Труды МАИ. 2022. № 126
Trudy MAI, 2022, no. 126

Научная статья
УДК 004.891
DOI: 10.34759/trd-2022-126-20

РАЗРАБОТКА РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ НА ОСНОВЕ СЕССИЙ С ИСПОЛЬЗОВАНИЕМ МНОГОУРОВНЕВОЙ СИСТЕМЫ ОТБОРА КАНДИДАТОВ

Мохов Алексей Игоревич[1], Кислинский Вадим Геннадьевич[2], Алексейчук Андрей Сергеевич[3]

[1,3]Московский авиационный институт (национальный исследовательский университет), МАИ, Москва, Россия
[2]Московский физико-технический институт (национальный исследовательский университет), МФТИ, Долгопрудный, Московская обл., Россия
[1]AIMokhov@mai.ru
[2]kislinskiy.vg@phystech.edu
[3]alexejchuk@gmail.com

Аннотация. В статье рассматривается рекомендательная система, основанная на сессиях – взаимодействиях вида «пользователь-объект», которые происходят в течение некоторого времени – и производится сравнение подходов к построению рекомендаций с ранжированием и без него. В работе рассматриваются парные критерии для задач ранжирования, а также методы подбора кандидатов на основе векторных представлений для объектов (товаров). В ходе исследования была

разработана рекомендательная система и ПО для сравнения оценки двух и более подходов к задаче рекомендации. Для оценки модели используются метрики качества, применяемые в задачах ранжирования и построения рекомендаций. Приведены результаты сравнения алгоритмов обучения ранжированию на реальных данных.

Ключевые слова: обучение ранжированию, машинное обучение, рекомендательная система, метрики ранжирования, CatBoost

Для цитирования: Мохов А.И., Кислинский В.Г., Алексейчук А.С. Разработка рекомендательной системы на основе сессий с использованием многоуровневой системы отбора кандидатов // Труды МАИ. 2022. № 126. DOI: 10.34759/trd-2022-126-20

Original article

SESSION BASED RECOMMENDER SYSTEM WITH MULTISTAGE CANDIDATE SAMPLING

Alexey I. Mokhov[1], Vadim G. Kislinskiy[2], Andrey S. Alekseychuk[3]

[1,3]Moscow Aviation Institute (National Research University), MAI,
Moscow, Russia

[2]Moscow Institute of Physics and Technology (National Research University), MIPT,
Dolgoprudny, Moscow Region, Russia

[1]AIMokhov@mai.ru

[2]kislinskiy.vg@phystech.edu

[3]alexejchuk@gmail.com

Abstract. Understanding users' preferences is a challenging task especially with a huge amount of items. Modern recommender systems are keen to solve this task by applying state-of-the-art methods of candidate sampling and simple heuristics in couple with Machine Learning ranking algorithms. This paper presents an algorithm of candidate sampling from three different sources followed by a ranking algorithm. These two stages form a session-based recommender system that is capable of building a user's probable preferences based on its current session. For candidate sampling, we use a language model (Word2Vec) and sparse vectors for item representations, and the most popular items from a dataset. Each stage is divided into multiple substages making it really simple to add new candidate sources or remove existing ones. The same technique can be easily applied to ranking algorithms – one can remove a ranking algorithm or add the new one in order to blend model predictions maximizing Precision or Recall metrics as well. We also show the importance of ranking algorithms in recommender systems by measuring Learning to Rank (L2R) specific metrics on test data. There are several ranking algorithms in this paper. All of them belong to the pairwise algorithms subclass. Such algorithms as LambdaRank, YetiRank, and StochasticRank are used in comparison to non-ranked recommendations. We use CatBoost implementation of gradient boosting and PyTorch to build a neural ranking net. As a result of the experiment, we get a ready end-to-end recommender system pipeline with flexible modules that are easy to add/remove and show the benefits of ranking models with recommendations on real data.

Keywords*:* learning to rank, recommender system, machine learning, ranking metrics, CatBoost

For citation: Mokhov A.I., Kislinskiy V.G., Alekseychuk A.S. Session based recommender system with multistage candidate sampling. *Trudy MAI*, 2022, no. 126. DOI: 10.34759/trd-2022-126-20

Рекомендательные системы стали фундаментальным инструментом для принятия решений, помогая делать более эффективный и умный выбор практически во всех сферах жизни. Их роль стала особенно важна в эпоху цифровой экономики, когда пользователи делают выбор из огромного множества товаров и услуг. Современные рекомендательные системы, основанные на контенте и на коллаборативной фильтрации, возникли благодаря успешным исследованиям в области рекомендательных систем. Однако они заточены на использование исторических данных пользователя с целью выявления статических предпочтений. Такой сценарий использования несёт в себе предположение о равноважных взаимодействиях пользователя с товарами на протяжении всего времени. В действительности данное предположение может быть неверно в силу двух факторов. Во-первых, предпочтения пользователя зависят не только от его долгосрочных действий, но и от краткосрочных, которые составляют лишь малую часть от первых. Во-вторых, предпочтения пользователя относительно товаров склонны меняться со временем и являются скорее динамичными, нежели статичными.

