ПРОГРАММИРОВАНИЕ /
PROGRAMMING AND COMPUTER SOFTWARE, 2025, № 6, с. 17–35

ИНФОРМАЦИОННЫЙ ПОИСК

УДК 004.822
DOI: 10.7868/S3034584725060021 Оригинальная статья / Original Article

ГИБРИДНЫЙ ПОДХОД К ФОРМИРОВАНИЮ РЕКОМЕНДАЦИЙ НА ОСНОВЕ ГРАФА ЗНАНИЙ И КОНТЕКСТНОГО ЭНКОДЕРА

© 2025 г. Т. А. Дмитриева[a], И. Ю. Каширин[a], Г. В. Овечкин[a, *], Д. И. Успенский[a]

[a] Рязанский государственный радиотехнический университет имени В.Ф. Уткина
390005 Рязань, ул. Гагарина, д. 59/1, Россия
** e-mail: ovechkin.g.v@rsreu.ru*

Аннотация. Решается задача построения цифровой рекомендательной системы на основе нейросетевой модели вывода информации из графа знаний. Предложен алгоритм выделения сущностей из неструктурированных текстовых атрибутов объектов предметной области для обогащения графа знаний. Рассмотрен процесс конструирования векторизованного представления моделируемых объектов. Созданная искусственная нейронная сеть выполняет генерацию индивидуальных рекомендаций из заданного множества товаров, основываясь на историческом поведении пользователя и семантических связях смоделированных сущностей, входящих в граф. Работа базируется на существующих исследованиях в области комбинирования графов знаний и глубокого обучения. Новизна идеи состоит в использовании текстового энкодера с архитектурой типа трансформер для извлечения дополнительных сущностей из текстового описания объектов предметной области и встраивания их в семантическую структуру графа. Проведено сравнение предлагаемого подхода и некоторых существующих решений. Результаты исследования могут быть использованы для создания рекомендательных сервисов товаров и услуг.

Ключевые слова: графы знаний, дедукция на основе путей, BERT, нейронные сети, системы рекомендаций

Финансирование. Исследование проведено без дополнительного финансирования.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Вклад авторов. Авторы данной работы внесли равнозначный вклад в разработку концепции, проведение исследования и подготовку статьи.

Ссылка для цитирования: *Т. А. Дмитриева, И. Ю. Каширин, Г. В. Овечкин, Д. И. Успенский,* Гибридный подход к формированию рекомендаций на основе графа знаний и контекстного энкодера // Программирование / Programming and Computer Software. 2025. № 6. С. 17–35.

Благодарности. Авторы благодарят компанию ООО “Альткрафт” за предоставленное для экспериментов оборудование и инженерную поддержку.

ПРОГРАММИРОВАНИЕ /
PROGRAMMING AND COMPUTER SOFTWARE, 2025, № 6, с. 17–35

DOI: 10.7868/S3034584725060021 Оригинальная статья / Original Article

RECOMMENDATION SYSTEM BASED ON JOINT APPLICATION OF TEXT BIDIRECTIONAL ENCODER AND KNOWLEDGE GRAPH

T. A. Dmitrieva, I. Y. Kashirin, G. V. Ovechkin, D. I. Uspenskiy

Ryazan State Radio Engineering University named after V.F. Utkin,
59/1, Gagarina street, Ryazan, 390005, Russian Federation

Abstract. In this paper we develop recommendation system based on reasoning neural network and knowledge graph. Some aspects of domain knowledge graph completion and entity and link embeddings creation are considered. Composed graph is used for personal recommendations reasoning. Neural network generates personal recommendations depending on historic user activity patterns and semantic connections between the simulated entities in the graph. Our work is based on some existing researches. The scientific novelty lies in implementation of a transformer based language model to extract additional entities from the text description of domain objects and embed them in the graph structure. Quality recommendation metrics are statistically boosted in comparison with the some existed approaches in the field of path-based reasoning on knowledge graphs. The research may be used to ameliorate recommendation services in online retail.

Keywords: knowledge graphs, path-based reasoning, BERT, neural networks, recommendation systems

Funding. The work was conducted without additional funding.

Conflict of interest. The authors of this work declare that they have no conflicts of interest.

Author contributions. The authors contributed equally to the work.

For Citation: *T. A. Dmitrieva, I. Y. Kashirin, G. V. Ovechkin, D. I. Uspenskiy*, Recommendation System Based on Joint Application of Text Bidirectional Encoder and Knowledge Graph // Programmirovanie / Programming and Computer Software. 2025. № 6. P. 17–35.

Acknowledgments. The authors thank LLC “Altkraft” for providing the equipment and engineering support for the experiments.

1. ВВЕДЕНИЕ

Глобальная цифровизация требует внедрения интеллектуальных систем принятия решений, основанных на моделях машинного и глубокого обучения, в бизнес-процессы предприятий. Все более возрастающую часть оборотов компаний составляют продажи продуктов, предложенных специализированными рекомендательными сервисами. Целью существования подобных систем является прогнозирование предпочтений пользователей в отношении определенного товара (или классов товаров) в целях формирования индивидуального набора предлагаемых к покупке продуктов и максимизации тем самым доходов от продаж.

Классический подход в создании подобных решений основан на алгоритмах коллаборативной фильтрации [1], матричных разложений [2], анализа портрета потенциального покупателя и/или предлагаемого контента [3], а также применения особых архитектур искусственных нейронных сетей [4]. Одной из основных проблем указанных методов является высокая разреженность данных в матрице “пользователь–объект”, в которой покупатели проставляют оценки купленным вещам. Как правило, количество предлагаемых продуктов огромно, в то время как большинство покупателей редко оставляет оценки. В результате матрица становится сильно разреженной, что ведет к проблемам вычислительного характера.

Существуют и другие сложности, например синонимия, заключающаяся в том, что большинство рекомендательных систем не способны обнаружить скрытые связи между объектами и поэтому относятся к этим предметам как к разным. Например, “фильмы для детей” и “детский фильм” относятся к одному жанру, но система воспринимает их как разные [5].

К недостаткам подобных моделей следует также отнести излишнюю концентрированность на исторических и статистических взаимосвязях между рекомендуемыми объектами. В то время как семантика само́й предметной области, равно как свойств и характеристик предлагаемых товаров, выходит из фокуса внимания рекомендатора. Алгоритмы на основе коллаборативной фильтрации или схожести контента подходят для продуктов, которые покупаются достаточно часто. Однако для товаров, продающихся довольно редко, результаты выглядят посредственно. В данном случае требуется анализ более глубоких взаимосвязей между предлагаемыми продуктами и потребительскими интересами потенциальных покупателей. Так возникли рекомендательные системы, основанные на извлечении знаний из самой предметной области задачи. Их отличительной чертой является толерантность к отсутствию явно проставленных товарных рейтингов или пользовательских оценок.

Создание модели продуктовых знаний является важнейшей задачей построения подобных систем. Нужная информация должна быть представлена в некоторой структурированной форме, воспринимаемой машиной. Например, онтологии [6] вполне могут быть использованы для вывода обоснования того, какие продукты отвечают текущим потребностям пользователя. Рекомендации в этом случае могут быть получены путем семантического сопоставления предпочтений конкретного покупателя и онтологического описания объектов в предлагаемом наборе продуктов [7]. Учесть степень влияния признаков можно с помощью их предварительной семантической структуризации [6].

Недавние исследования в области создания графов знаний и использования графовых нейронных сетей [8, 9] позволяют эффективно использовать эти инструменты для кодирования информации о предпочтениях пользователей, свойствах и характеристиках товаров. Структура знаний продуктовой области представляется в виде схемы ребер и вершин графа. Современные алгоритмы рекомендаций в своих решениях должны учитывать большое количество самых разнообразных данных, включая текстовые описания продуктов и отзывы покупателей. Информация, заключенная в этих документах, например, тональность комментария или извлеченный набор характеристик товара, может быть соответствующим образом внедрена в структуру графа знаний, дополняя и расширяя его. При этом точность решений, принимаемых алгоритмом, повышается.

Подходы в области прикладного рекомендательного машинного обучения эволюционировали от сравнительно простых моделей, таких как деревья решений и марковские цепи, к более сложным архитектурам – графовым нейронным сетям, сверточным структурам и, наконец, к трансформерам и крупномасштабным языковым моделям. Пристальное внимание

в последние годы уделяется графовым нейросетевым архитектурам, поскольку именно они способны моделировать сложные взаимосвязи между пользователями и товарами, включая многозначные отношения, контекстные зависимости и семантические связи. Однако, несмотря на значительные успехи, такие модели сталкиваются с рядом ограничений: высокая сложность архитектуры, трудная интерпретируемость, слабая обобщаемость на холодные объекты, а также ограниченность в понимании семантических закономерностей при наличии у сущностей неструктурированных данных, например, текстовых атрибутов.

В настоящей работе предлагается метод нейросетевого вывода рекомендаций по графу знаний, сочетающий несколько ключевых характеристик: возможность последовательной интерпретации решающих факторов за счет использования путевого (path-based) обхода графа, а также способность моделировать дополнительные семантические связи между некоторыми объектами предметной области. В своем решении мы основываемся на идеях: CAFE [10] и HeteroEmbed [11], которые реализуют многошаговое логическое рассуждение, используя набор полносвязных нейросетевых модулей для каждого типа отношений. Наш вклад заключается в следующем: во-первых, мы улучшили базовые подходы за счет ремоделирования и обогащения графа знаний новыми сущностями, извлеченными предобученным текстовым энкодером; во-вторых, мы формализовали и упростили исходный алгоритм построения векторных представлений и вывода рекомендаций; в-третьих, мы провели серию экспериментов на датасете Amazon Beauty (5-core) и продемонстрировали статистически подтвержденное улучшение качества рекомендаций.

