УДК 004.89
EDN: TXHGGP

Исследование методов построения RAG-систем на основе CSV-документов для повышения точности информационного поиска

Слободчиков И. Д., Плотников П. В. ✉

Санкт-Петербургский государственный университет телекоммуникаций им. проф. М. А. Бонч-Бруевича, Санкт-Петербург, 193232, Российская Федерация

Постановка задачи. *Неэффективность традиционных методов информационного поиска при работе с большими объемами структурированных данных (CSV-документов) в корпоративной среде актуализирует задачу разработки специализированных решений на базе архитектуры RAG (Retrieval-Augmented Generation). Существующие подходы, включая классический RAG, демонстрируют недостаточную точность при обработке табличных данных.* ***Цель работы*** *– разработка и экспериментальная валидация инженерных реализаций RAG-систем, адаптированных для точного поиска информации в структурированных CSV-документах.* ***Используемые методы.*** *В работе предложены и реализованы четыре подхода: 1) Эталонный классический RAG с векторной базой данных; 2) RAG с предварительной текстовой трансформацией CSV; 3) SQL-агент с точным соответствием запросов; 4) SQL-агент, интегрированный с фильтром нечеткого поиска. Экспериментальное исследование проведено на корпоративных данных.* ***Элементами новизны*** *являются адаптация архитектуры RAG для обработки строго структурированных CSV-документов и разработка гибридных инженерных решений, сочетающих большие языковые модели с генерацией SQL-запросов и механизмами нечеткого поиска, специально оптимизированными для табличных данных.* ***Результат.*** *Экспериментально доказано, что предложенные SQL-агенты (реализации 3 и 4) обеспечивают точность информационного поиска до 94,8 %. Данный результат на 29,4 % превышает точность эталонного классического RAG-подхода (65,4 %), что подтверждает эффективность специализированных решений для работы со структурированными данными.* ***Теоретическая / Практическая значимость.*** *Разработанные решения предназначены для внедрения в корпоративные системы аналитики и принятия решений. Они позволяют значительно повысить эффективность работы менеджеров и аналитиков со статистическими данными, хранящимися в табличных форматах, за счет обеспечения высокоточного и естественно-языкового доступа к информации. Предложенная архитектура открывает возможность интеграции с существующими бизнес-инструментами и базами данных.*

Ключевые слова: *RAG, векторные базы данных, SQL-агенты, CSV-документы, точность поиска, обработка структурированных данных*

Актуальность исследования

В условиях цифровой трансформации предприятий возникает важная задача эффективной обработки корпоративных структурированных данных с помощью систем искусственного интеллекта. Технология поисковой дополненной генерации (RAG – *аббр. от англ.* Retrieval-Augmented Generation)

Библиографическая ссылка на статью:
Слободчиков И. Д., Плотников П. В. Исследование методов построения RAG-систем на основе CSV-документов для повышения точности информационного поиска // Вестник СПбГУТ. 2025. Т. 3. № 4. С. 3. EDN: TXHGGP

Reference for citation:
Slobodchikov I., Plotnikov P. Research of RAG System Construction Methods Based on CSV Documents for Improving Information Retrieval Accuracy // Herald of SPbSUT. 2025. Vol. 3. Iss. 4. P. 3. EDN: TXHGGP

получила широкое распространение как инструмент создания интеллектуальных систем, способных работать с внутренними информационными базами организаций [1]. Однако стандартные RAG-системы разработаны и оптимизированы для работы с неструктурированным текстом. При работе со структурированными данными в текстовом формате для хранения табличных данных (CSV – *аббр. от англ*. Comma-Separated Values), широко распространенными в бухгалтерском учете, управлении персоналом и аналитике, возникают фундаментальные проблемы точности поиска [2].

Данное исследование сосредоточено не на теоретическом анализе существующих методов, а на разработке и апробации трех собственных инженерных решений для адаптации RAG-технологии к специфике табличных данных. Каждая реализация представляет собой самостоятельный подход, разработанный и оптимизированный для решения конкретных аспектов задачи обработки CSV-документов, таких как адаптация через текстовую трансформацию и явное указание контекста полей; адаптация через преобразование запросов в SQL-команды (*аббр. от англ*. Structured Query Language – язык структурированных запросов) для управления базами данных с точным соответствием; расширенная адаптация с использованием нечеткой логики для повышения гибкости обработки.

