МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра компьютерных технологий и систем

МИШИНА Мария Витальевна

# АРХИТЕКТУРА И РАЗРАБОТКА РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ ДЛЯ СОЦИАЛЬНЫХ ПЛАТФОРМ НА ОСНОВЕ МАШИННОГО ОБУЧЕНИЯ

Дипломная работа

Научный руководитель:
доктор педагогических наук, кандидат физико-математических наук,
профессор В.В. Казаченок

Допущена к защите
«__» _____ 2025 г.
Заведующий кафедрой компьютерных технологий и систем
доктор педагогических наук, кандидат физико-математических наук,
профессор В.В. Казаченок

Минск, 2025

# ОГЛАВЛЕНИЕ

# РЕФЕРАТ

Дипломная работа, 55 страниц, 11 рисунков, 6 таблиц, 2 приложения, 14 источников.

**Ключевые слова:** РЕКОМЕНДАТЕЛЬНЫЕ СИСТЕМЫ, МАШИННОЕ ОБУЧЕНИЕ, ГИБРИДНЫЕ АЛГОРИТМЫ, ПРОБЛЕМА ХОЛОДНОГО СТАРТА, ВЕКТОРНЫЕ ПРЕДСТАВЛЕНИЯ, МАТРИЧНАЯ ФАКТОРИЗАЦИЯ, ТАРГЕТИНГ.

**Объект исследования:** алгоритмы машинного обучения, архитектура рекомендательных систем.

**Цели работы:** создание высокопроизводительной рекомендательной системы, обеспечивающей точную персонализацию при обработке значительных массивов пользовательских данных.

**Методы исследования:** анализ существующих подходов к построению рекомендательных систем, разработка гибридной модели, тестирование с использованием стандартных метрик, экспериментальная оценка эффективности через A/B-тестирование.

**Результаты работы:** разработана готовая к внедрению рекомендательная система, которая эффективно решает проблему холодного старта, обеспечивает значительное повышение точности рекомендаций и реализует сбалансированный механизм распределения контента с полным набором интерфейсов для интеграции в существующие платформы. Система демонстрирует устойчивую работу в промышленных условиях и готова к масштабированию.

**Область применения результатов:** профессиональные социальные сети, корпоративные медиа-платформы, экспертные сообщества, образовательные ресурсы.

# РЭФЕРАТ

Дыпломная праца, 55 старонак, 11 малюнкаў, 6 табліц, 2 дадаткі, 14 крыніц.

**Ключавыя словы:** РЭКАМЕНДАЦЫЙНЫЯ СІСТЭМЫ, МАШЫННАЕ НАВУЧАННЕ, ГІБРЫДНЫЯ АЛГАРЫТМЫ, ПРАБЛЕМА ХАЛОДНАГА СТАРТУ, ВЕКТАРНЫЯ ПРАДСТАЎЛЕННІ, МАТРЫЧНАЯ ФАКТАРЫЗАЦЫЯ, ТАРГЕТЫНГ.

**Аб'ект даследавання:** алгарытмы машыннага навучання, архітэктура рэкамендацыйных сістэм.

**Мэта працы:** стварэнне высокапрадукцыйнай рэкамендацыйнай сістэмы, якая забяспечвае дакладную персаналізацыю пры апрацоўцы значных масіваў карыстацкіх даных.

**Метады даследавання:** аналіз існуючых падыходаў да пабудовы рэкамендацыйных сістэм, распрацоўка гібрыднай мадэлі, тэставанне з выкарыстаннем стандартных метрык, эксперыментальная ацэнка эфектыўнасці праз A/B-тэставанне.

**Вынікі працы:** распрацавана гатовая да ўкаранення рэкамендацыйная сістэма, якая эфектыўна вырашае праблему халоднага старту, забяспечвае значнае павышэнне дакладнасці рэкамендацый і рэалізуе збалансаваны механізм размеркавання кантэнту з поўным наборам інтэрфейсаў для інтэграцыі ў існуючыя платформы. Сістэма дэманструе ўстойлівую працу ў прамысловых умовах і гатовая да маштабавання.

**Вобласць прымянення вынікаў:** прафесійныя сацыяльныя сеткі, карпаратыўныя медыя-платформы, экспертныя супольнасці, адукацыйныя рэсурсы.

# ABSTRACT

Diploma thesis, 55 pages, 11 figures, 6 tables, 2 attachments, 14 sources.

**Keywords:** RECOMMENDER SYSTEMS, MACHINE LEARNING, HYBRID ALGORITHMS, COLD START PROBLEM, VECTOR REPRESENTATIONS, MATRIX FACTORIZATION, TARGETING.

**Object of research:** machine learning algorithms, recommender systems architecture.

**Objective:** development of a high-performance recommender system providing accurate personalization when processing significant volumes of user data.

**Research methods:** analysis of existing approaches to building recommender systems, development of a hybrid model, testing using standard metrics, experimental efficiency evaluation through A/B testing.

**Results:** a production-ready recommender system was developed that effectively solves the cold start problem, provides significant improvement in recommendation accuracy, and implements a balanced content distribution mechanism with a complete set of integration interfaces for existing platforms. The system demonstrates stable operation in industrial environments and is ready for scaling.

**Scope:** professional social networks, corporate media platforms, expert communities, educational resources.

# ВВЕДЕНИЕ

В условиях стремительного роста объемов пользовательского контента на социальных платформах возрастает необходимость в системах, способных эффективно управлять информацией и предоставлять пользователям персонализированные рекомендации. Согласно отчетам 1.6], в 2023 году совокупный объем данных, генерируемых пользователями на онлайн-платформах, достигал нескольких петабайт ежедневно, и подобная динамика сохраняется. Это приводит к проблеме «информационной перегрузки», когда пользователи оказываются не способны обрабатывать и сортировать массивы данных самостоятельно.

Рекомендательная система (РС) – это инструмент, использующий алгоритмы и методы анализа данных для предоставления пользователям персонализированных рекомендаций на основе их предпочтений и поведения. Эффективность РС подтверждена многочисленными исследованиями и примерами использования на крупных платформах. Согласно исследованию Google 1.14], рекомендации, основанные на методах машинного обучения, увеличили количество просмотров видеоконтента на YouTube более чем на 70%. Исследования Netflix 1.6] показывают, что рекомендации снижают отток пользователей на 30%, а персонализированные списки просмотра являются одним из основных факторов удержания аудитории. Несмотря на существующие достижения, многие социальные платформы по-прежнему сталкиваются с проблемами неэффективности и нерелевантности рекомендаций.

Цель работы – разработка архитектуры рекомендательной системы, которая с использованием методов машинного обучения способна обрабатывать большие объемы данных и формировать высокоточные персонализированные рекомендации. В работе рассматривается реализация гибридной модели, сочетающей преимущества трансформерных архитектур и методов матричной факторизации, интегрированной с механизмом мультипараметрического таргетинга. Применяется комплекс методов, включающий теоретический анализ современных алгоритмов рекомендательных систем, экспериментальную проверку гипотез с использованием A/B-тестирования, а также практическую реализацию на базе современных технологий машинного обучения и обработки больших данных. Практическая значимость исследования заключается в возможности непосредственного внедрения разработанных решений в существующие социальные платформы.

# ГЛАВА 1
# АКТУАЛЬНОСТЬ И ОБЗОР МЕТОДОВ

## 1.1 Актуальность и проблемы в разработке рекомендательных систем

Современные цифровые платформы активно используют рекомендательные системы, чтобы предлагать пользователям персонализированный контент, товары или услуги. Одна из основных проблем заключается в отсутствии универсального решения, подходящего для всех типов данных бизнес-задач. В большинстве случаев для достижения оптимальных результатов требуется комбинирование нескольких методов. При этом сложно сказать, как сочетать их наиболее эффективно в конкретной ситуации.

Одной из основных проблем является проблема холодного старта. Проблема холодного старта – это ситуация, когда система рекомендаций не может предоставить точные рекомендации из-за недостатка данных о новых пользователях или новых товарах/контенте.

Проблема разреженности данных возникает из-за того, что чаще всего пользователи оценивают только малую часть товаров или контента. В результате большинство взаимодействий "пользователь-объект" остаются незафиксированными, формируется разреженная матрица предпочтений. Точность алгоритмов коллаборативной фильтрации снижается из-за недостаточного количества пересечений в пользовательских предпочтениях. Система начинает учитывать случайные совпадения вместо реальных закономерностей.

С ростом числа пользователей или товаров системы сталкиваются с трудностями при обработке больших объемов данных в реальном времени. Существующие методы оптимизации не всегда могут гарантировать необходимую производительность при крайне высоких нагрузках.

Проблема диверсификации рекомендаций выражается в недостаточной вариативности предложений. Это ситуация, когда в рекомендации попадают только те товары, которые максимально совпадают с предыдущими предпочтениями пользователя. Пользователю предлагаются однотипные товары или материалы, что со временем снижает его удовлетворенность.

## 1.2 Типы систем рекомендаций

Коллаборативная фильтрация работает по принципу выявления закономерностей. Если группа пользователей проявляла сходные предпочтения в прошлом, их поведение с высокой вероятностью будет коррелировать в будущем.

Контентная фильтрация основывается на анализе характеристик объектов и профилей пользователей. Она актуальна в случаях, когда данные о других пользователях ограничены или недоступны, так как не требует сравнения интересов множества пользователей и вообще не учитывает поведение других участников системы. Предпочтения пользователей выявляются через его взаимодействия или явные отклики на контент. Алгоритмы не зависят от наличия исторических данных и, как следствие, дают возможность предлагать новые объекты сразу после их добавления в систему только на основе их описаний.

Знание-ориентированные рекомендательные системы эффективны в предметных областях, где требуется строгая формализация процессов. Такие системы обычно применяются в сложных профессиональных средах для задач, которые требуют строгого соблюдения нормативных требований и отраслевых стандартов. Знание-ориентированные подходы обеспечивают прозрачность и объяснимость принимаемых решений. Это важно для сфер, где ошибки могут иметь серьезные последствия.

Гибридные рекомендательные системы комбинируют преимущества нескольких подходов и минимизируют ограничения отдельных методов.

### 1.1.1 Коллаборативная фильтрация

Коллаборативная фильтрация анализирует действия пользователей с целью найти сходства между их предпочтениями. В контексте коллаборативной фильтрации используются явные и неявные данные. Явные данные предоставляют точную информацию о предпочтениях пользователей, но могут страдать от низкой плотности, поскольку не все пользователи оценивают объекты. Неявные данные, например, информация о покупках, просмотрах или времени взаимодействия с контентом, собираются в большем объеме. Они позволяют моделировать предпочтения в системах с высокой активностью пользователей. Неявные данные могут быть шумными и не всегда прямо отражать интересы пользователей, однако исследования 1.13] показывают, что использование неявных данных значительно улучшает точность рекомендаций.

Коллаборативная фильтрация делится на три основных типа: подходы, основанные на соседстве, на модели и гибридные методы.

Методы на основе соседства основываются на том, что предпочтения пользователя можно предсказать исходя из предпочтений пользователей с похожими интересами. Для повышения точности выбора соседей применяются пороговые значения сходства, ограничение числа соседей и, в крупных системах, item-based фильтрация. Рекомендации строятся на основе сходства между объектами, которые пользователь оценил ранее.

Методы на основе моделей ориентированы на построение прогностических моделей, которые обучаются обнаруживать скрытые паттерны предпочтений пользователей и свойства объектов. Они могут с высокой точностью предсказывать потенциально интересные пользователю объекты.

Для анализа пользовательских предпочтений применяется матричная факторизация – метод, декомпозирующий исходную матрицу взаимодействий на произведения матриц меньшей размерности. Классическое сингулярное разложение матриц (SVD) и его модификации (например, FunkSVD) позволяют снижать размерность данных и ускорять обновления при поступлении новых данных. Байесовские вероятностные модели и методы кластеризации повышают точность рекомендаций благодаря учету вероятностных распределений и естественной группировки пользователей. Вариационные автоэнкодеры и трансформеры выявляют латентные зависимости в предпочтениях пользователей и учитывают дополнительные контентные признаки объектов. Тензорные разложения расширяют классическую матричную факторизацию для работы с многомерными структурами данных. Методы кластеризации существенно снижают вычислительную сложность и помогают эффективно решать проблему разреженности данных.

Коллаборативная фильтрация сталкивается с рядом ограничений. Разреженность данных возникает вследствие того, что пользователи взаимодействуют с малой частью доступных объектов. Для решения применяются методы, ориентированные на извлечение скрытых зависимостей. Проблема холодного старта решается через интеграцию гибридных моделей. Традиционные методы плохо работают на больших данных. Вычислительная сложность многих алгоритмов растет непропорционально быстро с увеличением числа пользователей и объектов в системе 1.3].

Коллаборативная фильтрация остается одной из самых популярных и эффективных техник для создания рекомендательных систем, в том числе в

контексте социальных сетей. Использование эмбеддингов контента в сочетании с пользовательскими предпочтениями может компенсировать недостаток данных для новых пользователей или элементов и повысить точность рекомендаций.