2. ОБЗОР ПОХОЖИХ РАБОТ

Ключевой задачей при построении систем рекомендаций является извлечение качественных представлений (эмбеддингов) пользователей и товаров. Графы знаний предлагают эффективный подход к кодированию информационных взаимосвязей между разнотипными объектами предметной области. Рекомендательные системы, построенные на основе графов, можно разделить на три основные категории. Методы первой группы стремятся напрямую получить фиксированные векторы для пользователей, товаров и/или иных сущностей. Они не рассматривают структуру графа явно, а скорее инкапсулируют всю информацию от непосредственных соседей анализируемого объекта. Примером подобного решения может быть работа [12]. Второй класс методов использует идею распространения информации (message passing) в графе знаний по аналогии со сверточными сетями [13–17]. Представление узла обновляется с помощью агрегации информации от соседей различной степени близости, используя те или иные механизмы взвешивания. Третья категория (path-based-алгоритмы) делает акцент на извлечение данных во время обхода графовых путей, соединяющих пользователя и товар через другие сущности и отношения [10, 11, 18–23].

В настоящее время основное внимание исследователей сосредоточено на подходах, относящихся к последним двум категориям. Так, графовые нейронные сети (GNN) относятся ко второй группе и получили широкое признание в задачах вывода персональных предпочтений благодаря своей способности к агрегации информации из множества соседних вершин (включая непрямых соседей) объекта. Исследователи используют общий принцип первоначального построения эмбеддингов сущностей с захватом максимума полезной информации из объектных атрибутов и наложение коллаборативной составляющей в виде агрегации через GNN (либо ее модификации). KGTORe (Knowledge-Graph- and Tree-Oriented Recommendation) [14] использует деревья решений для персонализированного моделирования пользовательских предпочтений и графовую нейросеть для агрегации контекста и коллаборативного сигнала. Деревья решений в KGTORe выполняют интерпретируемый отбор признаков, определяя, какие семантические характеристики товаров важны для конкретного пользователя в конкретном взаимодействии. GNN обучает эмбеддинги этих признаков, а затем агрегирует их для построения векторных представлений узлов графа, оптимизируя точность рекомендаций. Подход обладает хорошей интерпретируемостью и персонализацией. Деревья решений позволяют объяснять, почему пользователь предпочел или отверг товар. С другой стороны, модель работает только с признаками первого порядка (предикат, объект). Это упрощает представление, но теряет семантику

многозвенных путей в графе. К недостаткам деревьев решений также можно отнести эффект переобучения на малом числе взаимодействий, что особенно актуально для новых покупателей с небогатой поведенческой историей. Модель GTR (Graph Transformer for Recommendation) [13] – это попытка объединить графовую сеть, учитывающую структуру взаимодействий, с гибким механизмом внимания для дифференцированной агрегации соседей. GTR предлагает архитектуру, которая не ограничивается локальной аггрегацией смежных узлов, как обычные GNN, а использует глобальное внимание по всему графу взаимодействий. Для выделения различий по типу ребер и вершин используются индивидуальные проекции и маски. Исследователи из [17] предлагают контрастивную рекомендательную модель, применяющую аугментации для создания модифицированных версий исходного графа. Шумовой сигнал генерируется вариационным автоэнкодером и способствует выводу более обобщенных и устойчивых представлений товаров и пользователей в процессе обучения. Обе архитектуры зарекомендовали себя на таких проверочных наборах данных как Yelp2018, LastFM и обладают хорошей обобщающей способностью. Однако способность классических GNN извлекать информативные представления может быть ограничена в реальных сценариях, где взаимодействия между пользователями и товарами крайне разрежены. В работе [15] подчеркивается важность равномерного распределения векторных представлений. Такое распределение снижает популяризационный сдвиг (смещение предпочтения модели к рекомендации популярных товаров) и способствует продвижению так называемого long-tail-контента (объектов с малым числом взаимодействий). Их работа основывается на идее аугментации исходного пользовательского продуктового графа путем структурных искажений, например, удаление ребер или узлов с определенной вероятностью. После чего нейронная сеть, применяя функцию контрастивных потерь, дополнительно обучается сближать модифицированные вектора, относящиеся к одинаковым объектам. Описанный подход продемонстрировал эффективность и получил множество дальнейших усовершенствований. Так, в [16] было предложено искажать исходные векторы путем добавления небольшого случайного шума, сгенерированного из равномерного распределения и нормированного на ε – масштаб шума. При этом искажения добавляются на каждом слое агрегации GNN, прямо внутри самой модели. Авторы [17] используют для модификации эмбеддингов две обучаемые нейросети: генератор шумов и модель для детекции сгенерированных аугментаций. Закономерным итогом является повышение качества рекомендаций за счет более устойчивой обобщающей способности эмбеддингов. Тем не менее указанные подходы, как и их предыдущие аналоги, обладают одной существенной особенностью. Базовые графовые сети предназначены для однородных двудольных графов, содержащих два типа узлов: товары и пользователи, и один тип ребер – взаимодействие (покупка). В современных рекомендательных системах используется множество сигналов (просмотры, добавления в корзину, добавления в избранное и т. д.). В приведенных работах агрегации вроде $E^{(l)} = AE^{(l-1)}$, где E – эмбеддинги сущностей, работают только при однородной матрице смежности графа A. Наша работа учитывает множественные сигналы и принимает во внимание как тип исходных объектов графа, так и тип связей (взаимодействий).

Другая большая группа path-based-методов (к которой мы относим и свою работу) извлекает информацию из последовательности переходов между узлами (графовыми путями) для моделирования распределения вероятностей рекомендуемых объектов. Инструменты вывода здесь варьируются от классических EM-алгоритмов и марковских процессов до рекуррентных, полносвязных нейросетевых модулей и языковых моделей. Классическим примером здесь может послужить модель RippleNet [18], которая напрямую использует граф знаний, чтобы моделировать распространение пользовательских предпочтений по графовым путям. В работе используется понятие “волна предпочтений”, которую можно представить как путь от вершины пользователя, проходящий через приобретенные товары к узлам других сущностей. RippleNet итеративно строит цепочки предпочтений глубиной K (обычно 2-3). На каждом шаге векторное представление интереса пользователя обновляется через механизм внимания с учетом эмбеддингов типа связи и конечного узла перехода. Таким образом, модель обращает внимание только на те сущности в графе, которые резонируют с текущими пожеланиями пользователя и способна моделировать ком-

плексные многосторонние предпочтения. С другой стороны, решение очень чувствительно с подбору гиперпараметров, в частности, количеству и длине волн. Оно может не учесть или переобобщить зависимости при неверно подобранном *K*. Авторы PGPR (Policy-Guided Path Reasoning) [19] формулируют задачу рекомендации как поиск графовых путей от пользователя к товару, используя принципы обучения с подкреплением. Модель должна не просто предсказать интерес, а объяснить его логикой маршрута. Под средой обучения понимается сам граф знаний. Агентом выступает нейросеть, пошагово принимающая решения о выборе траектории. На каждой итерации определяется следующее действие: тип связи и сущность для перехода. Агент получает награду, если он пришел к товару, с которым пользователь взаимодействовал. Для обновления весов нейросети используется градиент политики. В отличие от графовых сетей PGPR не агрегирует все соседние представления, а целенаправленно находит важные цепочки покупок, сокращая шум и чрезмерную обобщенность эмбеддингов. PGPR чувствителен к выбору алгоритма обхода: агент может не найти все релевантные пути, особенно в зашумленныих или неполных графах. В нашем подходе применяются экспертно заданные шаблонные траектории, что упрощяет процедуру движения по графу. Решения [20, 21] используют механизмы языкового моделирования, представляя ребра и узлы графа специальными токенами и генерируя рекомендации в виде путей-кандидатов с соответствующими совместными вероятностями в качестве ранжирующих оценок. Это позволяет моделировать сложные исторические зависимости между токенами – действиями пользователя. Генерация последовательностей путей выполняется в авторегрессионном стиле, т. е. декодирование одного токена за раз слева направо с учетом последовательности ранее наблюдавшихся токенов. Метод, изложенный в работе [22], использует практически те же принципы построения рекомендательной модели, что и базовые решения: CAFE и HeteroEmbed [10, 11], которые были положены в основу наших экспериментов. Оценка вероятности графового перехода осуществляется набором компактных нейросетевых модулей, состоящих из нескольких полносвязных слоев. Модель KPAR (Knowledge-aware Path-based Attentive Recommender), описанная в статье [23], представляет семантические связи между пользователем и товаром с помощью пути, представленного как последовательность триплетов вида (сущность, отношение, сущность). На первом этапе извлекается множество релевантных путей между покупателем и целевым объектом взаимодействия. Каждый путь кодируется с учетом структуры отношений, используя специальный механизм внимания, подобный тому, что применяется в стандартных трансформерах. Используя перекрестное внимание (cross attention), все пути перевзвешиваются в зависимости от их итоговой значимости, а именно: от факта наличия взаимодействия с конечным товаром либо его отсутствия. KPAR обеспечивает глубокое семантическое моделирование и одновременно объяснимость действий пользователя, что является неоспоримым преимуществом модели. Ключевым отличием нашего подхода от методов, описанных выше, является использование предобученных семантических эмбеддингов для некоторых типов сущностей. Это позволяет обогащать структуру знаний графа, усиливая смысловые связи между объектами и улучшая качество рекомендаций.

