МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

РАЗРАБОТКА API ДЛЯ СЕМАНТИЧЕСКОГО ПОИСКА ПО ТЕКСТУ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы
направления 02.03.03 Математическое обеспечение и администрирование
информационных систем
факультета компьютерных наук и информационных технологий
Гражданова Александра Константиновича

Научный руководитель:
к.ф.-м.н., доцент _____ Огнева М. В.
подпись, дата

Зав. кафедрой:
к.ф.-м.н., доцент _____ Огнева М.В.
подпись, дата

Саратов 2025

ВВЕДЕНИЕ

Актуальность темы. Семантический поиск по текстам представляет собой современный подход к обработке информации, направленный на понимание смысла запросов и документов. В отличие от традиционного подхода, основанного на точном совпадении ключевых слов, семантический поиск учитывает контекст и семантические связи, что позволяет улучшить релевантность результатов поиска.

Актуальность этой темы подтверждается в статье [1], авторы подчеркивают фундаментальные ограничения традиционных систем поиска на основе ключевых слов, которые не способны интерпретировать сложные пользовательские интенции, синонимы или контекстные нюансы. Взамен в данной статье предлагают использовать подход основанный на поиске по векторным представлениям текста.

В данной работе будет проведен анализ методов, лежащих в основе семантического поиска, с акцентом на векторное представление текста, методы измерения расстояний между векторами и использование специализированных баз данных, также будет разработано API (Application Programming Interface) для организации семантического поиска по документам.

Цель бакалаврской работы – разработать API для семантического поиска по текстам.

Поставленная цель определила **следующие задачи**:

1. Рассмотреть метод поиска по ключевым словам.
2. Рассмотреть способы представления текста в обработке естественных языков и информационном поиске.
3. Изучить методы измерения расстояния векторными представления текста.
4. Оценить эффективность векторных баз данных в условиях больших объёмов данных, включая анализ производительности, масштабируемости и точности.

5. Провести тестирование векторных представлений на реальных данных и измерить ключевые метрику.
6. Реализовать микросервисную архитектуру приложения с интеграцией предобученных моделей и специализированных хранилищ, обеспечив конвейер обработки данных: от индексации текстов до семантического поиска.

Методологические основы исследования в области семантического поиска представлены в работах А. Алувалия [1], Ц. Ван [2], Г. Чани [3], Д. Радев [4], М. Хагивара [5].

Практическая значимость бакалаврской работы заключается в комплексном исследовании методов семантического поиска, что проявляется в систематизации современных подходов к векторизации текста (TF-IDF, Word2Vec, SBert), сравнительном анализе их эффективности на реальных данных MS MARCO с использованием метрики MRR@10, а также в выявлении критериев выбора векторных баз данных на основе оценки производительности и точности. Практическая ценность подтверждается реализацией готового к внедрению API с микросервисной архитектурой, обеспечивающего конвейер обработки документов — от сегментации текстов и генерации эмбеддингов до семантического поиска в Qdrant, что позволяет интегрировать решение в корпоративные и научные системы для работы с неструктурированными текстами, продемонстрировано на примере поиска публикаций arXiv.org.

Структура и объём работы. Бакалаврская работа состоит из введения, 8 разделов, заключения, списка использованных источников и 4 приложений. Общий объем работы – 84 страниц, из них 46 страниц – основное содержание, включая 4 рисунков и 5 таблиц, список использованных источников информации – 22 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Поиск по ключевым словам» посвящен анализу традиционных подходов к информационному поиску, основанных на лексическом совпадении терминов. В рамках раздела систематизированы принципы работы алгоритмов Okapi BM25 и TF-IDF, включая формулы расчета релевантности с учетом частоты терминов, длины документов и нормализации. Рассмотрено сравнение эффективности данных методов на разнородных текстовых корпусах. Ключевыми понятиями являются *релевантность* (соответствие документа запросу), *нормализация длины* (компенсация смещения в сторону длинных текстов через параметр b в BM25), а также фундаментальные ограничения методов: игнорирование контекста, синонимии и семантических связей. Выводы раздела подтверждают, что традиционные подходы эффективны для задач точного совпадения ключевых слов, но неприменимы для семантического поиска, при этом BM25 демонстрирует преимущество перед TF-IDF в устойчивости к вариациям длины документов.