### 1.1.2 Контентная фильтрация

Контентные рекомендации основаны на использовании характеристик объектов для формирования рекомендаций, которые соответствуют предпочтениям пользователей. Современные методы контентной фильтрации активно используют факторизационные машины (FM) 1.3] и их расширенные версии, такие как Field-aware Factorization Machines (FFM) 1.10]. Они способны эффективно обрабатывать векторные представления признаков и выявлять скрытые зависимости в данных. Среди актуальных подходов выделяется использование глубокой факторизации. DeepFM 1.10] сочетает традиционные методы факторизационных машин с нейронными сетями для более сложной обработки признаков. В отличие от классических FM, которые используют фиксированные признаки, DeepFM способен автоматически извлекать скрытые зависимости и обучать систему на более высокоуровневых признаках. Это значительно улучшает результаты в задачах с большими и сложными наборами данных.

Современные достижения в области контентной фильтрации во многом связаны с внедрением архитектур трансформерного типа. В частности, моделей BERT и DSSM. Эти модели специализируются на анализе контекстных зависимостей и обработке текстовых данных, что делает их особенно полезными для рекомендательных систем, работающих с текстами. BERT эффективно справляется с пониманием контекста слов и фраз 1.8]. DSSM используется для вычисления семантической схожести между запросами и документами.

Контентная фильтрация полностью зависит от качества и полноты метаданных. Система не сможет строить точные рекомендации, если характеристики объектов описаны неполно. Также классическая контентная фильтрация неспособна учитывать контекст и ситуативные факторы.

Для того, чтобы сделать контентные фильтры более чувствительными к нюансам языка, все чаще применяются модели, ориентированные на контекстуальный анализ (например, BERT). Они способны учитывать не только явные признаки объектов, но и их глубокие семантические связи. Это позволяет избегать ошибок, связанных с неоднозначностью значений слов в различных контекстах. Модели на основе эмбеддингов создают плотные

векторные представления слов и фраз. Такие модели позволяют эффективнее учитывать смысловые связи между словами, даже если они не встречаются в одном контексте или не используются вместе в явной форме.

Системы, основанные исключительно на контентной фильтрации, ограниченны в обработке изображений, видео и аудио. Несмотря на то, что существуют подходы для извлечения признаков из мультимедийных данных (например, в музыке с использованием глубоких нейронных сетей), большинство контентных систем остаются ориентированными в основном на текстовые данные.

Контентная фильтрация также сталкивается с проблемой избыточной специализации. Чем точнее система подбирает контент по формальным критериям, тем уже становится круг рекомендаций. В музыкальных сервисах это проявляется в бесконечных вариациях похожих треков, в новостных лентах – в потоке материалов на одну и ту же тему. В результате рекомендации становятся предсказуемыми и однообразными.

### 1.1.3 Фильтрация, основанная на знаниях

Системы рекомендаций, основанные на знаниях, используют явно заданные онтологии, правила вывода и семантические модели предметной области для генерации рекомендаций 1.5]. Они могут эффективно работать в условиях ограниченной информации, и, как следствие, хорошо подходят для новых пользователей или товаров и услуг, для которых невозможно собрать большое количество пользовательских оценок.

Одним из наиболее распространенных подходов является метод ограничений, который использует заранее определенные правила, задаваемые экспертами или системой. Правила описывают требования к объектам, которые должны быть рекомендованы.

Другим распространенным подходом является кейс-ориентированная система рекомендаций, где решения принимаются на основе метрик сходства между запросами пользователя и атрибутами объектов.

Для упрощения взаимодействия с пользователем многие системы используют значения по умолчанию, которые помогают быстрее определить начальные параметры для рекомендаций. В социальных сетях система может предложить пользователю стандартные параметры для фильтрации контента (популярность, актуальность) на основании данных о предыдущих взаимодействиях. В случае, если предложенные рекомендации не соответствуют заданным ограничениям, применяются алгоритмы ослабления ограничений. Существуют более сложные подходы (например, методы

критики), которые позволяют пользователю уточнять свои предпочтения, не изменяя при этом запрос полностью. Пользователь может получить рекомендации по контенту, а затем изменить запрос, уточнив критерии.

Несмотря на значимость в области обработки информации, в контексте социальных сетей знание-ориентированные системы имеют ограниченное применение в силу их сложности и зависимости от формализованных моделей знаний. Основная проблема заключается в их жесткой зависимости от заранее формализованных правил, которые требуются постоянно поддерживать в актуальном состоянии. В динамичной среде социальных медиа такие системы не успевают адаптироваться к изменениям.

## 1.3 Инструменты для построения рекомендательных систем

Python остается ведущим языком программирования для разработки рекомендательных систем. Его популярность обусловлена богатой экосистемой специализированных библиотек. Для обработки сырых данных и их анализа используют Pandas (работа с таблицами) и NumPy (векторные операции). Для больших данных используют PySpark.

LightFM – одна из наиболее популярных библиотек для гибридных рекомендаций. Она объединяет коллаборативную фильтрацию и методы контентного анализа. Библиотека Surprise предоставляет реализацию классических методов матричной факторизации (SVD, SVD++) для быстрого создания и оценки базовых моделей рекомендаций. Sentence-Transformers ориентирована на преобразование текстов в векторные представления и активно используется в контентной фильтрации для задач рекомендаций. Она основана на трансформерах и позволяет строить эмбеддинги текста. Затем эти эмбеддинги могут быть использованы для сравнения и поиска схожих объектов.

Для более сложных решений выбирают TensorFlow Recommenders (TFRS), который интегрируется с фреймворком TensorFlow. Это инструменты для создания глубоких рекомендательных систем с использованием методов обучения с подкреплением и других подходов, ориентированных на большие объемы данных. FastFM специализируется на реализации методов факторизации матриц. Для работы с графами активно используется библиотека LightGCN. В ней для изучения сложных взаимодействий между пользователями и контентом применяются графовые нейронные сети.

В коммерческих приложениях Python может использоваться для создания микросервисов на легковесных фреймфорках (FastAPI, AioHTTP,

Flask, Sanic) с API рекомендаций для интеграции рекомендательных алгоритмов в более крупные системы. Главное конкурентное преимущество Python – это баланс между скоростью разработки и производительностью.

## 1.4 Метрики для оценки рекомендательных систем

Одной из основных метрик для оценки рекомендательных систем является точность (англ. accuracy) – метрика, показывающая долю релевантных объектов среди всех рекомендованных системой. Показатели RMSE, MAE и Precision@k позволяют оценить, насколько точно система рекомендует элементы 1.9]. Высокая точность не всегда свидетельствует о высоком уровне удовлетворенности пользователя, особенно если рекомендации слишком предсказуемы и не обеспечивают разнообразия.

Полнота (англ. recall) измеряет долю рекомендованных объектов среди всех доступных объектов. В социальных сетях высокая полнота означает, что алгоритм не зацикливается на популярном контенте и предлагает также нишевые и менее распространенные материалы. Увеличение полноты может привести к снижению точности, так как система будет предлагать более разнообразные, но менее релевантные объекты.

Новизна (англ. novelty) характеризует степень неожиданности и неизвестности предлагаемого контента для пользователя. позволяет оценить вероятность того, что система порекомендует контент, который является редким для пользователя. Метрика оценивает, насколько рекомендации отличаются от того, что пользователь уже видел или с чем взаимодействовал ранее. Она особенно актуальна в системах с высоким уровнем насыщенности контента (новостные ленты, стриминговые сервисы). В отличие от полноты, новизна поощряет рекомендации объектов, которые имеют более низкую вероятность быть уже известными пользователю.

Разнообразие (англ. diversity) отражает степень отличия рекомендуемых объектов друг от друга по ключевым характеристикам. Характеристика используется для предотвращения эффект пресыщения, когда однотипные предложения вызывают у пользователя усталость и снижение интереса к сервису.

Серендипность (англ. serendipity) – это способность рекомендательной системы предлагать пользователю неожиданный, но интересный контент. Для ее оценки часто используют разницу между прогнозами персонализированной модели и примитивной модели. Слишком высокий уровень серендипности может снизить доверие к системе. Оптимальным показателем считается 15-20% таких рекомендаций в ленте 1.5].

Ретенция (англ. retention) измеряет способность системы удерживать пользователей в долгосрочной перспективе. На практике ретенцию измеряют через регулярность возврата, глубину просмотра и частоту взаимодействий. Оптимальные рекомендательные алгоритмы демонстрируют стабильный прирост retention rate на 3-5% в квартал 1.9].

Метрика вовлеченности (англ. engagement) отражает глубину взаимодействия пользователя с платформой. Она учитывает как долго пользователь остается в системе, какие действия совершает помимо просмотра, возвращается ли к рекомендованному контенту повторно.

Для промышленной рекомендательной системы критичны операционные метрики: время отклика (англ. latency), пропускная способность (англ. throughput), частота ошибок (англ. error rate), загрузка CPU и оперативной памяти.

# ГЛАВА 2
# ТЕОРЕТИЧЕСКАЯ РАЗРАБОТКА СИСТЕМЫ ПЕРСОНАЛИЗИРОВАННЫХ РЕКОМЕНДАЦИЙ

## 2.1 Формализация требований к данным и их представление

Для реализации рекомендательной системы необходимо формализовать требования к данным, которые будут использованы для обучения модели. На этом этапе важно не только определить источники и структуру данных, но и обосновать выбор методов их хранения и обработки. В рамках проектируемой системы предполагается, что данные поступают из нескольких источников, хранятся в структурированных таблицах реляционной базы данных и проходят предварительную обработку перед созданием числовых векторов – эмбеддингов – для оценки взаимодействий пользователей с контентом.

Данные для системы можно условно разделить на три категории. Пользовательские взаимодействия фиксируются в реальном времени через систему событий. Лайки, репосты, комментарии, отметки “неинтересно” и другие взаимодействия поступают через API. Эта информация является основным источником для оценки вовлеченности пользователей. Тексты публикаций и их метаданные извлекаются из контентной базы данных. Данные профилей пользователей содержат текстовые описания и характеристики, отражающие их предпочтения и поведение.

В системе используется комбинация структурированных и неструктурированных данных. Структурированные данные – взаимодействия пользователей с контентом – представлены в таблице interaction_scores. Таблица содержит бинарные флаги для каждого типа взаимодействия (лайк, комментарий, репост и т.д.), а также вычисляемые поля, которые агрегируют различные типы взаимодействий. Эти данные затем используются для построения матрицы взаимодействий, в которой отражается степень вовлеченности пользователя с публикациями. Неструктурированные данные, такие как текстовая информация, связанная с пользователями и публикациями, загружаются из внешних систем управления контентом. Эти данные обрабатываются с использованием алгоритмов машинного обучения для извлечения признаков из текстов и преобразования их в числовые векторы. Эмбеддинги хранятся в таблицах account_embeddings и publication_embeddings и используются для вычисления схожести между пользователями и публикациями. Связи между таблицами представлены на рисунке 2.1.

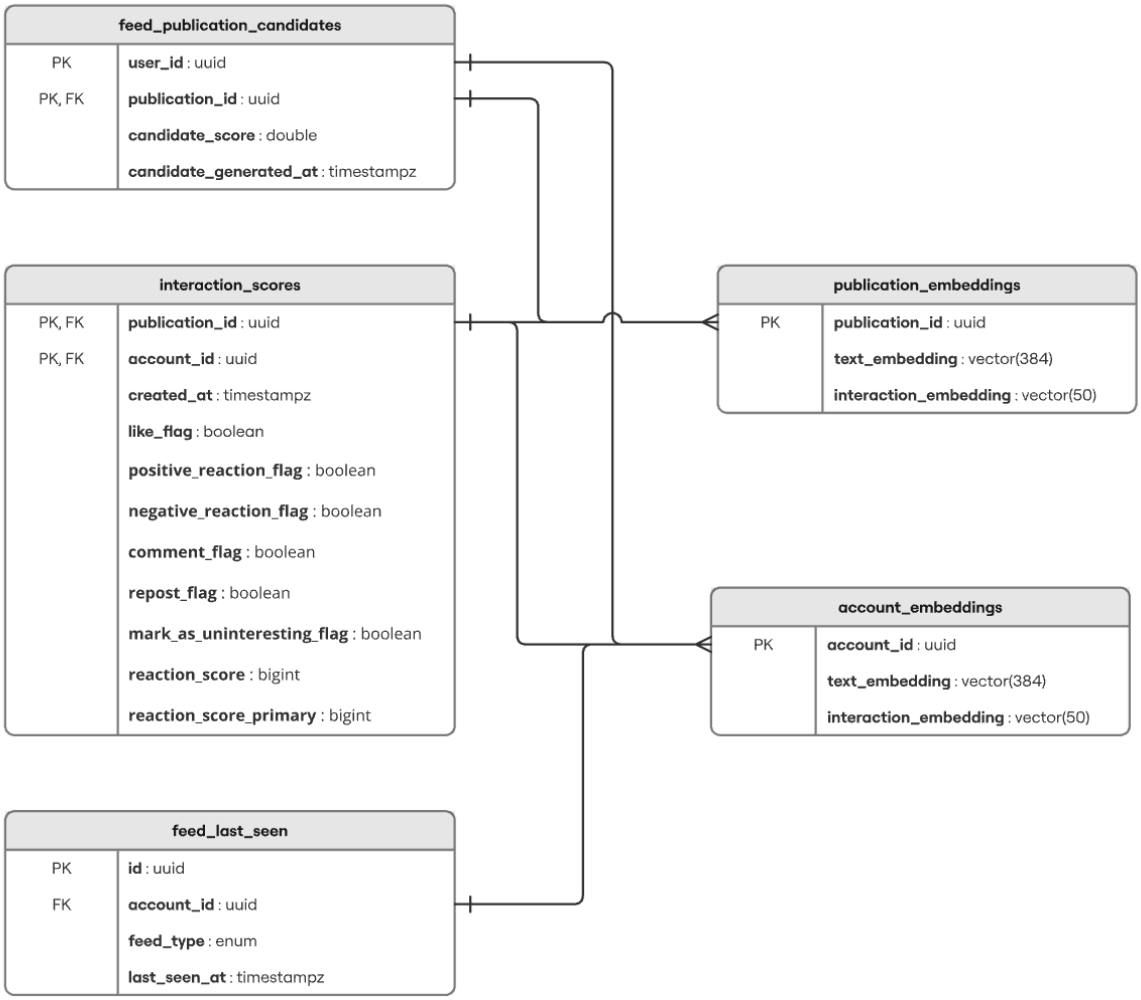


**Рисунок 2.1 – ER-диаграмма пользовательских взаимодействий и контентных эмбеддингов**

Матрица взаимодействий – один из основных элементов в традиционных рекомендательных системах. Она представляет собой таблицу, где строки соответствуют пользователям, а столбцы – публикациям. Каждый элемент матрицы отражает степень взаимодействия между пользователем и публикацией. В данном случае взаимодействия фиксируются через бинарные флаги, и каждому типу взаимодействия присваивается вес. Например, за лайк начисляется 5 баллов, за комментарий – 10, а за репост – 20. Баллы суммируются в поле reaction_score, чтобы позволить системе учитывать как явные, так и скрытые предпочтения пользователей.