Существенные результаты достигнуты в области представления сущностей с набором признаков из разных типов модальностей. В подобных работах, как и в нашей, для предварительного формирования эмбеддингов используются специализированные нейросетевые архитектуры. Авторы [24], используя наработки в домене компьютерного зрения, комбинируют мультимодальный трансформер (для предобработки текста и графических изображений) и графовую сеть. Трансформер формирует разнотипные векторные представления, сводя их в единое пространство. В то время как GNN эффективно объединяет эмбеддинги признаков в единую сущность, добавляя туда сигналы о взаимодействиях этого товара с покупателями. Визуальные и текстовые признаки агрегируются в единый эмбеддинг товара, а в графе остаются лишь два типа объектов: пользователи и товары. Напротив, в нашем решении трансформер используется для извлечения признаков, встраиваемых в граф в виде отдельных семантических сущностей. Эти сущности образуют связи с другими объектами графа, что позволяет сохранять структурированное представление доменной информации и расширять семантическое пространство графа.

3. ПОСТАНОВКА ПРОБЛЕМЫ

Формализуя задачу, обозначим U – множество пользователей, I – множество товаров, $G = \{(e, r, \acute{e}) | e, \acute{e} \in E, r \in R\}$ – граф знаний, построенный на основе наблюдаемых пользовательских взаимодействий и вспомогательной информации. Здесь E и R соответственно представляют множество сущностей и связей моделируемой области. Каждая вершина $e \in E$ представляет сущность предметной области (например, пользователь, товар, категория, бренд), а каждое ребро $r \in R$ описывает отношения между этими сущностями. Для каждого пользователя $u \in U$ определено множество уже приобретенных товаров $I_u \subseteq I$, а также подграф $G_u \subseteq G$, индуцированный соответствующими путями покупок: $l = (e_0, r_1, e_1, r_2, ..., e_{|l|-1}, r_{|l|}, e_{|l|} | e_0, ..., e_{|l|} \in E, r_1, ..., r_{|l|} \in R, (e_{t-1}, r_t, e_t) \in G, t = 1, ..., |l|)$. Совокупность всех возможных путей в графе G обозначим как L. Подмножество $L_u \in L$ называется путями покупок пользователя при условии, что для любого пути множества $e_0^u \in U$, $e_{|l|}^u \in I_u$, где I_u – множество товаров уже приобретенных покупателем u. То есть путь покупки всегда начинается в вершине типа “пользователь” и через множество вспомогательных сущностей заканчивается в вершине типа “товар”, входящей в множество покупок пользователя u. Для краткости обозначим такой путь как $l_{u \to i}$, $i \in I_u$. Тогда формализовать задачу выполнения рекомендаций можно следующим образом. Для каждого пользователя $u \in U$ на основании заданного L_u необходимо определить K наиболее вероятных путей покупки $L'_u = \{l_{u \to i}^{(k)'} \mid i \notin I_u, k \in [K]\}$. Множество I_u', составляющее конечные вершины путей L'_u, будет набором рекомендованных к покупке товаров для пользователя u.

Дополнительной особенностью задачи является наличие неструктурированных текстовых атрибутов у объектов: описания товаров $f_d(i)$ и пользовательские отзывы $f_r(u, i)$. Эти источники содержат скрытые семантические признаки, которые традиционно не представлены в графе знаний явным образом. Возникает необходимость дополнительно извлекать из этих текстов новые сущности $E*$ и связи $R*$, которые затем могут быть встроены в граф, расширяя его структуру до $G* = (\{E* \cup E\}, \{R* \cup R\})$ в целях более глубокого моделирования предметной области и повышения качества рекомендаций.

Применительно к исследуемой нами предметной области можно выделить следующий набор вершин: “товар”, “пользователь”, “бренд”, “производитель”, “категория продукта”, “характеристика”. Множество связей состоит из отношений: “товар продан”, “покупатель приобрел”, “упомянул” (товар через сущность “характеристика” упомянут в отзыве или комментарии пользователя), “принадлежит” (отражает принадлежность объекта к категории или бренду), “произведен” (связывает товар с его компанией-производителем), “куплен вместе” (указывает на другие продукты, которые пользователь приобрел в рамках одного заказа), “просмотрен” (товар был просмотрен незадолго до или сразу после сделки). Следует отметить, что подобный перечень ничем не ограничен. Инжиниринг новых сущностей и ввод их в знаниевую модель зависит только от возможностей исследователя к более тщательной проработке предметной области задачи. Визуально часть графа с уже известными путями, а также потенциальные рекомендации можно представить на рис. 1.

Конечной целью рекомендательной системы является генерация множества товаров $I_u' \subseteq I / I_u$, упорядоченного по убыванию релевантности для конкретного пользователя u. Для этого система должна эффективно извлекать, моделировать и учитывать сложные связи между сущностями графа G, в том числе через латентные траектории взаимодействия, проходящие через промежуточные вершины.

4. ПРЕДЛАГАЕМЫЙ МЕТОД РЕШЕНИЯ

Предлагаемый подход состоит из двух последовательных этапов. Сначала выполняется семантическое обогащение графа знаний за счет включения дополнительных сущностей и связей, извлеченных из неструктурированных текстовых атрибутов, таких как описания товаров и пользовательские отзывы. Эти данные служат источником латентной информации, неявно отражающей взаимосвязи объектов предметной области. Любое расширение данных является достаточно значимым фактором для path based-подходов, позволяющим находить новые рекомендательные маршруты. На втором этапе осуществляется построение обобщенных векторных представлений всех сущностей и типов связей в графе. Для этого приме-

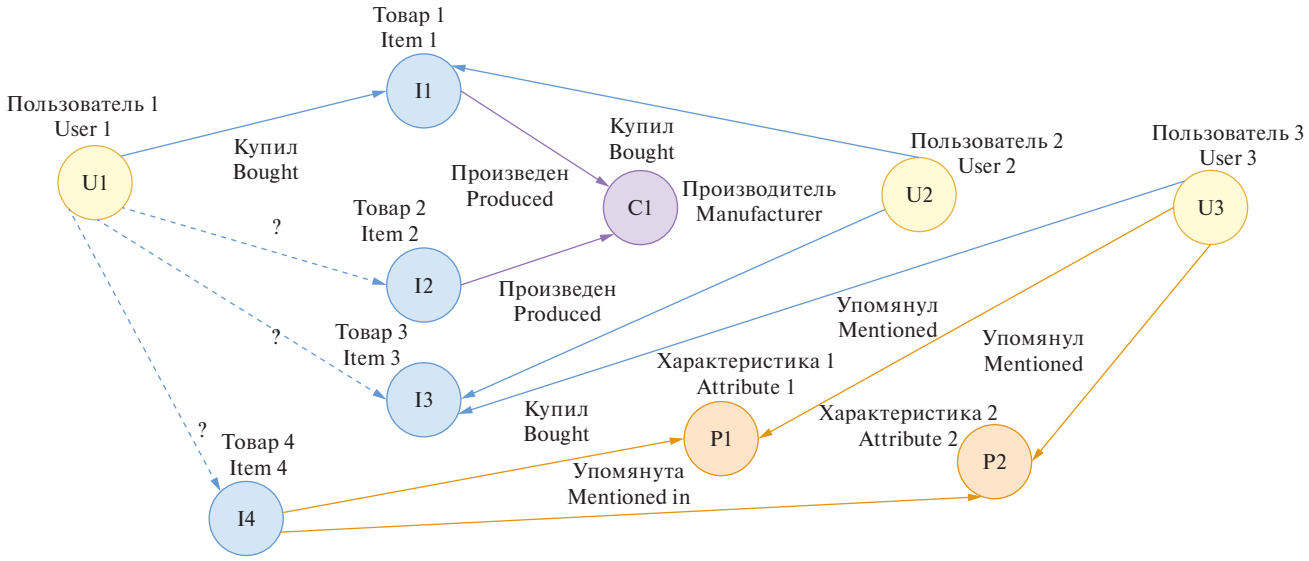

Рис. 1. Пример фрагмента структуры графа знаний рекомендательной системы. Вершины различного типа (пользователи, товары, производители, характеристики) соединены ребрами, отражающими семантические отношения: “купил”, “произведен”, “упомянул”, “упомянута”. Отображены как фактические взаимодействия (сплошные линии), так и потенциальные связи (пунктирные линии), которые могут быть предсказаны моделью. Цветами выделены различные типы сущностей.

няется обучаемая нейросетевая архитектура, принимающая во внимание экспертно заданные шаблоны путей покупок.

4.1. Применение текстового энкодера для обогащения графа

Современные трансформерные модели обработки естественного языка демонстрируют высокую эффективность в извлечении смысловых признаков из неструктурированного текста. Эти предобученные энкодеры способны формировать представления, отражающие скрытую семантику слов и предложений, составляющих такие атрибуты, как, например, товарные описания и отзывы пользователей. В рамках предлагаемого подхода текстовый энкодер используется для извлечения ключевых смысловых единиц (слов, фраз и их сочетаний), которые интерпретируются как новые сущности предметной области. Извлеченные элементы встраиваются в граф знаний в виде вершин, соединенных с уже существующими объектами (товарами и пользователями) специальными типами связей. Таким образом, структура графа расширяется и обогащается дополнительной семантической информацией, что способствует улучшению качества последующего обучения рекомендательной модели. Среди множества существующих энкодеров BERT-подобной архитектуры в данной работе была выбрана модель multilingual-e5-base [25], относящаяся к семейству моделей E5, специально разработанных для извлечения универсальных семантических представлений в задачах сопоставления текстов. Особенностью данной модели является ее способность обрабатывать тексты на различных языках, что делает ее применимой в многоязычных рекомендательных системах. Для адаптации под специфику рекомендательного домена модель прошла дообучение на корпусе текстов из Amazon Reviews [26] с использованием задачи восстановления маскированных токенов (Masked Language Modeling, MLM). Извлеченные сущности встраиваются в граф связями типа “характеризуется”, “характеризует”, когда речь идет о товаре, а также “упомянул” и “упомянут”, когда некоторое свойство предмета фигурирует в отзыве потребителя. Кроме того, мы используем для частичной инициализации графа эмбеддинги, сконструированные энкодером. Так, векторные представления сущности “характеристика” копируются в нейросетевую модель вывода на этапе ее создания. Это несколько ускоряет процесс схождения весов во время дальнейшего обучения.