Второй раздел «Поиск по векторным представлениям» исследует нейросетевые и статистические методы преобразования текста в векторные пространства. В разделе проанализированы статистические подходы (мешок слов, TF-IDF), нейросетевые архитектуры (Word2Vec, SBert) и ограничения векторных представлений на примере ElasticSearch и Word2Vec. Центральными понятиями выступают *эмбеддинг* (векторное представление текста), *контекстная чувствительность* (способность моделей учитывать семантику, где SBert превосходит Word2Vec) и роль гиперпараметров (размерность векторов, методы пулинга). Выводы подчеркивают, что современные трансформерные модели (SBert) обеспечивают глубокое понимание контекста, но требуют значительных ресурсов, тогда как статистические методы сохраняют актуальность для узких предметных областей.

Третий раздел «Расчет векторных представлений для больших текстов» посвящен оптимизации обработки объемных документов. Основное

внимание уделено алгоритмам сегментации: *chunking* (разбиение по семантическим границам) и *striding* (перекрытие блоков), а также методам сохранения контекстной целостности, таким как рекурсивное иерархическое разделение. Самостоятельная разработка включает адаптивный алгоритм сегментации текста на блоки по 256 токенов с 10% перекрытием, реализованный на Python. Выводы раздела свидетельствуют, что сегментация с перекрытием минимизирует потерю смысла на границах блоков, а оптимальная длина сегмента для моделей типа BERT составляет 200–300 токенов.

Четвертый раздел «Расстояние между векторами» рассматривает метрики сравнения векторных представлений. Проанализированы свойства евклидова расстояния, косинусной меры, манхэттенского расстояния, их чувствительность к выбросам, зависимость от размерности данных и вычислительная сложность. Экспериментально установлено, что для семантического поиска косинусное расстояние предпочтительнее благодаря инвариантности к масштабу векторов и ориентации на смысловую близость. Итоговые выводы указывают, что выбор метрики определяется задачей: косинусная мера оптимальна для поиска, евклидова — для кластеризации.

Пятый раздел «Метрика MRR@k» посвящен оценке качества ранжирования поисковых систем. Ключевые аспекты включают формулу расчета метрики (среднее обратного ранга первого релевантного результата в топ-k), зависимость от поведения пользователей (74% взаимодействуют с топ-5 выдачи) и эмпирическое обоснование выбора k=10 для тестирования. Выводы раздела доказывают, что MRR@k является критичной метрикой для оптимизации пользовательского опыта, а ее применение подтвердило эффективность SBert (MRR@10=0.51) против TF-IDF (0.09).

Шестой раздел «Векторные базы данных», данный раздел представляет детальный сравнительный анализ современных систем хранения векторных данных, включая Qdrant, Milvus, Weaviate и ElasticSearch.

Исследование проводилось на реальном датасете MS MARCO объемом 10 миллионов векторных представлений с использованием единой платформы тестирования Vector DB Benchmark. Ключевыми критериями оценки стали производительность (количество запросов в секунду RPS), время отклика (latency), стабильность работы (P95 latency), скорость индексации данных (Index Time) и точность поиска (Recall@10). Экспериментальные результаты выявили лидерство Qdrant по скорости обработки запросов — 1260 RPS при задержке 3.26 мс и точности 0.96, что в 4 раза превышает показатели ElasticSearch. Последний показал наивысшую точность (0.98), но критически высокое время индексации — 5.5 часов для 10 млн векторов. Redis обеспечил минимальную задержку в однопоточном режиме, но не справился с параллельными запросами. Технические преимущества Qdrant включают реализацию на языке Rust, обеспечивающую нулевые накладные расходы на сборку мусора, динамическое переключение между типами индексов, поддержку гибридных запросов и встроенные механизмы шардирования. Выводы раздела подтверждают, что Qdrant оптимален для промышленных решений, требующих обработки миллиардов векторов, тогда как ElasticSearch рекомендован для задач с приоритетом точности над скоростью.