Кроме того, исследование демонстрирует, что специально разработанные адаптации RAG-систем могут существенно превосходить базовый подход при обработке структурированных данных. При этом традиционные RAG-системы, основанные на векторных базах данных, показывают высокую эффективность при работе с неструктурированными текстовыми документами [1, 3]. Однако при обработке структурированных CSV-файлов возникают специфические проблемы, обусловленные самой природой векторизации.

Фундаментальные ограничения классических RAG для CSV-данных

Классические RAG-системы используют векторное представление (эмбеддинги) для преобразования текста в многомерные векторы, выполняя семантический поиск на основе близости в этом пространстве [1, 3]. При работе с CSV-документами этот подход сталкивается с серьезными ограничениями [4]:

– *семантическое сходство вместо точного соответствия*: векторные представления отражают смысловую близость, а не точное совпадение значений; система может найти «похожую» запись вместо точно соответствующей [4];

– *потеря структурной информации*: контекст связи между полями и значениями часто теряется при векторизации, особенно при работе с категориальными и числовыми данными;

– *проблемы с точными поисками*: исследования показывают, что точность векторного поиска в структурированных данных редко превышает 65–70 % даже с использованием современных моделей эмбеддингов [4, 5].

Эти ограничения делают классический RAG неприемлемым для корпоративных приложений, где ошибки при поиске финансовых данных, идентификаторов сотрудников или учетной информации могут привести к серьезным последствиям.

Разработанные реализации адаптаций RAG

Исследование включает экспериментальную апробацию четырех конфигураций: первая из них представляет собой эталонную реализацию классического RAG на базе готовой библиотеки и выступает в качестве базовой модели (эталона), тогда как остальные три являются авторскими адаптациями, разработанными с нуля.

Реализация 1: Эталонный классический RAG с векторной базой данных представляет собой классическую архитектуру RAG, используемую в качестве эталона для демонстрации ограничений векторных методов при работе с табличными данными.

CSV-данные загружаются и разбиваются на семантические единицы (фрагменты), каждый из которых преобразуется в векторное представление с использованием модели Sentence-

Transformers (многоязычной модели эмбеддингов) [2, 5]. Полученные векторы индексируются в векторной базе данных FAISS (FAISS, *аббр от англ.* Facebook AI Research Similarity Search – разработка команды Facebook AI Research для быстрого поиска ближайших соседей и кластеризации в векторном пространстве) с применением стандартного алгоритма индексирования для эффективного сопоставления запросов.

При обработке пользовательского запроса система преобразует его в вектор с помощью той же модели эмбеддингов и выполняет поиск K-ближайших соседей в векторном пространстве, возвращая наиболее похожие фрагменты.

Преимущества реализации состоят в высокой скорость поиска (в среднем 0,8 с), хорошей адаптивностью системы к семантическим запросам общего характера (точность 78,5 % на категориях с общими формулировками), масштабируемости благодаря эффективным индексам и устойчивости к синонимам и вариациям формулировок.

К числу ограничений реализации относятся: невысокая полнота совпадений при поиске по точным значениям (48,7 %), отсутствие гарантий строгого соблюдения числовых критериев, недетерминированность выдачи (один и тот же запрос может возвращать разные результаты), а также сложности при выполнении агрегирующих запросов.

Реализация 2: RAG с предварительной текстовой трансформацией CSV является авторской разработкой, направленной на сохранение структурной информации в CSV-данных путем их преобразования в структурированный текстовый формат перед векторизацией.

На этапе предварительной обработки каждая строка CSV преобразуется в явную текстовую форму со строго определенным форматом:

Исходная строка в CSV выглядит так: Иванова | инженер | 3 | отдел_IT.