Рассмотрим пошагово алгоритм 1 (algorithm 1). В строке 3 исходные документы (описания товаров) проходят очистку, т. е. из текста удаляется определенный список стоп-слов. Проводится лемматизация слов, т. е. приведение их к первоначальной словарной форме. Из очищен-

ных предложений формируется набор n-грамм. В своем исследовании мы использовали униграммы и биграммы. Именно для них экспериментальные результаты были наилучшими. Последовательности больших размеров становятся достаточно уникальными и не несут полезности в формировании связей между объектами модели знаний рекомендатора. В строке 6 энкодер выстраивает векторные представления документа и элемента. Идея заключается в том, что наиболее подходящие кандидаты должны отражать смысл документа как можно ближе, а значит, их векторы будут максимально сонаправлены с вектором самого текста. Хорошей мерой сонаправленности в высокоразмерном пространстве выступает косинусная близость. Выбираются те слова и фразы, которые преодолели заданный минимальный порог схожести. С точки зрения знаниевой модели они и будут являться сущностями, характеризующими товар либо упомянутыми в отзыве пользователя. Векторизованные представления выбранных n-грамм сохраняются и позднее встраиваются в граф рекомендательной системы.

При работе с биграммами или триграммами следует учесть следующий важный аспект. В выборку могут попасть фразы, отличающиеся друг от друга набором слов, но выделяющие смысл документа в одинаковой степени. Чтобы избежать подобных ситуаций, можно несколько увеличить выборку кандидатов, а затем, дополнительно рассчитав схожесть между самими фразами, оставить только те, у которых парная близость минимальна. В строке 10 происходит расчет схожести *ub* с другими элементами. В строке 13 мы удаляем n-граммы, близость которых к *ub* выше порога *ElemScore*.

Сгенерированные сущности и связи встраиваются в граф. Их представления сохраняются, чтобы в дальнейшем быть использованы в обучении рекомендатора.

4.2. Нейросетевая модель вывода

Задача предсказания существования путей в графе G решается через расчет так называемых внутренних векторизованных представлений сущностей и связей, называемых эмбеддингами знаний (Knowledge Base Embeddings, KBE). Основное требование к KBE заключается в корректном кодировании структурной и смысловой информации, проецирующей отношения между

Алгоритм 1. Обогащение графа знаний внешней семантикой из текстовых описаний

Вход: исходный граф $G = (E, R)$; F_r – множество отзывов пользователей о товарах; F_d – множество описаний товаров; *DocScore* – минимальный порог схожести документа и его элемента; *ElemScore* – максимальный порог схожести элементов друг на друга.

Выход: обогащенный граф $G* = (E*, R*)$; матрица представлений новых сущностей V.

1: $R^* \leftarrow R$, $E^* \leftarrow E$, $V \leftarrow []$
2: **Для каждой пары:** товар и описание $(i, f_d) \in F_d$
3: Из f_d получить биграммы и униграммы: $UB \leftarrow \{w_1, w_2, ..., w_n, w_{12}, w_{13}, ..., w_{ij} \mid i < j\}$
4: $U_{\text{scores}} \leftarrow []$
5: **Для каждого** $ub \in UB$
6: $Score \leftarrow \cos(\text{emb}(f_d), \text{emb}(ub))$
7: **Если** $Score < DocScore$:
8: $UB \leftarrow UB \setminus \{ub\}$
9: **Переход на:** 5
10: $U_{\text{scrs}}[ub, \cdot] \leftarrow \{\cos(\text{emb}(ub), \text{emb}(w)) \mid w \in UB\}$
11: $UB \leftarrow \text{sort}(UB)$ по убыванию $\cos(\text{emb}(f_d), \text{emb}(ub))$, $\forall ub \in UB$
12: **Для каждого** $ub \in UB$
13: $UB \leftarrow UB \setminus \{w \in UB \mid U_{\text{scrs}}[ub, w] > ElemScore\}$
14: $V \leftarrow V \cup \{ub(ub)\}$
15: $E^* \leftarrow E^* \cup \{ub\}$
16: $R^* \leftarrow R^* \cup \{\text{relation}(ub, i)\}$
17: **Повторить** шаги 2:16 для каждого $(i, u, f_r) \in F_r$
18: **Вернуть** $G* = (E*, R*)$; V

описываемыми объектами в низкоразмерное латентное пространство. Обозначив эмбеддинги исходной и конечной вершин как e_h и e_t соответственно, а закодированное представление связи между ними как r, можно выразить соотношение

$$e_t = trans(e_h, r) = e_h + r. \quad (4.1)$$

Здесь $e_h, e_t, r \in \mathbb{R}^d$, d – размерность латентного пространства. Отношение r фактически является линейным преобразованием из e_h в e_t [11]. На практике достичь выполнения подобного равенства для всех вершин и отношений графа невозможно. Как правило, находится решение нестрогих соответствий: $trans(e_h, r) \approx e_t$ для всех $(e_h, r, e_t) \in G$ и $trans(e_h, r) \neq e_t$ для любых $(e_h, r, e_t') \notin G$. Общий подход к решению был предложен *Mikolov at al.* [27] и заключается в оптимизации порождающей вероятности e_t при заданном $trans(e_h, r)$:

$$Score\left(e_t\right) = \exp\left(e_t \cdot trans\left(e_h, r\right)\right);$$

$$Z = \sum_{e_t' \in E_t'} \exp\left(e_t' \cdot trans\left(e_h, r\right)\right);$$

$$P\left(e_t \mid trans\left(e_h, r\right)\right) = \frac{Score\left(e_t\right)}{Z}, \quad (4.2)$$

где E_t' – множество вершин e_t' того же типа, что и e_t, при этом возможно, что $(e_h, r, e_t') \notin G$; Z – нормализующее множество. Операция $[\cdot]$ выполняет скалярное произведение векторов.

Применительно к задаче формирования рекомендаций вероятность появления связи между узлами a и b должна определяться на основе некоторой предшествующей последовательности переходов между вершинами, приводящей в итоге к a. Например, покупке определенного товара может способствовать факт его производства той же компанией, что и другие товары, приобретенные данным покупателем. Тогда путь покупки выглядит так: Пользователь → Купил → Товар_1 → Произведен → Компания_А → Произвела → Товар_2 → Куплен → Пользователь. То есть на вероятность появления последней связи “Куплен” влияет совокупная вероятность предшествующих переходов. Один из возможных подходов к решению предложен в [10]. Введем понятие некоторой траектории переходов $h_t = (u, r_1, e_1, r_2, \ldots, r_{t-1}, e_{t-1})$ для пользователя u. Тогда вероятность существования конечного пути покупки l_u, определяемую некоторым программным модулем Φ с набором подстраиваемых параметров Θ, обозначим как $P_\Theta\left(l_u \mid u\right) = \prod_{t=1}^{T} P_\Theta\left(r_t, e_t \mid h_t, u\right)$. В силу сложности дифференцирования данного выражения прибегают к взятию натурального логарифма:

$$\log\left(P_\Theta\left(l_u \mid u\right)\right) = \sum_{t=1}^{T} \log\left(P_\Theta\left(r_t, e_t \mid h_t, u\right)\right). \quad (4.3)$$

Модуль Φ архитектурно реализован в виде многослойной нейронной сети (МНС, MLP). В силу наличия весьма существенного количества узлов и ребер графа обучение единой МНС довольно затратно в вычислительном отношении. Основываясь на идеях в [10], мы также применяли ансамбль нейросетевых модулей, каждый из которых отвечает за кодирование параметров для определенного типа связи r. Каждый модуль включает в себя три матрицы параметров: $W_{r,1}, W_{r,2} \in \mathbb{R}^{2d \times 2d}$, $W_{r,3} \in \mathbb{R}^{2d \times d}$, т. е. $\Theta_r = \{W_{r,1}, W_{r,2}, W_{r,3}\}$. На вход модуля подаются эмбеддинги как самого пользователя u, так и его траекторий h_t. Последние рассчитываются итеративно. Так, на нулевом шаге ($t = 0$) в качестве h_t используется вектор u. Далее, для $t \geqslant 1$, h_t является выходом соответствующего связи r модуля Φ_r на шаге $t - 1$. Мы использовали следующую структуру слоев MLP:

$$\Phi_{r,1}\left(u, h, \Theta_r\right) = \sigma\left([u; h]\, W_{r,1}\right) + [u; h];$$

$$\Phi_{r,2}\left(u, h, \Theta_r\right) = \sigma\left(\Phi_{r,1}\left(u, h, \Theta_r\right) W_{r,2}\right) + \Phi_{r,1}\left(u, h, \Theta_r\right);$$

$$\Phi_r\left(u, h, \Theta_r\right) = \Phi_{r,3}\left(u, h, \Theta_r\right) = \sigma\left(\Phi_{r,2}\left(u, h, \Theta_r\right) W_{r,3}\right).$$

Здесь $\Phi_r(u, h, \Theta_r)$ является выходом модуля, операция [;] означает конкатенацию векторов, операция [+] выполняет поэлементное сложение матриц, а под функцией σ понимается какая-либо функция активации MLP (в нашем исследовании использовалась ReLU [31]). В отличие от модели, использованной в [10], в нашей сети добавляются остаточные связи, позволяющие улучшить общую сходимость решения.

