RAG Системы

Автор: Ганеев Рустам Марсович

План лекции

Проблематика LLM

Почему обычных моделей недостаточно для бизнеса.

Архитектура RAG

Как устроены современные поисковые системы.

Внедрение

Практические шаги и метрики.

Что такое LLM?

LLM (Large Language Model) — это большие языковые модели, то есть тип искусственного интеллекта, который «понимает», обрабатывает и генерирует человеческий язык.

Проблема 1: Галлюцинации

LLM — это не база знаний. Это вероятностная машина.

Модель стремится дать *правдоподобный*, а не *правдивый* ответ.

Риск: Юридические ошибки, дезинформация клиентов, репутационные потери.

"Я придумаю ответ, если не знаю"

Проблема 2: Актуальность

“Мои данные актуальны на 2023 год"

- **Устаревшие знания:** GPT-4 училась на данных до определенной даты. Она не знает курс доллара сегодня.

Вывод: Нам нужен способ "подкладывать" свежие данные в момент запроса.

Что такое RAG?

RAG (Retrieval-Augmented Generation) в переводе дополненная генерация поиска. RAG стал настоящим трендом среди ML-инженеров — и не зря. Он позволяет моделям не просто генерировать текст, а опираться на актуальные документы, инструкции и базы знаний. Это снижает количество «галлюцинаций» и делает ответы точными.

- **Retrieval:** Находим документы.
- **Augmented:** Добавляем в контекст.
- **Generation:** Модель отвечает.

Аналогия: Студент на практике

Обычная LLM

Студент, который пытается вспомнить материал по памяти. Может забыть или напутать факты.

RAG

Тот же студент, но с **открытым учебником** (вашей базой знаний). Ответ точный и с цитатами.

Кейсы: Внутренние процессы (B2E - Business-to-Employee)

HR-помощник

Мгновенные ответы на вопросы об отпусках, ДМС и зарплатных листах.

База знаний Dev

Поиск по технической документации, API и легаси-коду.

Юристы

Анализ договоров на риски и поиск по базе законодательства.

Кейсы: Клиенты (B2C - Business-to-Consumer)

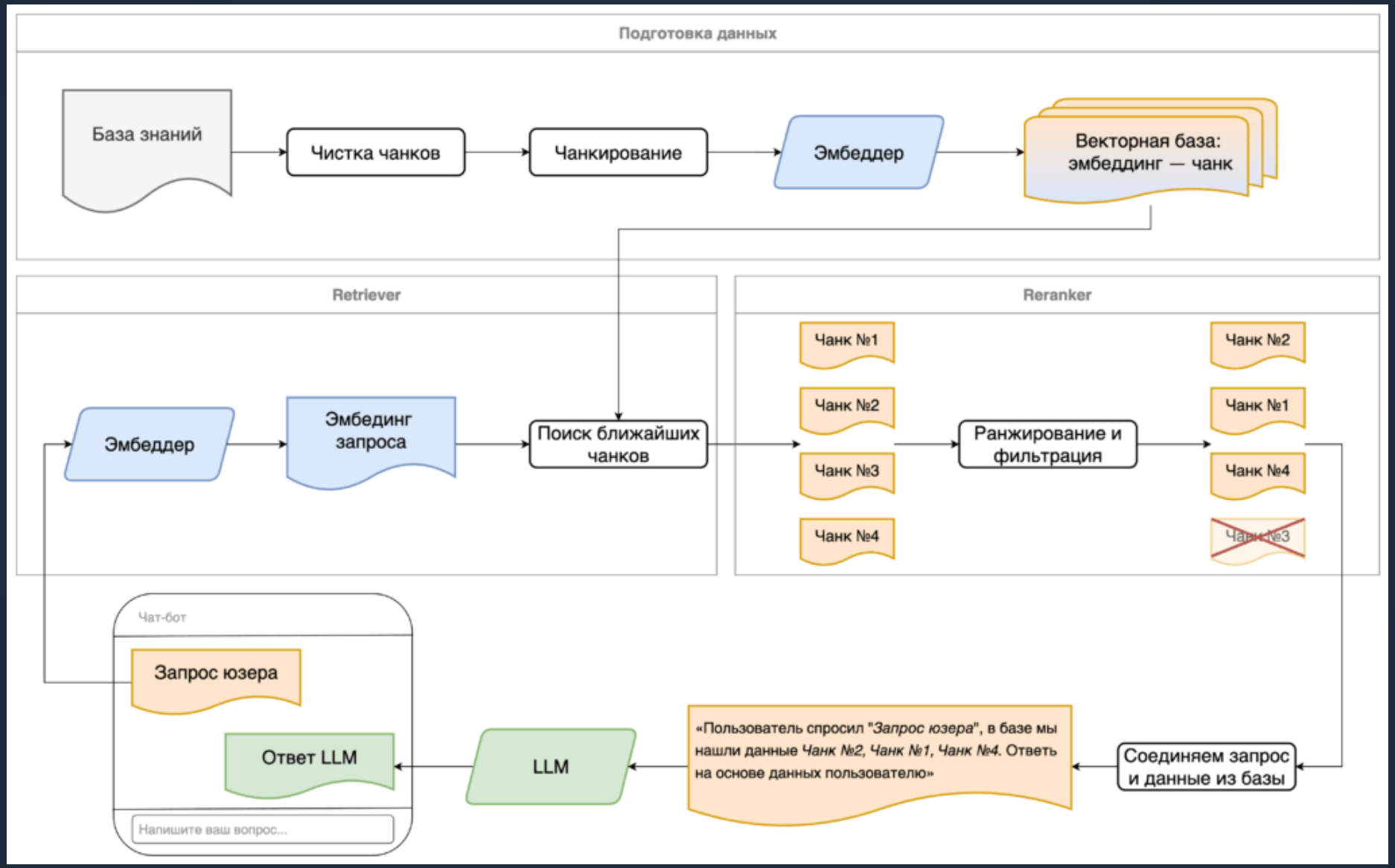
Умная техподдержка

Бот не просто кидает ссылки, а анализирует проблему и пишет пошаговое решение.

E-commerce

"Посоветуй ноутбук для игр до 100к". Сравнение характеристик и проверка наличия.