В связи с данными недостатками пристальное внимание уделяется рекомендательным системам на основе сессионной информации. В отличие от подходов, которые были упомянуты выше, такие системы учитывают предпочтения на основе сессий, которые создаются в процессе потребительской активности.

Каждая сессия состоит из некоторых взаимодействий вида «пользователь-товар», которые происходят в течение некоторого времени, например, добавление товара в корзину, просмотр товара и т.д. Принимая сессию в качестве входных данных, сессионная РС способна извлекать информацию как о кратковременных предпочтениях пользователя, так и о изменениях предпочтений с течением времени, что помогает делать своевременный и точный прогноз (рекомендацию).

В рекомендательных системах важным этапом построения рекомендаций является еще и ранжирование. Задача ранжирования является ключевой частью многих информационных систем: поисковые системы, системы оперативного анализа информации [20], системы поддержки принятия решений [21, 22] и др. К настоящему моменту разработано множество подходов к ранжированию. Самый простой из них – поточечное ранжирование (pointwise ranking), где каждый документ в поисковой выдаче рассматривается как единая сущность, а ранжирование строится на основе модели регрессии, обученной на оценках релевантости. Более сложные подходы основываются на отношениях для нескольких объектов в рамках одной поисковой выдачи – попарное ранжирование (pairwise ranking), списочное ранжирование (listwise ranking) [11, 12]. Регрессионные модели в свое время показали неэффективность, поскольку оптимизируемый функционал значительно отличается от метрик качества в задачах ранжирования.

Самые первые работы по рекомендательным системам освещали коллаборативную фильтрацию как инструмент для подбора кандидатов. Среди множества методов коллаборативной фильтрации наиболее популярным является матричная факторизация [9]. Это такой алгоритм, который проецирует пользователей

и объекты в одно пространство и оценивает предпочтения пользователей для конкретного продукта как скалярное произведение двух векторов. Другие подходы оценивают схожесть между товарами по заранее вычисленной матрице и таким образом выдают похожие товары как рекомендацию [9, 10]. Также в последнее время есть множество работ, которые показывают, как использование нейросетевых методов помогает улучшать качество рекомендательных систем. Зачастую алгоритм для подбора кандидатов строят на методах, которые используются для обработки естественного языка. Это объясняется тем, что история действий пользователя на сайте или история его покупок представляется в виде последовательности, аналогичной последовательности слов в предложениях. Например, в [16] в качестве алгоритма для подбора кандидатов используют BERT – нейросетевую модель для естественного языка на основе механизма внимания (attention), которая произвела прорыв в сфере обработки естественного языка и до сих пор остается наиболее частым выбором для задач классификации, перевода и извлечения векторных представлений из текста [5].

В задачах ранжирования принято выделять два вида объектов: q (запрос) и d (документ), а также функцию $f(q, d)$, с помощью которой строится ранжирование документов. Ранее такие задачи решались с помощью моделей без обучаемых параметров, например, BM25 [15]. Здесь $f(q, d)$ рассматривается как условная вероятность $P(r|q, d)$, где q и d обозначают запрос и документ соответственно, $r = = 1$, если объект является релевантным для данного запроса, и $r = 0$ в противном случае. В модели Language Model for IR (LMIR) $f(q, d)$ представлена как $P(q|d)$ [4].

Такие модели не требуют обучения, поскольку значения вероятностей могут быть вычислены по запросу и документу явным образом.

Применение моделей машинного обучения для получения ранжирующих моделей открыло новую область для исследования. В первом десятилетии XXI века было предложено множество решений задач ранжирования с использованием машинного обучения, и большинство из моделей, предложенных тогда, используются и по сей день [1, 3]. Были предложены парные функционалы качества *LambdaRank* и *RankNet*, которые изначально предполагались для использования с нейронными сетями, а затем появились их модификации с использованием градиентного бустинга [2].

На данный момент все ещё остается проблема оптимизации негладких функционалов качества и применения списочных (listwise) критериев в задачах ранжирования. Так, в [19] говорится о новом методе оптимизации для метрик ранжирования с использованием стохастического сглаживания и оценкой градиентов при помощи частного интегрирования. Поднимается проблема типичных метрик ранжирования, которые определены для списка объектов, но при этом их нельзя оптимизировать напрямую, так как они являются кусочно-постоянными функциями, а поэтому не являются ни выпуклыми, ни гладкими, а множество современных попыток аппроксимировать эти метрики гладкими функционалами страдают от смещения и не являются выпуклыми, поэтому существующие алгоритмы могут гарантировать нахождение только локального минимума. При этом в статье нет проблемы воспроизводимости [17], поскольку предложенный метод доступен в библиотеке CatBoost [6]. В другой статье про обучение ранжированию также

предлагается метод для оптимизации списочных критериев качества, причем упоминается отсутствие корреляции между функционалом качества и целевой метрикой. Используя результат другой работы, авторы получают дифференцируемую аппроксимацию функции сортировки с помощью релаксации реальной функции качества [7, 14].

