ФГАОУВО «Московский физико-технический институт
(национальный исследовательский университет)»
Физтех-школа прикладной математики и информатики
Кафедра «Интеллектуальные системы»

Гришанов Алексей Владимирович

Построение рекомендательной системы, основанной на обучении с подкреплением

03.03.01 – Прикладные математика и физика

Выпускная квалификационная работа бакалавра

Научный руководитель:
д. ф.-м. н. Воронцов Константин Вячеславович

Консультант:
Янина Анастасия Олеговна

Москва
2020

Содержание

Аннотация

В последнее время задачу рекомендаций часто формулируют и решают, сводя её к задаче обучения с подкреплением. Важной проблемой является баланс между эксплуатированием известных знаний о пользователе и исследованием его предпочтений с целью своевременного подстраивания под новые интересы. В работе изучаются стратегии исследования среды в полученной постановке. Для стимулирования агента к разнообразию рекомендаций предлагается использовать случайные процессы Орнштейна-Уленбека. Полученные модели тестируются на наборе данных Movielens (1M). Эксперименты показывают, что при использовании предложенного подхода ускоряется сходимость и улучшаются итоговые результаты модели.

Ключевые слова: *рекомендательные системы, обучение с подкреплением, Deep Deterministic Policy Gradient (DDPG).*

1 Введение

Взаимодействие рекомендательной системы и пользователя — это сложный процесс. Часто он является многошаговым, в процессе рекомендаций могут возникать новые интересы пользователя или возобновляться старые. Особенный интерес представляют поисково-рекомендательные сценарии, когда на любом шаге стратегии пользователя могут измениться, нет четко определенной цели поиска или нет четкого понимания, какая именно информация поможет ответить на пользовательский поисковый запрос. Для такого рода задач подходящим методом решения представляется обучение с подкреплением.

Длительное время для задачи рекомендаций использовались многорукие и контекстуальные бандиты, например [1], однако они имеют ряд ограничений. В классических многоруких бандитах считается, что на награду влияет только сделанное действие, в контекстуальных — добавляется учёт текущего состояния среды, однако и эта группа алгоритмов не предназначена для сложных, «многоходовых» стратегий.

В последнее время возрастает интерес к полноценным постановкам обучения с подкреплением для рекомендательных систем. Поскольку множество объектов (товаров, фильмов, статей и т.д.), доступных для рекомендаций, обычно велико, пространство состояний в обучении с подкреплением выгодно делать непрерывным. В этих условиях перспективные алгоритмы из обучения с подкреплением, применимые к рекомендациям, можно разделить на 2 типа. В первом содержатся продвинутые on-policy методы, такие как PPO [2], trulyPPO [3]. Во втором — такие off-policy методы как DDPG [4] и TD3 [5].

В данной работе исследуются алгоритмы второго типа. В настоящее время они чаще встречаются в публикациях, например [6–9]. Их рекомендации получаются детерминированными как в процессе тренировки, так и при применении обученной модели. Это является недостатком, поскольку для построения ценных рекомендаций важно исследовать рекомендательную среду.

В данной работе для поощрения исследования среды предлагается добавлять к детерминированным предсказаниям агента шум, сгенерированный из процесса Орнштейна — Уленбека. В отличие от гауссовского

шума его значения к текущему моменту времени коррелируют с предыдущими. Он более точно соотносится с целями исследования среды и поэтому больше подходит для применения в задаче рекомендаций. Насколько известно, до этого в обучении с подкреплением применительно к рекомендательным системам этот подход не использовался.

Таким образом, основной целью данной работы является анализ и устранение недостатков текущих подходов к обучению с подкреплением в рекомендательных средах, вызванных недостаточным исследованием среды агентом из-за детерминированности предсказаний. В рамках поставленной цели необходимо решить следующие задачи:

1. Найти и подготовить датасет для рекомендательного моделирования.

2. Реализовать алгоритм тренировки модели на основе подхода актор-критик (предлагается использовать Deep Deterministic Policy Gradient [4]).

3. Стимулировать алгоритм из пункта 2 (DDPG) к исследованию среды за счет добавления к детерминированным предсказаниям агента шум, сгенерированный из процесса Орнштейна — Уленбека.