Пайплайн RAG



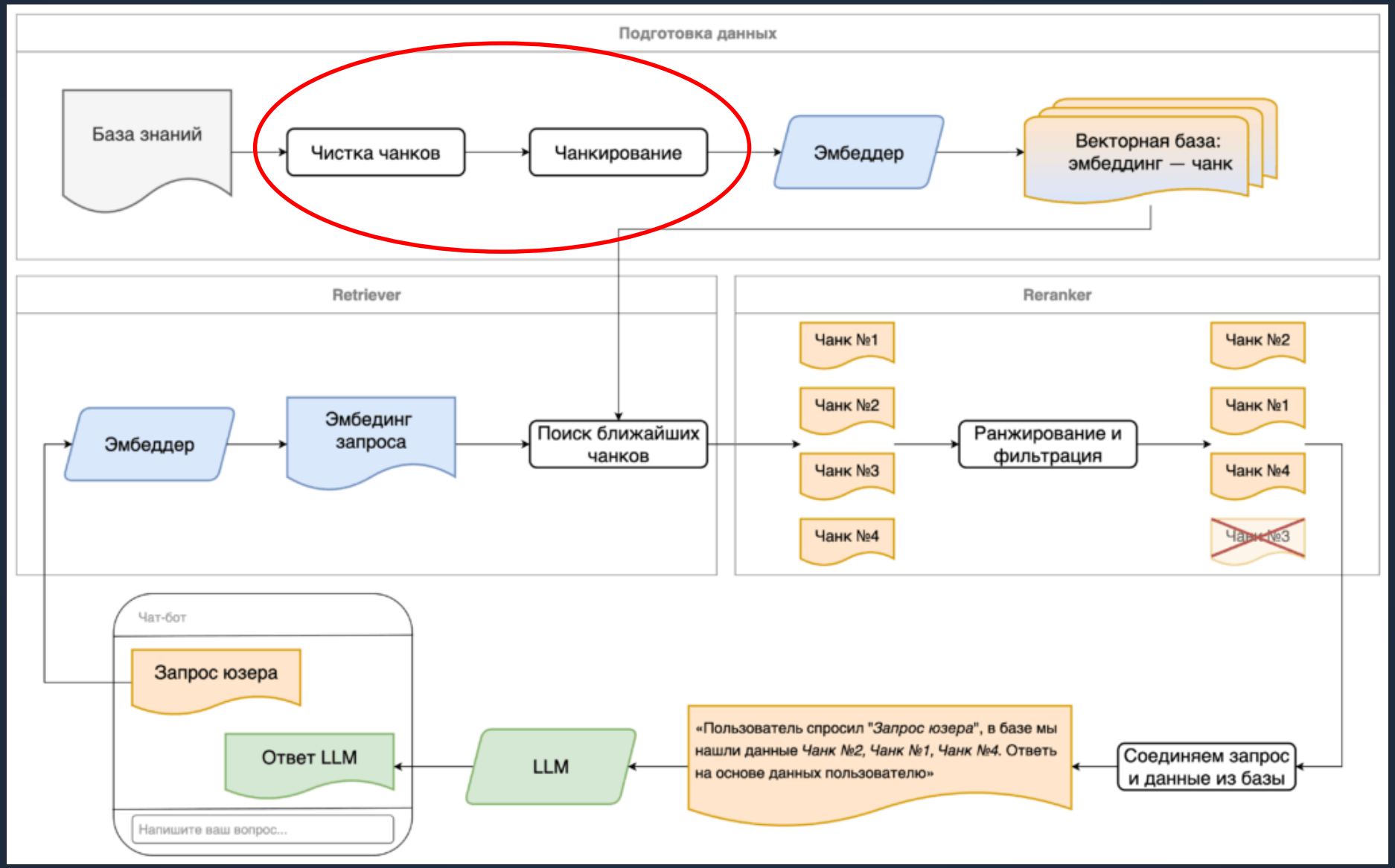
Чистка (Data Cleaning)

Garbage In — Garbage Out

Важный этап. Если в базе мусор, AI не поможет.

- Удаление HTML-тегов и скриптов
- Исправление кодировок
- Удаление дубликатов
- Нормализация

Пайплайн RAG

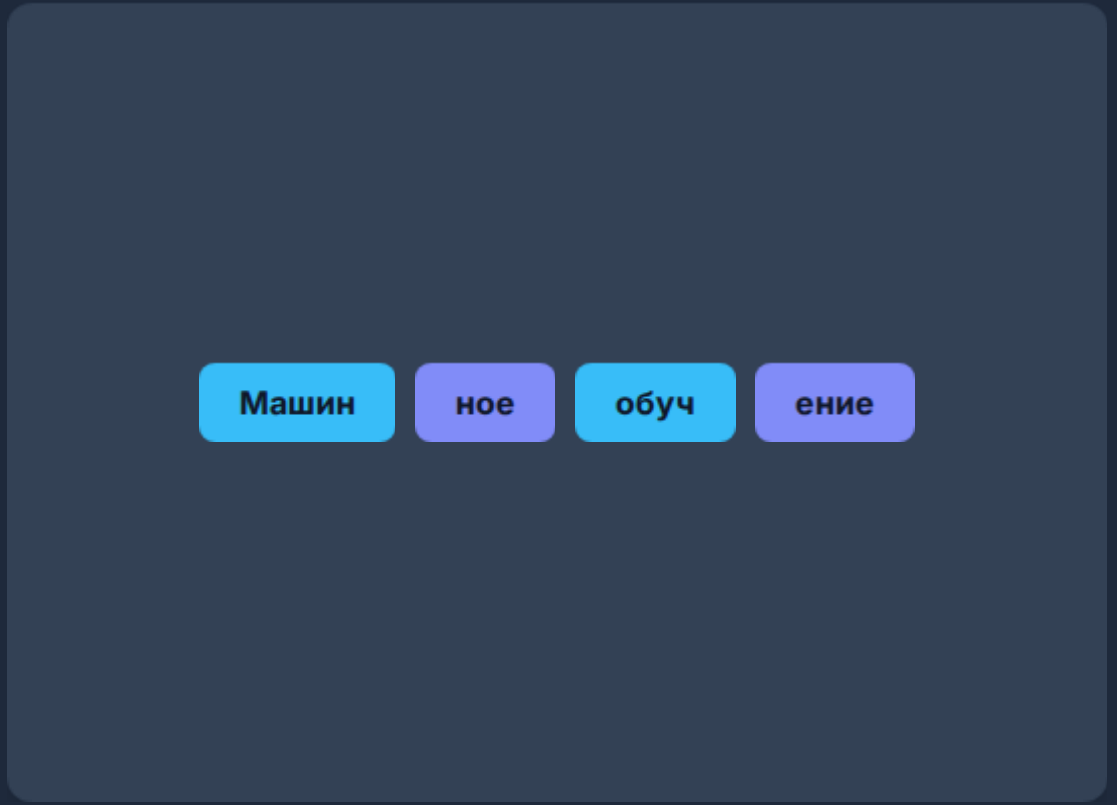


Токенизация

Модели не читают слова, они читают **токены**. Токен — это минимальная единица смысла (слово, часть слова или даже символ).