2. Математическая постановка задачи

Рассмотрим построение рекомендательной системы, предназначенной для выработки рекомендаций и ранжирования рекомендуемых товаров для пользователей интернет-магазина. Предполагается, что последовательность взаимодействий пользователя является неупорядоченной с неравномерными интервалами времени между взаимодействиями, то есть подходит под определение сессии в сессионных рекомендательных системах (СРС).

Исходными данными являются множество сессий S ($|S| = l$) и множество товаров I. Сессия S задаётся набором из трёх подмножеств $i_{view} \in I, i_{cart} \in I, i_{order} \in I$, каждое из которых указывает на тип взаимодействия во время сессии: «просмотр», «добавление в корзину», «заказ». Требуется для каждой сессии $s \in S$ построить список из N (задаваемый параметр) наиболее релевантных товаров $i \in I$ по входным данным i_{view}, i_{cart}. Каждый из построенных списков необходимо отсортировать в порядке убывания релевантности товаров. Релевантными в данном случае считаются товары, которые принадлежат i_{order} для конкретной сессии. Такие товары имеют значение релевантности 1 внутри сессии, остальные – 0.

Задачу построения рекомендаций можно разбить на два этапа: этап подбора кандидатов и этап ранжирования.

Этап подбора кандидатов заключается в выборе алгоритма подбора рекомендаций g, осуществляющего фильтрацию списка товаров в соответствии с предполагаемыми интересами пользователя:

$$g\left(\{i_{view}^k\}, \{i_{cart}^k\}\right) = \{\hat{\imath}_1^k, \dots, \hat{\imath}_N^k\} = R,$$

где $\{i_{view}^k\}$, $\{i_{cart}^k\}$ – списки просмотренных и добавленных в корзину товаров для k-ой сессии, а $\{\hat{\imath}_1^k, \dots, \hat{\imath}_N^k\}$ – подобранные рекомендации согласно заданным действиям.

На этапе ранжирования необходимо определить алгоритм ранжирования f, позволяющий упорядочить множество объектов в списке рекомендаций в порядке убывания их релевантности. Для этого для каждого объекта $\hat{\imath}_i$ в списке рекомендаций R вычисляется вектор вещественных значений X_i – вектор признаков, по которому в дальнейшем будет определяться положение элемента в упорядоченном списке R_{ranked}. Алгоритм f отображает каждый вектор признаков X_i в вещественное значение r_i:

$$f(X_0, \dots, X_N) = \{r_0, \dots, r_N\} = r.$$

Таким образом, список всех векторов признаков $X = (X_0, \dots, X_N)$ отображается в вектор вещественных значений r. Соответственно, каждому элементу $R_i \in R$ будет соответствовать значение r_i, которое указывает на его порядок в итоговом списке R_{ranked}.

Список R_{ranked} обладает свойством:

$$r_{R_i} \geq r_{R_j} : i, j = 1, \dots, N, i < j,$$

то есть в рамках конкретной сессии список R_{ranked} будет содержать элементы, упорядоченные по убыванию значения r_i.

Основная задача, рассматриваемая в настоящей статье – разработка сквозной (end-to-end) рекомендательной системы с ранжированием списка рекомендаций по релевантности с использованием функционала ранжирования на основе нейронных сетей.

3. Модели подбора товаров-кандидатов

На первом этапе для подбора товаров-кандидатов проведем сравнение трех моделей:

- векторное представление товаров на основе языковых моделей;
- векторное представление товаров с использованием разреженных векторов;
- выбор наиболее популярных товаров.

В качестве наиболее популярных товаров выбирается заданное количество наиболее заказываемых товаров за месяц за исключением промо-товаров (например, тех, которые кладутся в заказ в качестве подарка).

3.1. Векторное представление товаров на основе языковых моделей

Векторное представление товаров основано на вычислении векторных представлений (embeddings) – векторов заданной размерности, сопоставленных каждому товару. Подобный подход широко используется в моделях обработки текстов на естественном языке. Для каждого слова строится его векторное представление, отражающее отношения семантического сходства между ним и другими словами, входящими в словарь выбранного языка. Для построениия векторных представлений слов разработан ряд моделей обработки больших корпусов

текстов, в частности, модель Word2Vec [13]. Аналогичную модель можно использовать и для построения векторных представлений товаров. Введем метрику косинусной схожести d для пары векторов (x, y), где $x \in R^n, y \in R^n$:

$$d(x, y) = \frac{(x, y)}{|x||y|},$$

где (x, y) – скалярное произведение двух векторов x и y, а $|x|$ – норма вектора x. Эта метрика отражает отношение близости между векторами в R^n и, соответственно, наличие связи между соответствующими товарами с точки зрения покупателя. Обучая модель Word2Vec, используя данные сессий в качестве обучающей выборки, можно получить векторные представления, отражающие частоту встречаемости товаров в одной сессии. Таким образом, векторы пар товаров, встречающихся часто в одной сессии, будут иметь значение метрики косинусной схожести выше, чем для товаров, которые появлялись в одной сессии редко или вовсе не появлялись. Для подбора кандидатов выбираются товары с наибольшим значением этой метрики.