Седьмой раздел «Практическая часть» описывает В данном разделе представлена комплексная разработка API для семантического поиска, где ключевым аспектом стало экспериментальное сравнение алгоритмов векторизации текста на датасете MS MARCO (102,000 запросов и 755,000 пассажей). Оценка проводилась с использованием метрики MRR@10, отражающей пользовательский опыт взаимодействия с топ-10 результатами.

Модель SBert (all-MiniLM-L6-v2) продемонстрировала наивысшую точность (MRR@10=0.51) благодаря способности учитывать контекст предложений через механизм mean-pooling скрытых состояний, генерируя 384-мерные векторы. Её ключевое преимущество — сохранение семантических нюансов и порядка слов, хотя это сопровождается

ресурсоемкостью (2.8 сек/1000 текстов на GPU Tesla T4). В сравнении, Word2Vec (Google News 300) показал значительно более низкую эффективность: метод усреднения векторов слов достиг MRR@10=0.06, а PCA-проекция (выделение главного смыслового направления) улучшила результат лишь до 0.11, что выявило проблемы потери контекста при объединении векторов.

TF-IDF с биграммами продемонстрировал стабильный результат (MRR@10=0.09-0.10), но с критической зависимостью от предобработки: удаление стоп-слов повысило точность на 11%, при этом наблюдался экспоненциальный рост потребления памяти (>32 ГБ для 500K документов). Интересной аномалией стало влияние предобработки на разные алгоритмы: лемматизация *снизила* эффективность SBert (0.46 против 0.51) из-за потери контекстных сигналов, тогда как для TF-IDF она оказалась полезной, а в Word2Vec с PCA-агрегацией привела к деградации точности до 0.04.

На основе этих результатов для реализации API был выбран SBert как оптимальное решение. Техническая архитектура системы включила асинхронный сервис векторизации на Python с поддержкой динамической загрузки моделей, алгоритм сегментации больших текстов с 10% перекрытием блоков (256 токенов)

Веб-интерфейс, разработанный как клиентская часть системы, предоставляет интуитивный доступ к функционалу API. Интерфейс включает: панель выбора коллекций с фильтрами по тематикам, двойное поле ввода для основного и негативного запросов, переключатель уникализации результатов (distinct) и визуализацию топ-10 документов с превью метаданных.

Восьмой раздел «Пример использования API» демонстрирует применение системы для поиска научных статей на arXiv.org. В рамках практической реализации системы семантического поиска была разработана комплексная инфраструктура для обработки научных публикаций. Основой

послужил специализированный парсер, осуществляющий сбор метаданных с платформы arXiv.org за 90-дневный период. Парсер обеспечивал получение структурированной информации, включающей названия статей, имена авторов, аннотации и ссылки на полные тексты в формате PDF.

Для обработки полученных PDF-документов был создан многоэтапный конвейер преобразования. На первом этапе происходила конвертация PDF-файлов в растровые изображения с высоким разрешением (300 dpi), что обеспечивало качественное распознавание текста. Затем с использованием библиотеки Tesseract OCR осуществлялось извлечение текстового содержимого с последующей постобработкой, включающей исправление переносов строк и нормализацию специальных символов.

Особое внимание было уделено проблеме обработки объемных научных статей. Разработанный алгоритм сегментации текста обеспечивал разбиение документов на семантически целостные блоки с перекрытием в 10%, что позволяло сохранять контекст при последующей векторизации. Каждый текстовый сегмент преобразовывался в векторное представление с использованием предобученной модели SBert, что обеспечивало глубокое понимание смыслового содержания.