- **Английский:** "Machine learning" -> ["Machine", "learning"]
- **Русский:** "Машинное обучение" -> ["Машинное", "обучение"] или же как на примере справа

*Все лимиты и стоимость API считаются зачастую в токенах.



Методы токенизации

Whitespace

По пробелам. Применяется для языков с чёткой границей между словами

BPE / WordPiece

Разбиение на частотные подслова.

Морфологическая

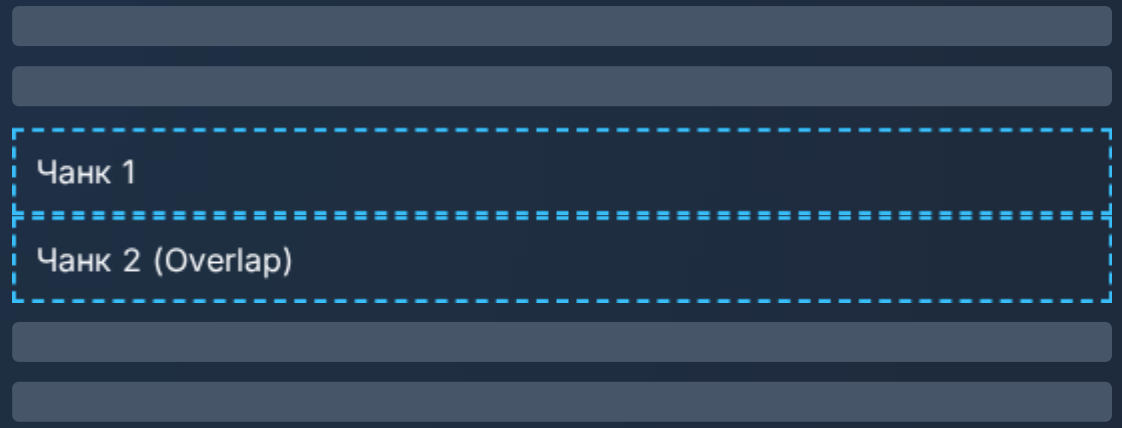
С учетом грамматики. Сложно, редко применяется.

Чанкинг (Нарезка)

Мы не можем подать в модель всю книгу целиком. Мы "нарезаем" текст на небольшие фрагменты — чанки.

Оптимальный размер: 256–1024 токена.

> **Overlap (Перекрытие):** Конец одного чанка должен повторяться в начале следующего, чтобы не разорвать мысль.



Стратегии Чанкинга

Фиксированная длина

По символам. Быстро, но рвет предложения.

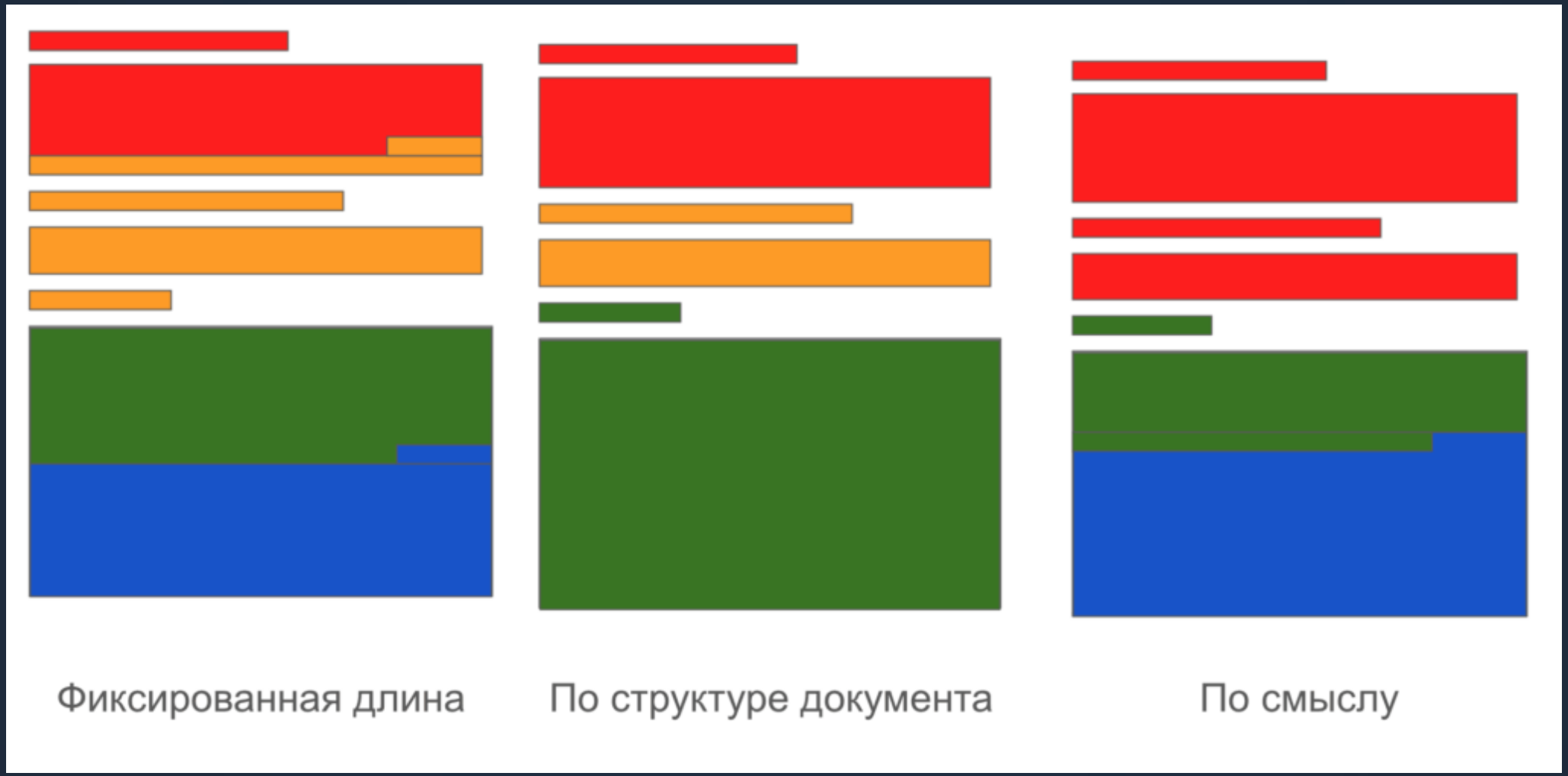
По структуре

Сначала по абзацам, потом по предложениям.