3.2. Векторное представление товаров с использованием разреженных векторов

Модель представления товаров с использованием разреженных (sparse) векторов основана на построении матрицы, отражающей наличие товаров в рамках каждой сессии. В качестве примера рассмотрим следующий список сессий и участвующих в них товаров (рис. 1).

	session
0	[0, 1, 2, 5]
1	[1, 3, 4, 5]
2	[3, 4, 1, 2]

Рисунок 1. Список сессий

Составим такую матрицу A, где строки будут отвечать за сессии, а столбцы – за товары. Тогда элемент матрицы с индексами i, j будет равен 1, если товар с индексом j встречался в сессии i, и 0 в противном случае (рис. 2).

	0	1	2	3	4	5
session_0	1	1	1	0	0	1
session_1	0	1	0	1	1	1
session_2	0	1	1	1	1	0

Рисунок 2. Разреженная матрица

В реальных приложениях матрица A является разреженной, поскольку в рамках каждой сессии участвует лишь малая доля всего ассортимента товаров, и большинство элементов матрицы равно нулю. Используя матрицу A, можно посчитать косинусную схожесть между любой парой товаров:

$$sim(a, b) = \frac{(a, b)}{|a||b|},$$

где a, b – произвольные столбцы A. Применив данную формулу ко всем возможным парам a, b из A, получаем матрицу A' размерности $N \times N$, где N – число уникальных товаров, а элементами этой матрицы являются попарные схожести между товарами.

В дальнейшем для подбора кандидатов используется только матрица A' путем нахождения наиболее близких товаров по метрике косинусной схожести.

Все три рассмотренные модели подбора товаров-кандидатов можно объединить в следующий алгоритм. Параметрами алгоритма являются значения m – число рекомендаций на товар и K – общее число рекомендаций на сессию.

1. На вход подается сессия $a = \{id_0, \ldots, id_n\}$, где каждый элемент id_i представляет собой идентификатор товара из данной сессии.

2. Для каждого товара id_i находится m наиболее близких товаров в соответствии с выбранной моделью. Так как используется три модели, то для каждого товара выходит $3m$ схожих товаров. В итоге для всей сессии находится $3mn$ товаров.

3. Каждая такая рекомендации сортируется по убыванию значения схожести r_i. Из сортированного списка выбирается максимум K товаров.

4. Методы ранжирования рекомендаций

Для получения окончательной рекомендации нужно, получив список товаров-кандидатов, найти среди них наиболее релевантные, а затем ранжировать список по значению релевантности. В таком случае лучше использовать либо парные (pairwise), либо списочные (listwise) подходы к ранжированию, так как при обучении они способны напрямую учитывать релевантность объектов и их расположение относительно друг друга. В данной статье рассматриваются только парные критерии ранжирования, поскольку они требуют значительно меньше ресурсов, но при этом их качество зачастую не отличается со списочных критериев. Применение ранжирования позволяет значительно повысить метрики ранжирования и тем самым улучшить качество рекомендаций.

Рассмотрим 4 метода возможной поисковой выдачи:

- без ранжирования (просто порекомендуем популярные товары);
- LambdaRank;
- YetiRank;
- StochasticRank.

4.1. Модели ранжирования

Модель LambdaRank использует идею множителя λ, который в оригинальной статье [3] был получен как изменение (Δ) значения метрики *NDCG* при перестановке двух элементов ранжированного списка. Исследователи выяснили, что при домножении градиентов на значение $\lambda = \Delta NDCG$ можно значительно быстрее оптимизировать желаемую метрику, которой в случае задач информационного поиска, зачастую является *NDCG*. В качестве реализации используется собственный код, написанный с использованием библиотеки машинного обучения pyTorch.

Авторы модели YetiRank продолжают результаты исследования [3] и используют также парный подход к ранжированию, определяя оптимизируемую функцию как

$$L = -\sum_{(i,j)} w_{ij} log \frac{e^{x_i}}{e^{x_i} + e^{x_j}},$$

где индексы (i, j) – это все возможные индексы пар документов для конкретного запроса, w_{ij} – вес конкретной пары из запроса, x_i, x_j – предсказанные значения функции ранжирования для документов i и j соответственно. Дальнейшие

рассуждения приведены в оригинале статьи [8]. Данный метод реализован в проекте с открытым исходным кодом CatBoost [6].