Алгоритм учитывает временную актуальность данных. В таблице interaction_scores фиксируется время каждого взаимодействия через поле creation_time, чтобы учитывать свежесть данных при формировании рекомендаций. Это важно, поскольку предпочтения пользователей могут изменяться со временем. Для поддержания актуальности рекомендаций

используется механизм обработки исторических данных backfill, который обновляет данные с определенной временной точки, обеспечивая динамическую адаптацию ленты пользователя.

Реляционные базы данных были выбраны для хранения данных в силу их способности к быстрому выполнению запросов и эффективному управлению связями между различными сущностями. Использование эмбеддингов для представления текстов и взаимодействий мотивировано их относительной компактностью и способностью передавать семантику.

## 2.2 Применение трансформеров BERT для контентной фильтрации и построения векторных представлений

Для решения задачи создания эмбеддингов текстов в рамках разработки системы был выбран подход, основанный на модели BERT (Bidirectional Encoder Representations from Transformers) 1.7]. Подход относится к контентной фильтрации, основная цель заключается в анализе и создании эмбеддингов текстового контента. Это позволяет системе делать более точные рекомендации на основе содержимого публикаций, описаний аккаунтов и других текстовых данных. Выбор модели BERT обусловлен ее способностью учитывать контекст каждого слова, анализируя его значение в зависимости от слов, расположенных как слева, так и справа от него в предложении, что значительно повышает качество семантического представления. В отличие от более традиционных моделей, которые обучаются на локальных контекстах и игнорируют более широкий контекст в предложении, BERT использует двустороннее внимание, что позволяет учитывать весь контекст слова в предложении одновременно, как слева, так и справа от него 1.7].

Векторные представления текстов становятся основой для дальнейшей работы системы, позволяя эффективно анализировать и сравнивать тексты в высокоразмерных пространствах. Основной задачей является построение эмбеддингов, которые максимально точно отражают семантическое содержание текста, а также создание методов для их агрегации с учетом различных факторов, таких как веса, основанные на реакции пользователя на публикации.

Преобразование текста в эмбеддинг, осуществляемое моделью Sentence-BERT, представляет собой задачу сопоставления текста с векторным пространством, в котором текст $T = \{t_1, t_2, \ldots, t_n\}$ из n-словных последовательностей сопоставляется с вектором $v_T \in R^d$, где d – размерность эмбеддинга. Sentence-BERT использует архитектуру трансформеров с механизмами внимания, что позволяет учесть контекст

каждого слова относительно других слов в пределах всего текста. Результат работы трансформера можно записать как $v_T = BERT(T)$, где BERT(T) обозначает результат применения модели BERT к тексту T, а вектор $v_T$ – выход этой модели.

При агрегации текстовых эмбеддингов используется взвешенная сумма с динамическими весами. Веса вычисляются как нелинейная функция от пользовательских взаимодействий (клики, длительность). Пусть $v_T \in R^d$ – это эмбеддинг i-го текста, а $w_i \in R$ – вес, отражающий силу реакции пользователя на соответствующий текст. Для получения агрегации эмбеддингов нескольких текстов используется взвешенное среднее, которое вычисляется по формуле 2.1.

$$v_{agg} = \frac{\sum_{i=1}^{n} w_{i} * v_i}{\sum_{i=1}^{n} |w_i|} \tag{2.1}$$

Нормализация предотвращает доминирование отдельных текстов с большими весами.

При необходимости комбинировать эмбеддинги, относящиеся к различным типам объектов (например, к публикациям и описаниям аккаунтов), применяются аналогичные методы взвешенной агрегации, которые учитывают важность каждого типа. Пусть $v_{desc} \in R^d$ и $v_{pub} \in R^d$ – эмбеддинги для описания аккаунта и публикации, соответственно, а $w_{desc}$ и $w_{pub}$ – соответствующие веса. Итоговый комбинированный эмбеддинг вычисляется с использованием формулы взвешенного среднего 2.2.

$$v_{combined} = \frac{w_{desc} v_{desc} + w_{pub} v_{pub}}{w_{desc} + w_{pub}} \tag{2.2}$$

В формуле $v_{combined}$ является итоговым вектором, который учитывает как описание аккаунта, так и его публикации. Такой подход в дальнейшем помогает системе точнее учитывать контекст.

В случае отсутствия эмбеддинга для одного из текстов или типов объектов (например, если текст или реакция отсутствуют), применяется техника замещения отсутствующих данных. В таких случаях для эмбеддингов, которые не могут быть получены, используется нулевой вектор.

Процесс обучения модели Sentence-BERT заключается в минимизации функции потерь, которая описывает отклонение между предсказанным эмбеддингом $v_{pred}$ и истинным эмбеддингом $v_{true}$. Одним из распространенных вариантов функции потерь является функция, измеряющая евклидово расстояние между векторами

$$L(v_{pred}, v_{true}) = \frac{1}{2}||v_{pred} - v_{true}||^2, \quad (2.3)$$

где $||.||$ – это евклидова норма, которая измеряет расстояние между предсказанным и истинным векторами. Минимизация этой функции позволяет Sentence-BERT обучаться так, чтобы эмбеддинги максимально точно отражали семантическое содержание текста.

Математические методы, лежащие в основе работы системы генерации и агрегации эмбеддингов, включают использование трансформеров для представления текста в векторном пространстве, агрегацию векторов с учетом различных весов, а также оптимизацию модели с использованием функции потерь, основанной на евклидовой норме. Эти методы позволяют системе эффективно обрабатывать и анализировать текстовую информацию и генерировать эмбеддинги, которые точно отражают семантику текста.

## 2.3 Использование Alternating Least Squares для коллаборативной фильтрации

Алгоритм Alternating Least Squares (ALS) – это метод коллаборативной фильтрации, основанный на разложении матрицы взаимодействий между пользователями и элементами на две матрицы меньшей размерности. В ходе работы алгоритм поочередно обновляет эти матрицы, минимизируя ошибку предсказания взаимодействий с использованием метода наименьших квадратов.

Подход применим в случаях, когда необходимо извлечь скрытые закономерности из взаимодействий пользователей с контентом (публикации, продукты или услуги). В рекомендательных системах, как правило, вводится матрица R, где строки соответствуют пользователям, а столбцы – публикациям или другим элементам контента. Каждый элемент этой матрицы отражает степень взаимодействия пользователя с публикацией (лайк, комментарий или просмотр). Задача заключается в том, чтобы на основе существующих взаимодействий предсказать, какие публикации будут интересны пользователю в будущем. Алгоритм ALS решает эту задачу путем разложения исходной матрицы взаимодействий на две матрицы меньшей

размерности: матрицу факторов для пользователей U и матрицу факторов для публикаций V, которые вместе аппроксимируют исходную матрицу взаимодействий.

Математически задача оптимизации для ALS сводится к минимизации ошибки предсказания между наблюдаемыми значениями взаимодействий и аппроксимированными значениями. Ошибка описывается через функцию потерь, которая включает не только разницу между предсказанным значением и реальным значением, но и регуляризацию, направленную на предотвращение переобучения модели. В процессе оптимизации алгоритм корректирует матрицы латентных факторов пользователей и объектов и постепенно минимизирует общую функцию потерь. Формально задача минимизации может быть записана как:

$$L(U,V) = \sum_{i,j\in\Omega}(R_{ij} - U_i * V_j^T)^2 + \lambda(||U||_F^2 + ||V||_F^2), \qquad (2.4)$$

где $R_{ij}$ – значение взаимодействия пользователя i с публикацией j,
$U_i$ и $V_j$ – векторы факторов для пользователя и публикации соответственно,
$\lambda$ – параметр регуляризации,
$||.||_F$ – фробениусова норма, которая служит для регуляризации параметров.

Процесс работы ALS заключается в чередовании двух шагов. На первом шаге фиксируется матрица V, и происходит оптимизация матрицы факторов для пользователей U. На втором шаге фиксируется матрица U, и происходит оптимизация матрицы факторов для публикаций V. Шаги повторяются до тех пор, пока ошибка не стабилизируется. Применение регуляризации на каждом шаге обучения снижает риск переобучения, особенно в случаях, когда данные о взаимодействиях ограничены или содержат шум.

## 2.4 Проблема холодного старта

Для решения проблемы холодного старта в условиях недостаточных данных рекомендательная система обрабатывает два основных случая: новые публикации, еще не набравшие взаимодействий, и новые пользователи без поведенческой информации.

Для пользователей, не имеющих взаимодействий в системе, используется метод на основе ALS. Он позволяет строить эмбеддинги пользователей, даже если эти пользователи только начинают

взаимодействовать с системой. Подход создает векторы на основе взаимодействий с похожими пользователями, что исключает необходимость пересчета всех векторов в реальном времени. Вектор для нового пользователя строится на основе усредненных векторов пользователей с аналогичной активностью или интересами. Это позволяет системе предложить персонализированные рекомендации даже без предварительного взаимодействия с контентом.

Для публикаций, не имеющих откликов, применяется метод, основанный на BERT с генерацией эмбеддингов текстов публикаций. Даже если публикация не имеет взаимодействий или откликов, она будет иметь вектор. Вектор получается на основе заголовка и основного текста публикации. В случае отсутствия этих элементов (например, для публикаций только с изображениями), система использует текст из описания изображения, сообщества или автора.

Предложенный гибридный подход сочетает ALS-оптимизацию для обработки новых пользователей и BERT-эмбеддинги для анализа нового контента для решения проблемы холодного старта. ALS обрабатывает пользователей без истории взаимодействий через латентные факторы, BERT анализирует семантику нового контента. Подход позволяет системе сразу после появления новых пользователей или публикаций генерировать релевантные рекомендации.

## 2.5 Концептуальная модель и архитектура системы рекомендаций

Архитектура системы разделена на онлайн- и оффлайн-компоненты для оптимизации производительности. Такой подход минимизирует задержки при формировании рекомендаций в реальном времени за счет переноса ресурсоемких вычислений в оффлайн-режим. На рисунке 2.2 проиллюстрирована организация взаимодействия между этими компонентами.

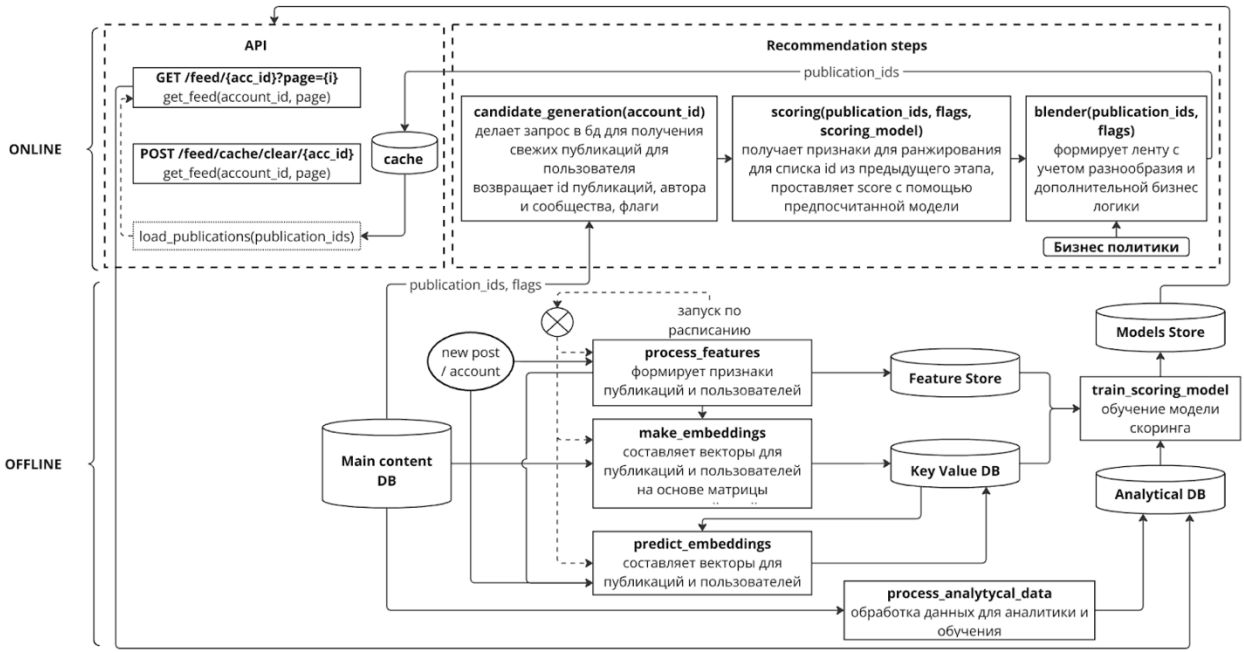


**Рисунок 2.2 – Разделение системы на онлайн- и оффлайн-компоненты**

Онлайн-компонент системы предназначен для генерации рекомендаций в реальном времени. Процесс начинается с этапа формирования списка кандидатов. Система извлекает из базы данных публикации, которые могут быть потенциально интересны пользователю. Проводится предварительная фильтрация контента, чтобы исключить публикации из заблокированных сообществ, скрытые или удаленные материалы, а также те, с которыми пользователь уже взаимодействовал. Для решения проблемы холодного старта, возникающей у новых пользователей или пользователей с ограниченной историей взаимодействий, используется подход на основе векторных представлений, которые формируются с помощью алгоритма ALS.