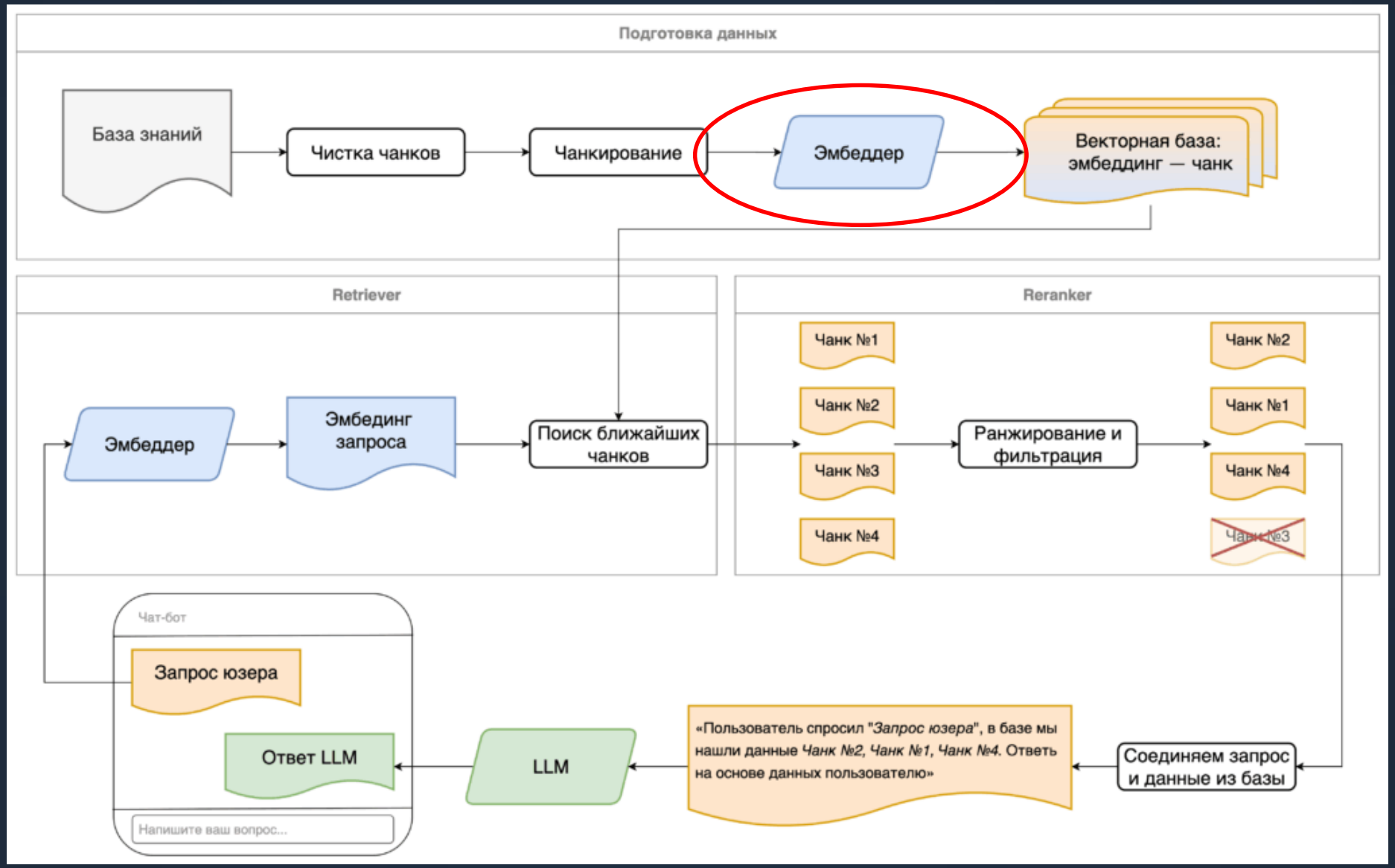
Семантическая

ИИ определяет смысловые переходы. Качественно, но дорого.