Преобразованная форма имеет вид: ФИО: Иванова Ирина Олеговна | Должность: инженер | Категория: 3 | Подразделение: отдел_IT.

Такое преобразование выполняется для всех строк CSV, после чего трансформированный текст подлежит стандартной векторизации и индексированию в FAISS. Явное указание названий полей в текстовом представлении позволяет векторным моделям лучше понимать структуру и семантику данных.

Преимуществами реализации являются: улучшенное понимание контекста векторными моделями (точность возрастает до 78,2 %), лучшая производительность на категориальных запросах (Precision: 74,8 %, Recall: 81,3 %), простота внедрения (требует минимальных модификаций стандартной RAG-системы), а также F1-мера 0,784 – метрика, представляющая собой гармоническое среднее точности и полноты, что означает улучшение сбалансированной эффективности на 19,8 % относительно базовой реализации.

Ограничения реализации состоят в сохранении фундаментальных проблем точного поиска, присущих векторным методам, увеличении объема индексируемых данных за счет метаинформации, дополнительных вычислительных затратах на предварительную обработку, ограниченной эффективности для числовых диапазонов и условных запросов.

Основные инженерные решения здесь – это использование детерминированного преобразования для воспроизводимости, кэширование трансформированных текстов для оптимизации и дополнительная нормализация значений перед текстовым преобразованием.

Реализация 3: SQL-агент с точным соответствием запросов представляет собой авторскую архитектуру, где RAG-парадигма адаптируется через использование агентов на основе больших языковых моделей для преобразования естественно-языковых запросов в структурированные SQL-команды.

CSV-данные загружаются в реляционную базу данных (PostgreSQL) с явно определенной схемой. Большая языковая модель (LLM, *аббр от англ*. Large Language Model) использует описание схемы базы данных для преобразования пользовательского запроса на естественном языке в синтаксически корректный SQL-запрос [6, 7], который выполняется на реальных данных в базе, гарантируя точное соответствие и полноту результатов.

К ключевым компонентам реализации относятся: модуль определения схемы, анализирующий структуру CSV и формирующий описание таблиц и полей для LLM; агент преобразования

NL-to-SQL, использующий концепцию prompt engineering для обеспечения корректного преобразования запросов на примерах; валидатор SQL-запросов, который проверяет синтаксис и безопасность перед выполнением; модуль постобработки результатов, форматирующий результаты запроса в естественный язык для пользователя.

Преимущества реализации состоят в повышении уровня точности до 94,8 % (улучшение на 29,4 % относительно базовой реализации), обеспечении высокой точности поиска (Precision: 96,2 %, Recall: 93,5 %), поддержке сложных условий и агрегаций (точность 97,1 % на агрегирующих запросах), получении детерминированных результатов (т. е. один и тот же запрос всегда возвращает одинаковый результат) и сохранении гарантии точного соответствия критериям.

Отметим ряд ограничений реализации: увеличенное время отклика (в среднем 2,1 с), необходимо учитывать точное соответствие терминологии в пользовательских запросах, модель может испытывать трудности при обработке сложно сформулированных или двусмысленных запросов, система требует явного определения схемы базы данных.

При реализации были предложены следующие основные инженерные решения: в запросы к модели (промпты) добавлено несколько готовых примеров «вопрос → правильный результат» (few-shot), чтобы модель стабильнее выполняла нужное преобразование; предусмотрен запасной сценарий обработки (fallback) на случай, если преобразование не удалось, – система перезапускает попытку с упрощенными правилами или возвращает безопасный результат вместо ошибки; для ускорения работы заранее подготовлены быстрые обращения к наиболее востребованным данным (индексирование часто используемых полей); также введены ограничения на «тяжесть» формируемых запросов (например, на число условий и соединений), чтобы выполнение не прерывалось из-за превышения лимита времени на один запрос.

Реализация 4: SQL-агент, интегрированный с фильтром нечеткого поиска, расширяет третий вариант путем интеграции методов нечеткой логики для повышения гибкости обработки пользовательских запросов при сохранении высокой точности.

