_3 = 13/23 = 0{,}5652$$

При бесконечной работе заданного τ - детерминированного вероятностного автомата, на его выходе формируется двоичная последовательность с вероятностью появления единицы 0,5652

Многорукие бандиты

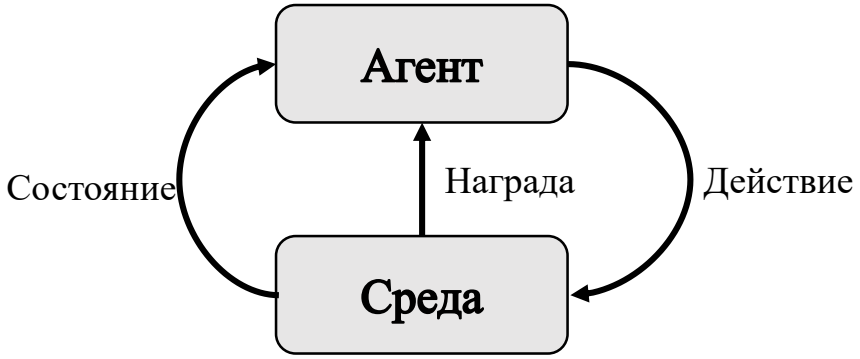
В какой последовательности и как часто дергать ручки автоматов, чтобы максимизировать выигрыш.

k – количество автоматов, $P_i, \mathrm{i} = 1, \dots, \mathrm{k}$ – вероятность выигрыша.

В t агент выбирает 1 из автоматов и получает награду $r_{i,t}$ - случайную величину из P_i с мат. ожиданием μ_i.
Все случайные величины автоматов независимы. После получения награды агент обновляет знания о среде и переходит к следующей итерации.

Цель – максимизация выигрыша.

Функция потерь R = $\sum_{t=1}^{T} r_{i,t}^{*} - r_{i,t}$ - минимизируя эту функцию алгоритм находит оптимальное действие с наибольшей наградой.



Пример:
Интернет магазин. Контекст – пользовательские признаки, действия – признаки рекомендуемых товаров, агент – алгоритм, награда +1 за клик по товару, 0 – за отсутствие клика.

Виды алгоритмов

Название алгоритма	**Описание**	**Особенности и недостатки**
Наивный	1. Совершает каждое действие N раз 2. Составляем эмпирическое $\widehat{\mu_{*,t}}$	1. Нет исследования в процессе 2. Нет уверенности в оптимальном выборе
ε –greedy	1. С вероятностью ε выбираем случайное действие 2. С вероятностью 1-ε – действие с максимальным мат. ожиданием $\widehat{\mu_{i,t}}$ 3. Если ε = 0 – жадный алгоритм (этап использования) 4. Если ε= 1 – случайный алгоритм (этап исследования	После найденного оптимального действия ε становится не нужен (и его минимизируют)
UCB	1. Алгоритм выбирает действие и применяет его; 2. Если успешно – используем действие дальше; 3. Если не успешно – уменьшаем величину доверительного интервала (чем чаще выбираем a_i, тем точнее $\widehat{\mu_{i,t}}$, тем уже доверительный интервал); 4. Ширина доверительного интервала строится на основе неравенства Хёфдинга (отклонения мат. ожидания от истины)	Оценивается не только $\widehat{\mu_{i,t}}$, но и ширину доверительного интервала $\widehat{\mu_{i,t}}$ (на сколько алгоритм уверен в выборе $\widehat{\mu_{i,t}}$. Используется принцип оптимизации в условиях неопределённости. *https://habr.com/ru/articles/689364/*
Thomson Sampling	В большинстве задач награды распространяются по закону Бернулли и принимают значение $r_i \epsilon \{0,1\}$. Алгоритм составляет априорное предположение о μ_i, совершает действие и строит апостериорное Бета-распределение с плотностью вероятности $p(\theta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha,\beta)}$, где α и β – параметры B-распределения, $B(\alpha,\beta)$ – бета функция (нужна для нормализации вероятности и θ – вероятность успеха в испытаниях Бернулли.	*https://habr.com/ru/companies/vk/articles/673914/*