Стратегии Чанкинга



Пайплайн RAG



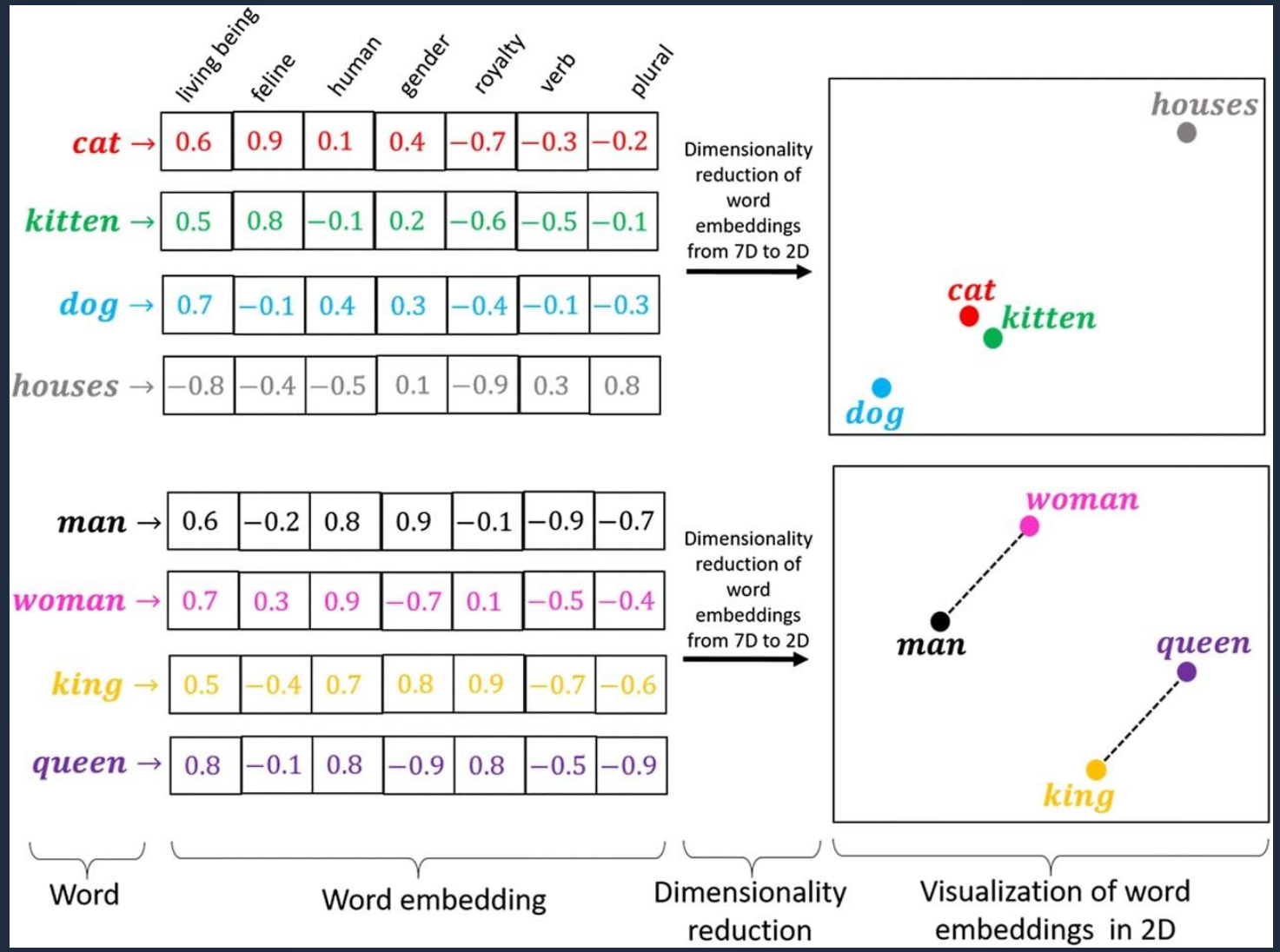
Эмбеддинг

Перевод текста в многомерный вектор. отображает чанки в векторы (эмбеддинги) для оценки схожести двух текстов не по словам, а через сравнение их векторов (например, с помощью оценки косинусного расстояния).

Суть: Тексты с похожим смыслом находятся рядом в математическом пространстве.

"Король" ближе к "Царь", чем к "Капуста".