Основываясь на выражении (4.2), вероятность связи r_t с вершиной e_t можно определить как

$$\mathrm{RelationScore}\left(e_t\right) = \exp\left(\left\langle \Phi_r\left(u, h_t, \Theta_r\right), e_t \right\rangle\right);$$

$$Z = \sum_{e_t' \in E_t'} \exp\left(\left\langle \Phi_r\left(u, h_t, \Theta_r\right), e_t' \right\rangle\right);$$

$$P_\Theta\left(r_t, e_t \mid u, h_t\right) = \frac{\mathrm{RelationScore}\left(e_t\right)}{Z}. \quad (4.4)$$

В данном выражении операция $\langle \cdot, \cdot \rangle$ является скалярным произведением векторов.

Для обучения сети необходимо задать функцию потерь. С одной стороны, здесь требуется

поощрять модель за формирование правильного множества путей покупок L_u, т. е. максимизировать $P_\Theta(l_u|u)$. Используя (4.3), запишем функцию потерь:

$$l_p(\Theta; l_u) = -\log\left(P_\Theta(l_u \mid u)\right);$$

$$l_p(\Theta; l_u) = -\sum_{t=1}^{T} \log\left(P_\Theta(r_t, e_t \mid u, h_t)\right). \quad (4.5)$$

Одновременно необходимо штрафовать рекомендатор за неверные решения, т. е. попытку завышения вероятности покупки товаров, которые так и не были приобретены. Для решения этой задачи был выбран алгоритм негативного семплирования наиболее популярных товаров (Popularity-Based Negative Sampling, PNS). По сравнению со случайным семплированием PNS пытается извлечь как можно больше информации из несостоявшихся покупок. Акцентируя внимание именно на популярных продуктах, модель выстраивает решающую плоскость более аккуратно. Имея множество товаров I, мы рассчитываем вес каждого товара w_i как частоту покупок на исторической ретроспективе. Для семплирования рассчитывается значение $k_i = \text{rand}(0, 1)^{1/w_i}$, где rand – функция генерации случайного числа из равномерного распределения от 0 до 1. Для пользователя u отбираются M товаров с наибольшим k_i при условии их отсутствия в истории покупок u: $I_u^- = \{i^{(m)} \mid i^{(m)} \notin I_u, m \in [M]\}$. Как было отмечено в начале раздела, существует также множество L_u исторических путей покупок пользователя u. Обозначим i_{lu}^+ товар, купленный пользователем в конце некоторого пути $l_u \in L_u$. Тогда функция потерь для решения итоговой задачи будет выглядеть следующим образом:

$$S^+ = \left\langle i_{l_u}^+, \Phi_r(u, h, \Theta_r) \right\rangle;$$

Алгоритм 2. Обучение нейросетевой модели вывода рекомендаций

Вход: обогащенный граф $G = (E, R)$; множество пользователей U; множество всех товаров I; множество купленных товаров I_u^+ для каждого $u \in U$; множество шаблонов графовых путей M; матрица предрасчитанных представлений текстовых сущностей V_{kn}; множество обучаемых нейросетевых модулей $\Phi = \{\Phi_1, \Phi_2, ..., \Phi_n\}$ и соответствующих параметров $\Theta = \{\Theta_1, \Theta_2, ..., \Theta_n\}$, где $n = |\{\text{type}(r)\}| \; r \in R|$; гиперпараметр λ ; параметры обучения нейросети.

Выход: обученные модули и параметры (Φ, Θ).

$\Theta \leftarrow V_{kn} \cup \{\text{init\_emb}(s) \mid s \in E \cup R, s \notin V_{kn}\}$
$l_{\text{total}} \leftarrow 0, l_p \leftarrow 0, l_r \leftarrow 0$
Для каждого $u \in U$
  Для каждого шаблона $m \in M$
    Для каждого купленного товара пользователя $i_u^+ \in I_u^+$
      Семплируем пути покупок графа $L_u \sim \text{traverse}(G, u, i_u^+, m)$
      Для каждого $l_u \in L_u$
        $t \leftarrow 0$
        Для каждого триплета $(u, r_i, e_i) \in l_u$, $0 \le i \le \text{len}(l_u)$
          $h_t \leftarrow \begin{cases} \Phi_{r_i}(\Theta_{r_i}, u, u), & \textbf{если } t = 0 \\ \Phi_{r_i}(\Theta_{r_i}, u, h_{t-1}), & \textbf{если } t > 0 \end{cases}$
        Вычислить вероятность $P_\Theta(r_t, e_t \mid u, h_t)$ по формуле (4.4)
        Вычислить потери $l_p(\Theta; l_u)$ по формуле (4.5)
        $l_p \leftarrow l_p + l_p(\Theta; l_u)$; $t \leftarrow t + 1$
      Семплируем множество негативных товаров I_u^-
      Вычислить $l_r(\Theta; l_u)$ по формуле (4.6)
      $l_r \leftarrow l_r + l_r(\Theta; l_u)$
    Если достигнут размер батча:
      $l_{\text{total}} \leftarrow \lambda\, l_p + (1 - \lambda)\, l_r$
      Обновить параметры Θ по градиенту $\nabla_\Theta l_{\text{total}}$
      $l_p \leftarrow 0, l_r \leftarrow 0$
Вернуть Φ, Θ

$$S^- = \left\langle i_u^-, \Phi_r\left(u, h, \Theta_r\right)\right\rangle;$$

$$l_r\left(\Theta; l_u\right) = -\sum_{i_u^- \sim I_u^-} \log\left[\sigma\left(S^+ - S^-\right)\right]. \quad (4.6)$$

Поясним данное выражение. Во время обучения нейросети происходит минимизация функции потерь. Минимум l_r достигается в случае, когда модули Φ_r рассчитывают эмбеддинги, максимально сонаправленные с эмбеддингами i_{lu}^+, т. е. товара, купленного пользователем u, и одновременно максимально отличающиеся от кодированного представления невостребованных товаров I_u^-. Под функцией $\sigma(\cdot)$ в данном контексте понимается сигмоида, нормирующая разницу эмбеддингов в диапазон (0, 1). Общая функция потерь складывается из суммы (4.5) и (4.6) по всем пользователям и их путям покупок в графе. В силу ограниченности вычислительных ресурсов для больших моделей возможно семплирование N путей для каждого покупателя:

$$l(\Theta) = \sum_u \sum_{l_u \sim L_u} \left[\lambda l_p\left(\Theta; l_u\right) + (1-\lambda) l_r\left(\Theta; l_u\right)\right]. \quad (4.7)$$

Здесь λ выражает весомость вклада каждой ошибки и является гиперпараметром модели.

Формально, весь процесс обучения представлен в виде алгоритма 2 (algorithm 2). Функция init_emb выполняет начальную инициализацию эмбеддингов сущностей. В качестве инициализатора используется равномерное распределение $U(-0.5, 0.5)$. Метод traverse реализует алгоритм двунаправленного поиска в ширину (bidirectional BFS), обеспечивая эффективный перебор возможных путей в графе между заданными парами начальной и конечной вершин.

4.3. Процесс построения рекомендаций

На этом этапе имеется сконструированный граф знаний G. Также обучен набор нейросетевых модулей $\{\Phi_r, r \in R\}$, каждый из которых принимает на вход пару (u, h_t), $u, h_t \in \mathbb{R}^d$, где u – созданные в ходе обучения эмбеддинги пользователя, а h_t – эмбеддинги истории некоторой траектории в графе G, начатой в узле u и приведшей к вершине e_{t-1}. Применяя (4.4) к выходу модуля Φ_r, получаем вероятности $P_\Theta(r_t, e_t \mid u, h_t)$ существования связи r_t из e_{t-1} в конечное множество однотипных (отражают сущности одного типа) вершин $\{e_t^{(i)}, i \in [N]\}$, где N – количество таких вершин. Согласно (4.4) P_Θ являются нормализованными вероятностями переходов. Кроме того, если рассматривать отдельно взятую траекторию, P_Θ также можно принять как условную вероятность перехода между узлами e_{t-1} и e_t этой траектории, зависящую от u и h_t.

Таким образом, задача построения рекомендаций заключается в выборе K наиболее вероятных траекторий $u \rightsquigarrow i$, которые могут привести к приобретению пользователем u товара i, при условии $i \notin I_u$, т. е. отсутствия товара в предыдущих покупках. На практике подобный подход имеет существенный недостаток. При наличии достаточно весомого перехода в начале траектории вероятности конечных связей нивелируются. То есть возможен вариант выбора K траекторий, которые ничем не отличаются в начале и имеют небольшие вариации лишь в своем конце. Как следствие это может привести к формированию однотипных рекомендаций, выведенных практически одинаковым путем. В качестве возможного решения предлагается использование алгоритма стохастического лучевого поиска (Stochastic Beam Search, SBS) [29]. В этом методе на каждом уровне перехода применяется стохастический оператор (Gumbel-Max[30]), добавляющий шум из распределения Гумбеля к логарифмированным вероятностям возможных переходов. Выбор узлов для спуска происходит так:

$$\tilde{z}_j = \log p_j + g_j,\ g_j \sim \text{Gumbel}(0, 1);$$

$$B = \text{Top}\, K_j\left(\tilde{z}_j\right).$$

Здесь p_j – вероятность перехода в узел j с предыдущей вершины; g_j – случайная величина, сэмплированная из распределения Гумбеля с параметрами (0, 1); B – множество K лучших узлов, отобранных по убыванию значений $\tilde{z}_j$.