Система строится на основе SQL-агента (как в реализации 3), но дополняется функциями нечеткого поиска для обработки запросов с возможными опечатками, вариантами написания или неточностями в исходных данных [8]. Для этого используются операторы LIKE (поиск по шаблону с символами подстановки) и ILIKE (аналог LIKE без учета регистра), а также метрика редакционного расстояния (edit distance) – мера минимального количества односимвольных изменений (вставок, удалений, замен), необходимых для превращения одной строки в другую, что позволяет находить близкие по написанию, но не идентичные значения. Применение принципов нечеткой логики позволяет системе оперировать приближенными значениями в контексте реляционных операций [9, 10].

Архитектура системы построена на последовательном взаимодействии нескольких технических компонентов. В первую очередь за обработку пользовательского ввода отвечает модуль коррекции опечаток, который оценивает сходство строк с помощью алгоритма Левенштейна. На основе очищенных данных адаптивный SQL-генератор формирует запросы к базе, заменяя оператор строгого равенства «=» на условие LIKE, если в исходном тексте обнаруживаются потенциальные неточности. После извлечения информации в работу включается модуль ранжирования, сортирующий итоговую выдачу по степени релевантности первоначальному запросу. При этом чувствительность такого нечеткого поиска можно настраивать под требования конкретного приложения с помощью системы пороговых регуляторов.

Такая архитектура обеспечивает ряд существенных преимуществ. Главным из них является высокая устойчивость системы к опечаткам и вариативности формулировок, благодаря чему общая точность преобразования достигает 91,3 %. Гибкая обработка неточных запросов позволяет модели демонстрировать отличные метрики качества: точность (Precision) составляет 89,7 %, а полнота (Recall) – 92,8 %. Встроенное ранжирование результатов помогает алгоритму находить оптимальный компромисс между строгой математической логикой SQL и естественной неоднозначностью человеческого языка.

Тем не менее у предложенного подхода существуют и объективные ограничения. Вычислительная сложность нечеткого поиска и генерации приводит к наибольшему времени выполнения

среди всех рассмотренных методов – в среднем 2,8 с на запрос. Кроме того, алгоритм требует сложной ручной настройки параметров фильтрации и пороговых значений. Если задать слишком мягкие критерии, значительно возрастает риск ложных срабатываний. Из-за этого при внедрении системы в критически важные бизнес-процессы результаты ее работы обязательно требуют дополнительной валидации со стороны пользователя.

Основные инженерные решения в реализации – это адаптивные пороги в зависимости от типа поля (строковые или числовые), кэширование результатов нечеткого поиска для оптимизации, явное указание уровня доверия к результатам в ответах системы.

Экспериментальное исследование и результаты. Методология оценки

Для объективной оценки эффективности разработанных реализаций использовались стандартные метрики качества информационного поиска [11, 12]:

Accuracy (точность) – доля правильно классифицированных результатов от общего числа запросов;

Precision (точность извлечения) – доля действительно релевантных результатов среди всех извлеченных;

Recall (полнота) – доля найденных релевантных результатов от всех существующих в базе;

F1-мера – гармоническое среднее Precision и Recall;

Время отклика – среднее время обработки запроса (критично для корпоративных приложений).

Тестирование проводилось на наборе из 500 подготовленных запросов различной сложности, разделенных на следующие категории:

Простые точные поиски (150 запросов): поиск по одному критерию, требующий точного соответствия;

Категориальные запросы (125 запросов): фильтрация по одному или нескольким категориальным полям;

Числовые диапазоны (100 запросов): запросы с условиями на диапазоны значений;

Агрегирующие запросы (75 запросов): подсчет, суммирование, вычисление среднего;

Сложные комбинированные запросы (50 запросов): комбинация нескольких условий различных типов.

Результаты оценки представлены в таблице 1, а итоги тестирования разных типов запросов – в таблице 2.