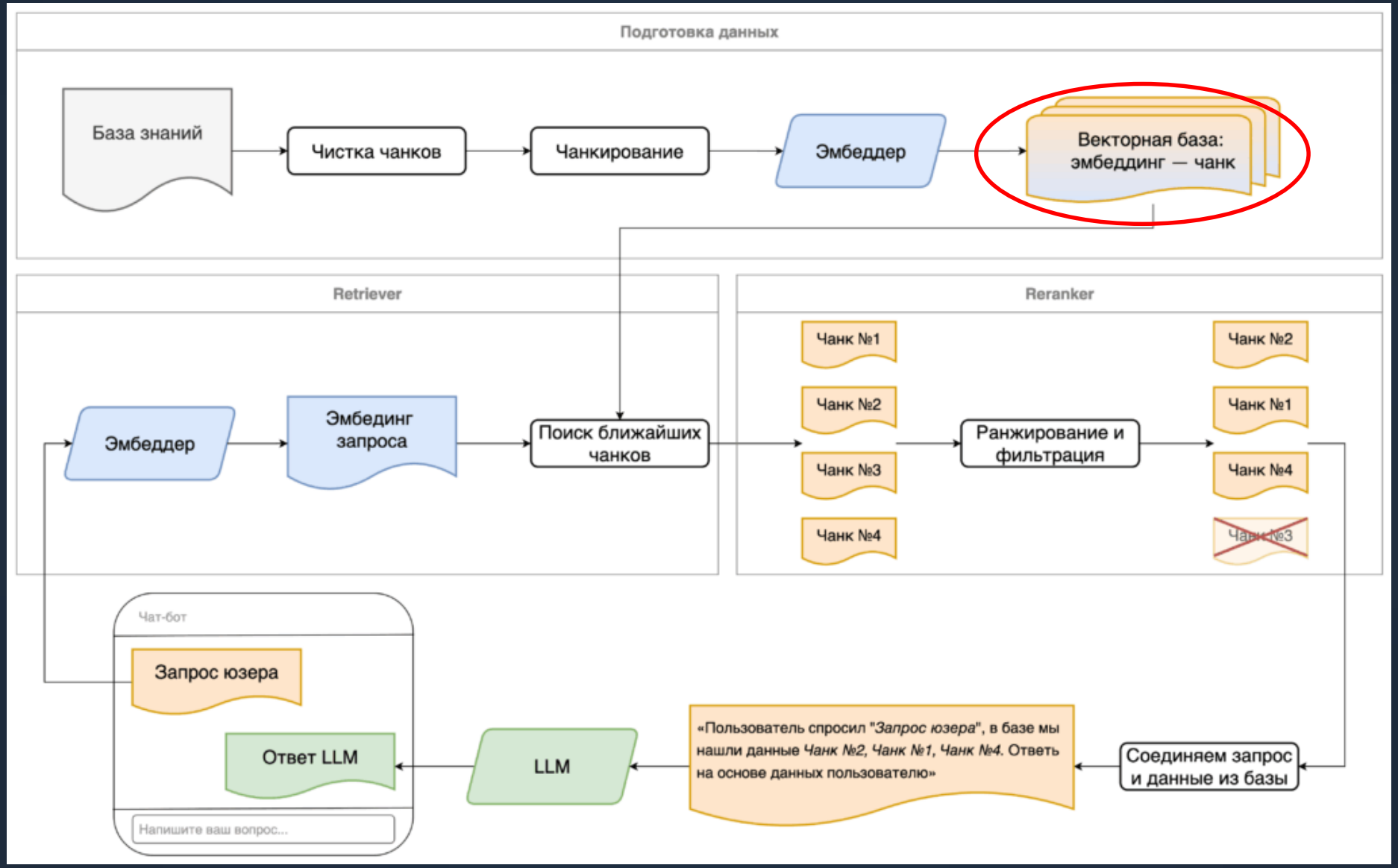
Эмбеддинг


living being
feline
human
gender
royalty
verb
plural
cat → 0.6 0.9 0.1 0.4 −0.7 −0.3 −0.2
kitten → 0.5 0.8 −0.1 0.2 −0.6 −0.5 −0.1
dog → 0.7 −0.1 0.4 0.3 −0.4 −0.1 −0.3
houses → −0.8 −0.4 −0.5 0.1 −0.9 0.3 0.8
Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
man → 0.6 −0.2 0.8 0.9 −0.1 −0.9 −0.7
woman → 0.7 0.3 0.9 −0.7 0.1 −0.5 −0.4
king → 0.5 −0.4 0.7 0.8 0.9 −0.7 −0.6
queen → 0.8 −0.1 0.8 −0.9 0.8 −0.5 −0.9
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

Выбор модели эмбеддинга

Тип	Примеры	Особенности
OpenAI	text-embedding-3-large	Стандарт индустрии. Платно. Легко стартовать.
Open Source	BGE-M3, E5-Large	Бесплатно. Можно запускать на своих серверах.
Русский язык	DeepPavlov, bge-multilingual-gemma2	Нужны мультиязычные или дообученные модели.

Пайплайн RAG



Векторная База

Зачем?

Обычные SQL базы ищут точные совпадения. Векторные базы ищут **смысл** (ближайших математических соседей).

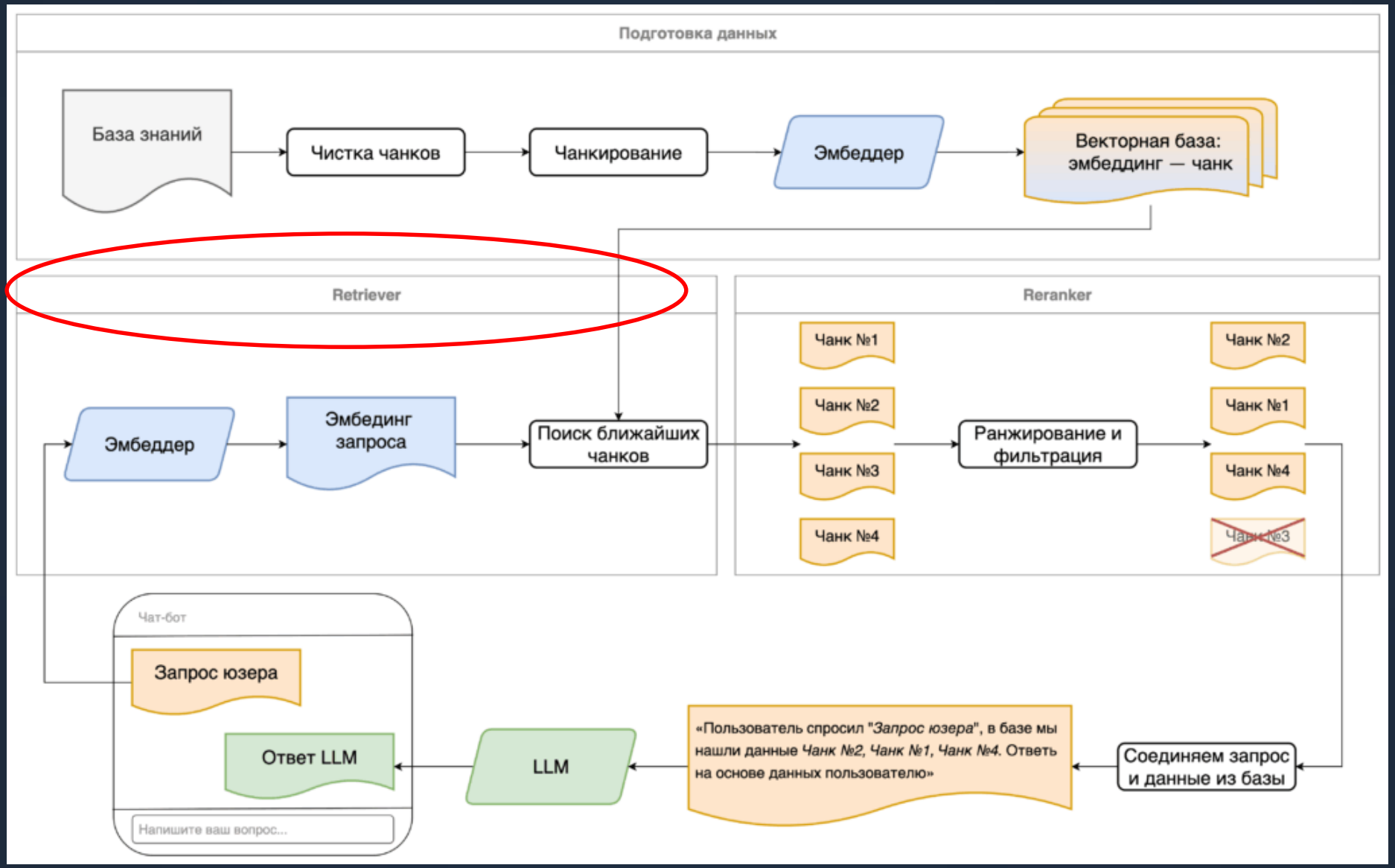
Скорость

Оптимизированы для поиска среди миллионов векторов за миллисекунды.

Инструменты

Pinecone, Qdrant, Weaviate, ChromaDB, FAISS.

Пайплайн RAG



Логика поиска (Retrieval)

1. Пользователь пишет вопрос.
2. Превращаем вопрос в вектор.
3. База ищет чанки с векторами, близкими к вектору вопроса.
4. Используем метрику **Cosine Similarity**.

```
> Query: "Как вернуть товар?"
 > Vectorizing... [0.12, 0.98, ...]
 > Searching DB...
 > Found: "Policy_Return.pdf" (Score: 0.95)
 > Found: "Refund_Steps.txt" (Score: 0.92)
```

Проблемы чистого вектора

Векторы ищут смысл, но теряют детали.