Для каждого пользователя u строится множество траекторий, соответствующих заданному шаблону. Это множество можно рассматривать как дерево с вершиной в u. На каждом шаге происходит спуск на следующий уровень дерева. Пусть B_{t-1} – множество узлов, выбранных на предыдущей итерации. Для каждого узла $b \in B_{t-1}$ формируется множество кандидатов $C_b' = \{ b' \mid (b, r, b') \in G \}$. Для всех $b' \in C_b$ по формуле (5) вычисляются лог вероятности выбора узлов. Далее, по (9) и (10) семплируем кандидатов для текущего шага, формируем из них B_t. Повторяем процесс, пока не пройдем все уровни шаблонной траектории. Визуально это можно представить на рис. 2.

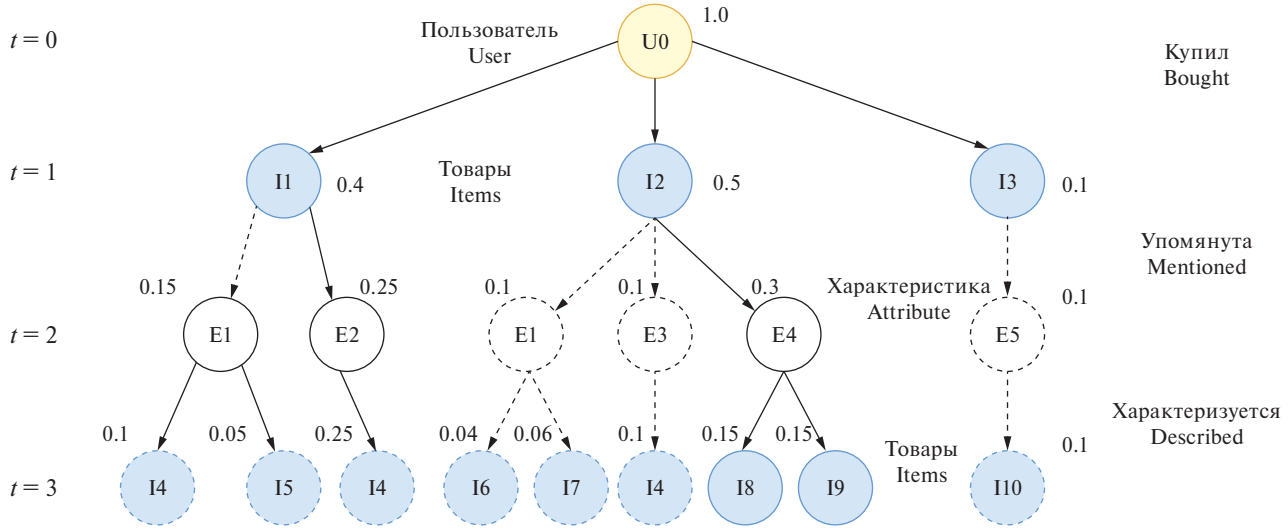

Рис. 2. Выбор траекторий рекомендуемых товаров для шаблона: “пользователь” → “купил” → “товар” → “упомянута” → “характеристика” → “характеризуется” → “товар”. Дерево имеет четыре уровня. Цифры около узлов обозначают вероятность выбора соответствующего элемента в пределах уровня. Значение K принимается равным 3, т.е. на каждом уровне, используя (4.8) и (4.9), выбирается по три следующих узла. Пунктиром очерчены не попавшие в выборку объекты. Различным цветом проиллюстрированы разные типы сущностей. Финально выбраны товары I4, I8, I9.

Нетрудно заметить, что все траектории, принадлежащие одному шаблону, имеют одинаковое количество уровней. Здесь SBS применяются только в пределах групп однотипных шаблонных траекторий. Получив K сэмплирований для каждой группы, мы нормируем кумулятивные вероятности путей на их длину, возведя в степень $\frac{1}{T}$. После чего из всех цепочек еще раз, теперь уже финально, отбираются K лучших. Их концевые узлы, представляющие товары, и будут текущими рекомендациями для пользователя u. Далее процесс повторяется для всех других покупателей.

4.4. Ограничения предлагаемого подхода

Существенным ограничением алгоритма обогащения графа является его фокус на текстовой модальности. Остается открытым вопрос интеграции других источников информации, таких как изображения или аудиальные данные. Отсутствие поддержки мультимодальных признаков может ограничить полноту представления знаний о товарах, особенно в тех случаях, когда визуальная составляющая (например, фотографии упаковки или внешний вид продукта) играет важную роль в пользовательском восприятии и принятии решений. Алгоритм может быть использован в тех доменных областях, где представлены сущности со значимыми текстовыми атрибутами.

Модель вывода также обладает рядом ограничений. Во-первых, ее эффективность существенно зависит от корректности зафиксированной схемы шаблонных путей обхода графа. Эти шаблоны, определяющие допустимые траектории между пользователями и товарами, задаются вручную и требуют участия эксперта, хорошо понимающего структуру предметной области. Если потенциально важные связи между сущностями не будут отражены в шаблонах, модель утратит возможность обрабатывать соответствующие семантические зависимости, что приведет к потере качества рекомендаций. Во-вторых, процесс обучения модели требует итеративного обхода графа. Для каждого пользователя необходимо просемплировать пути по всем шаблонам, собрать последовательности триплетов и применить соответствующие модули трансформации. Такой подход оказывается вычислительно затратным и не масштабируется на большие графы в режиме реального времени. Это делает невозможным онлайн-обновление модели. Актуализация возможна лишь в офлайн-режиме с периодическим запуском переобучающей процедуры.

5. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

В данном разделе приведены результаты проведенных испытаний. В силу того, что настоящее исследование базируется на материалах, изложенных в [10], авторы, в первую очередь, ставили перед собой цель предложить улучшения исходной модели.

Программный код подготовлен на языке программирования Python 3.10.12. Список основных библиотек программ включал: Torch 2.3.0, Numpy 1.26.4, NetworkX 3.3. Аппаратная часть состояла из процессора Intel Core i9, выполняющего базовые вычислительные операции, и графического ускорителя NVIDIA GeForce GTX 1660 для обучения нейросетевых модулей. В целях соблюдения корректности сравнения были использованы те же наборы данных, что и в оригинальной статье. Указанные датасеты доступны для загрузки по ссылке [26]. Все они относятся к проблемной области создания рекомендательных систем и содержат различную описательную информацию о предлагаемых товарах, покупателях, а также истории заказов. В работе приведены метрики экспериментов по набору данных Amazon Beauty Small 5-core. Для проверки существуют отдельно собранные файлы с перечнем покупок каждого пользователя на контрольном периоде времени. Предполагается, что подготовленная и обученная модель должна пройти оценку своих рекомендаций на предоставленных валидационных данных. В качестве метрик качества использовались четыре показателя. Точность для лучших K рекомендованных товаров (Precision @K, P@K) оценивает соотношение количества реально купленных пользователем товаров к K рекомендованным: $P@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K}\frac{hit\left(i_j,u\right)}{K}$, где n – количество покупателей, u – текущий оцениваемый покупатель, i_j – рекомендованный товар, а функция $hit(i_j, u)$ возвращает 1, если товар i_j был куплен пользователем u, и 0 в противном случае. Полнота для лучших K рекомендованных товаров (Recall @K, R@K) показывает, как много товаров из реального множества покупок мы смогли включить в рекомендации: $R@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K}\frac{hit\left(i_j,u\right)}{|relevant(u)|}$, где $relevant(u)$ – множество покупок пользователя u в оценочных данных. Дополнительной мерой оценки может являться абсолютное число покупок из списка рекомендованных товаров, нормированное на число пользователей системы: HitRate@K = $= \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K}hit\left(i_j,u\right)$. Нормализованный дисконтированный совокупный выигрыш (Normalized DiscountedCumulativeGain@K, NDCG@K) оценивает способность рекомендательной системы выдавать список продуктов в порядке их релевантности для покупателя. Фактически NDCG сравнивает порядок рекомендаций с идеально упорядоченным списком покупок пользователя, в котором наиболее релевантные товары стоят на первых местах:

$$NDCG@K = \frac{DCG@K}{IDCG@K};$$

$$DCG@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K}\frac{hit\left(i_j,u\right)}{\log_2(j+1)};$$

$$IDCG@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{|relevant(u)|}\frac{hit\left(i_j,u\right)}{\log_2(j+1)}.$$

Используя описанные метрики, мы оценили нашу модель на тех же наборах данных, что и исходная рекомендательная система в [10]. В табл. 1–3 приведены результаты для $K = 5$, $K = 10$, $K = 20$ рекомендованных товаров. Для устранения эффекта случайности было проведено 10 независимых итераций. Для каждого испытания можно сравнить показатели оригинального алгоритма CAFE и нашего подхода. Для удобства анализа метрики по всем экспериментам сведены в табл. 4.