С точки зрения практического исполнения данная работа включает в себя имплементацию алгоритма DDPG [4]. Код, воспроизводящий эксперименты, размещен по ссылке https://github.com/shashist/recsys-rl

2 Рекомендательная система

2.1 Постановка задачи

Заданы:

- $U = \left\{u_j |\ u_j \in \mathbb{R}^k,\ j \in 1, \dots n_{users}\right\}$ — множество субъектов (пользователей/users), для удобства преобразованных в векторы. Это преобразование можно сделать например с помощью матричной факторизации или VAE [10].
- $I = \left\{i_j |\ i_j \in \mathbb{R}^k,\ j \in 1, \dots n_{items}\right\}$ — множество объектов (рекомендуемых фильмов/items), преобразованных в векторы той же размерности, что и пользователи.
- $R = \|r_{ui}\|$ — матрица рейтингов размера $n_{users} \times n_{items}$, $r_{ui} \in \overline{1,5}$

В рекомендательных системах существуют различные постановки задач, такие как прогнозирование неизвестных рейтингов r_{ui}, оценивание сходства $\rho(u, u'),\ \rho(i, i'),\ \rho(u, i)$ и т. д. В данной работе исследуются методы, формирующие список рекомендаций для пользователей.

Более формально, для каждого пользователя $u \in U$ требуется построить список $y = \{y_j\}_{j=1}^{N}$ из наиболее релевантных объектов, отсортированный по убыванию.

Для того, чтобы определять, насколько построенный алгоритмом список соответствует истинным значениям релевантности, введём два критерия качества:

$$HR@p(y) = \sum_{j=1}^{p} rel_{y_j};$$

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

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

Первый критерий (Hit Rate) считает количество релевантных объектов среди первых p элементов списка. Второй (Discounted Cumulative Gain) дополнительно дисконтирует это количество, учитывая позицию в рекомендованном списке. Поскольку важнее правильно выдавать наиболее релевантные объекты, для знаменателя больше подходит логарифм порядкового номера из списка, а не этот номер напрямую.

В дальнейшем будем оптимизировать усреднённые значения этих критериев по всем подвыборкам (батчам) из тестовой выборки.

2.2 Сведение к задаче обучения с подкреплением

Традиционные подходы к решению задачи рекомендаций используют коллаборативную фильтрацию [11, 12] или контентно-основанные рекомендации [13, 14]. Эти методы имеют ряд недостатков: проблема холодного старта, отсутствие возможности онлайн-обучения, необходимость хранить в памяти всю матрицу рейтингов. Кроме того, такие подходы основываются на исторических данных и не учитывают постоянно возникающих новых интересов пользователей.

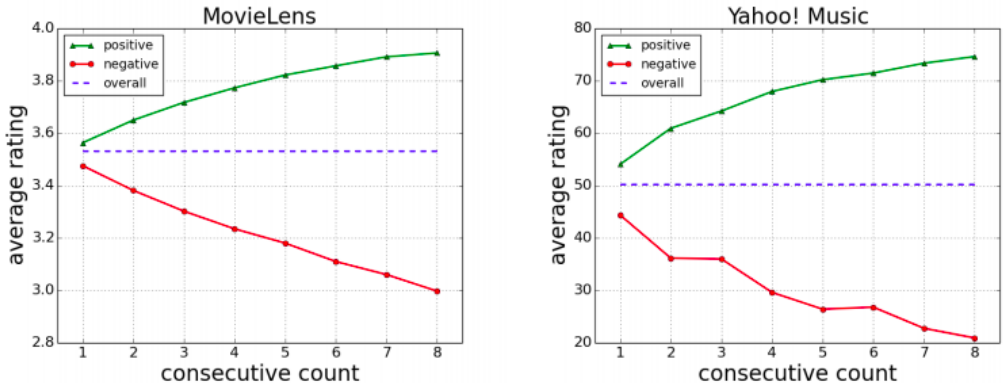

Рис. 1: Анализ последовательных рекомендаций. Взято из [15]

В работе [15] (рис. 1) показан пример того, как начальные рекомендации влияют на дальнейшие оценки пользователей. Видно, что если система последовательно рекомендует пользователю релевантные товары, то он будет склонен ставить более высокие оценки и наоборот. Это показывает, что рекомендации пользователю следует рассматривать как последовательный процесс принятия решений.