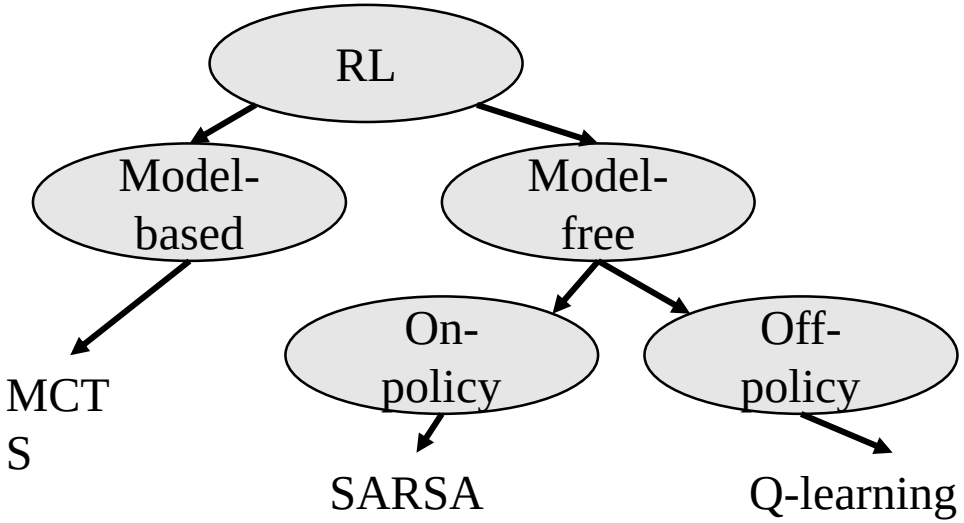
Обучение с подкреплением. RL.

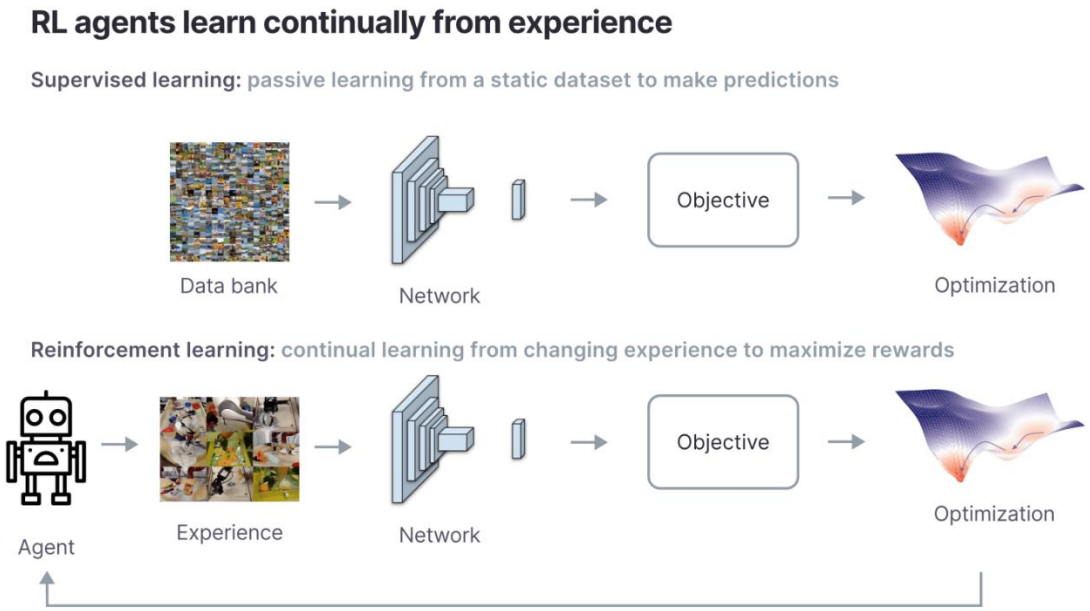
В отличии от многоруких бандитов:

- Наблюдения. Агент исследует среду и использует знания для выбора следующего действия.
- Отсроченный выигрыш. Все действия взвешиваются не только на основе сиюминутного выигрыша, но и на основе отсроченной награды.

Чем оперируем:

1. Действия – все возможные действия агента в среде;
2. State (S,s) – текущее состояние, возвращается окружением (средой);
3. Reward – мгновенная награда. Синтезируется окружением, как оценка последнего действия;
4. Policy – стратегия (политика), которую использует агент для синтеза следующего действия;
5. Estimation – ожидаемая награда. Синтезируется стратегией для текущего состояния s.
6. Q-value – Q-оценка стратегии на основе исполнения действия «a»





Источник: https://habr.com/ru/companies/wunderfund/articles/667654/

Материал к изучению:
https://habr.com/ru/companies/newprolab/articles/343834/

https://habr.com/ru/articles/443240/

Пример взят из https://habr.com/ru/articles/443240/

Пример. Q-learning.

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma max_{a'} Q(s',a') - Q(s,a)]$$

- α – скорость обучения (каждый раз, когда следующее действие a^{-} выбирается для максимизации Q –значения следующих состояний, вместо того, чтобы следовать текущей стратегии. (его основная цель — избежать больших изменений в одном обновлении);
- ε – коэффициент изучения окружающей среды;
- γ – коэффициент влияния отложенной награды.

