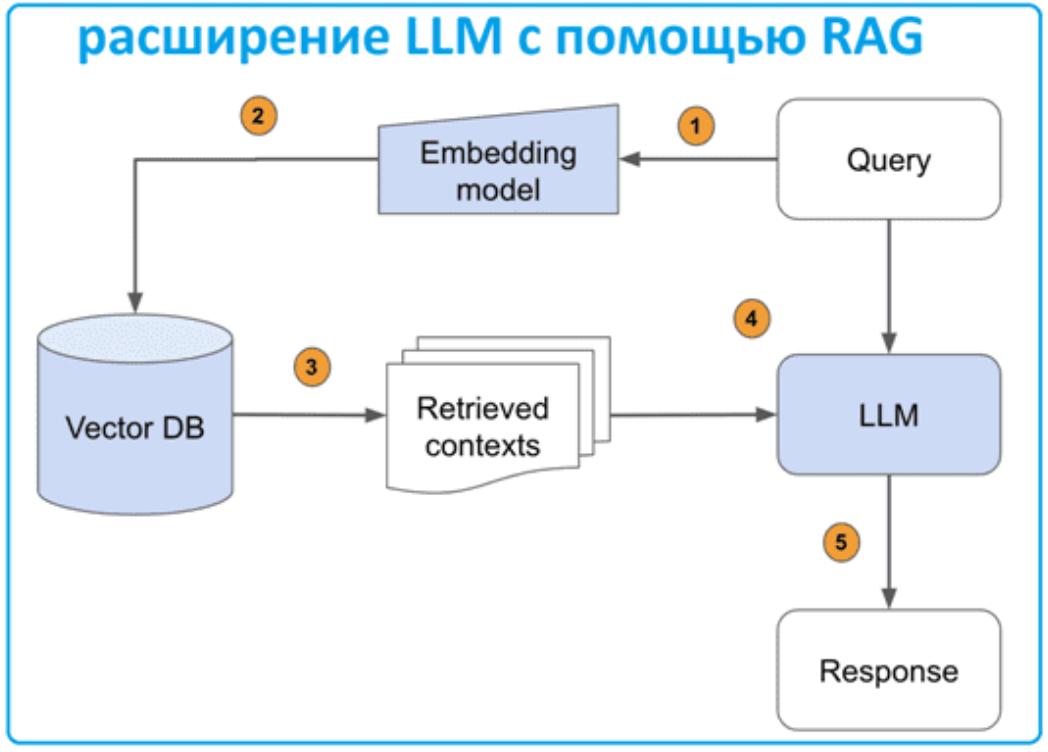
State-of-the art техники RAG на примере победного решения Enterprise RAG Challenge и его адаптации