Стандартным подходом к решению подобных задач является обучение с подкреплением. Оно позволяет явно использовать последовательность действий пользователя, чтобы лучше подстраиваться под его интересы.

3 Обучение с подкреплением

3.1 Постановка задачи

В задачах обучения с подкреплением рассматривается агент, взаимодействующий со средой. Конечная цель — научить агента совершать оптимальные действия для достижения заданной цели.

Задано множество $\mathcal{S}$ состояний среды и множество $\mathcal{A}$ доступных действий агента. Введём также функцию переходов $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$.

Согласно гипотезе Р. Саттона, автора книги [16], произвольные цели и задачи могут быть сформулированы как максимизация математического ожидания суммы последовательно полученного скалярного сигнала. В обучении с подкреплением сигнал, получаемый от среды, называется функцией 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.

Часто удобно задавать стратегию агента в параметрическом виде: $\pi_\theta = \mathbb{P}(a|s, \theta)$. Решить задачу обучения с подкреплением означает найти стратегию, максимизирующую дисконтированную сумму наград:

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

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

3.2 Среда для рекомендательной системы

1. Вектор состояния среды будем описывать аналогично [15]. Он состоит из вектора текущего пользователя u, вектора, отражающего n последних релевантных для него объектов и вектора, содержащего их попарные произведени (рис. 2).

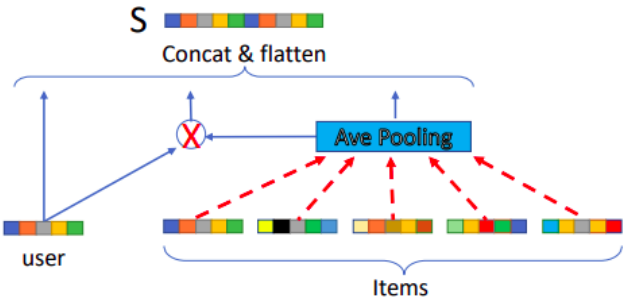

Рис. 2: Описание состояния среды. Взято из [15]

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

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

2. Действия агента удобно задавать с помощью вектора $a \in \mathbb{R}^k$, следуя статье [17]. Пользователю рекомендуется объект, скалярное произведение которого с вектором a наибольшее:

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

3. Наконец награды будем задавать следующим образом $(r_t \in \mathcal{R}, r_{ui} \in R)$:

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

3.3 Методы решения

В обучении с подкреплением можно выделить 2 типа методов (см. рис. 3).

В первой группе (model-based) выучивается модель среды $p(s_{t+1}|s_t, a_t)$, что требует большого количества наблюдений и времени. В ранних работах, например в [18] предпринимались попытки использовать model-based подходы, однако это мало применимо для рекомендательных си-

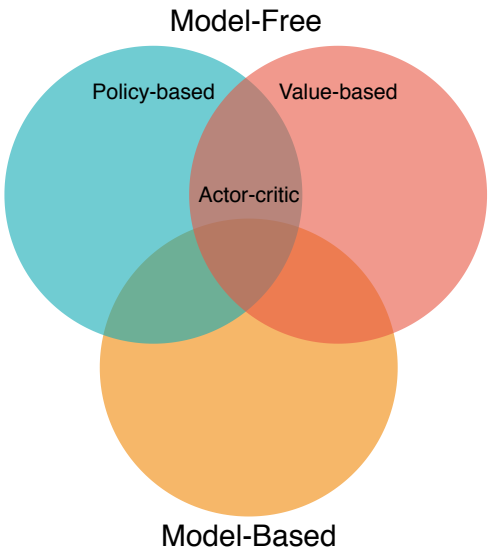

Рис. 3: Группы алгоритмов обучения с подкреплением

стем из-за высокой ресурсозатратности, далее этот подход рассматриваться не будет.

Алгоритмы из второй группы либо выучивают функцию ценности и используют её для формирования стратегии (value-based), либо параметризуют стратегию и обновляют её напрямую (policy-based), либо делают и то и то (actor-critic). В работе исследуется метод, относящийся к классу актор-критик. Для целостности изложения кратко опишем все три класса алгоритмов.