После предварительной фильтрации отобранные публикации поступают на следующий этап: оценку. Контент обогащается дополнительными признаками, которые были предварительно сформированы на оффлайн-этапе. Для извлечения текстовой информации из публикаций используется модель BERT. Она генерирует контекстно-зависимые векторные представления, которые учитывают семантические особенности текста. Эти представления, наряду с другими признаками, такими как реакции пользователей или временные характеристики, подаются на вход модели оценки. Модель оценки присваивает каждой публикации численную оценку, которая отражает ее релевантность для конкретного пользователя. Полученные оценки затем используются для ранжирования публикаций, то есть для сортировки контента по степени его значимости.

Полная формула скоринга:

$$score = (1 + \frac{r}{10})(20cos(e_{pi}, e_{ai}) + 30cos(e_{pt}, e_{at}) + 10\frac{l}{v} + 10\frac{c}{v} + 20(\frac{1}{a+1})^2), \quad (2.4)$$

где r – коэффициент бустинга (от 0 до 10),
$e_{pi}$ – вектор-эмбеддинг публикации, полученный из ALS,
$e_{ai}$ – вектор-эмбеддинг пользователя, полученный из ALS,
$e_{pt}$ – вектор-эмбеддинг публикации, полученный из BERT,
$e_{at}$ – вектор-эмбеддинг пользователя, полученный путем взвешенного усреднения векторов-эмбеддингов публикаций полученных из BERT,
$l$ – количество реакций у публикации,
$v$ – количество просмотров публикации,
$c$ – количество комментариев публикации,
$a$ – возраст публикации в днях.

Если для пользователя не предпосчитан какой-либо из эмбеддингов, то соответствующий аргумент в сумме приравнивается к 0.

Финальным этапом в процессе онлайн-обработки является комбинирование (блендинг), на котором результаты оценки перераспределяются с учетом заранее заданных бизнес-правил. Этот этап необходим для того, чтобы обеспечить разнообразие контента в ленте пользователя, минимизировать повторяющиеся публикации и учитывать пропорции публикаций разных типов (реклама, новостные сообщения). В результате формируется итоговая лента рекомендаций, которая готова для отображения пользователю. Ранее вычисленные данные могут быть сохранены в кэш для сокращения времени отклика при повторных запросах. Приложение А содержит код, реализующий этап блендинга.

Оффлайн-компонент системы выполняет более ресурсоемкие операции, связанные с подготовкой данных и обучением моделей. Происходит извлечение признаков из публикаций и пользователей. Этот процесс включает в себя извлечение реакций пользователей на публикации, временных зависимостей и другие контекстных данные. Извлеченные признаки используются для обучения моделей оценки, а также для формирования векторных представлений публикаций и пользователей. Признаки, полученные на этом этапе, сохраняются в хранилище признаков.

После того, как данные были обработаны, обучаются модели оценки. Затем обученные модели сохраняются в хранилище моделей, чтобы далее

использоваться на онлайн-этапе для быстрого определения релевантности публикаций в реальном времени.

Для обработки и анализа данных на оффлайн-этапе используется аналитическая база данных. Она используется для агрегации данных о поведении пользователей и их взаимодействиях с контентом. В хранилище ключ-значение сохраняются векторные представления пользователей и публикаций. Это позволяет значительно ускорить предсказания и уменьшить задержку в процессе онлайн-обработки.

Разделение между онлайн- и оффлайн-обработкой позволяет обеспечить высокую скорость обработки запросов пользователей в условиях больших объемов данных.

# ГЛАВА 3
# РЕАЛИЗАЦИЯ СИСТЕМЫ РЕКОМЕНДАЦИЙ

## 3.1 Нефункциональные требования к системе рекомендаций

Нефункциональные требования определяют ограничения и характеристики системы, которые не связаны непосредственно с функциональностью, но обеспечивают надежность, производительность, масштабируемость и удобство сопровождения. Нефункциональные требования служат основой для создания архитектуры, способной эффективно функционировать в условиях реального использования 1.1].

Архитектура системы должна быть спроектирована так, чтобы минимизировать усилия по сопровождению и обновлению компонентов. Каждый компонент системы должен быть изолирован, легко обновляться без необходимости полной остановки системы и предоставлять набор метрик, характеризующих его производительность. Метрики обычно включают время выполнения API-запросов, время выполнения задач вычислений, уровень загрузки системных ресурсов. Все компоненты системы должны обеспечивать централизованный сбор логов. Логи должны быть структурированными, включать временные метки и контекст выполнения, чтобы облегчить анализ инцидентов. Система должна быть интегрирована с инструментами мониторинга и диагностики (APM, Application Performance Monitoring).

Система должна обеспечивать обработку запросов на получение релевантных публикаций с задержкой не более 500 миллисекунд. Для контроля можно использовать мониторинг времени ответа с визуализацией на дашбордах в системах мониторинга. Хорошей практикой является настройка алертов, чтобы своевременно реагировать на аномалии.

Архитектура системы должна быть устойчива к изменениям нагрузки: поддерживать автоматическое масштабирование и возможность быстрого восстановления (англ. disaster recovery) через автоматическое создание резервных копий.

Система должна обеспечивать уровень доступности не ниже 99.9% (доступность в течение месяца не менее 43 минут простоя) и использовать механизмы отказоустойчивости (репликация данных, резервное копирование, балансировщики нагрузки, автоматическое переключение на резервные компоненты).

Для защиты от атак типа DDoS и других злоупотреблений рекомендуется использовать WAF (Web Application Firewall), лимитирование запросов (rate limiting) и подписи запросов (X-Signature).

## 3.2 Технический стек

В качестве фреймворка для разработки API используется FastAPI. Он является высокопроизводительным решением для создания веб-приложений и API на языке Python и обеспечивает асинхронную обработку запросов. Асинхронность снижает задержки и позволяет обрабатывать запросы в реальном времени с минимальной нагрузкой на сервер.

Для работы с данными и хранения структурированной информации выбрана реляционная база данных PostgreSQL. Redis применяется в качестве кеширующего слоя для ускорения обработки запросов и снижения нагрузки на базу данных за счет хранения часто запрашиваемых данных в памяти, а также для организации работы с фоновыми задачами и крон-задачами через библиотеку ARQ (Asynchronous Redis Queue). Аналитическая база данных построена на основе Hadoop.

Система оркестрации и управления контейнерами основана на Kubernetes. Масштабируемость обеспечивается встроенным горизонтальным автоскейлингом Kubernetes (Horizontal Pod Autoscaler, HPA), который настраивается по среднему использованию CPU и оперативной памяти. Подход позволяет системе автоматически добавлять или удалять поды в зависимости от текущей нагрузки, при этом сохранять производительность и экономить ресурсы. Простейшее решение для быстрого восстановления в случае сбоев в Kubernetes основывается на использовании снапшотов для PersistentVolume. Оно позволяет автоматически создавать резервные копии данных в облачных хранилищах или локальных файловых системах и восстанавливать их при сбоях.

Для обработки текстовых данных и построения векторных представлений используется библиотека Sentence-Transformers, которая основана на трансформерах и позволяет эффективно преобразовывать текстовую информацию в эмбеддинги. Для хранения и обработки больших объемов данных (таких как обученные модели) используется облачное хранилище Amazon S3. ALS из библиотеки implicit обучает модели на бинарных данных через итеративное обновление матриц пользователей и объектов.

Для взаимодействия с облачными сервисами Amazon Web Services (AWS) используется библиотека aioboto3. Для автоматизации процессов и управления задачами используется Apache Airflow.

Для централизованного логирования и мониторинга производительности системы используется ELK-стек (Elasticsearch, Logstash, Kibana). Elasticsearch предоставляет возможности для быстрого поиска и анализа логов, Logstash отвечает за сбор и обработку данных, Kibana используется для визуализации метрик и логов. Для мониторинга ключевых показателей и создания дашбордов используется Grafana. Для диагностики ошибок в реальном времени используется Sentry.

## 3.3 Функциональная схема системы

Кодовая база системы состоит из двух ключевых компонентов: RecommendationApiApp и RecommendationMLApp. Первый отвечает за предоставление рекомендаций через REST API, а второй – за формирование эмбеддингов, признаков и обучение моделей.

Пользователь взаимодействует с системой через клиентский интерфейс, который передает запросы на серверную часть через API Gateway. Шлюз выполняет роль основного маршрутизатора запросов, он направляет их к различным модулям системы. Рисунок 3.1 иллюстрирует ключевые компоненты системы и их взаимодействие.

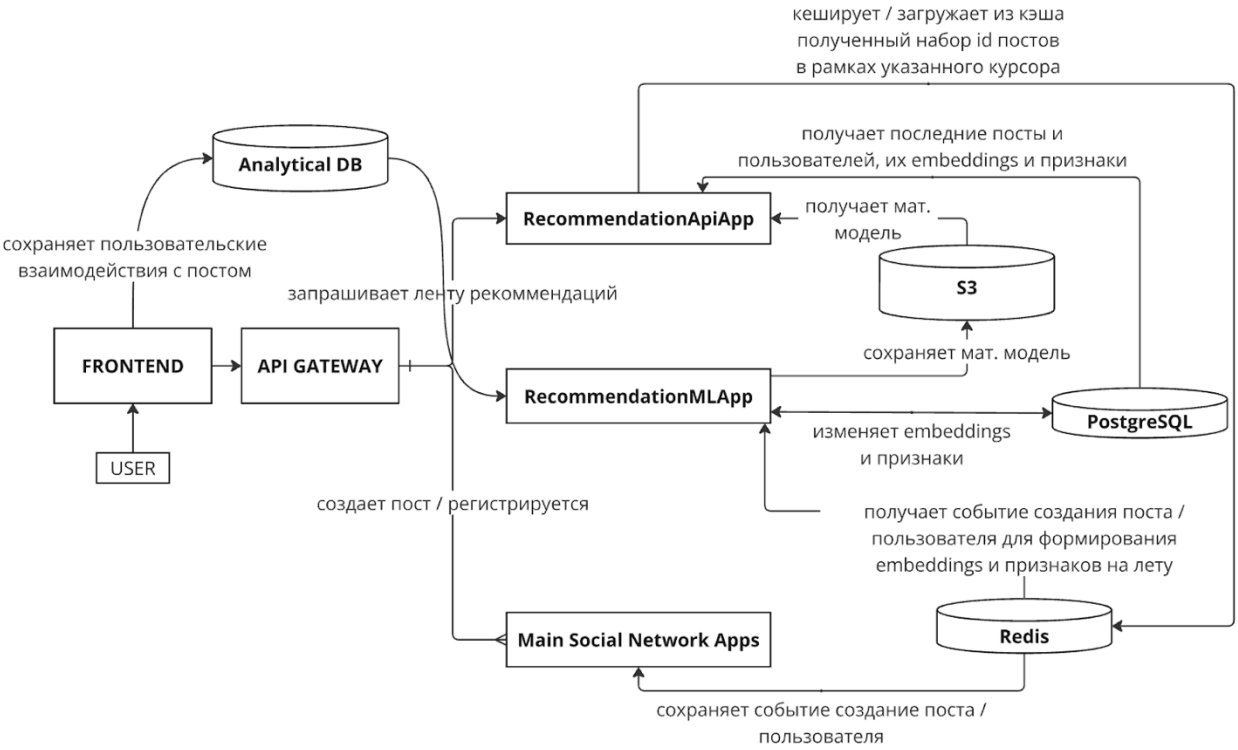


**Рисунок 3.1 – Взаимодействие между основными компонентами**

RecommendationApiApp обрабатывает запросы на получение ленты рекомендаций. Компонент обращается к аналитической базе данных для извлечения пользовательских данных, данных о постах и эмбеддингов. Пользователь идентифицируется по заголовку X-USER-ID, который передается на уровне API Gateway. Модуль взаимодействует с хранилищем Redis, которое используется для кеширования идентификаторов постов в рамках указанных курсоров для того, чтобы ускорить обработку запросов и повысить производительность.

