Построение рекомендательной системы, основанной на обучении с подкреплением

Гришанов Алексей

Московский физико-технический институт
Физтех-школа прикладной математики и информатики
Факультет управления и прикладной математики
Кафедра «Интеллектуальные системы»

Научный руководитель д.ф.-м.н., К. В. Воронцов

Научный консультант, А. О. Янина

Москва 2020 г.

Цель исследования

Проблема

Задачу рекомендаций удобно формулировать и решать, сводя её к задаче обучения с подкреплением. Ряд подходящих алгоритмов имеют недостаток в том, что их предсказания детерминированы. Это является недостатком, поскольку для построения качественных рекомендаций важно исследовать рекомендательную среду.

Цель

Стимулировать агента к разнообразию рекомендаций.

Предлагается

Подобрать семейство случайных процессов для «зашумления» предсказаний агента

Постановка задачи рекомендаций

Заданы:

- $U = \{u_j |\ u_j \in \mathbb{R}^k,\ j \in 1, \dots n_{users}\}$ — множество субъектов (пользователей/users), для удобства преобразованных в векторы.
- $I = \{i_j |\ i_j \in \mathbb{R}^k,\ j \in 1, \dots n_{items}\}$ — множество объектов (рекомендуемых фильмов/items), преобразованных в векторы той же размерности, что и пользователи.
- $R = \|r_{ui}\|$ — матрица рейтингов размера $n_{users} \times n_{items}$, $r_{ui} \in \overline{1,5}$

Для каждого пользователя $u \in U$ требуется построить список объектов $y = \{y_j\}_{j=1}^{N}$, ранжированных по релевантности.

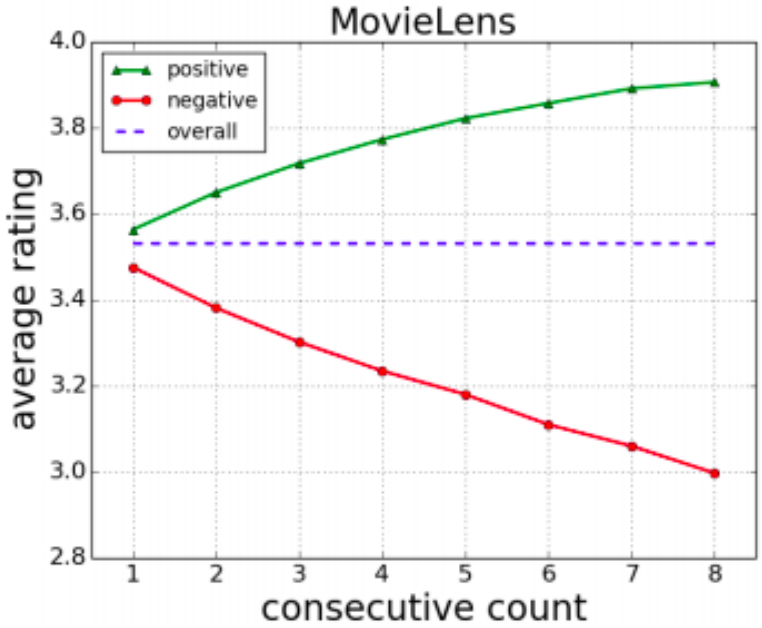
Критерии качества:

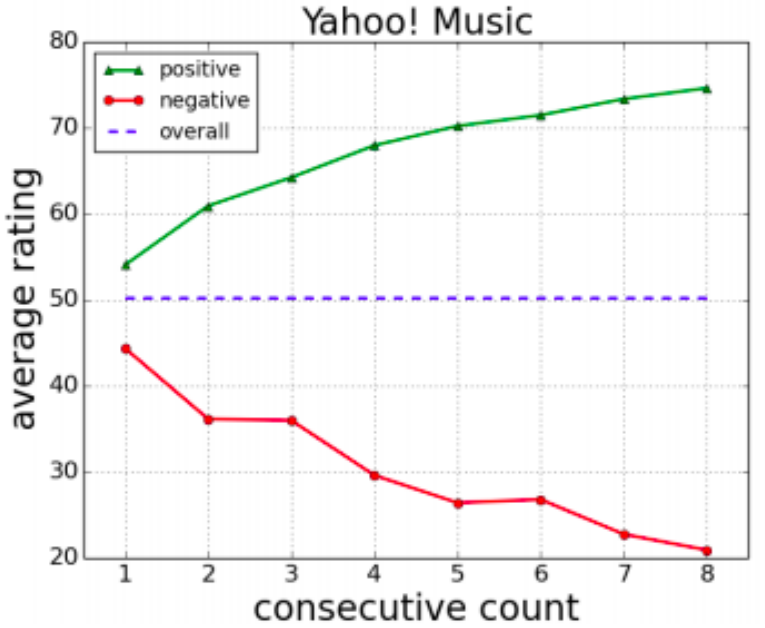
$$HR@p(y) = \sum_{j=1}^{p} rel_{y_j}; \qquad DCG@p(y) = \sum_{j=1}^{p} \frac{rel_{y_j}}{\log_2(j+1)},$$

где $rel_{y_j} = 1$, если объект y_j релевантен ($r_{ui} > 3$), иначе 0.

Пример

Рекомендации влияют на дальнейшие предпочтения пользователя:





Если система последовательно рекомендует пользователю релевантные товары, то он будет склонен ставить более высокие оценки (и наоборот). Для подобных задач подходящей областью представляется обучение с подкреплением.

Постановка задачи для обучения с подкреплением

Заданы: множество $\mathcal{S}$ состояний среды, множество $\mathcal{A}$ доступных действий агента, функция награды $\mathcal{R}: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$.

В момент времени t агент наблюдает состояние среды $s_t \in \mathcal{S}$, совершает действие $a_t \in \mathcal{A}$ в соответствии со свой стратегией (политикой, policy) $\pi: \mathcal{S} \times \mathcal{A} \to [0,1] = \mathbb{P}(a_t|s_t)$, переходит в состояние s_{t+1} и получает награду r_t.

Цель:

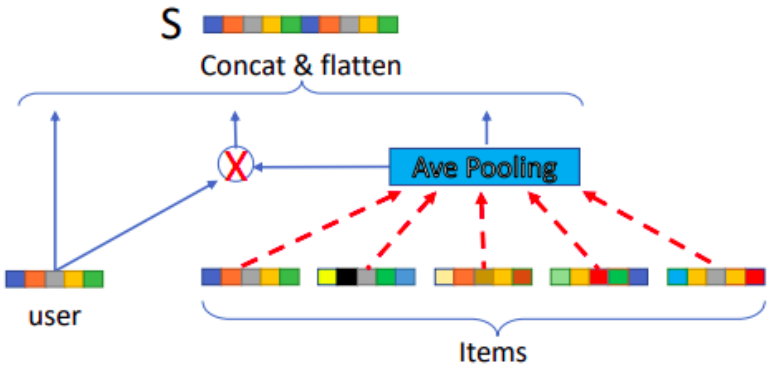
$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k}\right] \to \max_{\pi_\theta},$$

где $\gamma \in [0,1)$ — параметр, гарантирующий, что бесконечная сумма не будет расходиться при конечных значениях награды.

Рекомендательная среда

1. Состояние

$$s = [u, u \otimes \{w_l i_l \mid l = 1, ..., n\}, \{w_l i_l \mid l = 1, \ldots, n\}] \in \mathbb{R}^{3k},$$



где w_l — веса понижающего размерность слоя, а символом $\otimes$ обозначено поэлементное произведение.

2. Действия задаются с помощью вектора параметров $a \in \mathbb{R}^k$. Рекомендуется объект, скалярное произведение которого с вектором a наибольшее:

$$i = \underset{i_j \in \mathcal{A}}{\operatorname{argmax}}\ i_j a^T,$$

3.