3.3.1 Value-based

Введём функцию ценности:

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

и Q-функцию:

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

Выучив оптимальное значение Q-функции можно задать стратегию, например как $\pi(s_t) = \underset{a_t}{\operatorname{argmax}}\, Q(s_t, a_t)$

В непрерывных средах трудно применять напрямую данный подход, поскольку это требует жадной оптимизации на каждом шаге. Вместо этого воспользуемся подходом актор-критик, где критик будет выучивать Q - функцию, а актор — политику. Для этого предварительно опишем часть, связанную с актором в следующем подразделе.

3.3.2 Policy-based

Алгоритмы, основанные на оптимизации политики, обновляют параметры π_θ, максимизируя ожидаемую награду $J(\theta)$. $\nabla_\theta J(\theta)$ можно рассчитать, используя результат из [19] (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 дополнительно введено распределение состояний с учётом дисконтирования $\rho^\pi(s) = \sum_{t=1}^{\infty} \gamma^{t-1} \mathbb{P}(s_t = s|s_0, \pi)$

Истинные значения Q - функции обычно неизвестны, их можно заменить например на дисконтированные суммы полученных в сессии наград.

3.3.3 Actor-Critic

В подходе актор-критик обучается и параметризованная Q-функция $Q_{\theta^Q}(s,a)$ (критик) и политика π_θ (актор), так что для вычисления $\nabla_\theta J(\theta)$ используется значение $Q_{\theta^Q}(s,a)$.

Заметим, что вычисление $\nabla_\theta J(\theta)$ требует интегрирования и по пространству состояний, и по пространству действий, что требует большего количества данных, особенно при больших размерностях. Традиционным является подход с использованием детерминированной политики, для неё градиент ожидаемой награды $\nabla_\theta J(\theta)$ имеет вид [20]:

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

В результате получим следующий алгоритм:

Алгоритм 1 Deep Deterministic Policy Gradient (DDPG).

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

Инициализировать Q' весом $\theta^{Q'} = \theta^Q$ и π' весом $\theta^{\pi'} = \theta^\pi$
Инициализировать буфер B
for episode $= 1, \ldots, M$ **do**
  Инициализировать случайный процесс P
  for $t = 1, \ldots, N$ **do**
    Выбрать действие $a_t = \pi(s_t) + P_t$ в соответствии с текущей политикой и добавочным шумом
    Сделать действие a_t, получить награду r_t, перейти в состояние s_{t+1}
    Сохранить (s_t, a_t, r_t, s_{t+1}) в B
    Сэмплировать N штук (s_i, a_i, r_i, s_{i+1}) из B
    Вычислить $y_i = r_i + \gamma Q'(s_{i+1}, \pi'(s_{i+1}))$
    Обновить критик, минимизируя $L = \frac{1}{N} \sum_i (y_i - Q_{\theta^Q}(s_i, a_i))^2$
    Обновить актор, используя сэмплированный градиент политики:

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

    Обновить веса:

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

  end for
end for

Далее мы остановимся подробнее именно на этом алгоритме и зададимся целью стимулировать его к исследованию среды.

4 Исследование среды

4.1 Постановка задачи

Введём семейство $\mathscr{P} = \{P_\alpha\}$ случайных процессов. Итоговой целью будет подобрать такой случайный процесс $P \in \mathscr{P}$, при использовании которого обучение модели по алгоритму 1 будет максимизировать критерии качества DCG@10 и HR@10.

В качестве семейства $\mathscr{P}$ рассмотрим процессы Орнштейна - Уленбека.

4.2 Процесс Орнштейна — Уленбека

Рассмотрим следующее стохастическое дифференциальное уравнение:

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

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

Таким уравнением задаётся процесс O_t Орнштейна — Уленбека [21].