RecommendationMLApp отвечает за обработку данных, связанных с действиями пользователей, такими как создание постов или регистрация нового пользователя. Эти данные поступают из основного компонента социальной сети и используются для обновления эмбеддингов пользователей и признаков публикаций. Обновленные данные сохраняются в базе данных PostgreSQL, а обученные модели выгружаются в хранилище S3 для последующего использования. Это позволяет системе непрерывно адаптироваться к изменениям пользовательских предпочтений. На рисунках 3.2 и 3.3 представлены ключевые этапы обработки событий и подготовки данных.

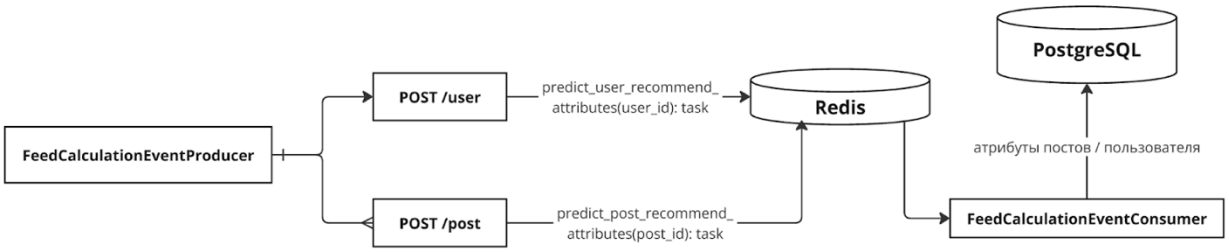


**Рисунок 3.2 – Схема обработки событий для формирования рекомендаций**

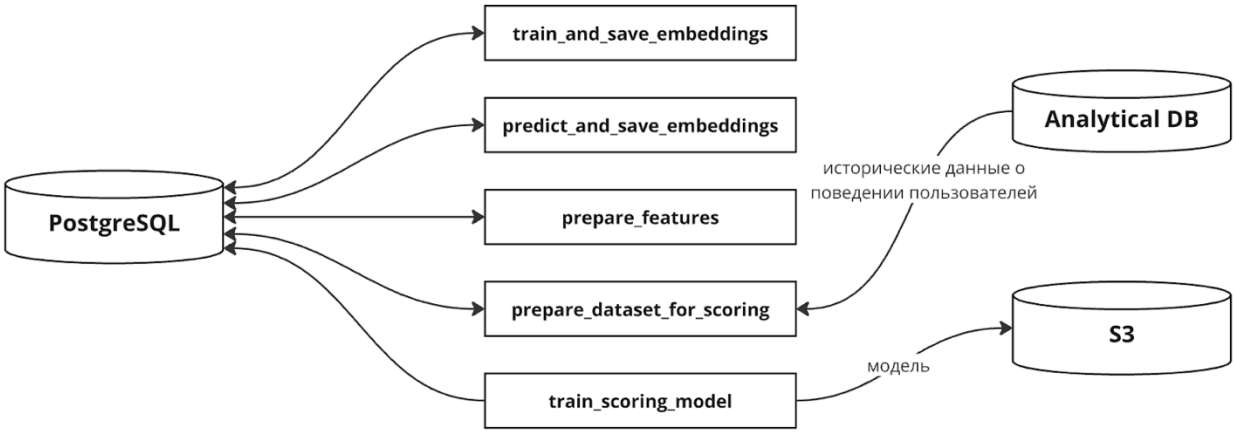


**Рисунок 3.3 – Процессы подготовки данных и обучения моделей для рекомендаций**

Основные компоненты социальной сети регистрируют события взаимодействия пользователей с контентом. Эти события передаются в том числе в рекомендательные модули системы. RecommendationMLApp обновляет признаки пользователей и постов в режиме реального времени. Аналитическая база данных накапливает статистические данные о взаимодействиях.

Система должна поддерживать возможность указания параметров запроса, таких как ограничение количества возвращаемых публикаций (limit, по умолчанию 20, максимум 1000) и курсора для постраничного получения данных. Курсор – это строковое представление объекта, который хранит закодированные параметры user_id, start_time (время начала выборки) и offset (смещение), которые используются для отслеживания текущей позиции в наборе данных или потоке данных.

Результат запроса – ответ в формате JSON, включающий метаинформацию о публикациях: идентификатор поста, текст, дату создания, число просмотров, список прикрепленных изображений, реакции пользователей, и другие параметры (теги, информация о взаимодействиях текущего пользователя, ссылки на публикации). В случае отсутствия публикаций система возвращает пустой JSON, а при некорректных данных возвращается сообщение об ошибке.

## 3.4 Формирование рекомендаций: процессы и алгоритмы

Формирование рекомендаций реализовано через три ключевых этапа: генерацию кандидатов, ранжирование и переранжирование. Процесс формирования рекомендаций с этапами генерации ленты публикаций и компонентной структурой рекомендательного сервиса представлен на рисунках 3.4 и 3.5.

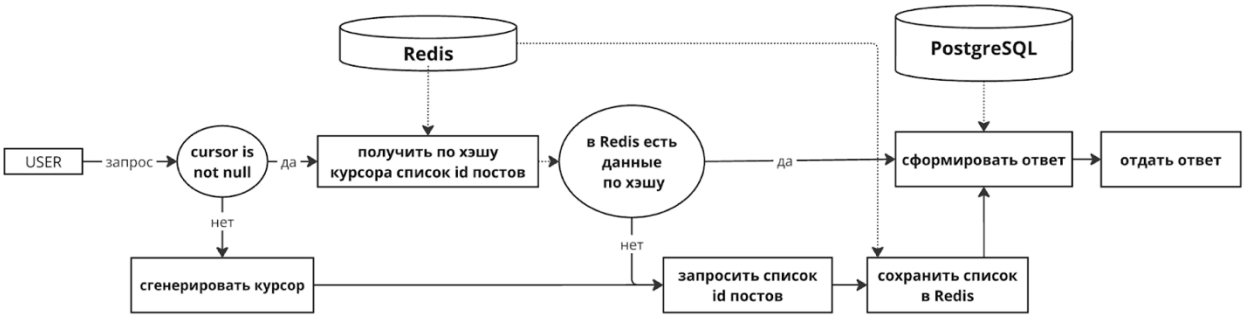


**Рисунок 3.4 – Процессы генерации ленты публикаций**

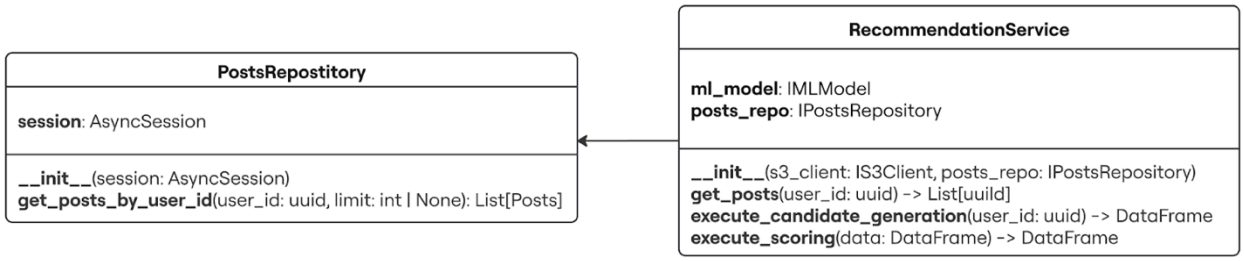


**Рисунок 3.5 – Компонентная структура рекомендательного сервиса**

Первый этап заключается в выборе релевантного набора публикаций, которые могут быть предложены пользователю. Для этого используется метод, принимающий идентификатор пользователя и возвращающий список из ограниченного числа публикаций. Каждая публикация сопровождается флагами, которые отражают специфические аспекты взаимодействия пользователя с контентом. Флаги могут указывать, подписан ли пользователь на автора публикации, видел ли он ее ранее или существуют ли лайки или комментарии от пользователей, на которых он подписан. На начальном этапе реализации кандидатами выступают последние по дате публикации посты, что является приемлемым подходом при ограниченном объеме данных. В качестве улучшений можно рассматривать внедрение методов на основе Approximate Nearest Neighbors для ускорения процесса отбора.

После отбора кандидатов выполняется их ранжирование, в ходе которого публикации оцениваются по степени их релевантности для конкретного пользователя. Этот этап включает генерацию кросс-признаков, отражающих схожесть эмбеддингов пользователя и публикации, а также контекстные признаки (например, время взаимодействия). Используемые модели обучаются предсказывать вероятность конкретных действий пользователя, таких как лайк, комментарий или репост. Доступна возможность изменения набора признаков и типов моделей.

Этап переранжирования предназначен для учета бизнес-правил. Алгоритмы учитывают разнообразие контента, баланс между различными источниками публикаций и предотвращение подряд идущих публикаций одного типа. На начальном этапе в ленте должно быть 50% публикаций от авторов и 50% от сообществ и также 50% публикаций от сообществ или авторов, на которых подписан пользователь, чтобы поддерживать разнообразие и актуальность контента. Соотношение публикаций динамически изменяется в зависимости от возможности выполнения вышеописанных требований.

Компонентная структура сервиса обучения модели, описывающая обработку событий и обучение моделей, представлена на рисунке 3.6.

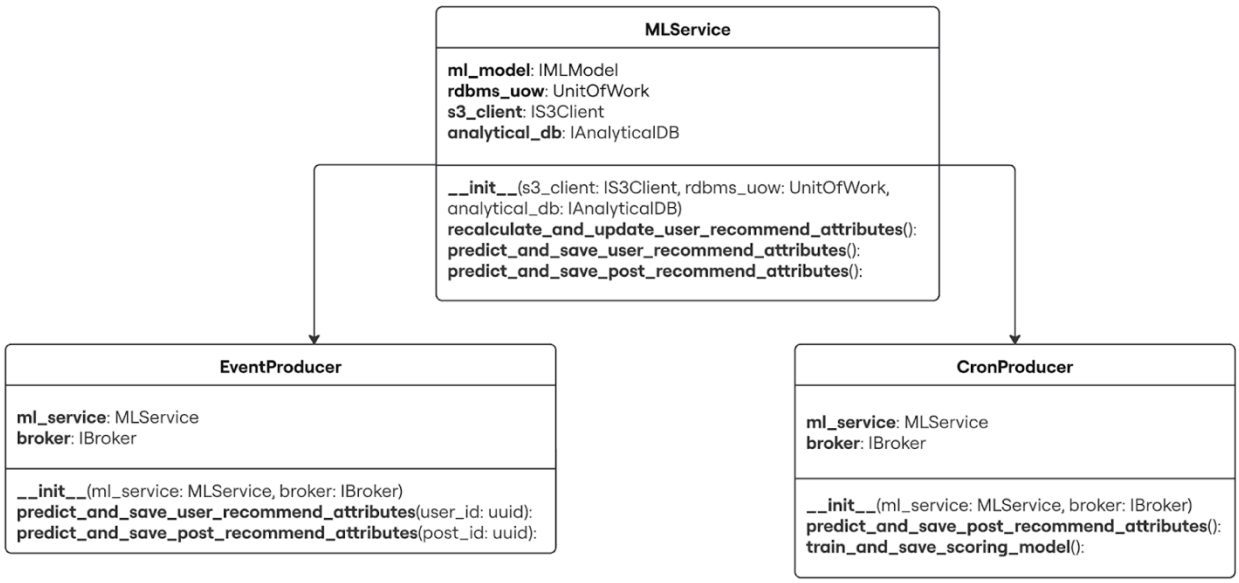


**Рисунок 3.6 – Компонентная структура сервиса обучения модели**

## 3.5 Оценка качества рекомендаций

Для оценки качества рекомендательной системы был проведен эксперимент, в котором использовались данные реальной социальной сети. Для тестирования была выбрана часть базы данных, включающая несколько сотен пользователей и около десяти тысяч постов. Чтобы гарантировать конфиденциальность использовались обфусцированные данные. Все идентифицируемые данные, такие как имена пользователей или другие личные данные, были заменены на случайные или зашифрованные значения.

Для оценки качества разработанной рекомендательной системы был проведен эксперимент, в котором тестировались три подхода: случайный метод рекомендаций, метод рекомендаций на основе популярности и персонализированная модель. Была сформирована тестовая выборка пользователей. Для каждого пользователя из выборки три алгоритма генерировали рекомендации. Случайный метод предлагал набор из случайно выбранных постов, популярный предлагал наиболее популярные посты по общему числу взаимодействий, а разработанный алгоритм формировал персонализированные рекомендации.

Были определены релевантные посты, которые служили эталоном для оценки качества рекомендаций. Релевантные посты для каждого пользователя определялись на основе его взаимодействий с контентом в течение тестового периода. Взаимодействия включали лайки, комментарии и просмотры, и каждому типу взаимодействия был присвоен определенный

вес. Актуальность поста оценивалась с учетом лайков и просмотров за весь период, количества комментариев, активности за последние 24 часа, возраста поста и его популярности среди пользователей. Учитывался штраф за повторные просмотры постов, чтобы избежать рекомендаций, которые уже были просмотрены.

Данные были разделены на две части: тренировочную и тестовую. Тренировочная выборка содержала данные о взаимодействиях, произошедших до определенной даты, тестовая выборка – после этой даты.