Таблица 1. Общие результаты точности реализаций

Реализация	Accuracy, %	Precision, %	Recall, %	F1-мера	Время отклика, с
1. Эталонный классический RAG с векторной базой данных	65,4	61,2	70,0	0,655	0,8
2. RAG с предварительной текстовой трансформацией CSV	78,2	74,8	81,3	0,784	0,9
3. SQL-агент с точным соответствием запросов	94,8	96,2	93,5	0,949	2,1
4. SQL-агент, интегрированный с фильтром нечеткого поиска	91,3	89,7	92,8	0,913	2,8

Таблица 2. Детальное сравнение точности реализаций по категориям запросов

Категория запроса	Реализация 1	Реализация 2	Реализация 3	Реализация 4
Простые точные поиски, %	48,7	68,2	98,3	96,7
Категориальные запросы, %	72,0	81,6	96,4	94,2
Числовые диапазоны, %	52,3	71,4	95,8	93,1
Агрегирующие запросы, %	81,2	85,3	97,1	92,4
Сложные комбинированные запросы, %	65,4	76,8	91,7	89,6

Анализ результатов

Реализация 1 (Эталонный классический RAG с векторной базой данных) достигает 65,4 % точности, демонстрируя фундаментальные ограничения векторных методов для структурированных данных. Наиболее критичные результаты получены на простых точных поисках (48,7 %) и числовых диапазонах (52,3 %), что объясняется природой векторного поиска. На семантических запросах общего характера (категориальные – 72,0 %, агрегирующие – 81,2 %) система показывает лучшие результаты, подтверждая ее применимость для неструктурированного поиска.

Реализация 2 (RAG с предварительной текстовой трансформацией CSV) повышает общую точность до 78,2 % (улучшение на 12,8 %), что демонстрирует значимость контекстной информации для векторных моделей. Наибольший прирост наблюдается на простых точных поисках (с 48,7 до 68,2 %) и числовых диапазонах (с 52,3 до 71,4 %), где явное указание названий полей помогает модели лучше понимать запросы. F1-мера 0,784 свидетельствует о хорошем балансе между полнотой и точностью для векторного метода.

Реализация 3 (SQL-агент с точным соответствием запросов) показывает существенное преимущество с точностью 94,8 %, что на 29,4 процентных пункта превышает базовый подход. Результаты практически идентичны на всех категориях запросов: простые точные поиски – 98,3 %, категориальные – 96,4 %, числовые диапазоны – 95,8%, агрегирующие – 97,1%. Высокие значения метрик Precision (96,2 %) и Recall (93,5 %) подтверждают, что SQL-агент обеспечивает как точность классификации, так и полноту извлечения. Увеличенное время отклика (2,1 с) остается приемлемым для корпоративных приложений, где точность важнее скорости.

Реализация 4 (SQL-агент, интегрированный с фильтром нечеткого поиска) достигает 91,3 % точности при сохранении гибкости обработки неточных запросов. Результаты немного ниже реализации 3 на всех категориях (простые поиски – 96,7 %, категориальные – 94,2 %), однако система предоставляет бóльшую гибкость при обработке опечаток и вариаций. Precision 89,7 % и Recall 92,8 % указывают на хороший баланс между точностью и полнотой в контексте нечеткого поиска.

Анализ компромисса между точностью скоростью

Экспериментальные данные обнаруживают явный компромисс между точностью результатов и скоростью обработки. Классический RAG обеспечивает минимальное время (0,8 с), но имеет низкую точность (65,4 %). Текстовая адаптация также имеет минимальные затраты времени (0,9 с) при значительном улучшении точности (78,2 %). SQL точное соответствие позволяет достигнуть баланса в пользу точности (94,8 %) с приемлемым временем (2,1 с). SQL нечеткий поиск – это гибкий и точный (91,3 %) подход при максимальном времени отклика (2,8 с).

Для корпоративных приложений, предъявляющих требования к безопасности (финансовые отчеты, персональные данные), преимущество точности реализаций 3 и 4 явно перевешивает увеличение времени обработки. Для поисков, где скорость имеет приоритетное значение, реализации 1 и 2 остаются приемлемыми вариантами.

Практическое значение и рекомендации

На основе полученных результатов предложены следующие рекомендации для внедрения в различные корпоративные сценарии.