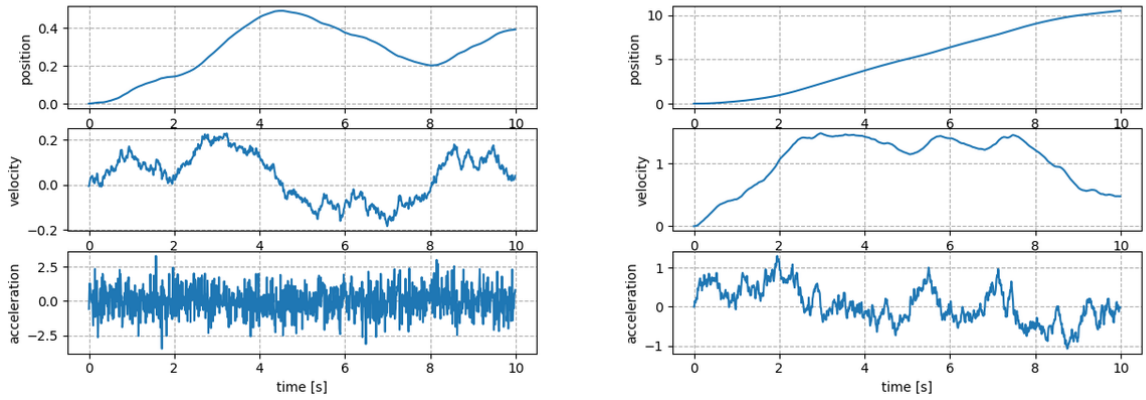

Рис. 4: Сравнение гауссовского шума (слева) и шума из процесса Орнштейна — Уленбека (адаптировано с сайта quora.com)

В отличие от гауссовского шума, для процесса Орнштейна — Уленбека шум, сгенерированный к текущему моменту, коррелирует с предыду-

щим, поэтому знак шума сохраняется на более длительном промежутке времени, а не меняется скачкообразно.

В частности процесс Орнштейна — Уленбека используется для моделирования движения броуновской частицы с трением. Параметром μ описывается равновесное состояние, σ — дисперсия, вызванная соударениями, θ — «сила притяжения» к исходному состоянию.

Далее в эксперименте зафиксируем $\mu = 0$, $\theta = 0.15$, а параметр θ будем подбирать, максимизируя DCG@10 и HR@10.

Для практической реализации удобно воспользоваться методом Эйлера–Маруямы и дискретизировать стохастическое дифференциальное уравнение, приводя его в к виду:

$$O_{n+1} = O_n + \theta(\mu - O_n)\,\Delta t + \sigma\,\Delta W_n,$$

здесь $\Delta W_n = W_{t_{n+1}} - W_{t_n} \sim \mathcal{N}(0, \Delta t) = \sqrt{\Delta t}\,\mathcal{N}(0, 1)$

5 Вычислительный эксперимент

5.1 Данные

Для экспериментов был выбран набор данных с рейтингами фильмов Movielens (1M) [22], содержащий:

- 6040 пользователей;
- 3952 фильмов;
- 1000209 выставлений рейтингов.

В обучающая выборку были случайным образом отложены 80% рейтингов, оставшиеся 20% — в тестовую.

Из данных были убраны пользователи с менее чем 20 рейтингами в обучающей выборке. После этого оставшиеся пользователи и фильмы были преобразованы в векторы размерности $k = 8$, сгенерированными из нормального распределения с математическим ожиданием $m_{normal} = 0$ и среднеквадратическим отклонением $\sigma_{normal} = 0.01$, которые далее обновлялись вместе с другими параметрами модели.

5.2 Агент

В качестве актора и критика для алгоритма 1 использовались двухслойные персептоны со скрытым слоем размера 16.

Актор:

$$\mu_{\theta^\mu}(s) = (\theta_3^\mu)^T \cdot ReLU\left((\theta_1^\mu)^T s + \theta_2^\mu\right) + \theta_4^\mu,$$

где $\theta_1^\mu \in \mathbb{R}^{24\times16},\ \theta_2^\mu \in \mathbb{R}^{16},\ \theta_3^\mu \in \mathbb{R}^{16\times8}, \theta_4^\mu \in \mathbb{R}^8$

Критик:

$$Q_{\theta^Q}(s,a) = \left(\theta_3^Q\right)^T \cdot ReLU\left(\left(\theta_1^Q\right)^T \begin{bmatrix} s \\ a \end{bmatrix} + \theta_2^Q\right) + \theta_4^Q,$$