Авторы метода StochasticRank [18] предлагают новый метод для оптимизации ранжирующего функционала, при этом адаптируя результаты своих предыдущих работ. Например, в качестве модели оптимизации используется SGLB (Stochastic Gradient Langevin Boosting), который гарантирует нахождение глобального минимума для мультимодальных функций, а также превосходит классический градиентный бустинг по результатам экспериментов. В своей работе авторы предлагают метод Coordinate Conditional Sampling, который позволяет получать оценки градиентов для любых ранжирующих функционалов с низкой дисперсией, а также метод для ускорения сходимости алгоритма (Scale-Free Acceleration) [19]. Метод также реализован в проекте CatBoost [6].

4.2. Метрики ранжирования

Для определения метрик ранжирования введем следующие обозначения. Пусть $Q = \{q_1, \dots, q_n\}$ – множество запросов на составление рекомендаций, отправленных информационной системе, $D_q = \left\{d_{q1}, \dots, d_{qm_q}\right\}$ – документы, которые получены для запроса q, а $L_q = \left\{l_{q1}, \dots, l_{qm_q}\right\}$ – истинные значения релевантности для документов из D_q. Необходимо получить такую функцию $f = f\left(d_{qi}\right) = x_{qi}$, которая выдает значение релевантности x_{qi} для документа d_{qi}. Затем полученные значения x_{qi} используются для ранжирования документов D_q путем их сортировки по убыванию значения x_{qi}.

Для оценки качества ранжирования будем использовать метрики *NDCG*, *MAP* и *MRR*. Это классические метрики качества в задачах ранжирования, их особенность заключается во взвешивании каждого из объектов в списке рекомендаций (или поисковой выдаче) таким образом, чтобы больший вес имели объекты, которые находятся в начале списка. Все они принимают значения от 0 до 1 и показывают отклонение действительного ранжирования от идеального ранжирования, при котором все релевантные объекты находятся выше в выдаче, чем нерелевантные. *Normalized Discounted Cumulative Gain* (*NDCG*) определяется для запроса q и набора документов d_q, полученных для запроса q, как

$$DCG(q) = \sum_{k=1}^{m_q} \frac{l_{qk}}{(k+1)},$$

где l_{qk} – это известная релевантность для документа под индексом k в выдаче для поискового запроса q. Порядок определяется сортировкой по значению релевантности в порядке убывания $x_i = f(d_{qi})$ для документа под индексом i из выдачи D_q. В таком случае значение *NDCG* определяется как отношение *DCG* полученной выдачи к значению *DCG* для идеальной выдачи (если бы сортировка была по действительным значениям релевантности, а не по полученным).

Используя такие же обозначения, введем метрику *Average Precision*, которая определяется через классическую метрику задач классификации – *Precision*:

$$Precision(d_i, k) = \frac{\sum_{k=1}^{m_q} I(d_{qi}, k)}{|d_i|},$$

где $I(d_{qi}, k)$ показывает, является ли данный документ релевантным для данного запроса, если рассматривать первые k элементов выдачи. *Average Precision* регулируется целочисленным параметром K и определяется так:

$$AP@K = \frac{\sum_{k=1}^{K} \quad Precision(d_i, k)}{K}.$$

Метрика *Mean Average Precision at K* ($MAP@K$) определяется как среднее всех значений $AP@K$ для всего множества запросов.

Ещё одной метрикой для задач ранжирования является *MRR* (*Mean Reciprocal Rank*). Эта метрика проста и в реализации, и для интуитивного понимания. Она показывает средний ранг первого релевантного документа в выдаче. Рангом в данном случае называется порядок элемента $i \in \{1, \ldots, n\}$, где n – размер ответа на поисковый запрос:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \quad \frac{1}{rank_i},$$

где $rank_i$ – ранг первого релевантного документа в выдаче, а Q – всё множество запросов.

5. Экспериментальное сравнение методов выдачи рекомендаций

Для сравнительного анализа приведенных методов выдачи рекомендаций проведен ряд экспериментов с использованием данных о сессиях одного из действующих интернет-магазинов. Каждая сессия представляет собой список просмотренных, добавленных в корзину и заказанных товаров. Время с момента

начала сессии до ее завершения составляет не более 24 часов. Таким образом, мы предполагаем, что у пользователя в рамках одной сессии сохраняются предпочтения и он смотрит схожие товары. В полной выборке содержится 1 млн уникальных товаров, а также 6 млн сессий.

Для получения репрезентативных оценок метрик эксперимент проводится несколько раз с усреднением полученных результатов. Из всей выборки каждый раз выбирается срез из 50 тысяч случайных сессий, которые разбиваются на обучающую и тестовую выборки в соотношении 4:1 (40000 на обучение и 10000 на тест).

Проведение эксперимента состоит из 5 этапов:

1. Обработка исходных данных.
2. Получение моделей для векторных представлений.
3. Подбор кандидатов и генерация признаков.
4. Обучение моделей ранжирования.
5. Оценка моделей ранжирования.

Эти пункты применяются при каждом запуске эксперимента, после чего проводится оценка полученных моделей на тестовой выборке. В результате для каждой конфигурации эксперимента получается 12 итоговых значений (по 3 метрики на каждую из четырех моделей ранжирования).