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

```
Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(terminal\text{-}state, \cdot) = 0$
Repeat (for each episode):
    Initialize $S$
    Repeat (for each step of episode):
        Choose $A$ from $S$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
        Take action $A$, observe $R$, $S'$
        $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
        $S \leftarrow S'$
    until $S$ is terminal
```

	0	1	2	3	4	5	6	7	8	9	10	11
	U: -6.76 D: -6.73 R: -6.75 L: -6.71	U: -6.70 D: -6.74 R: -6.60 L: -6.62	U: -6.42 D: -6.53 R: -6.34 L: -6.34	U: -6.14 D: -6.09 R: -6.06 L: -6.12	U: -5.82 D: -5.77 R: -5.74 L: -5.78	U: -5.51 D: -5.37 R: -5.36 L: -5.53	U: -5.12 D: -4.97 R: -4.94 L: -5.32	U: -4.58 D: -4.49 R: -4.49 L: -4.69	U: -4.01 D: -4.02 R: -3.94 L: -4.28	U: -3.57 D: -3.37 R: -3.36 L: -3.70	U: -2.65 D: -2.69 R: -2.65 L: -3.01	U: -2.26 D: -1.90 R: -2.07 L: -2.15
1	U: -6.81 D: -6.96 R: -6.89 L: -6.89	U: -6.80 D: -6.71 R: -6.70 L: -6.75	U: -6.51 D: -6.43 R: -6.45 L: -6.67	U: -6.17 D: -6.08 R: -6.09 L: -6.39	U: -5.76 D: -5.68 R: -5.68 L: -5.98	U: -5.55 D: -5.21 R: -5.21 L: -5.63	U: -4.73 D: -4.68 R: -4.68 L: -4.92	U: -4.37 D: -4.09 R: -4.09 L: -4.23	U: -3.92 D: -3.44 R: -3.44 L: -3.98	U: -3.80 D: -2.71 R: -2.71 L: -2.93	U: -2.84 D: -1.90 R: -1.90 L: -3.24	U: -1.72 D: -1.00 R: -1.43 L: -2.27
2	U: -7.06 D: -7.40 R: -6.86 L: -7.14	U: -6.96 D: -99.95 R: -6.51 L: -7.15	U: -6.71 D: -93.75 R: -6.13 L: -6.86	U: -6.37 D: -96.88 R: -5.70 L: -6.16	U: -6.09 D: -99.61 R: -5.22 L: -6.12	U: -5.60 D: -99.22 R: -4.69 L: -5.68	U: -5.07 D: -99.22 R: -4.10 L: -5.18	U: -4.60 D: -99.22 R: -3.44 L: -4.07	U: -4.07 D: -96.88 R: -2.71 L: -3.98	U: -3.34 D: -98.44 R: -1.90 L: -3.35	U: -2.64 D: -98.44 R: -1.00 L: -2.63	U: -1.72 D: 0.00 R: -1.00 L: -1.81
3	U: -7.18 D: -7.46 R: -99.22 L: -7.45	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00	U: 0.00 D: 0.00 R: 0.00 L: 0.00

```
# Initialize Q arbitrarily, in this case a table full of zeros
q_values = np.zeros((num_states, num_actions))

# Iterate over 500 episodes
for _ in range(500):
    state = env.reset()
    done = False

    # While episode is not over
    while not done:
        # Choose action
        action = egreedy_policy(q_values, state, epsilon=0.1)
        # Do the action
        next_state, reward, done = env.step(action)
        # Update q_values
        td_target = reward + gamma * np.max(q_values[next_state])
        td_error = td_target - q_values[state][action]
        q_values[state][action] += learning_rate * td_error
        # Update state
        state = next_state
```

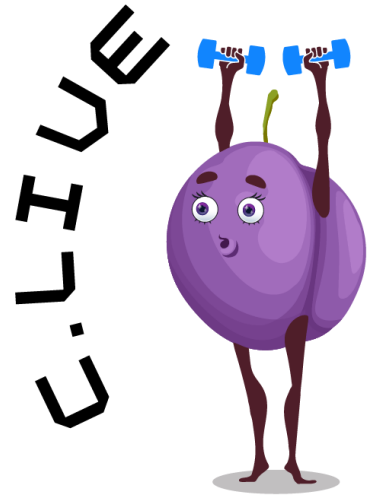
Пример применения RL-агентов:

- Симулятор реакций человеческого тела: https://www.researchgate.net/publication/341036633_A_closed-loop_healthcare_processing_approach_based_on_deep_reinforcement_learning

- Улучшение распознавания медицинских снимков: https://aapm.onlinelibrary.wiley.com/doi/full/10.1002/acm2.13898

- Увеличение точности постановки диагноза (диабет 2 типа): https://www.sciencedirect.com/science/article/pii/S0950705122009704?dgcid=rss_sd_all

Спасибо за внимание!



Руководитель проекта Когнитивный ассистент
старший преподаватель, к.т.н. Киселёв Г.А.
+79067993329
kiselev@isa.ru