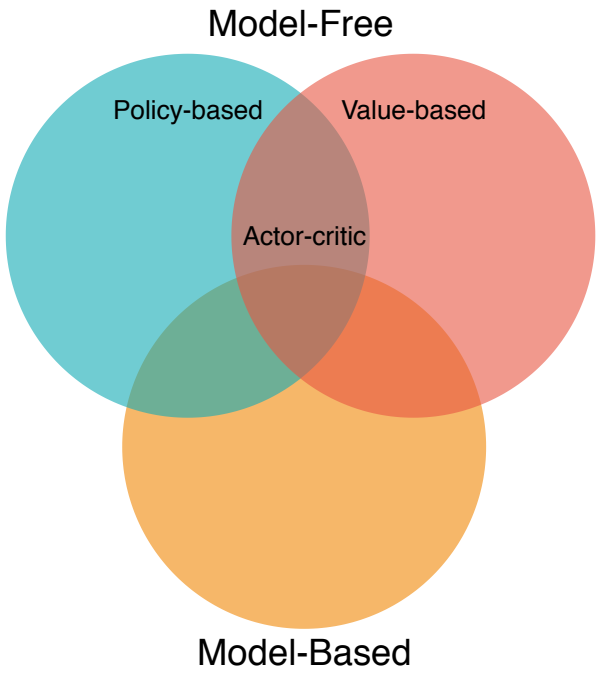
$$r_t = \begin{cases} 1, & \text{если } r_{ui} > 3 \\ 0, & \text{иначе} \end{cases}$$

здесь $r_t \in \mathcal{R}$, $r_{ui} \in R$.

Методы обучения с подкреплением

Агенты в обучении с подкреплением моделируют хотя бы одну из трёх компонент:

- политика (policy)
- функция ценности (value function)
- модель среды (environment model)



Преимущество детерминированной политики

Введём функцию ценности:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+1} \,\middle|\, s_t = s\right]$$

и Q-функцию:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+1} \,\middle|\, s_t = s,\ a_t = a\right]$$

$\nabla_{\theta} J(\theta)$ можно рассчитать следующим образом (считая π_{θ} дифференцируемой по θ):

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{s\sim\rho^{\pi}, a\sim\pi_{\theta}}\ \left[\nabla_{\theta} \log \pi_{\theta}(a|s) Q^{\pi}(s,a)\right],$$

где $\rho^{\pi}(s) = \sum_{t=1}^{\infty}\gamma^{t-1}\mathbb{P}(s_t = s|s_0,\pi)$

Детерминированная политика требует интегрирования только по состояниям среды:

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{s\sim\rho^{\pi}}\ \left[\nabla_{a} Q(s,a)|_{a=\pi(s)} \nabla_{\theta^{\pi}}\pi(s)\right]$$

Алгоритм 1 Deep Deterministic Policy Gradient (DDPG).

1: Инициализировать критик $Q_{\theta^Q}(s, a)$ весом θ^Q и актор $\pi_{\theta^\pi}(s)$ весом θ^π

2: Инициализировать Q' весом $\theta^{Q'} = \theta^Q$ и π' весом $\theta^{\pi'} = \theta^\pi$

3: Инициализировать буфер B

4: **for** episode $= 1, \dots M$ **do**

5: Инициализировать случайный процесс P

6: **for** $t = 1, \dots N$ **do**

7: Выбрать действие $a_t = \pi(s_t) + P_t$ в соответствии с текущей политикой и добавочным шумом

8: Сделать действие a_t, получить награду r_t, перейти в состояние s_{t+1}

9: Сохранить (s_t, a_t, r_t, s_{t+1}) в B

10: Сэмплировать N штук (s_i, a_i, r_i, s_{i+1}) из B

11: Вычислить $y_i = r_i + \gamma Q'(s_{i+1}, \pi'(s_{i+1}))$

12: Обновить критик, минимизируя $L = \frac{1}{N} \sum_i (y_i - Q_{\theta^Q}(s_i, a_i))^2$

13: Обновить актор, используя сэмплированный градиент политики:

$$\nabla_{\theta^\pi} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a)|_{s=s_i, a=\pi(s_i)} \nabla_{\theta^\pi} \pi(s)|_{s=s_i}$$