где $\theta_1^Q \in \mathbb{R}^{32\times16},\ \theta_2^Q \in \mathbb{R}^{16},\ \theta_3^Q \in \mathbb{R}^{16\times1}, \theta_4^Q \in \mathbb{R}^1$

5.3 Валидация модели

Процесс оценивания качества модели описан в алгоритме 2

Алгоритм 2 Схема валидации

1: Разбить тестовую выборку на батчи $\{X_j\}_{j=1}^{M}$ по 100 элементов, где 1 релевантный, 99 случайно без повторов выбраны из нерелевантных
2: **for** t $= 1, \dots M$ **do**
3: Получить текущее состояние среды s_t
4: Вычислить вектор предсказаний модели a_t
5: 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
6: Вычислить DCG@10(y), HR@10(y)
7: **end for**
Выход: средние значения DCG@10, HR@10 за M батчей

5.4 Результаты

В ходе обучения критерии качества измерялись двумя способами. Раз в 10000 шагов измерялось качество на всей тестовой выборкее, раз в 100 шагов — на одном случайно выбранном изначально пользователе.

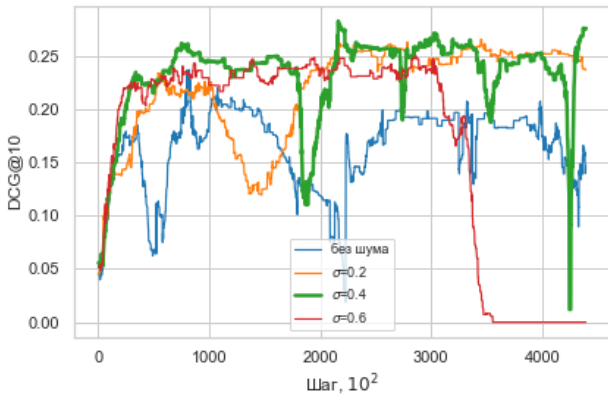

Рис. 5: Кривая обучения DCG@10 для одного пользователя

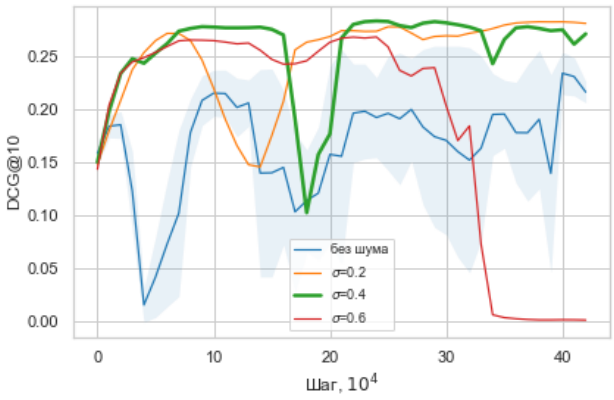

Рис. 6: Кривая обучения DCG@10 для всех пользователей

Синим затемнением на графике отмечено стандартное отклонение по трём запускам модели без шума.