Для высокочувствительных приложений (финансовых и учетных данных, HR) использование реализации 3 (SQL-агент с точным соответствием запросов) обязательно. Точность 94,8 % и гарантированная детерминированность результатов критичны при предотвращении ошибок с серьезными последствиями.

Для приложений с потенциально неточными запросами наилучшим образом подойдет реализация 4 (SQL-агент, интегрированный с фильтром нечеткого поиска), обеспечивающая баланс между точностью (91,3 %) и устойчивостью к пользовательским ошибкам, что делает ее оптимальной для систем с широкой аудиторией пользователей.

Если приложение требовательно к скорости, стоит использовать реализацию 2 (RAG с предварительной текстовой трансформацией CSV), которая предоставляет значительное улучшение точности (12,8 % над базовым подходом) при минимальном увеличении времени отклика (дополнительно всего 0,1 с).

При решении задач поиска в больших объемах неструктурированного контента наиболее эффективным будет эталонный классический RAG с векторной базой данных (реализация 1), остающийся оптимальным выбором благодаря скорости и высокой масштабируемости, несмотря на низкую точность на структурированных данных.

Исследование демонстрирует, что парадигма RAG может быть успешно адаптирована для обработки структурированных CSV-данных при условии специализированного инженерного подхода к архитектуре системы. Разработанные подходы показывают, что простые модификации в обработке данных (реализация 2) дают существенное улучшение точности, но не снимают фундаментальные ограничения векторных методов. Переход к структурированным методам обработки (реализации 3 и 4) кардинально изменяет результаты: точность возрастает с 65–78 % до 91–95 %. Гибридный подход (реализация 4) обеспечивает практический компромисс между точностью и гибкостью для реальных корпоративных применений.

Заключение

Проведенное исследование подтверждает, что парадигма RAG может быть эффективно адаптирована для обработки структурированных данных в формате CSV при применении специализированных инженерных подходов. Разработанные и апробированные реализации демонстрируют качественное превосходство структурированных методов обработки данных (реализации 3 и 4) над традиционными векторными подходами (реализация 1), достигая точности 91,3–94,8 % против 65,4 % у эталонной системы.

Практическая ценность исследования заключается в том, что разработанные реализации могут быть применены в корпоративной среде для решения критически важных задач информационного поиска в табличных данных. Результаты показывают, что инженерные адаптации RAG-систем под специфику структурированных данных могут улучшить точность на 29–43 % по сравнению с классическими методами.

Литература

1. Lewis P., Piktus A., Petroni F., Karpukhin V., Goyal N., et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // Advances in Neural Information Processing Systems. Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS, December 06–12, 2020). 2020. Vol. 33. EDN: JPMMQE

2. Herzig J., Nowak P. K., Müller T., Piccinno F., Eisenschlos J. TaPas: Weakly Supervised Table Parsing via Pre-Training // Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL, July 5–10, 2020). 2020. DOI: 10.48550/arXiv.2004.02349

3. Karpukhin V., Oguz B., Min S., Lewis P., Wu L., Edunov S., Chen D., Yih W. T. Dense Passage Retrieval for Open-Domain Question Answering // Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP, November 16–20, 2020). 2020. DOI: 10.48550/arXiv.2004.04906

4. Liang X., Hu R., Liu Y., Zhu K. Open-Domain Question Answering over Tables with Large Language Models // Advanced Intelligent Computing Technology and Applications: Proceedings of the 20th International Conference on Intelligent Computing (ICIC, August 5–8, 2024, Tianjin, China). 2024. PP. 347–358.

5. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks // Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP–IJCNLP, November 3–7, 2019, Hong Kong, China). 2019. DOI: 10.48550/arXiv.1908.10084

6. Yu T., Zhang R., Yang K., Yasunaga M., Wang D., et al. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task // Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP, October 31 – November 4, 2018, Brussels, Belgium). 2018. DOI: 10.48550/arXiv.1809.08887

7. Pourreza M., Rafiei D. DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction // Advances in Neural Information Processing Systems (NeurIPS, December 10–16, 2023, New Orleans, USA). 2023. DOI: 10.48550/arXiv.2304.11015

8. Левенштейн В. И. Двоичные коды с исправлением выпадений, вставок и замещений символов // Доклады Академии наук СССР. 1965. Т. 163. № 4. С. 845–848.