14: Обновить веса:

$$\theta^{Q'} = \tau\theta^{Q'} + (1 - \tau)\theta^Q$$

$$\theta^{\pi'} = \tau\theta^{\pi'} + (1 - \tau)\theta^\pi$$

15: **end for**

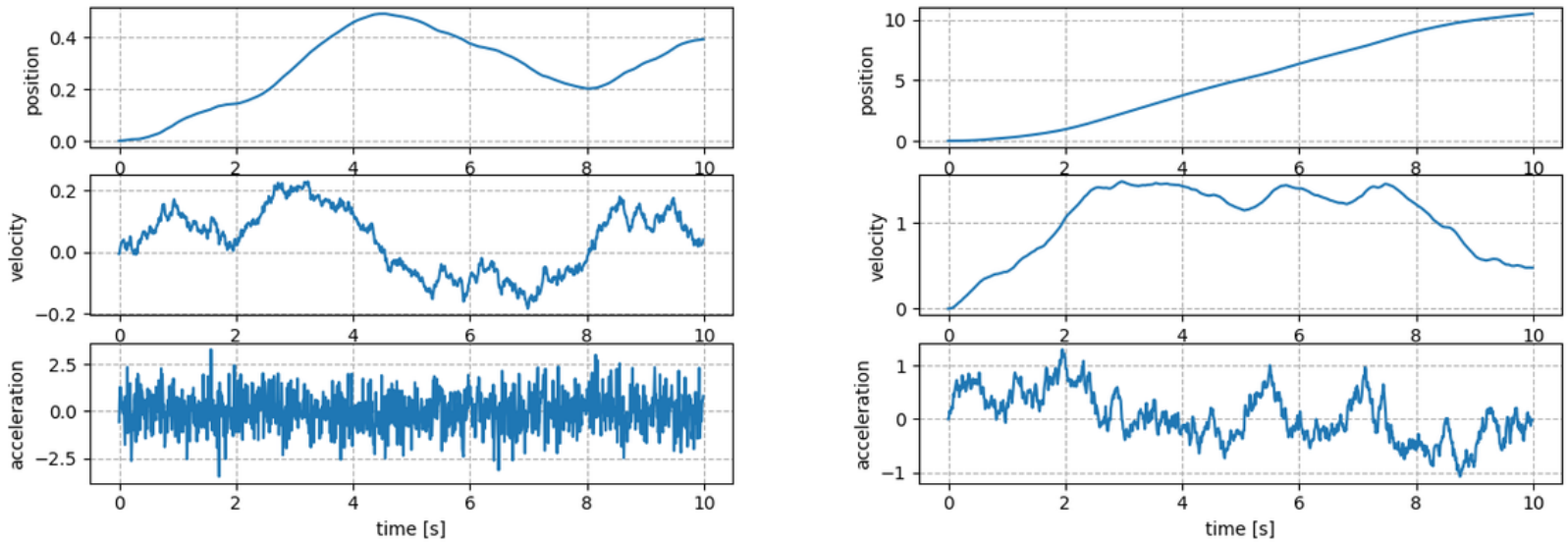
16: **end for**

Семейство случайных процессов для шума

Процесс O_t Орнштейна — Уленбека задаётся следующим стохастическим дифференциальным уравнением:

$$dO_t = \theta(\mu - O_t)\,dt + \sigma\,dW_t$$

где $\theta > 0$, $\sigma > 0$ и $\mu \in \mathbb{R}$ — параметры, а W_t — винеровский процесс.



Сравнение гауссовского шума (слева) и шума из процесса Орнштейна — Уленбека (адаптировано с сайта quora.com)

Данные

Movielens (1M)

- 6040 пользователей
- 3952 фильмов
- 1000209 выставленных рейтингов

Обучающая выборка содержит 80% рейтингов, тестовая — 20 %
Не учитывались пользователи с менее чем 20 рейтингами