Эффективность системы была продемонстрирована на практическом примере поиска по запросу "improving data for processing pictures". В отличие от традиционных keyword-методов, система успешно идентифицировала и ранжировала научные работы, посвященные аугментации изображений и методам улучшения данных для компьютерного зрения, даже при отсутствии точных лексических совпадений в текстах статей. Это стало возможным благодаря способности алгоритма выявлять семантические связи между концепциями.

Полученные результаты подтвердили универсальность разработанного решения. Система продемонстрировала высокую эффективность при работе с

разнородными документами - от научных публикаций до технической документации и корпоративных материалов. Особую ценность представляет способность решения адаптироваться к специфическим предметным областям без необходимости существенной модификации алгоритмов, что открывает широкие возможности для практического применения в различных отраслях.

ЗАКЛЮЧЕНИЕ

Были проведены оценка и сравнения нескольких алгоритмов получения векторных представлений текста. Из рассмотренных методов векторного представления текстов SBERT оказался самым точным методом для выделения смысла текста, потому что он учитывает контекст слов, но в тоже время этот метод является самым требовательным к вычислительным ресурсам. Также модели sbert гораздо проще масштабировать, чем модели основанные на концепции "мешка слов". Если предполагается работать с текстами ограниченной тематики, где заранее можно составить словарь синонимов, то такие методы как мешок слов и TF-IDF будут лучшим решением, чем нейросетевые модели. Таким образом были выбраны технологии для реализации семантического поиска.

Также реализованно API для семантического поиска с микросервисной архитектурой, что поможет в будущем масштабировать и усложнять приложение. Все сервисы были запакованы в Docker контейнеры, что обеспечит быструю развертку приложения на сервере или персональном компьютере.

Для эффективного поиска и хранения векторных представлений текстов была использована специализированная векторная база данных Qdrant. База данных предоставляет широкий функционал для работы с векторными представлениями текстов, а также хранит дополнительную информацию о векторах.

Также практический пример использования разработанного API показал его возможности для упрощения поиска публикаций на сайте arxiv.org.

Семантический поиск — это мощный инструмент для анализа текстов, основанный на современных достижениях машинного обучения и обработки данных. Его успешная реализация требует использования передовых методов векторного представления, точных мер расстояния и специализированных хранилищ данных. В дальнейшем развитие технологий, таких как SBert и Qdrant, сделает семантический поиск еще более точным и эффективным.

Основные источники информации:

1. Алувалия А., Сутрадхар Б., Гош К. и др. Гибридный семантический поиск: раскрытие пользовательских интенций за пределами ключевых слов // arXiv [cs.IR]. — 2024. — Ст. 2408.09236. [Электронный ресурс]. URL: https://arxiv.org/abs/2408.09236 (дата обращения: 17.11.2024).
2. Ван Ц., Донг И. Измерение сходства текстов: обзор // Information. — 2020. — Т. 11, № 9. — С. 421. DOI: 10.3390/info11090421.
3. Чани Г. М. и др. От черновиков фактов к операционным системам: семантический поиск в юридических решениях с использованием факт-драфтов // Big Data and Cognitive Computing. — 2024. — Т. 8, № 12. — С. 185. DOI: 10.3390/bdcc8120185.
4. Радев Д. Р. и др. Оценка веб-систем вопросно-ответного поиска // Proceedings of LREC. — 2002. — С. 1–8. [Электронный ресурс]. URL: https://aclanthology.org/L02-1109 (дата обращения: 13.02.2025).
5. Хагивара М. Реальное применение обработки естественного языка: практическое использование глубокого обучения. — Manning, 2021. — 352 с. — ISBN 978-1-61729-544-8.
6. Qdrant: векторная база данных для production-ready приложений // Официальная документация. — 2023. [Электронный ресурс]. URL: https://qdrant.tech/documentation (дата обращения: 22.11.2024).
7. Бондарь А., Захаров В. Сравнительный анализ векторных баз данных: Qdrant vs Milvus // Журнал системной информатики. — 2023. — Т. 15, № 4. — С. 45–62. DOI: 10.18287/2687-1308-2023-15-4-45-62.