Для оценки качества были использованы четыре метрики: Precision@K, Recall@K, NDCG и MAP. Precision@K измеряет долю релевантных постов среди первых K рекомендаций, показывая, насколько точно система рекомендует контент. Recall@K оценивает, какой процент всех релевантных постов был найден в топ K рекомендаций и отражает полноту системы. NDCG учитывает как точность, так и порядок рекомендаций, при этом более высокие позиции релевантных постов получают больший вес. MAP усредняет точность по всем пользователям и всем позициям рекомендаций. Метрика показывает общую оценку качества работы системы.

Результаты эксперимента представлены в таблице 3.1.

Таблица 3.1 – Результаты эксперимента

| Метрика | Случайный | Популярный | Собственный |
|---|---|---|---|
| Precision@K | 0.12 | 0.18 | 0.35 |
| Recall@K | 0.09 | 0.25 | 0.45 |
| NDCG | 0.07 | 0.02 | 0.5 |
| MAP | 0.11 | 0.2 | 0.38 |

Случайный метод рекомендаций продемонстрировал низкие значения всех метрик, так как он вообще не учитывает интересы пользователей. Популярный показал улучшение, но его эффективность все равно была ниже, чем у персонализированной модели, так как рекомендации основывались на популярности постов, а не на предпочтениях пользователей. Результаты показывают, что персонализированная модель значительно превосходит случайный и популярный подходы. Несмотря на относительно высокие значения, метрики указывают на наличие значительного потенциала для улучшения.

# ГЛАВА 4
# ВВЕДЕНИЕ СИСТЕМЫ МУЛЬТИПАРАМЕТРИЧЕСКОГО ТАРГЕТИНГА КОНТЕНТА

## 4.1 Постановка задачи улучшения рекомендательной системы через таргетинг контента

Развитие современных социальных платформ требует создания интеллектуальных систем, способных одновременно решать задачи вовлечения как авторов контента, так и конечных потребителей – читателей. Одной из основных проблем в этом процессе становится поиск баланса между интересами обеих сторон платформы.

В процессе развития рекомендательной системы разрабатываемой социальной платформы была выявлена необходимость решения следующих проблем:

1. Низкая мотивация авторов к созданию контента из-за отсутствия прозрачных механизмов таргетирования на целевую аудиторию.

   Таргетинг – это механизм точного определения и выбора целевой аудитории для показа контента, рекламы или рекомендаций на основе заданных параметров.
2. Снижение релевантности ленты для конечных пользователей из-за недостаточной персонализации.

В результате была предложена и внедрена система таргетинга публикаций, реализующая комплексный подход к персонализации контента. В отличие от существующих решений (например, таргетинга рекламных публикаций в Facebook), данный механизм ориентирован на органический контент, что формирует дифференцирующее преимущество платформы.

Разработка велась с учетом специфики профессиональных социальных сетей. Концепция строится на принципе многоуровневой разметки контента, позволяющей авторам определять целевую аудиторию по нескольким взаимодополняющим критериям. Механизм одновременно учитывает три ключевых параметра целевой аудитории: профессиональную принадлежность (585 специализаций), корпоративный контекст (более 20 миллионов компаний) и отраслевую специализацию (18 индустрий).

Система таргетинга решает три основные задачи: повышает вовлеченность авторов за счет точного позиционирования контента, увеличивает глубину аналитики платформы через 3D-сегментацию аудитории, и улучшает пользовательский опыт за счет релевантных профессиональных рекомендаций. Для авторов она предоставляет

инструменты точного попадания в целевую аудиторию, а для платформы открывает новые возможности монетизации. В результате все участники получают выгоду: авторы – больший охват, пользователи – качественный контент, платформа – рост вовлечения и новые бизнес-модели.

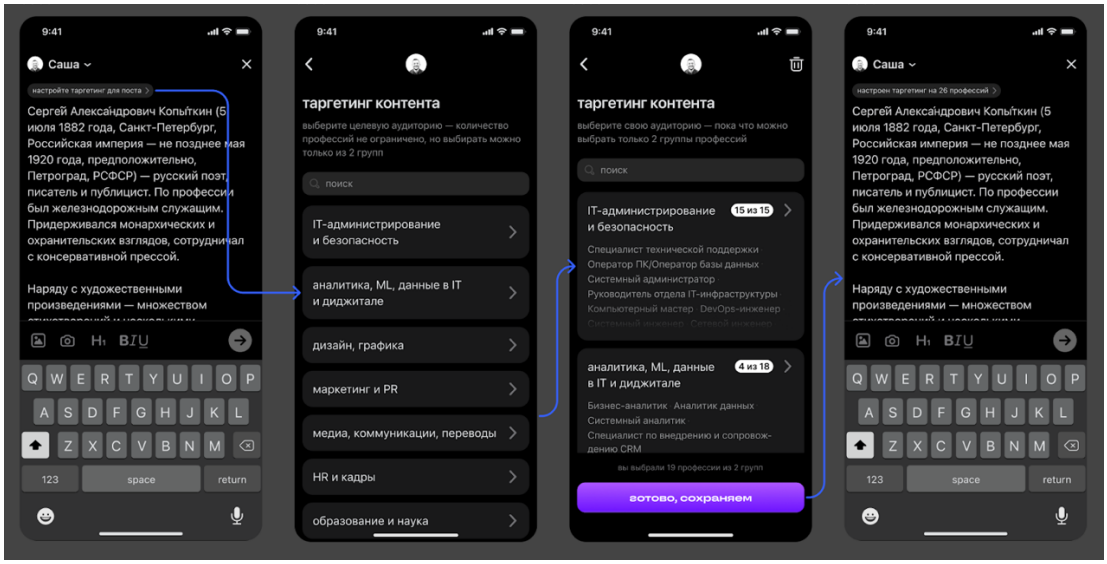


**Рисунок 4.1 – Интерфейс настройки таргетинга контента**

Для оценки эффективности системы используются количественные и качественные метрики: охват размеченного контента (целевой уровень >80%) и его распределение по профессиональным, корпоративным и отраслевым сегментам, а также ключевые показатели вовлеченности – CTR (click-through rate, коэффициент кликабельности) и время взаимодействия с таргетированными материалами, дополненные анализом роста аудитории авторов, применяющих инструменты таргетинга.

Ключевые этапы внедрения:

1. Разработка интерфейса разметки для авторов (выбор категорий при публикации).

2. Модификация рекомендательной модели. Введение весов для признаков "роль автора", "роль читателя", "компания автора", "компания читателя", "индустрии автора", "индустрии читателя" и использование коллаборативной фильтрации для сопоставления.

3. Создание дашборда аналитики для мониторинга баланса контента/аудитории по ролям, компаниям и индустриям и алерты при отклонении метрик (например, <80% размеченных публикаций).

Техническая реализация потребовала создания схемы данных для всех типов таргетинга и разработки механизма каскадного наследования настроек (от профиля или сообщества к отдельным публикациям). Большое внимание

было уделено оптимизации запросов для работы с комбинированными критериями для сохранения высокой производительности системы при увеличении сложности алгоритмов рекомендаций.

Таргетинг существенно расширяет возможности платформы по точному позиционированию контента и обеспечивает новый уровень персонализации для пользователей. Такой механизм дает значительное конкурентное преимущество платформы в сегменте профессиональных социальных сетей.

## 4.2 Функциональные требования и реализация

Функционал системы строится вокруг принципа адаптивного продвижения публикаций среди релевантной профессиональной аудитории. Доступ к функции таргетинга предоставляется авторам, опубликовавшим не менее пяти материалов в рамках платформы. При активации таргетинга система инициирует первоначальное продвижение публикации (буст) среди пользователей выбранных профессиональных ролей. Дальнейшая судьба контента определяется алгоритмами машинного обучения, которые анализируют поведенческие реакции целевой аудитории. Положительные сигналы, такие как лайки и комментарии, приводят к продолжению продвижения, тогда как негативные реакции (отметки "не интересно" или блокировки) служат сигналом для прекращения буста.

Механизм модерации автоматически выявляет недобросовестных авторов по характерным паттернам негативных реакций со стороны целевой аудитории. В таких случаях система генерирует сигналы для модераторов платформы, позволяя оперативно принимать решения о возможных ограничениях.

Интерфейсная часть решения, представленная на рисунке 4.1, демонстрирует инструменты для разметки контента, где авторы могут выбирать до двух групп профессиональных ролей или конкретные профессии из вложенных категорий. Каталог ролей с 585 профессиями организован в 18 тематических групп для удобной навигации. Система ранжирует предлагаемые группы ролей, основываясь на профиле автора. Это значительно упрощает процесс выбора целевой аудитории.

Унифицированный интерфейс настройки активируется после достижения пользователем определенного уровня активности на платформе. Для личных профилей функция становится доступной после пяти публикаций. При создании нового поста пользователь получает возможность настроить таргетинг через специальную кнопку, которая ведет на страницу с

тремя блоками параметров: выбор профессий, компаний и индустрий. Система интеллектуально предзаполняет эти параметры, если они были ранее указаны в настройках профиля, для упрощения процесса для постоянных авторов. Для сообществ и каналов действуют аналогичные правила активации, но с дополнительным требованием: все пять публикаций должны быть сделаны от имени сообщества.

Система предусматривает временное окно в 30 минут после публикации, в течение которого автор может скорректировать параметры таргетинга. По истечении этого срока изменения блокируются. Для управления списком ролей, категорий, индустрий и компаний, настройки флага is_targetable и связывания ролей с категориями используется административная панель.

Механизм таргетинга потребовал изменений в организации базы данных. Для хранения категорий ролей для таргетинга была добавлена таблица role_categories. В таблице position были добавлены флаг is_targetable для пометки таргетируемых позиций и связь с таблицей categories через внешний ключ category_id. Были добавлены таблицы channel_targeting_settings, post_targeting_settings и account_targeting_settings, хранящие настройки таргетинга для каналов, публикаций и аккаунтов соответственно, их структура детально описана в таблицах 4.1, 4.2 и 4.3.

Таблица 4.1 – Описание полей таблицы настроек таргетинга для каналов

| Название поля | id | channel_id | roles, companies, industries | created_at |
|---|---|---|---|---|
| Тип данных | UUID | UUID | list[UUID] | datetime |
| Описание | Уникальный идентификатор настроек для таргетинга | ID канала для которого настраивается таргетинг | Список ID ролей / компаний / индустрий для таргетинга | Время создания настроек для таргетинга |
| Ограничения | nullable = False primary_key = True | FK("channel.id") nullable = False | nullable = False | nullable = False |

Таблица 4.2 – Описание полей таблицы настроек таргетинга для постов

| Название поля | id | post_id | roles, companies, industries | created_at |
|---|---|---|---|---|
| Тип данных | UUID | UUID | list[UUID] | datetime |
| Описание | Уникальный идентификатор настроек для таргетинга | ID поста для которого настраивается таргетинг | Список ID ролей / компаний / индустрий для таргетинга | Время создания настроек для таргетинга |
| Ограничения | nullable = False primary_key = True | FK("post.id") nullable = False | nullable = False | nullable = False |

Таблица 4.3 – Описание полей таблицы настроек таргетинга для пользователей

| Название поля | id | post_id | roles, companies, industries | created_at |
|---|---|---|---|---|
| Тип данных | UUID | UUID | list[UUID] | datetime |
| Описание | Уникальный идентификатор настроек для таргетинга | ID пользователя для которого настраивается таргетинг | Список ID ролей / компаний / индустрий для таргетинга | Время создания настроек для таргетинга |
| Ограничения | nullable = False primary_key = True | FK("account.id") nullable = False | nullable = False | nullable = False |

Рекомендательная система была модифицирована через введение квотного механизма, который резервирует часть выдачи под таргетируемые посты и одновременно применяет штрафные коэффициенты к нерелевантному контенту в основной ленте. Это позволяет четко разделить поток рекомендаций на два канала: органическую выдачу, формируемую

традиционными алгоритмами, и управляемую квоту таргетированного контента с гарантированным охватом целевой аудитории.

Развитие системы может включать введение платного снятия штрафов для буста контента и расширение параметров таргетинга (геолокация, статус работы).

## 4.3 Внедрение системы таргетинга в рекомендательную систему

В рекомендательной системе реализован двухуровневый механизм формирования ленты, который сочетает органическую выдачу и управляемый таргетинг. Основной поток контента формируется традиционными алгоритмами рекомендаций, при этом для таргетируемых постов, не соответствующих профессиональной роли пользователя, применяется понижающий коэффициент. Отдельно выделена квота таргетированного контента, гарантирующая показ одного релевантного поста каждые пять позиций в ленте. Внутри выделенного блока посты распределяются случайным образом с приоритетом для материалов, наиболее соответствующих специализации пользователя.

Изначально в рамках экспериментальной реализации рекомендательная система учитывала исключительно параметр профессиональных ролей при таргетировании контента. На этом этапе 20% ленты пользователя формировалось из публикаций, таргетированных на его профессиональную принадлежность. При этом ранжирование таких публикаций осуществлялось по стандартным алгоритмам рекомендательной системы, за исключением случаев явного несоответствия целевой аудитории.