Оценка качества модели

Алгоритм 1 Схема валидации

Разбить тестовую выборку на батчи $\{X_j\}_{j=1}^{M}$ по 100 элементов, 1 из которых релевантный, 99 — случайно без повторов выбраны из нерелевантных

for t $= 1, \ldots M$ **do**

Получить текущее состояние среды s_t

Вычислить вектор предсказаний модели a_t

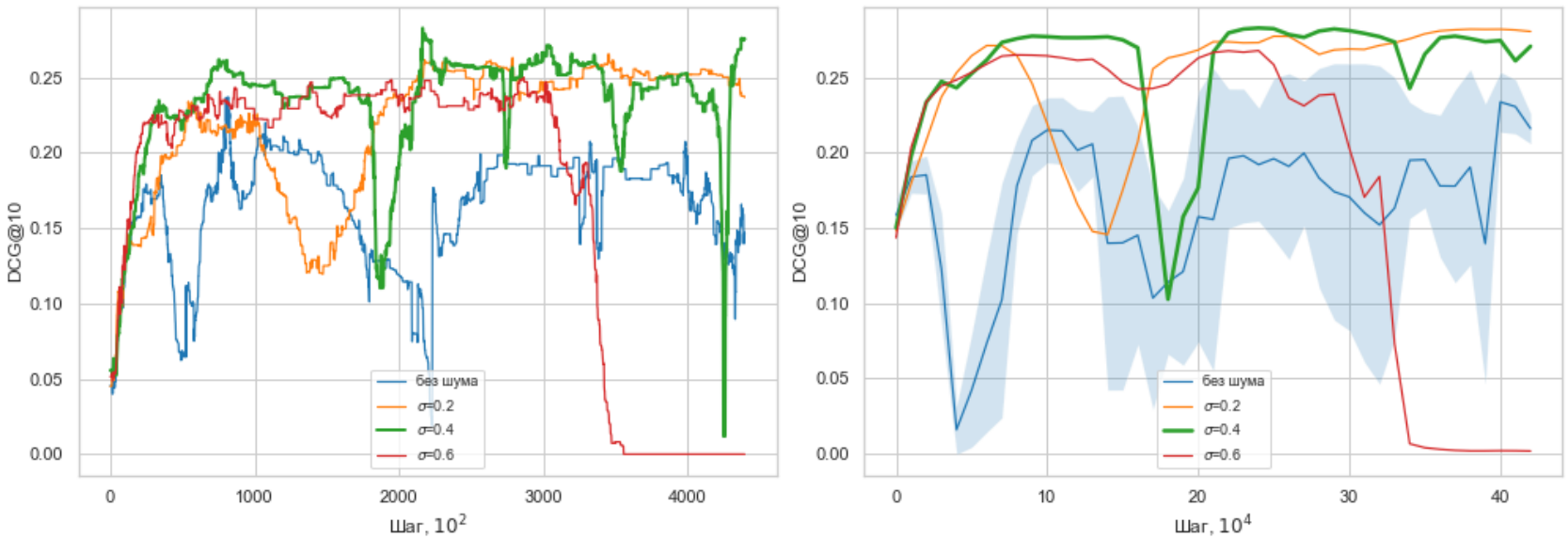
Составить список рекомендаций $y = \underset{i_j \in X_t}{argtop_{10}} \left(i_j a_t^T\right)$,

где $argtop_k$ — операция, возвращающая k наибольших элементов, отсортированных по убыванию

Вычислить DCG@10(y), HR@10(y)

end for **Выход:** средние значения DCG@10, HR@10 за M батчей

Графики обучения



Слева качество измерялось по рейтингам одного случайного фиксированного пользователя, справа — по всем тестовых данных. Синее затемнение — стандартное отклонение по 3 запускам.

Результаты эксперимента

Модель	DCG@10	HR@10
$\sigma = 0.6$	0.268	0.487
$\boldsymbol{\sigma = 0.4}$	**0.282**	**0.509**
$\sigma = 0.2$	**0.282**	0.504
без шума	0.254	0.454
Случайные рекомендации	~ 0.05	~ 0.1

Таблица: Сравнение разных вариантов шума

Результаты, выносимые на защиту

- Разработана модель ранжирования рекомендаций на основе алгоритма обучения с подкреплением актор-критик с использованием стохастических процессов Орнштейна — Уленбека
- Показано, что оптимизация дисперсии процессов Орнштейна — Уленбека улучшает качество рекомендаций по критериям DCG и HR.
- Показано, что детерминированные предсказания затрудняют исследование среды агентом.