В результате экспериментирования и подсчета метрик получены следующие результаты (табл. 1).

Таблица 1

Сравнительный анализ метрик ранжирования

Метод	Метрика	Значение метрики
YetiRank	NDCG@10	0.974323
StochasticRank	**NDCG@10**	**0.975185**
LambdaRank	NDCG@10	0.967434
No Rank	NDCG@10	0.09074
YetiRank	MAP@10	0.106497
StochasticRank	**MAP@10**	**0.108269**
LambdaRank	MAP@10	0.083618
No Rank	MAP@10	0.001056
YetiRank	MRR	0.134089
StochasticRank	**MRR**	**0.136322**
LambdaRank	MRR	0.104541
No Rank	MRR	0.001326

При рассмотрении примеров рекомендаций можно увидеть, что при использовании моделей ранжирования (рис. 4) пользователю рекомендуются товары, которые часто покупают вместе и они могут быть полезны пользователю, а, следовательно, выше вероятность их покупки. Если ранжирование не используется (рис. 5), то в рекомендации попадают товары, очевидно не релевантные текущей сессии.

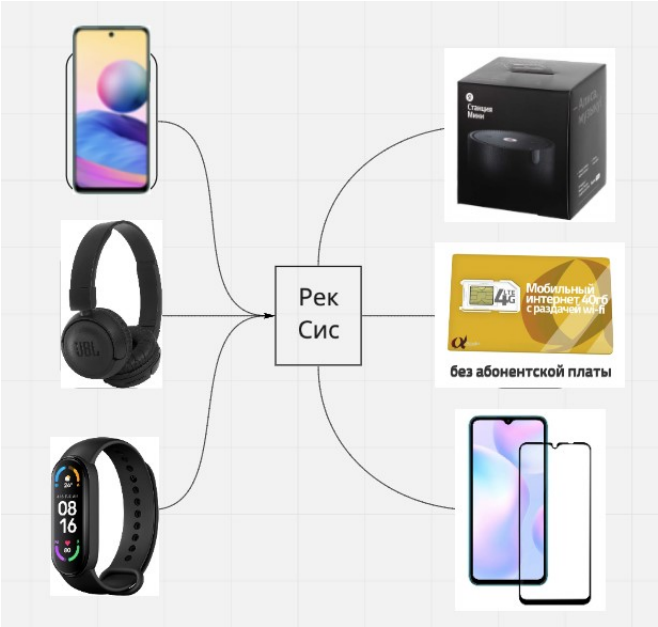

Рисунок 4. Рекомендация с ранжированием

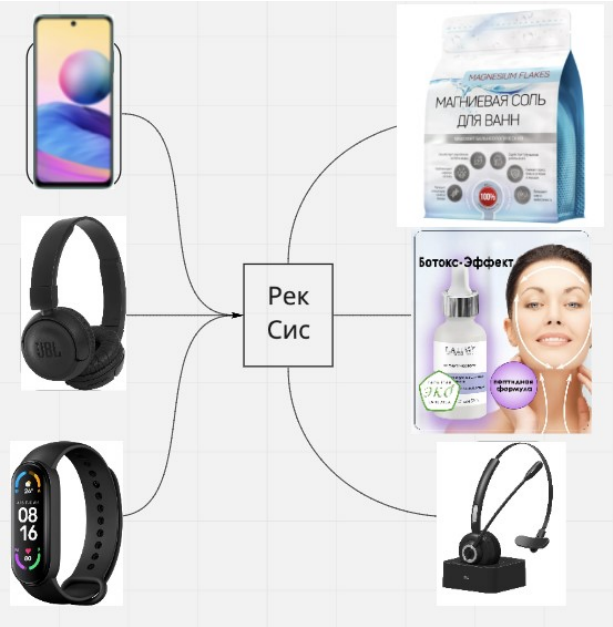

Рисунок 5. Рекомендация без ранжирования

Исходный код программ, использованных при проведении эксперимента, опубликован по ссылке https://github.com/HuviX/my_recom.

6. Заключение

По результатам эксперимента можно сделать вывод, что разработанная рекомендательная система способна формировать довольно качественные рекомендации на основе пользовательских сессий. Использование ранжирования значительно помогает повысить целевые метрики по сравнению с моделью без ранжирования, следовательно, ранжирование позволяет показывать более релевантные объекты на первых позициях списка рекомендуемых товаров. При этом значения метрик качества с использованием ранжирующих моделей получаются довольно близкими, что свидетельствует о сравнимом качестве всех трех моделей ранжирования.

Разработанное ПО может быть использовано для исследования качества работы рекомендательных систем, систем подбора товаров-кандидатов или алгоритмов ранжирования, а также в качестве готового сквозного решения, способного обеспечивать высокое качество рекомендаций при обучении на сессионных данных.