9. Ротштейн А. П., Штовба С. Д. Нечеткая надежность алгоритмических процессов. Винница: Континент-ПРИМ, 1997. 142 с.

10. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы. М.: Горячая линия–Телеком, 2006. 384 с.

11. Manning C. D., Raghavan P., Schütze H. Introduction to Information Retrieval. Cambridge: Cambridge University Press, 2008. 569 p.

12. Foody G. M. Challenges in the Real World Use of Classification Accuracy Metrics: From Recall and Precision to the Matthews Correlation Coefficient // PLOS ONE. 2023. Vol. 18. Iss. 10. DOI: 10.1371/journal.pone.0291908. EDN: LRZNQD

Статья поступила 19 ноября 2025 г.
Одобрена после рецензирования 20 декабря 2025 г.
Принята к публикации 24 декабря 2025 г.

Информация об авторах

Слободчиков Иоан Дмитриевич – студент 4-го курса факультета информационных технологий и программной инженерии Санкт-Петербургского государственного университета телекоммуникаций им. проф. М. А. Бонч-Бруевича. E-mail: slobodchikov.id@sut.ru

Плотников Павел Владимирович – кандидат физико-математических наук, доцент, заведующий кафедрой высшей математики Санкт-Петербургского государственного университета телекоммуникаций им. проф. М. А. Бонч-Бруевича. E-mail: plotnikov.pv@sut.ru

Research of RAG System Construction Methods Based on CSV Documents for Improving Information Retrieval Accuracy

I. Slobodchikov, P. Plotnikov ✉

The Bonch-Bruevich Saint Petersburg State University of Telecommunications,
St. Petersburg, 193232, Russian Federation

Purpose. *The inefficiency of traditional information retrieval methods when working with large volumes of structured data (CSV documents) in the corporate environment actualizes the task of developing specialized solutions based on the RAG (Retrieval-Augmented Generation) architecture. Existing approaches, including classical RAG, demonstrate insufficient accuracy when processing tabular data. The objective of the work is to develop and experimentally validate engineering implementations of RAG systems adapted for accurate information retrieval in structured CSV documents.* ***Methods.*** *In the work, four approaches are proposed and implemented: 1) Reference classical RAG with a vector database; 2) RAG with preliminary text transformation of CSV; 3) SQL agent with exact query match; 4) SQL agent integrated with a fuzzy search filter. The experimental study was conducted on corporate data.* ***The elements of novelty*** *include adapting the RAG architecture to process strictly structured CSV documents and developing hybrid engineering solutions that combine large language models with SQL query generation and fuzzy search mechanisms specifically optimized for tabular data.* ***Results.*** *It has been experimentally proven that the proposed SQL agents (implementations 3 and 4) provide information retrieval accuracy of up to 94.8 %. This result exceeds the accuracy of the reference classical RAG approach (65.4 %) by 29.4 %, confirming the effectiveness of specialized solutions for working with structured data.* ***Practical relevance.*** *The developed solutions are intended for implementation in corporate analytics and decision-making systems. They significantly improve the efficiency of managers and analysts with statistical data stored in tabular formats by providing highly accurate and natural-language access to information. The proposed architecture opens the possibility of integration with existing business tools and databases.*

Key words: *RAG, vector databases, SQL-agents, CSV-documents, retrieval accuracy, structured data processing*

Information about Authors

Slobodchikov Ioan – 4th-year student of Faculty of Information Technology and Software Engineering (The Bonch-Bruevich Saint Petersburg State University of Telecommunications).
E-mail: slobodchikov.id@sut.ru

Plotnikov Pavel – Ph. D. of Physical and Mathematical Sciences, Associate Professor, Head of the Department of Higher Mathematics (The Bonch-Bruevich Saint Petersburg State University of Telecommunications). E-mail: plotnikov.pv@sut.ru