Запрос: "Ошибка error-504"

Результат вектора: "Ошибка 502" (похожий смысл — ошибка сервера).

Но нам нужен точный код!

Keyword Search (BM25)

Старый добрый поиск

Алгоритм BM25 ищет точные совпадения слов (как Ctrl+F, но умнее).

- **Плюсы:** Идеально для артикулов, фамилий, терминов.
- **Минусы:** Не понимает синонимы ("Машина" != "Автомобиль").

Гибридный поиск

Vector Search

Ищем смысл

+

BM25

Ищем точность

Результаты объединяются с помощью **Reciprocal Rank Fusion (RRF)**.

Query Rewriting

Помогаем пользователю спросить правильно.

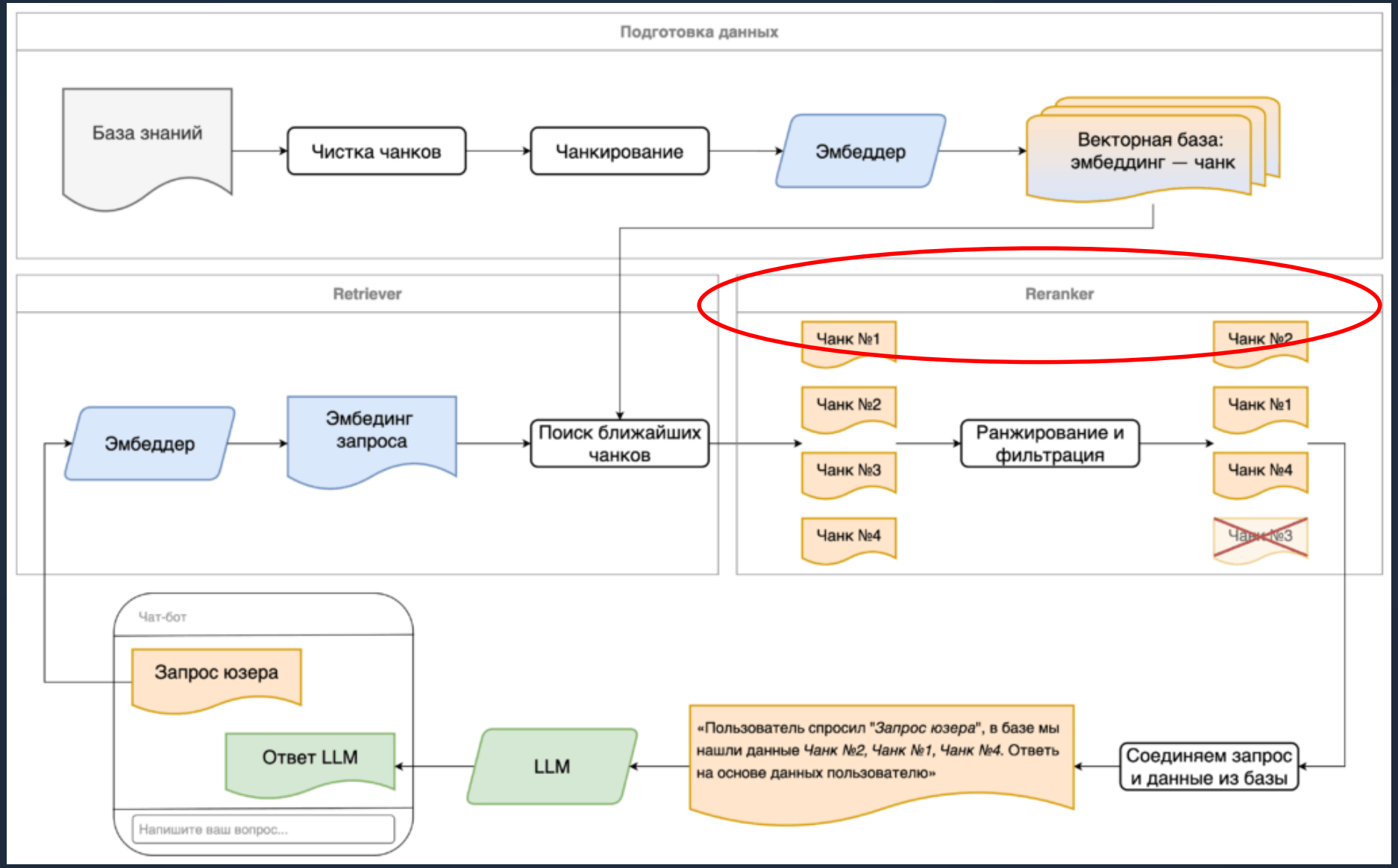
До

“Приветули! Че там с доставкой?))"

После (LLM)

"Каковы условия и сроки курьерской доставки?"

Пайплайн RAG



Реранкер (Reranker)