Предложенное нами решение демонстрирует прирост значений ключевых показателей практически на всех итерациях. Однако отличие метрик не столь выразительно и требует проведения комплексной проверки статистической значимости. Сперва с помощью теста Шапиро–Уилка оценим нормальность распределения выборок. По результатам из табл. 5 на всех показателях значение P-value более 0.05, что позволяет не отклонять гипотезу о требуемом характере распределения. Используя критерий Левена, оценим схожесть дисперсий. Результаты отображены в той же таблице. Принимаем гипотезу о равенстве дисперсий. Так как выборки нормально распределены и их дисперсии равны, то для оценки равенства средних применяется параметрический двухвыборочный t-критерий Стьюдента для независимых выборок с равными дисперсиями. Оценим t-статистику и P-value на уровне значи-

Таблица 1. Результаты работы рекомендаторов при $K = 5$. Датасет Amazon Beauty 5 Core

Итерация (Iteration)	Precision@5		Recall@5		Hitrate@5		NDCG@5	
	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)
1	2.735	2.875	10.812	11.105	18.589	18.900	7.059	7.132
2	2.740	2.860	10.780	11.150	18.590	18.760	7.035	7.100
3	2.730	2.850	10.770	11.100	18.540	18.730	7.020	7.110
4	2.750	2.880	10.790	11.130	18.580	18.790	7.040	7.140
5	2.720	2.890	10.740	11.140	18.530	18.810	7.010	7.110
6	2.750	2.875	10.765	11.100	18.590	18.790	7.045	7.120
7	2.730	2.865	10.780	11.130	18.520	18.760	7.030	7.110
8	2.745	2.880	10.800	11.140	18.560	18.800	7.050	7.120
9	2.735	2.860	10.765	11.120	18.570	18.780	7.025	7.110
10	2.740	2.875	10.790	11.130	18.580	18.770	7.035	7.110
Среднее (Mean)	**2.738**	**2.870**	**10.779**	**11.120**	**18.565**	**18.787**	**7.035**	**7.124**
Дисперсия (Variance)	**0.00009**	**0.00014**	**0.00040**	**0.00179**	**0.00038**	**0.00219**	**0.00017**	**0.00013**

Таблица 2. Сравнение результатов рекомендаций при $K = 10$. Датасет Amazon Beauty 5 Core

Итерация (Iteration)	Precision@10		Recall@10		Hitrate@10		NDCG@10	
	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)
1	2.573	2.605	10.869	10.941	18.406	18.500	7.062	7.065
2	2.564	2.618	10.943	11.013	18.446	18.487	7.059	7.062
3	2.569	2.600	10.950	10.973	18.506	18.520	7.064	7.067
4	2.574	2.607	10.896	10.938	18.447	18.495	7.059	7.062
5	2.571	2.603	10.885	10.951	18.457	18.513	7.062	7.065
6	2.567	2.615	10.917	10.966	18.436	18.510	7.061	7.070
7	2.576	2.611	10.958	10.981	18.494	18.524	7.065	7.068
8	2.566	2.602	10.933	10.965	18.471	18.488	7.058	7.060
9	2.568	2.606	10.897	10.966	18.462	18.502	7.062	7.066
10	2.572	2.615	10.936	10.980	18.473	18.519	7.066	7.071
Среднее (Mean)	**2.570**	**2.610**	**10.919**	**10.974**	**18.456**	**18.512**	**7.064**	**7.067**
Дисперсия (Variance)	**0.00015**	**0.00018**	**0.00055**	**0.00036**	**0.00082**	**0.00016**	**0.00006**	**0.00007**

Таблица 3. Сравнение результатов рекомендаций при $K = 20$. Датасет Amazon Beauty 5 Core

Итерация (Iteration)	Precision@20		Recall@20		Hitrate@20		NDCG@20	
	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)
1	2.244	2.271	10.667	10.751	17.757	17.976	6.793	6.787
2	2.235	2.259	10.630	10.696	17.694	17.896	6.783	6.786
3	2.254	2.266	10.700	10.714	17.802	17.864	6.805	6.788
4	2.256	2.270	10.731	10.772	17.846	17.963	6.803	6.828
5	2.248	2.272	10.705	10.784	17.761	17.976	6.783	6.831
6	2.256	2.273	10.724	10.760	17.837	17.954	6.795	6.821
7	2.244	2.266	10.642	10.765	17.739	17.985	6.793	6.798
8	2.243	2.279	10.669	10.802	17.784	18.034	6.819	6.821
9	2.254	2.264	10.746	10.796	17.833	17.943	6.818	6.823
10	2.239	2.270	10.646	10.729	17.775	17.905	6.787	6.800
Среднее (Mean)	**2.247**	**2.269**	**10.686**	**10.757**	**17.783**	**17.950**	**6.798**	**6.810**
Дисперсия (Variance)	**0.00005**	**0.00003**	**0.00149**	**0.00109**	**0.00208**	**0.00222**	**0.00015**	**0.00039**

Таблица 4. Итоговые результаты работы предложенной модели для различных значений K. Датасет Amazon Beauty 5 Core

K	Precision@K	Recall@K	Hitrate@K	NDCG@K
5	2.870	11.120	18.787	7.124
10	2.610	10.974	18.512	7.067
20	2.269	10.757	17.949	6.810

мости 0.05. Результаты также приведены в табл. 5. Для всех метрик кроме NDCG@20 подтверждена статистическая значимость. Это позволяет сделать вывод, что при $K = 5$ и $K = 10$ наша модель опережает CAFE. На 20 лучших рекомендациях качество становится практическо одинаковым. Причем по точности и полноте работы даже в этом случае мы находимся в выигрыше.

В табл. 6 показано относительное сравнение некоторых подходов, использующих принципы

Таблица 5. Результаты теста Шапиро–Уилка на нормальность распределения, критерия Левена на равенство дисперсий и t-критерия Стьюдента на равенство выборочных средних

Метрика (Metric)	Тест Шапиро–Уилка (Shapiro–Wilk Test)				Критерий Левена (Levene′s Test)		t-критерий (t-Test)	
	Статистика (Statistic)		P-value		Статистика (Statistics)	P-value	Статистика (Statstics)	P-value
	CAFE	Наша модель (Our model)	CAFE	Наша модель (Our model)				
Precision@5	0.9526	0.9519	0.6990	0.6914	0.2275	0.6391	–27.6207	≈ 0.0
Recall@5	0.9759	0.9062	0.9394	0.2560	0.0940	0.7626	–40.3526	≈ 0.0
Hitrate@5	0.8655	0.8506	0.0885	0.0590	0.4510	0.5104	–13.5299	1.0e–10
NDCG@5	0.9953	0.8665	0.9998	0.0909	0.3805	0.5451	–13.6333	1.0e–10
Precision@10	0.9800	0.9326	0.9653	0.4735	1.8923	0.1858	16.5826	≈ 0.0
Recall@10	0.9419	0.9356	0.5739	0.5049	2.3754	0.1407	–4.1566	0.0006
Hitrate@10	0.9811	0.9349	0.9706	0.4977	2.8433	0.1090	–4.5889	0.0002
NDCG@10	0.9463	0.9663	0.6255	0.8553	0.9584	0.3406	–2.7021	0.0146
Precision@20	0.9109	0.9738	0.2876	0.9240	1.9346	0.1812	–7.3896	7.5e–07
Recall@20	0.9387	0.9557	0.5390	0.7358	0.9365	0.3460	–4.1785	0.0006
Hitrate@20	0.9563	0.9659	0.7430	0.8504	0.0009	0.9764	–7.6284	4.8e–08
NDCG@20	0.9022	0.8898	0.2317	0.1686	3.7505	0.0687	–1.4667	0.1597

Таблица 6. Результаты сравнения path-based-моделей вывода рекомендаций на графах знаний с нашим подходом. $K = 10$. Датасет Amazon Beauty 5 Core

Алгоритм (Algorithm)	Precision@K	Recall@K	Hitrate@K	NDCG@K
RippleNet	1.699	8.127	14.681	5.162
PGPR	1.707	8.324	14.401	5.449
HeteroEmbed	1.986	10.411	17.498	6.399
CAFE	2.570	10.919	18.456	7.064
Наша модель (Our model)	**2.610**	**10.974**	**18.512**	**7.067**

глубокого обучения для вывода рекомендаций. Выбранные решения, как и наша работа, относятся к группе классических path-based-методов прогнозирования пользовательских предпочтений. В обзор вошли: RippleNet [18], PGPR [19], HeteroEmbed [11], CAFE [10], наша модель. Показатели качества большинства алгоритмов взяты из работы [10] в связи с отсутствием у авторов возможности в точности воспроизвести архитектуры указанных моделей. Результаты показывают превосходство предлагаемого решения над большинством методов подобного класса. Эксперименты проводились с использованием датасета Amazon Beauty 5-core [31]. Значение K равно 10.

6. ЗАКЛЮЧЕНИЕ

Задача построения эффективных систем рекомендаций имеет высокое прикладное значение для многих современных цифровых компаний, особенно в сегменте “бизнес для потребителя” (B2C). Увеличение объемов онлайн-продаж, сокращение оттока клиентов, целевое позиционирование товарных брендов – вот лишь некоторые макроцели, исполнение которых напрямую зависит от качества формируемых продуктовых наборов для индивидуальных покупателей.

Не оспаривая применимость классических подходов, приведенные результаты подчеркивают эффективность применения разработанной нейросетевой архитектуры, выстраивающих векторные представления сущностей путем обхода графов знаний для решения задач рекомендации товаров. В основу настоящей статьи были положены идеи, изложенные в работах [10, 11, 27]. Авторами статьи предложен ряд модификаций, позволивших улучшить качество работы исходного алгоритма, что продемонстрировано и доказано сравнительными испытаниями. Наиболее существенной является идея использования контекстного трансформера для конструирования эмбеддингов графовых сущностей из текстовых описаний объектов предметной области. Гипотетически уровень развития современных моделей глубокого обучения позволяет схожим образом извлечь и встроить в граф мультимодальные знания, выведенные из источников информации самых разнообразных типов, например, графических или видеопредставлений товаров. Это является предметом планируемых в будущем исследований.