В процессе дальнейшего развития системы был реализован качественно новый подход к многоцелевому таргетированию. Использовались уже три взаимосвязанных параметра: профессиональные роли, компании и отраслевая принадлежность. Это потребовало существенной модификации алгоритмов ранжирования, так как появилась необходимость учитывать:

1. Степень соответствия публикации параметрам пользователя (количество совпадающих критериев таргетинга).
2. Значимость различных типов таргетинга. Иерархия важности: компании > роли > индустрии.
3. Узость целевой аудитории (специфичность таргетируемой группы).

Для реализации нового подхода была разработана система бонусных коэффициентов, модифицирующих исходный скоринг рекомендаций.

Коэффициент попадания увеличивает вес публикации пропорционально количеству совпадающих параметров таргетинга (3/3 > 2/3 > 1/3).

Коэффициент значимости отражает приоритетность разных типов таргетинга, основанную на конкретности целевой группы

Коэффициент узости учитывает специфичность таргетируемой аудитории (преимущество узких профессиональных или корпоративных групп).

Такие модификации позволили сохранить баланс в ленте (20% таргетированного контента) и одновременно значительно повысить точность попадания публикаций в целевую аудиторию.

## 3.4 Экспериментальное исследование системы многоцелевого таргетинга

Экспериментальное исследование системы многоцелевого таргетинга было направлено на комплексную проверку эффективности предложенного подхода. Анализировалось влияние трех ключевых аспектов: степень воздействия комбинированного учета корпоративных, профессиональных и отраслевых параметров на качество рекомендаций, определение значимости различных типов таргетинга и их оптимальной иерархии, а также оценка зависимости эффективности таргетирования от степени узости выбранной целевой аудитории.

Экспериментальное исследование проводилось на четырех тестовых группах по 15 000 пользователей в каждой, где группа A получала рекомендации с базовым бонусом за попадание в целевую аудиторию без дополнительных коэффициентов, группа B – с учетом базового бонуса и коэффициента значимости таргетинга, группа C – с базовым бонусом и коэффициентом узости аудитории, а группа D тестировала полную модель, включающую все три параметра: базовый бонус, значимость и узость таргетинга.

Эффективность различных подходов оценивалась по трем основным метрикам, расположенным в порядке убывания важности: частота подписок на контент (subscription rate) как основной показатель долгосрочной вовлеченности, уровень лайков (like rate) как индикатор непосредственной реакции пользователей и показатель охвата целевой аудитории (ERV, Effective Reach Value), отражающий точность попадания в целевую группу.

Для объективной оценки эффективности был разработан комплексный алгоритм скоринга публикаций, который учитывает три параметра. Базовый коэффициент попадания рассчитывается по формуле 4.1:

$$score = hits / (groups \wedge C), \quad (4.1)$$

где *hits* отражает количество совпадений параметров (от 1 до 3),
*groups* – общее количество критериев таргетинга,
*C* – эмпирически подобранная константа для оптимальной градации оценок.

Также введены весовые коэффициенты значимости разных типов таргетинга: максимальный вес 1.4 для корпоративной принадлежности, средний 1.225 для профессиональных ролей и базовый 1.0 для отраслевой принадлежности. Это отражает приоритетность корпоративного контекста над профессиональным и отраслевым. Для учета размера целевой аудитории используется адаптивный бонус, линейно возрастающий от 1.0 для широких групп (>100 000 пользователей) до 1.3 для узких (<1 000 пользователей) по формуле 4.2

$$bonus = 1 + (0.3 * (1 - group\_size / max\_size)) \quad (4.2)$$

для более точного ранжирования контента для нишевых профессиональных сообществ.

Эксперимент проводился в несколько этапов.

Подготовительный этап включал разработку DAG в Airflow для ежедневного расчета размеров групп, модификацию методов получения публикаций с учетом новых параметров и настройку флагов для управления экспериментальными группами. Точность кластеризации $R^2$=0.91.

Тестовый эксперимент включал проверку работоспособности системы на ограниченной выборке и корректировку коэффициентов на основе предварительных результатов.

Основной эксперимент включал полноценный A/B/C/D тест на всей пользовательской базе и сбор и анализ данных по ключевым метрикам

В ходе эксперимента использовались числовые параметры, представленные в таблице 4.4. Все они были подобраны в результате предварительных исследований. Скрипт, с помощью которого были подобраны значения представлен в приложении Б. Он не включает непосредственно выбор значения, все итоговые выборы были сделаны вручную. Все указанные параметры прошли верификацию в ходе

многоэтапного A/B-тестирования, где демонстрировали статистически значимое улучшение ключевых метрик вовлеченности по сравнению с базовой версией алгоритма.

Таблица 4.4 – Числовые параметры эксперимента

| Параметр | Группа A (базовая) | Группа B (значимость) | Группа C (узость) | Группа D (полная) |
|---|---|---|---|---|
| Базовый коэффициент C | 0.612 | 0.612 | 0.612 | 0.612 |
| Значимость компаний | - | 1.4 | - | 1.2 |
| Значимость ролей | - | 1.225 | - | 1.11 |
| Значимость индустрий | - | 1.0 | - | 1.0 |
| Lower bound (узость) | - | - | 1.0 | 1.0 |
| Upper bound (узость) | - | - | 1.3 | 1.3 |

Базовый коэффициент C=0.612 был выбран после серии итеративных тестов как оптимальное значение. Оно обеспечивает выполнение фундаментального условия приоритезации: полное совпадение всех трех параметров таргетинга (3/3) должно давать более высокий результат, чем парное совпадение (2/2), которое в свою очередь превосходит одиночное (1/1). Аналогичная градация соблюдается и для частичных совпадений, где конфигурация 2/3 получает преимущество перед 1/2, а та – перед 1/3. Данный коэффициент создает устойчивую разницу около 10% между соседними уровнями совпадений, что было определено как оптимальный баланс между чувствительностью системы и устойчивостью к шумам в данных.

Система весовых коэффициентов значимости отражает разработанную иерархию важности различных типов таргетинга, где корпоративная принадлежность (компании) получает наибольший вес 1.4 (в группе B) или 1.2 (в финальной группе D), профессиональные роли – промежуточное значение 1.225 и 1.11 соответственно, а отраслевая принадлежность

(индустрии) остается базовым уровнем с коэффициентом 1.0. Разница между уровнями в 15-20% была определена как достаточная для обеспечения значимого различия в ранжировании, но не настолько большая, чтобы полностью нивелировать влияние других факторов.

Параметры узости аудитории используют граничные значения lower bound=1.0 и upper bound=1.3, где нейтральное влияние на широкие группы (более 100 000 пользователей) постепенно усиливается до максимального бонуса для узкоспециализированных аудиторий (менее 1 000 пользователей). Подход позволяет сохранять баланс между релевантностью контента для конкретных небольших сообществ и общим охватом платформы.

## 3.5 Результаты эксперимента

Таблица 4.5 – Результаты эксперимента

| Метрика | Группа A (базовая) | Группа B (значимость) | Группа C (узость) | Группа D (полная) |
|---|---|---|---|---|
| Subscription rate | 12.3%±0.4% | 14.7%±0.5% | 15.1%±0.6% | 17.9%±0.5% |
| Like rate | 8.2%±0.3% | 9.5%±0.4% | 9.8%±0.4% | 11.4%±0.5% |
| ERV ЦА | 0.78±0.02 | 0.82±0.03 | 0.85±0.03 | 0.91±0.02 |
| CTR | 4.1% | 4.8% | 5.2% | 6.0% |

Результаты эксперимента демонстрируют последовательный рост ключевых метрик по мере усложнения модели таргетинга. Полная модель (группа D) показала наилучшие результаты по всем показателям: прирост subscription rate на 5.6 п.п. (45.5% относительно базовой группы), увеличение like rate на 3.2 п.п. (39%), рост ERV ЦА на 0.13 (16.7%) и CTR на 1.9 п.п. (46.3%), что подтверждает эффективность комплексного учета значимости и узости таргетинга.

Наибольший вклад в улучшение показателей внес коэффициент узости аудитории (Группа C), особенно заметный в метриках ERV ЦА (+0.07 против группы B) и CTR (+0.4 п.п.), что свидетельствует о важности точного попадания в нишевые профессиональные сообщества. При этом добавление коэффициента значимости (Группа B) дало существенный прирост subscription rate (+2.4 п.п.) по сравнению с базовой моделью. Это подтверждает гипотезу о приоритетности корпоративного контекста (вес 1.4) перед профессиональными (1.225) и отраслевыми (1.0) параметрами.

Итоговый выбор обусловлен подтвержденной эффективностью и технической устойчивостью. Полученные данные легли в основу финальной реализации системы многоцелевого таргетинга.

## 3.6 Анализ влияния таргетинга на пользовательскую активность

Исследование эффективности системы таргетинга выявило несколько важных закономерностей. Основной фокус работы заключался в оценке влияния таргетинга на ключевые показатели вовлеченности и охватов. Особое внимание уделялось устранению эффектов, связанных с высокой дисперсией данных.

Методология исследования основывалась на сравнении пар "близнецов" – пользователей и сообществ со схожими характеристиками, но разным статусом использования таргетинга. Для анализа применялись два подхода: прямое сравнение показателей и метод различий в различиях (diff-in-diff). В выборку вошли 500 активных пользователей и 1000 сообществ, которые публиковали контент как минимум раз в четыре месяца до внедрения функции.

Отбор близнецов проводился по четырем параметрам: среднее количество взаимодействий с контентом, обычный охват публикаций, численность аудитории при старте тестирования и регулярность размещения постов, измеренная в 48-дневных периодах.

Результаты diff-in-diff анализа представлены в таблицах, отражающих влияние таргетинга на вовлеченность (действия) и охваты.

В первой таблице (рисунок 4.2), посвященной действиям пользователей, коэффициент Intercept=0.2635 показывает базовый уровень вовлеченности до внедрения таргетинга. Коэффициент test=0.0084 не является статистически значимым (p=0.537), что свидетельствует об отсутствии систематических различий между тестовой и контрольной группами до вмешательства. Значимый отрицательный коэффициент post=-0.2095 отражает общее снижение вовлеченности в обеих группах после момента анализа, возможно связанное с сезонными факторами или изменениями платформы. Показатель test:post=0.0839 отражает, что в тестовой группе, использовавшей таргетинг, вовлеченность оказалась существенно выше, чем в контрольной ($p<0.001$), с 95% доверительным интервалом от 0.037 до 0.131.

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| Intercept | 0.2635 | 0.010 | 25.871 | 0.000 | 0.244 | 0.284 |
| test | 0.0084 | 0.014 | 0.617 | 0.537 | -0.018 | 0.035 |
| post | -0.2095 | 0.019 | -10.908 | 0.000 | -0.247 | -0.172 |
| test:post | 0.0839 | 0.024 | 3.503 | 0.000 | 0.037 | 0.131 |

**Рисунок 4.2 – Результаты diff-in-diff анализа вовлеченности**

Во второй таблице (рисунок 4.3), анализирующей охваты, аналогичная структура коэффициентов показывает более выраженные эффекты. Базовый уровень охватов (Intercept=10.4936) значительно выше, чем для действий. Коэффициент test (1.7637) указывает на изначально большие охваты в тестовой группе ($p<0.001$), что требует осторожности при интерпретации. Общее снижение охватов (post=-5.6238) компенсируется значимым положительным эффектом таргетинга (test:post=2.9216), который обеспечил средний прирост около 3 дополнительных просмотров на пост ($p<0.001$) с доверительным интервалом от 1.536 до 4.307.

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| Intercept | 10.4936 | 0.301 | 34.913 | 0.000 | 9.904 | 11.083 |
| test | 1.7637 | 0.402 | 4.386 | 0.000 | 0.976 | 2.552 |
| post | -5.6238 | 0.567 | -9.924 | 0.000 | -6.734 | -4.513 |
| test:post | 2.9216 | 0.707 | 4.132 | 0.000 | 1.536 | 4.307 |

**Рисунок 4.3 – Результаты diff-in-diff анализа охватов**

Стандартные ошибки (std err) для всех значимых коэффициентов относительно малы по сравнению с величиной эффектов. Значения t-статистики (от 3.503 для действий до 4.132 для охватов) и соответствующие p-значения (все $<0.001$ для ключевых параметров) свидетельствуют о высокой статистической значимости выявленных эффектов. Доверительные интервалы [0.025, 0.975] не включают ноль для ключевого параметра test:post, что дополнительно подтверждает наличие реального положительного влияния таргетинга на обе метрики.

Важным аспектом исследования стало изучение влияния бустов. Исключение пользователей, использовавших платное продвижение, не изменило основных выводов, хотя и несколько уменьшило величину эффектов. Для сообществ ситуация оказалась сложнее – после исключения каналов с бустами значимый рост сохранился только для охватов, в то время как показатели вовлеченности перестали демонстрировать статистически значимые изменения.

Полученные результаты позволяют сделать несколько практических выводов. Во-первых, подтверждена эффективность таргетинга как инструмента роста охватов для всех типов аккаунтов. Во-вторых, влияние на вовлеченность более выражено у личных профилей, что требует дифференцированного подхода к развитию функции. В-третьих, эффекты бустов необходимо учитывать отдельно, особенно при анализе сообществ.

Целесообразно добавить индикатор конкурентности групп в интерфейсе таргетинга. Авторы смогут оперативно оценивать риски низкого ERV и перенаправлять трафик в менее насыщенные, но релевантные сегменты. Для авторов с устойчиво высокими метриками вовлеченности следует тестировать автоматизированный таргетинг с последующим уведомлением об эффективности кампании. Одновременно необходимо разработать триггеры, отключающие таргетинг для постов с ERV ниже натурального охвата более чем на 15-20%.