Список источников

1. Burges C., Shaked T., Renshaw E., Lazier A., Deeds M., Hamilton N., Hullender G. Learning to rank using gradient descent // 22nd International Conference on Machine learning, 2005, pp. 89-96. DOI: 10.1145/1102351.1102363

2. Burges C.J. From ranknet to lambdarank to lambdamart: An overview // Learning, 2010, vol. 11, no. 81. URL: https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/

3. Burges C., Ragno R., Le Q. Learning to rank with nonsmooth cost function // Advances in neural information processing systems, 2006, vol. 19, pp. 193-200. DOI: 10.5555/2976456.2976481

4. Croft W.B., Metzler D., Strohman T. Search engines: Information retrieval in practice, Reading: Addison-Wesley, 2010, vol. 520, pp. 131-141. URL: https://www.isi.edu/people/metzler/publications/search_engines_information_retrieval_practice

5. Devlin J., Chang M.W., Lee K., Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding // Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018, vol. 1, pp. 4171-4186. DOI: 10.18653/v1/N19-1423

6. Prokhorenkova L., Gusev G., Vorobev A., Dorogush A., Gulin A. CatBoost: unbiased boosting with categorical features // Advances in Neural Information Processing Systems 31 (NeurIPS 2018), 2008, pp. 6638-6648. URL: https://arxiv.org/pdf/1706.09516.pdf

7. Grover, A., Wang, E., Zweig, A., and Ermon, S. Stochastic optimization of sorting networks via continuous relaxations // Proceedings of the International Conference on Learning Representations, 2019. URL: https://arxiv.org/pdf/1903.08850.pdf

8. Gulin A., Kuralenok I., Pavlov D. Winning the transfer learning track of Yahoo!'s learning to rank challenge with yetirank // Learning to Rank Challenge, 2011, pp. 63-76. DOI: 10.5555/3045754.3045761

9. Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model // 14th ACM SIGKDD international conference on Knowledge discovery and data mining, 2008, pp. 426-434. DOI: 10.1145/1401890.1401944

10. Koren Y., Bell R., Volinsky C. Matrix factorization techniques for recommender systems // Computer, 2009, vol. 42 (8), pp. 30-37. DOI: 10.1109/MC.2009.263

11. Li H. A short introduction to learning to rank // IEICE Transactions on Information and Systems, 2011, vol. 94 (10), pp. 1854-1862. DOI: 10.1587/transinf.E94.D.1854

12. Liu T.Y. Learning to rank for information retrieval // Foundations and Trends® in Information Retrieval, 2009, vol. 3, no. 3, pp 225-331. DOI: 10.1561/1500000016

13. Mikolov T., Chen K., Corrado G., Dean J. Efficient estimation of word representations in vector space // In 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013.

14. Felix Petersen, Christian Borgelt, Hilde Kuehne, Oliver Deussen. Learning with Algorithmic Supervision via Continuous Relaxations // Proceedings of the 35th Conference

on Neural Information Processing Systems (NeurIPS 2021). URL: esearch.ibm.com/publications/learning-with-algorithmic-supervision-via-continuous-relaxations

15. Robertson S., Zaragoza H. The probabilistic relevance framework: BM25 and beyond, Now Publishers, Inc. 2009. DOI: 10.1561/1500000019

16. Sun F., Liu J., Wu J., Pei C., Lin X., Ou W., Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer // 28th ACM international conference on information and knowledge management, 2019, pp. 1441-1450. DOI: 10.1145/3357384.3357895

17. Sun Z., Yu D., Fang H., Yang J., Qu X., Zhang J., Geng C. Are we evaluating rigorously? Benchmarking recommendation for reproducible evaluation and fair comparison // In Fourteenth ACM Conference on Recommender Systems, 2020, pp. 23-32, DOI: 10.1145/3383313.3412489

18. Ustimenko A., Prokhorenkova, L. StochasticRank: Global Optimization of Scale-Free Discrete Functions // International Conference on Machine Learning, 2020, pp. 9669-9679, URL: https://proceedings.mlr.press/v119/ustimenko20a.html

19. Ustimenko A., Prokhorenkova L. SGLB: Stochastic Gradient Langevin Boosting // International Conference on Machine Learning, 2021, pp. 10487-10496. URL: https://deepai.org/publication/sglb-stochastic-gradient-langevin-boosting

20. Проценко П.А., Хуббиев Р.В. Методика ранжирования космических аппаратов дистанционного зондирования Земли с целью оперативного мониторинга чрезвычайных ситуаций // Труды МАИ. 2021. № 119. URL: http://trudymai.ru/published.php?ID=159756. DOI: 10.34759/trd-2021-119-18

21. Смерчинская С.О., Яшина Н.П. Агрегирование предпочтений с учетом важности критериев // Труды МАИ. 2015. № 84. URL: http://trudymai.ru/published.php?ID=62973

22. Заковряшин А.И. Особенности интеллектуальной поддержки принятия решений // Труды МАИ. 2012. № 61. URL: http://trudymai.ru/published.php?ID=35494

References

1. Burges C., Shaked T., Renshaw E., Lazier A., Deeds M., Hamilton N., Hullender G. Learning to rank using gradient descent, *22nd International Conference on Machine learning*, 2005, pp. 89-96. DOI: 10.1145/1102351.1102363