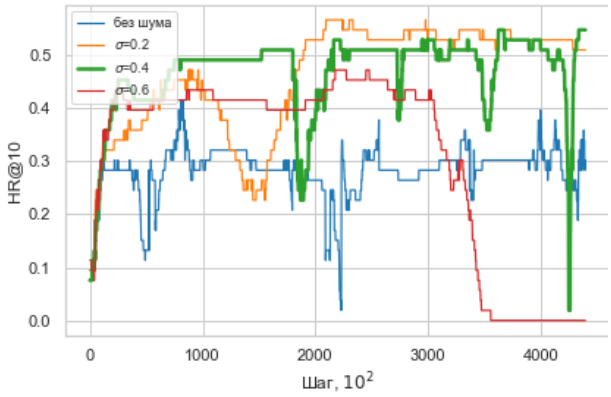

Рис. 7: Кривая обучения HR@10 для одного пользователя

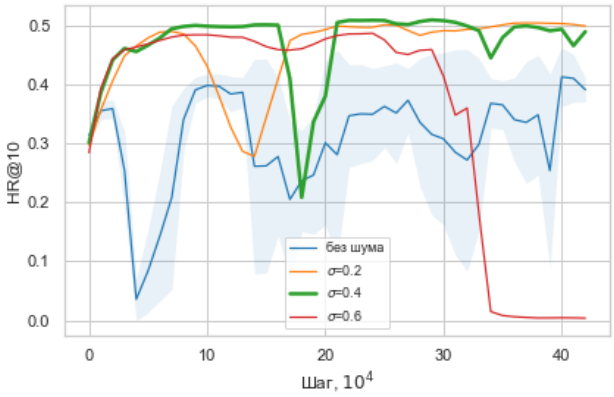

Рис. 8: Кривая обучения HR@10 для всех пользователей

Для итогового вычисления качества модели использовались наилучшие веса из истории оценивания по всем пользователям. Результаты представлены в таблице 1

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 1: Сравнение разных вариантов шума

Видно, что использование шума повышает как итоговые критерии качества, так и их промежуточные значения в процессе обучения.

6 Заключение

6.1 Итоги работы

В рамках проведенного исследования была достигнута поставленная цель и решены сформулированные в начале исследования задачи. На защиту выносятся следующие результаты:

1. Разработана модель ранжирования рекомендаций на основе алгоритма обучения с подкреплением актор-критик с использованием стохастических процессов Орнштейна — Уленбека

2. Показано, что оптимизация дисперсии процессов Орнштейна — Уленбека улучшает качество рекомендаций по критериям DCG и HR.

3. Показано, что детерминированные предсказания затрудняют исследование среды агентом.

6.2 Дальнейшие исследования

Рассматриваются следующие возможные варианты развития данной работы:

1. Изучить подходы к построению состояний среды.

2. Сравнить рассмотренный в данной работе метод с другими упомянутыми перспективными алгоритмами, такими как trulyPPO [3].

3. Исследовать более сложные постановки задач рекомендательного моделирования, включая многошаговые рекомендательные сценарии, где на каждой итерации происходит переформулировка или уточнение запроса. Такая постановка задачи близка к разведочному поиску. Обычно разведочный поиск включает в себя несколько итераций поисковых запросов, а также используется в случаях, когда пользователь не имеет четкого запроса или представления о

требуемом результате поиска. Цель такого поиска — не только найти информацию, точно соответствующую запросу, но и осознать, изучить новую тему. Обучение с подкреплением — подходящий метод для решения такой задачи. Таким образом, данное исследование может быть продолжено не только в рамках рекомендательного моделирования, но и в рамках алгоритмов поиска текстовых документов.

Также в дальнейшем планируется перейти на использование фреймворка Catalyst (включен в Pytorch Ecosystem) [23].

Список литературы

[1] A Contextual-Bandit Approach to Personalized News Article Recommendation / Lihong Li, Wei Chu, John Langford, Robert E. Schapire // Proceedings of the 19th International Conference on World Wide Web. — WWW '10. — New York, NY, USA : Association for Computing Machinery, 2010. — P. 661–670. — Access mode: https://doi.org/10.1145/1772690.1772758.

[2] Schulman John, Wolski Filip, Dhariwal Prafulla et al. Proximal Policy Optimization Algorithms. — 2017. — 1707.06347.

[3] Wang Yuhui, He Hao, Wen Chao, Tan Xiaoyang. Truly Proximal Policy Optimization. — 2019. — 1903.07940.

[4] Continuous control with deep reinforcement learning. / Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel et al. // ICLR / Ed. by Yoshua Bengio, Yann LeCun. — 2016.

[5] Fujimoto Scott, van Hoof Herke, Meger David. Addressing Function Approximation Error in Actor-Critic Methods // Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 / Ed. by Jennifer G. Dy, Andreas Krause. — Vol. 80 of Proceedings of Machine Learning Research. — PMLR, 2018. — P. 1582–1591. — Access mode: http://proceedings.mlr.press/v80/fujimoto18a.html.

[6] DRN: A Deep Reinforcement Learning Framework for News Recommendation / Guanjie Zheng, Fuzheng Zhang, Zihan Zheng et al. // Proceedings of the 2018 World Wide Web Conference. — 2018.