СПИСОК ЛИТЕРАТУРЫ

1. *Ji R., Tian Y., Ma M.* Collaborative Filtering Recommendation Algorithm Based on User Characteristics // 2020 5th International Conference on Control, Robotics and Cybernetics (CRC). Wuhan, China. 2020. P. 56–60. DOI: 10.1109/CRC51253.2020.9253466.
2. *Brunton S.L., Nathan K.J.* Data-Driven Science and Engineering. Cambridge University Press. 1st edition. 2019. 43 p. DOI: 10.1017/9781108380690.002.
3. *Sarwar B., Karypis G., Konstan J., Riedl J.* Item-based collaborative filtering recommendation algorithms // Proceedings of the 10th International Conference on World Wide Web. April 2001. P. 285295. DOI: 10.1145/371920.372071.
4. *He X., Liao L., Zhang H., Nie L., Hu X., Chua T.* Neural Collaborative Filtering // Proceedings of the 26th International Conference on World Wide Web. April 2017. P. 173–182. DOI: 10.1145/3038912.305256.
5. *Su X., Khoshgoftaar T.M.* A Survey of Collaborative Filtering Techniques // Advances in Artificial Intelligence. Hindawi Publishing Corporation. 2009. P. 1–19. DOI: 10.1155/2009/421425.
6. *Каширин И.Ю.* Применение теории иерархических чисел в проектировании ICF-таксономии для оптимизации нейронных сетей // Вестник

Рязанского государственного радиотехнического университета. 2022. № 80. С. 118–127.
DOI: 10.21667/1995-4565-2022-80-118-126.

7. *Ayesha A.* Knowledge based Recommendation System in Semantic Web – A Survey // International Journal of Computer Applications. March 2019. No. 43. P. 20–25.
DOI: 10.5120/ijca2019918538.
8. *Бхатт Ш., Чжао Ц., Сетх А., Шалин В.* Графы знаний как средство улучшения искусственного интеллекта // Открытые системы. СУБД. 2020. № 3. С. 24–26.
https:// http://elibrary.ru/item.asp?id=43925226elibrary.ru/item.asp?id=43925226 (дата обращения: 14.03.2024)
9. *Кузнецов М.* Графовые нейронные сети.
https://education.yandex.ru/handbook/ml/https://education.yandex.ru/handbook/ml/ article/grafovye-nejronnye-seti (дата обращения: 31.03.2024)
10. *Yun X., Fu Z., Zhao H., Ge Y., Chen X., Huang Q., Geng S., Qin Z., Gerard de Melo, Muthukrishnan S., Zhang Y.* CAFE: Coarse-toFine Neural Symbolic Reasoning for Explainable Recommendation // Proceedings of the 29th ACM International Conference on Information and Knowledge Management. October 2020. P. 16451654.
DOI: 10.1145/3340531.3412038.
11. *Ai Q., Azizi V., Chen X., Zhang Y.* Learning heterogeneous knowledge base embeddings for explainable recommendation // Algorithms. 2018. Vol. 11. No. 9. Article 137.
DOI: 10.3390/a11090137.
12. *Zhang F., Yuan N.J., Lian D., Xie X., Ma W.-Y.* Collaborative knowledge base embedding for recommender systems // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 353–362.
13. *Li C., Xia L., Ren X., Ye Y., Xu Y., Huang C.* Graph Transformer for Recommendation // Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'23). 2023. P. 16801689.
DOI: 10.1145/3539618.3591723.
14. *Mancino A.C.M., Ferrara A., Bufi S., Malitesta D., Di Noia T., Di Sciascio E.* KGTORe: Tailored Recommendations through Knowledge-aware GNN Models // Proceedings of the 17th ACM Conference on Recommender Systems (RecSys'23). 2023. P. 576–587.
DOI: 10.1145/3604915.3608804.
15. *Wu J., Wang X., Feng F., He X., Chen L., Lian J., Xie X.* Self-supervised Graph Learning for Recommendation // Proceedings of the 44th International ACM SIGIR Conference. 2021. P. 726–735.
DOI: 10.1145/3404835.3462862.
16. *Yu J., Xia X., Chen T., Cui L., Hung N.Q.V., Yin H.* XSimGCL: Towards Extremely Simple Graph Contrastive Learning for Recommendation // IEEE Trans. on Knowl. and Data Eng. 2024. Vol. 36. No. 2. P. 913–926.
DOI: 10.1109/TKDE.2023.3288135.
17. *Jiang Y., Huang C., Huang L.* Adaptive Graph Contrastive Learning for Recommendation // Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'23). 2023. P. 4252–4261.
DOI: 10.1145/3580305.3599768.
18. *Wang H., Zhang F., Wang J., Zhao M., Li W., Xie X., Guo M.* RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems // Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM'18). 2018. P. 417426.
DOI: 10.1145/3269206.3271739.
19. *Xian Y., Fu Z., Muthukrishnan S., Gerard de Melo, Zhang Y.* Reinforcement Knowledge Graph Reasoning for Explainable Recommendation // Proceedings of the 42nd International ACM SIGIR Conference (SIGIR'19). 2019. P. 285–294.
DOI: 10.1145/3331184.3331203.
20. *Geng S., Fu Z., Tan J., Ge Y., Gerard de Melo, Zhang Y.* Path Language Modeling over Knowledge Graphs for Explainable Recommendation // Proceedings of the ACM Web Conference 2022 (WWW'22). 2022. P. 946–955.
DOI: 10.1145/3485447.3511937.
21. *Li M., Zeng Q., Lin Y., Cho K., Ji H., May J., Chambers N., Voss C.* Connecting the dots: Event graph schema induction with path language modeling // Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020. P. 684–695.
22. *Zhu Y., Xian Y., Fu Z., Gerard de Melo, Zhang Y.* Faithfully Explainable Recommendation via Neural Logic Reasoning // Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics. 2021. P. 3083–3090.
23. *Eytan L., Bogina V., Ben-Gal I., Koenigstein N.* KPAR: Knowledge-aware Path-based Attentive Recommender with Interpretability // ACM Trans. Recomm. Syst. 2025. Vol. 3. No. 3. Article 35. 23 p.
DOI: 10.1145/3673243.
24. *Yi Z., Ounis I.* A Unified Graph Transformer for Overcoming Isolations in Multi-modal Recommendation // Proceedings of the 18th ACM Conference on Recommender Systems (RecSys'24). 2024. P. 518–527.
DOI: 10.1145/3640457.3688096.
25. multilingual-e5-base (языковая модель).
https://huggingface.co/intfloat/https://huggingface.co/intfloat/ multilingual-e5-base (дата обращения: 20.09.2024)

26. *Julian McAuley.* Amazon Review Data (2018). https://cseweb.ucsd.edu/ jmcauley/https://cseweb.ucsd.edu/jmcauley/ datasets/amazon_v2/ (дата обращения: 02.02.2024)
27. *Mikolov T., Sutskever I., Chen K., Corrado G.S., Dean J.* Distributed representations of words and phrases and their compositionality // Advances in Neural Information Processing Systems. December 2013. Vol. 2. P. 3111–3119. DOI: 10.48550/arXiv.1310.4546.
28. Rectifier (neural networks). https: //en.wikipedia.org/wiki/Rectifier_//en.wikipedia.org/wiki/Rectifier_(neural_ networks) (дата обращения: 03.04.2024)
29. *Kool W., van Hooe H., Weleling M.* Stochastic Beams and Where To Find Them: The Gumbel-Top-k Trick for Sampling Sequences Without Replacement // Proceedings of the 36th International Cenference on Machine Learning (ICML'19). 2019. P. 3499–3508.
30. Gumbel distribution. https://enhttps://en. http://wikipedia.org/wiki/Gumbel_distributionwikipedia.org/wiki/Gumbel_distribution (дата обращения: 23.04.2024)
31. *Julian McAuley.* Amazon Beauty Dataset. https://mcauleylab.ucsd.edu/https://mcauleylab.ucsd.edu/ public_datasets/data/amazon_v2/categoryFilesSmall/All_Beauty_5.json.gz (дата обращения: 02.02.2024)

Сведения об авторах.

Татьяна Александровна Дмитриева – кандидат технических наук, доцент, Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Рязань, Российская Федерация;
e-mail: dmitrieva.t.a.@rsreu.ru;
https://orcid.org/0000-0002-3560-0482

Игорь Юрьевич Каширин – доктор технических наук, профессор, Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Рязань, Российская Федерация;
e-mail: igor_kashirin@mail.ru;
https://orcid.org/0000-0003-1694-7410

Геннадий Владимирович Овечкин – доктор технических наук, профессор, заведующий кафедрой, Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Рязань, Российская Федерация;
e-mail: ovechkin.g.v@rsreu.ru;
https://orcid.org/0000-0001-6887-2217

Денис Иванович Успенский – научный сотрудник, Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Рязань, Российская Федерация;
e-mail: denisuspenskiy@mail.ru;
https://orcid.org/0009-0009-2878-6459

Author information.

Tatiana A. Dmitrieva – Ph.D. (Tech.), Associate Professor, Ryazan State Radio Engineering University named after V.F. Utkin, Ryazan, Russian Federation;
e-mail: dmitrieva.t.a.@rsreu.ru;
https://orcid.org/0000-0002-3560-0482

Igor Yu. Kashirin – D.Sc. (Tech.), Professor, Ryazan State Radio Engineering University named after V.F. Utkin, Ryazan, Russian Federation;
e-mail: igor_kashirin@mail.ru;
https://orcid.org/0000-0003-1694-7410

Gennady V. Ovechkin – D.Sc. (Tech.), Professor, Head of Department, Ryazan State Radio Engineering University named after V.F. Utkin, Ryazan, Russian Federation;
e-mail: ovechkin.g.v@rsreu.ru;
https://orcid.org/0000-0001-6887-2217

Denis I. Uspensky – Researcher, Ryazan State Radio Engineering University named after V.F. Utkin, Ryazan, Russian Federation;
e-mail: denisuspenskiy@mail.ru;
https://orcid.org/0009-0009-2878-6459

Поступила в редакцию 08.08.2025 г.
После доработки 13.09.2025 г.
Принята к публикации 13.09.2025 г.

Received Augest 08, 2025
Revised September 13, 2025
Accepted Septemver 13, 2025