2. Burges C.J. From ranknet to lambdarank to lambdamart: An overview, *Learning,* 2010, vol. 11, no. 81. URL: https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/

3. Burges C., Ragno R., Le Q. Learning to rank with nonsmooth cost function, *Advances in neural information processing systems,* 2006, vol. 19, pp. 193-200. DOI: 10.5555/2976456.2976481

4. Croft W.B., Metzler D., Strohman T. *Search engines: Information retrieval in practice,* Reading: Addison-Wesley, 2010, vol. 520, pp. 131-141. URL: https://www.isi.edu/people/metzler/publications/search_engines_information_retrieval_practice

5. Devlin J., Chang M.W., Lee K., Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding, *Proceedings of the 2019 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018, vol. 1, pp. 4171-4186. DOI: 10.18653/v1/N19-1423

6. Prokhorenkova L., Gusev G., Vorobev A., Dorogush A., Gulin A. CatBoost: unbiased boosting with categorical features, *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, 2008, pp. 6638-6648. URL: https://arxiv.org/pdf/1706.09516.pdf

7. Grover, A., Wang, E., Zweig, A., and Ermon, S. Stochastic optimization of sorting networks via continuous relaxations, *Proceedings of the International Conference on Learning Representations*, 2019. URL: https://arxiv.org/pdf/1903.08850.pdf

8. Gulin A., Kuralenok I., Pavlov D. Winning the transfer learning track of Yahoo!'s learning to rank challenge with yetirank, *Learning to Rank Challenge*, 2011, pp. 63-76. DOI: 10.5555/3045754.3045761

9. Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model, *14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 426-434. DOI: 10.1145/1401890.1401944

10. Koren Y., Bell R., Volinsky C. Matrix factorization techniques for recommender systems, *Computer*, 2009, vol. 42 (8), pp. 30-37. DOI: 10.1109/MC.2009.263

11. Li H. A short introduction to learning to rank, *IEICE Transactions on Information and Systems*, 2011, vol. 94 (10), pp. 1854-1862. DOI: 10.1587/transinf.E94.D.1854

12. Liu T.Y. Learning to rank for information retrieval, *Foundations and Trends® in Information Retrieval*, 2009, vol. 3, no. 3, pp 225-331. DOI: 10.1561/1500000016

13. Mikolov T., Chen K., Corrado G., Dean J. Efficient estimation of word representations in vector space, *In 1st International Conference on Learning Representations, ICLR 2013*, Scottsdale, Arizona, USA, May 2-4, 2013.

14. Felix Petersen, Christian Borgelt, Hilde Kuehne, Oliver Deussen. Learning with lgorithmic Supervision via Continuous Relaxations, *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*. URL: esearch.ibm.com/publications/learning-with-algorithmic-supervision-via-continuous-relaxations

15. Robertson S., Zaragoza H. *The probabilistic relevance framework: BM25 and beyond,* Now Publishers, Inc. 2009. DOI: 10.1561/1500000019

16. Sun F., Liu J., Wu J., Pei C., Lin X., Ou W., Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer, *28th ACM international conference on information and knowledge management*, 2019, pp. 1441-1450. DOI: 10.1145/3357384.3357895

17. Sun Z., Yu D., Fang H., Yang J., Qu X., Zhang J., Geng C. Are we evaluating rigorously? Benchmarking recommendation for reproducible evaluation and fair comparison, *In Fourteenth ACM Conference on Recommender Systems*, 2020, pp. 23-32, DOI: 10.1145/3383313.3412489

18. Ustimenko A., Prokhorenkova, L. StochasticRank: Global Optimization of Scale-Free Discrete Functions, *International Conference on Machine Learning*, 2020, pp. 9669-9679, URL: https://proceedings.mlr.press/v119/ustimenko20a.html

19. Ustimenko A., Prokhorenkova L. SGLB: Stochastic Gradient Langevin Boosting, *International Conference on Machine Learning,* 2021, pp. 10487-10496. URL: https://deepai.org/publication/sglb-stochastic-gradient-langevin-boosting

20. Protsenko P.A., Khubbiev R.V. *Trudy MAI,* 2021, no. 119. URL: http://trudymai.ru/eng/published.php?ID=159756. DOI: 10.34759/trd-2021-119-18

21. Smerchinskaya S.O., Yashina N.P. *Trudy MAI,* 2015, no. 84. URL: http://trudymai.ru/eng/published.php?ID=62973

22. Zakovryashin A.I. *Trudy MAI*, 2012, no. 61. URL: http://trudymai.ru/eng/published.php?ID=35494

Статья поступила в редакцию15.03.2022

Статья после доработки 20.03.2022

Одобрена после рецензирования 28.08.2022

Принята к публикации 12.10.2022

The article was submitted on 15.03.2022; approved after reviewing on 28.08.2022; accepted for publication on 12.10.2022