[7] End-to-End Deep Reinforcement Learning Based Recommendation with Supervised Embedding / Feng Liu, Huifeng Guo, Xutao Li et al. // Proceedings of the 13th International Conference on Web Search and Data Mining. — WSDM '20. — New York, NY, USA : Association for Computing Machinery, 2020. — P. 384–392. — Access mode: https://doi.org/10.1145/3336191.3371858.

[8] Deep reinforcement learning for page-wise recommendations / Xiangyu Zhao, Long Xia, Liang Zhang et al. // Proceedings of the 12th ACM Conference on Recommender Systems - RecSys 18. — ACM Press, 2018. — Access mode: https://doi.org/10.1145/3240323.3240374.

[9] Deep Reinforcement Learning for List-wise Recommendations / Xiangyu Zhao, Liang Zhang, Zhuoye Ding et al. // ArXiv. — 2018. — Vol. abs/1801.00209.

[10] Kingma Diederik P, Welling Max. Auto-Encoding Variational Bayes. — 2013. — 1312.6114.

[11] Koren Yehuda, Bell Robert, Volinsky Chris. Matrix Factorization Techniques for Recommender Systems // Computer. — 2009. — Aug. — Vol. 42, no. 8. — P. 30–37. — Access mode: https://doi.org/10.1109/MC.2009.263.

[12] Google News Personalization: Scalable Online Collaborative Filtering / Abhinandan S. Das, Mayur Datar, Ashutosh Garg, Shyam Rajaram // Proceedings of the 16th International Conference on World Wide Web. — WWW '07. — New York, NY, USA : Association for Computing Machinery, 2007. — P. 271–280. — Access mode: https://doi.org/10.1145/1242572.1242610.

[13] Philip Simon, Shola Peter, Abari Ovye. Application of Content-Based Approach in Research Paper Recommendation System for a Digital Library // International Journal of Advanced Computer Science and Applications. — 2014. — 10. — Vol. 5.

[14] Kompan Michal, Bieliková Mária. Content-Based News Recommendation // E-Commerce and Web Technologies / Ed. by Francesco Buccafurri, Giovanni Semeraro. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2010. — P. 61–72.

[15] Deep Reinforcement Learning based Recommendation with Explicit User-Item Interactions Modeling / Feng Liu, Ruiming Tang, Xutao Li et al. // ArXiv. — 2018. — Vol. abs/1810.12027.

[16] Sutton Richard S., Barto Andrew G. Reinforcement Learning: An Introduction. — Cambridge, MA, USA : A Bradford Book, 2018. — ISBN: 0262039249.

[17] Dulac-Arnold Gabriel, Evans Richard, van Hasselt Hado et al. Deep Reinforcement Learning in Large Discrete Action Spaces. — 2015. — 1512.07679.

[18] Shani Guy, Brafman Ronen I., Heckerman David. An MDP-Based Recommender System // Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence. — UAI'02. — San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2002. — P. 453–460.

[19] Policy Gradient Methods for Reinforcement Learning with Function Approximation / Richard S. Sutton, David McAllester, Satinder Singh, Yishay Mansour // Proceedings of the 12th International Conference on Neural Information Processing Systems. — NIPS'99. — Cambridge, MA, USA : MIT Press, 1999. — P. 1057–1063.

[20] Deterministic Policy Gradient Algorithms / David Silver, Guy Lever, Nicolas Heess et al. // Proceedings of the 31st International Conference on Machine Learning / Ed. by Eric P. Xing, Tony Jebara. — Vol. 32 of Proceedings of Machine Learning Research. — Bejing, China : PMLR, 2014. — 22–24 Jun. — P. 387–395. — Access mode: http://proceedings.mlr.press/v32/silver14.html.

[21] Uhlenbeck G. E., Ornstein L. S. On the theory of the Brownian motion // Phys. Rev. — 1930. — Vol. 36, no. 3. — P. 823–841.

[22] Harper F. Maxwell, Konstan Joseph A. The MovieLens Datasets: History and Context // ACM Trans. Interact. Intell. Syst. — 2015. — Dec. — Vol. 5, no. 4. — Access mode: https://doi.org/10.1145/2827872.

[23] Kolesnikov Sergey. Accelerated DL RD. — https://github.com/catalyst-team/catalyst. — 2018.