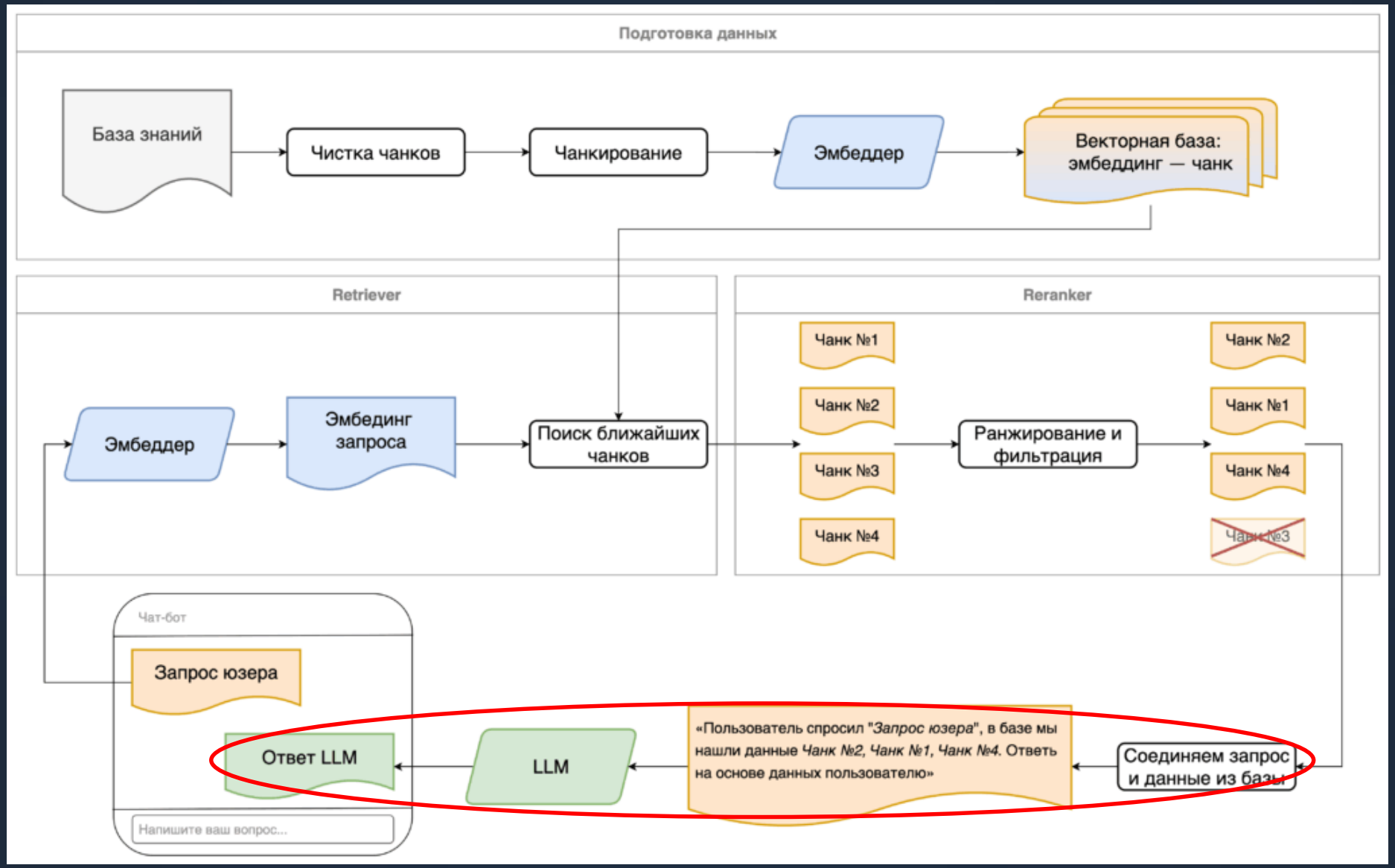
Фильтр качества

Поиск возвращает 50 кандидатов, но там много мусора.

Реранкер — это "тяжелая" модель, которая медленно оценивает каждый чанк: *"Отвечает ли этот текст на вопрос?"*

Оставляет только топ-5 лучших.

Пайплайн RAG



Генерация ответа

```
"Ты — полезный помощник. Ответь на вопрос пользователя, используя ТОЛЬКО предоставленную ниже информацию. Если не знаешь ответа, скажи 'Я не знаю'. Не выдумывай."
```

Пример системного промпта

Борьба с галлюцинациями

Grounding

Требовать указывать ID документа-источника.

"Я не знаю"

Прямая инструкция признаваться в незнании.

Температура 0

Минимальная креативность модели.

RAG vs Fine-tuning

Fine-tuning

Меняет **поведение** и **стиль**.

Плохо запоминает факты. Дорого.

RAG

Дает **знания**.

Дешево. Легко обновлять данные.

Когда нужен Fine-tuning?

Специфический язык

Медицинский, юридический, древнеславянский.

Формат

Жесткий JSON или специфический стиль кода.

Связка

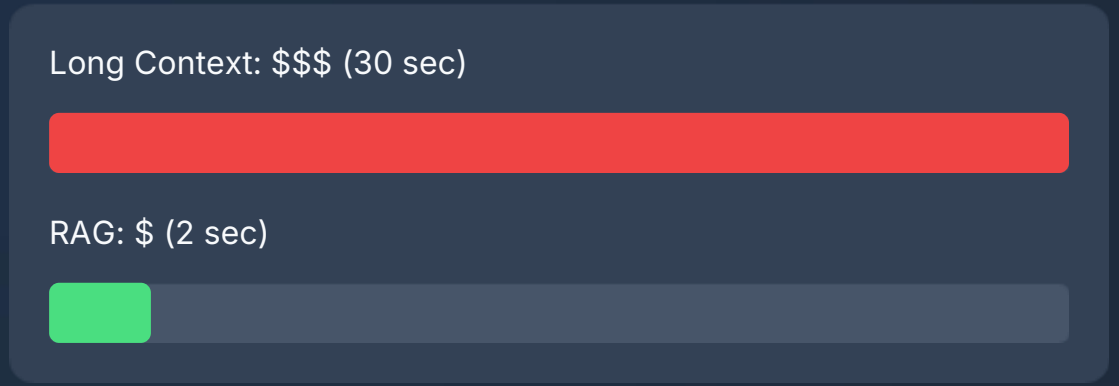
Часто используют RAG + Fine-tuned модель.

RAG vs Long Context

Можно "скормить" книгу целиком в промпт (1M токенов).

- **Плюсы:** Видит все связи, лучше аналитика.
- **Минусы:** Цена и Ресурсы.

RAG выигрывает на масштабе.



RAG vs Long Context

Метрики RAG

Нельзя оценивать "на глаз"

Нужен **Golden Set** (Набор из 100 пар "Вопрос — Эталонный ответ").

Триада метрик (RAG Triad)

Context Relevance

Нашел ли поиск нужные документы?

Faithfulness

Не придумала ли модель лишнего?

Answer Relevance

Полезен ли ответ пользователю?

Где будет больно?

Таблицы

Парсинг сложных таблиц из PDF.

Многоязычность

Смесь языков в документах.

Противоречия

Старые и новые приказы в одной базе.

С чего начать?

MVP

OpenAI API + Простая нарезка + ChromaDB.

Тесты

Сбор "золотого сета" вопросов и ручная проверка.

Тюнинг

Добавление Reranker и Гибридного поиска.

Prod

Кэширование, мониторинг и авто-обновление базы.

Главные выводы

- **RAG** — стандарт для внедрения AI в бизнес.
- **Данные** важнее модели. Чистите базу знаний!
- Начинайте с **простого пайплайна** и усложняйте по мере необходимости.
- Оценивайте качество на метриках, а не "на глаз".

Вопросы?

Спасибо за внимание!