# ЗАКЛЮЧЕНИЕ

Несмотря на значительный прогресс в области разработки рекомендательных систем, проблемы их эффективности и адаптивности остаются актуальными. Текущие подходы требуют совершенствования методов персонализации контента и разработки более гибких и устойчивых алгоритмов для повышения их эффективности в реальных условиях.

В ходе выполнения работы были рассмотрены и решены следующие задачи:

1. Проанализированы современные подходы к построению рекомендательных систем, включая методы коллаборативной фильтрации, контентной фильтрации и гибридные модели. Исследованы возможности их применения в социальных платформах, оценены сильные и слабые стороны каждого подхода.
2. Определены основные характеристики данных, необходимые для построения эффективной системы рекомендаций. Предложены способы их структурирования для дальнейшей обработки.
3. Исследована проблема холодного старта. Предложены методы минимизации этого эффекта.
4. Разработан алгоритм для формирования рекомендаций, который включает этапы предобработки данных, обучения моделей и выбора соответствующих методов фильтрации.
5. Спроектирована архитектура системы рекомендаций, детализированы взаимодействия между ключевыми компонентами и способы их интеграции.
6. Осуществлен выбор технического стека для реализации рекомендательных алгоритмов с учетом требований к производительности, масштабируемости и возможностей интеграции с внешними сервисами.
7. Разработана функциональная схема системы, которая отображает структуру и управление потоком данных в рамках всей системы.
8. Разработан прототип рекомендательной системы, реализующий все этапы обработки данных.
9. Проведены тестирование и оценка прототипа с использованием принятых в области метрик, выявлены области для дальнейшего улучшения системы.
10. Внедрена система мультипараметрического таргетинга, реализован механизм сегментации контента по множеству параметров. Проведены A/B-тесты, подтвердившие эффективность решения.

11. Проведено экспериментальное исследование улучшений системы, которое выявило повышение точности доставки контента. Положительная динамика подтверждена ростом метрик вовлеченности.

Результаты работы имеют практическую ценность в области разработки рекомендательных систем и могут использоваться в качестве основы для создания эффективных алгоритмов рекомендаций на социальных платформах.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Клеппман М., Высоконагруженные приложения. Программирование, масштабирование, поддержка / М. Клеппман / Литрес [Электронный ресурс]. – Питер, 2018 – Режим доступа: https://www.litres.ru/book/martin-kleppman-1733/vysokonagruzhennye-prilozheniya-programmirovanie-mass-39100996/ – Дата доступа: 15.03.2025.
2. Мартин Р., Чистая архитектура. Искусство разработки программного обеспечения / Мартин Р. // Bol. [Электронный ресурс]. – Питер, 2022 – Режим доступа: https://www.bol.com/nl/nl/s/?searchtext=9785446107728 – Дата доступа: 17.04.2025.
3. Фальк К., Рекомендательные системы на практике / К. Фальк // ДМК-Пресс [Электронный ресурс]. – ДМК-Пресс, 2020. – 448 с. – Режим доступа: https://dmkpress.com/catalog/computer/data/978-5-97060-774-9/?srsltid=AfmBOooN_wpyz69S-4uAZUr0JPDzKY-Ky9QJR-DyxVVG8I8UiY2GUn1w – Дата доступа: 24.04.2025.
4. Фаулер М., Шаблоны корпоративных приложений / М. Фаулер // Литрес [Электронный ресурс]. – Диалектика-Вильямс, 2003 – Режим доступа: https://www.litres.ru/book/martin-fauler/shablony-korporativnyh-prilozheniy-48637483/ – Дата доступа: 05.05.2025.
5. Aggarwal C. C., Recommender Systems / C. C. Aggarwal // Springer [Electronic resource]. – New York, 2016 – Mode of access: https://link.springer.com/book/10.1007/978-3-319-29659-3 – Date of access: 21.02.2025.
6. Annual Meeting of Stockholders of Netflix, Inc. [Electronic resource]. – https://ir.netflix.net/financials/annual-reports-and-proxies/default.aspx.
7. Campesato O., Transformer, Bert, and Gpt / O. Campesato // Degruyter [Electronic resource]. – MLI Generative AI Series, 2023 – Mode of access: https://www.amazon.com/Getting-Started-Google-BERT-state/dp/1838821597 – Date of access: 16.02.2025.
8. Devlin J., Lee M. K., Toutanova K., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / J. Devlin, M. K. Lee, K. Toutanova // ACL Anthology [Electronic resource]. – Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1. – 2019. – p. 4171–4186. – Mode of access: https://aclanthology.org/N19-1423/ – Date of access: 19.05.2025.
9. Fayyaz Z., Recommendation Systems: Algorithms, Challenges, Metrics, and Business Opportunities / Z. Fayyaz // ResearchGate [Electronic resource]. – 27th International Conference ICCBR, 2019 – Mode of access:

https://www.researchgate.net/publication/335551223_Personalized_Case-Based_Explanation_of_Matrix_Factorization_Recommendations – Date of access: 27.03.2025.
10. Guo, J., DeepFM: A Factorization-Machine based Neural Network for CTR Prediction / J. Guo // Arxiv [Electronic resource]. – Proceedings of the 26th International Conference on World Wide Web, 2017 – Mode of access: https://arxiv.org/abs/1703.04247 – Date of access: 12.02.2025.
11. Jorro-Aragoneses J. L., Recio-García J. A., Agudo B. D., Personalized case-based explanation of matrix factorization recommendations / J. L. Jorro-Aragoneses, J. A. Recio-García, B. D. Agudo // MDPI [Electronic resource]. – Appl. Sci., 2020 – Mode of access: https://doi.org/10.3390/app10217748 – Date of access: 28.03.2025.
12. Ravichandiran S., Getting Started with Google BERT: Build and train state-of-the-art natural language processing models using BERT / S. Ravichandiran // Amazon [Electronic resource]. – Packt Publishing, 2021 – Mode of access: https://www.degruyter.com/document/doi/10.1515/9781683928973/html – Date of access: 14.05.2025.
13. Rozanski N., Woods E., Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives / N. Rozanski, E. Woods E. // Amazon [Electronic resource]. – Addison-Wesley Professional, 2005 – Mode of access: https://www.amazon.com/Software-Systems-Architecture-Stakeholders-Perspectives/dp/0321112296 – Date of access: 12.01.2025.
14. YouTube, On YouTube's recommendation system [Electronic resource]. – https://blog.youtube/inside-youtube/on-youtubes-recommendation-system/ – Date of access: 13.05.2025.

## ПРИЛОЖЕНИЕ А

Код модуля программы, реализующего этап блендинга публикаций при формировании ленты рекомендаций

```
class PostsRetrievingService:
    def __init__(
        self, uow: RDBMSUnitOfWork, config: FeedApiConfig, cache: AbstractCache
    ) -> None:
        self._uow = uow
        self._config = config
        self._cache = cache
        self.rerank_proportions = {
            "is_subscription": 0.5,
            "is_community": 0.5,
            "is_question": 0.2,
            "is_vacancy_or_resume": 0.3,
        }
        self.special_publications_step = 5

    def blend_by_proportions(
        self, publications_with_flags_data: list[tuple[str, str, bool, bool, bool, bool]]
    ) -> list[tuple[str, str]]:
        columns = ["publication_id"] + list(self.rerank_proportions.keys())
        pub_id_to_original_pub_id_dict = {x[1]: x[0] for x in
publications_with_flags_data}
        flags_df = pd.DataFrame([x[1:] for x in publications_with_flags_data],
columns=columns)
        flag_columns = list(self.rerank_proportions.keys())
        flags_df["group"] = (
            flags_df[flag_columns].astype(int).astype(str).agg("".join, axis=1).apply(lambda
x: int(x, 2))
        )
        grouped_publications = flags_df.groupby("group").publication_id.apply(list)

        group_masks = {
            group: [bool(int(i)) for i in bin(group)[2:].zfill(len(self.rerank_proportions))]
            for group in range(2 ** len(self.rerank_proportions))
            if group in grouped_publications.index
        }
        group_probabilities = {
            group: prod(p if m else 1 - p for p, m in zip(self.rerank_proportions.values(),
mask))
            for group, mask in group_masks.items()
            if group in grouped_publications.index
        }
```

```
        result = []

        for _ in range(len(flags_df)):
            groups = list(group_probabilities.keys())
            probabilities = list(group_probabilities.values())

            group = np.random.choice(a=groups, p=(np.array(probabilities) /
np.array(probabilities).sum()))

            publication_group_list = grouped_publications[group]
            publication_id = publication_group_list.pop(0)
            result.append(publication_id)

            if not publication_group_list:
                del group_probabilities[group]

        return [(pub_id_to_original_pub_id_dict[x], x) for x in result]
```

## ПРИЛОЖЕНИЕ Б

Код программы для подбора числовых параметров для экспериментального исследования

```
tol0_type_cs = []
tol1_type_cs = []
biggest_type_tol1_diffs = {}
for C in good_base_C:
    print(f"Checking small c's for C={C:.2f}")
    biggest_type_tol1_diffs[C] = (0, 0, 0)
    tol0_before = len(tol0_type_cs)
    tol1_before = len(tol1_type_cs)
    for c1_int in range(4, 56, 2):
        c2_int = c1_int
        for c2_int in range(2, c1_int, 1):
            if c1_int - c2_int < 3 or c1_int - c2_int > 30:
                continue
            c3_int = 0
            if c2_int - c3_int < 3 or c2_int - c3_int > 30:
                continue
            c1 = 1 + c1_int / 100
            c2 = 1 + c2_int / 100
            c3 = 1 + c3_int / 100
            max_required_tolerance = check_rule_3(C, c1, c2, c3)
            if max_required_tolerance == 0:
                tol0_type_cs.append((C, c1, c2, c3))
            elif max_required_tolerance == 1:
                tol1_type_cs.append((C, c1, c2, c3))

                best_c1, best_c2, best_c3 = biggest_type_tol1_diffs[C]
                cur_eveness = (c1 - c2) + (c2 - c3)
                best_eveness = (best_c1 - best_c2) + (best_c2 - best_c3)
                if c1 - cur_eveness > best_c1 - best_eveness:
                    biggest_type_tol1_diffs[C] = (c1, c2, c3)

    print(f"Found {len(tol0_type_cs) - tol0_before} good c's with tolerance 0")
    print(f"Found {len(tol1_type_cs) - tol1_before} good c's with tolerance 1")
    print()  # max c's for C=0.65

print(len(tol0_type_cs), len(tol1_type_cs))
tol0_size_bounds = []
tol1_size_bounds = []
for C in good_base_C:
    print(f"Checking size bounds for C={C:.2f}")
    tol0_before = len(tol0_size_bounds)
```

```
    tol1_before = len(tol1_size_bounds)
    lb_int = 0
    for ub_int in range(1, 41, 1):
        lb = 1 + lb_int / 100
        ub = 1 + ub_int / 100
        max_required_tolerance = check_rule_4(C, lb, ub)
        if max_required_tolerance == 0:
            tol0_size_bounds.append((C, lb, ub))
        elif max_required_tolerance == 1:
            tol1_size_bounds.append((C, lb, ub))
    print(f"Found {len(tol0_size_bounds) - tol0_before} good size bounds with tolerance
0")
    print(f"Found {len(tol1_size_bounds) - tol1_before} good size bounds with tolerance
1")
    print()  # less 1's for C=0.50
# group results by C and tolerance
results = {
    'tol0': {
        'type': {},
        'size': {}
    },
    'tol1': {
        'type': {},
        'size': {}
    }
}
for C, c1, c2, c3 in tol0_type_cs:
    if C not in results['tol0']['type']:
        results['tol0']['type'][C] = []
    results['tol0']['type'][C].append((c1, c2, c3))

for C, c1, c2, c3 in tol1_type_cs:
    if C not in results['tol1']['type']:
        results['tol1']['type'][C] = []
    results['tol1']['type'][C].append((c1, c2, c3))

for C, lb, ub in tol0_size_bounds:
    if C not in results['tol0']['size']:
        results['tol0']['size'][C] = []
    results['tol0']['size'][C].append((lb, ub))

for C, lb, ub in tol1_size_bounds:
    if C not in results['tol1']['size']:
        results['tol1']['size'][C] = []
    results['tol1']['size'][C].append((lb, ub))

for C in good_base_C:
    print(f"Checking rule 5 for C={C:.2f}")
```

```
    for tol_type, tol_size in [
                            ('tol0', 'tol0'),
                            ('tol0', 'tol1'), ('tol1', 'tol0'), ('tol1', 'tol1')
                            ]:
        for _ in range(100):
            # get random combination of type and size parameters
            type_params = random.choice(results.get(tol_type, {}).get('type', {}).get(C,
[False]))
            size_params = random.choice(results.get(tol_size, {}).get('size', {}).get(C,
[False]))
            if not type_params or not size_params:
                break
            if check_rule_5(C, *type_params, *size_params, iterations=200):
                print(f"Found good parameters for C={C:.2f}, {(tol_type, tol_size)}:
c1={type_params[0]:.2f}, c2={type_params[1]:.2f}, c3={type_params[2]:.2f},
lb={size_params[0]:.2f}, ub={size_params[1]:.2f}")

print(biggest_type_tol1_diffs)
```