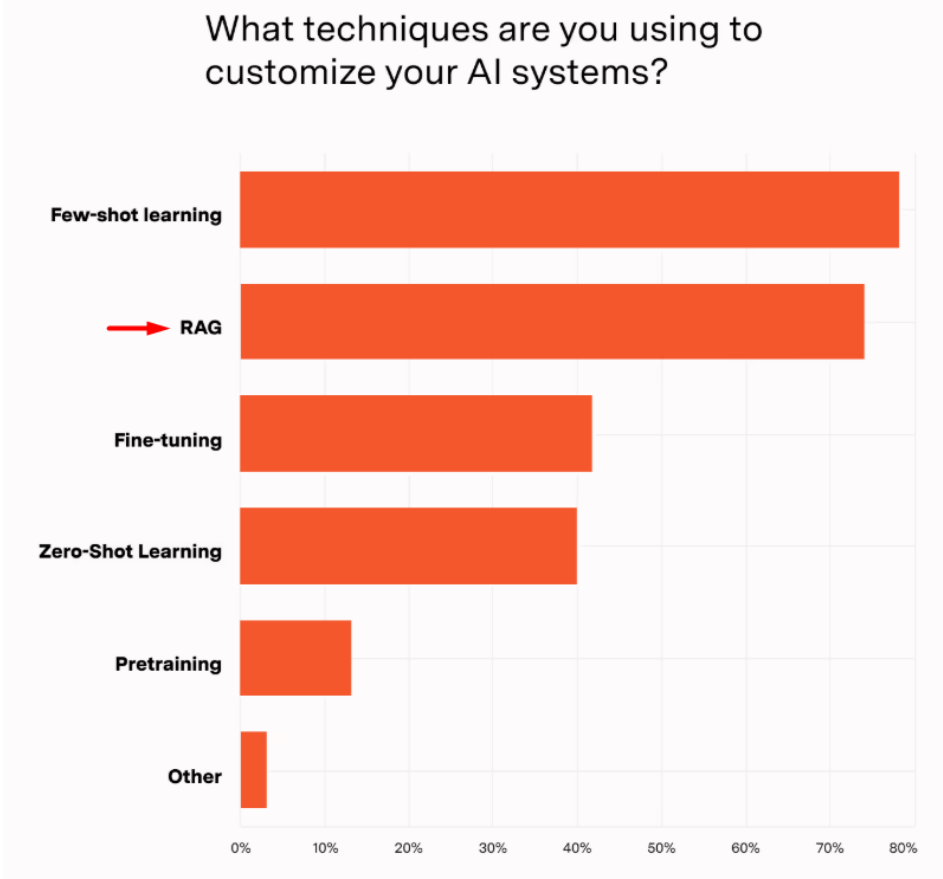
Акимов Дмитрий Андреевич

Актуальность RAG

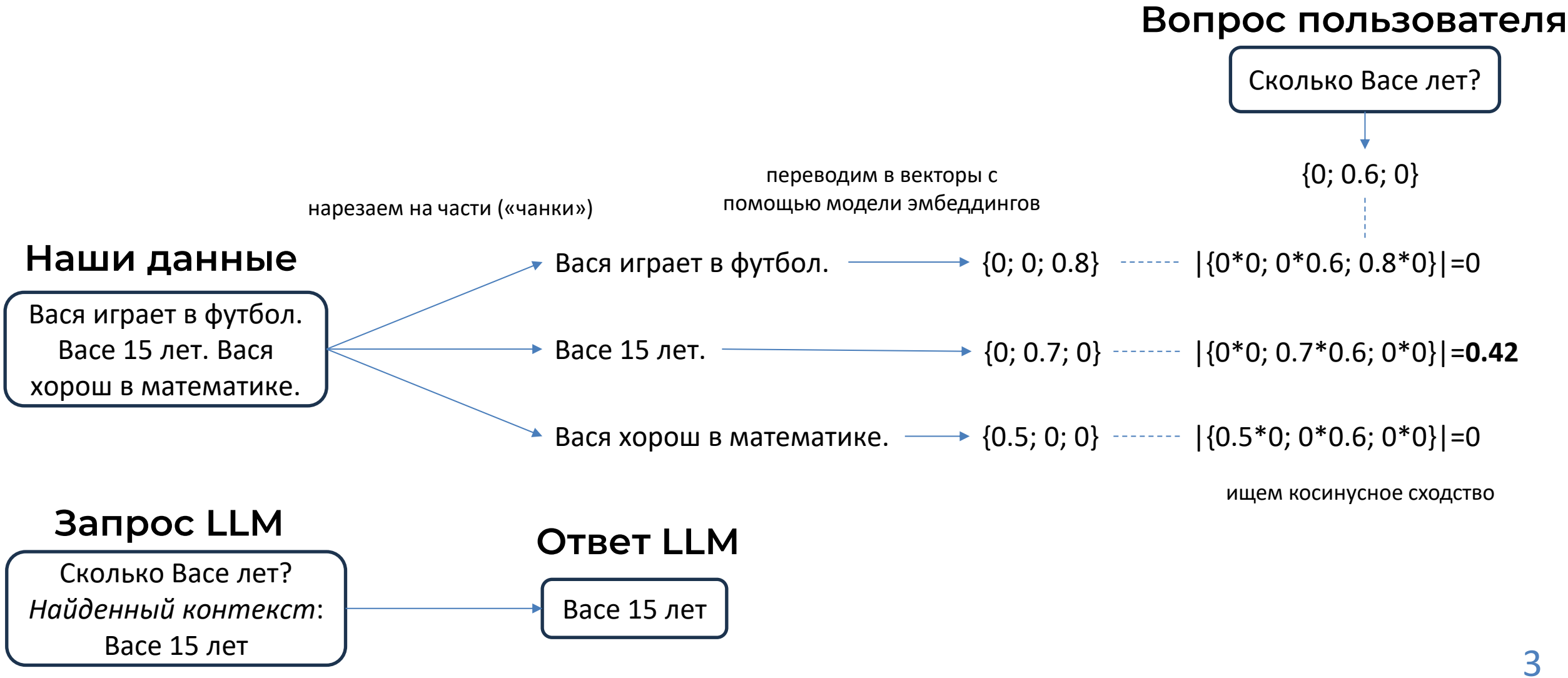


Engineering and Infrastructure

The 2025 AI Engineering Report



Retrieval Augmented Generation (на пальцах)



Retrieval Augmented Generation (если погружаться)

HyDE

RAG-Fusion

Haystack

LlamaIndex

Langchain

Knowledge Graph

MMR

TrueLens

RAGAS

Agentic RAG

Recall@k

BM25 + TF-IDF

Contextual Retrieval

Neo4j

Cross-Encoder Reranking

HNSW

IVF-Flat

AutoRAG

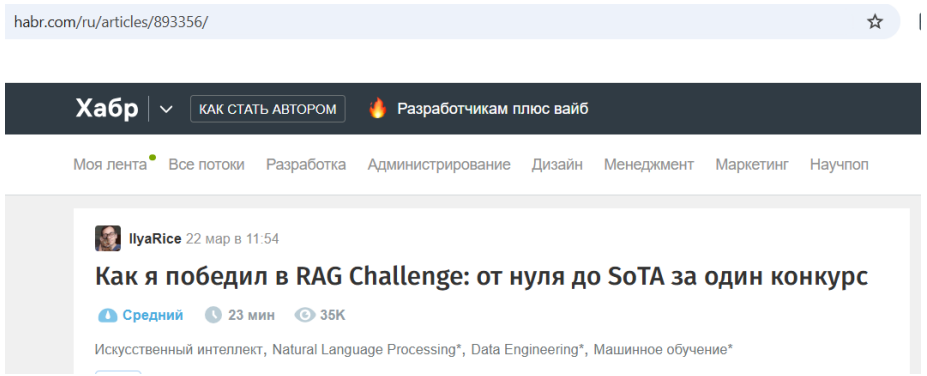
Multi-Query

ColBERT

DeepEval

CRAG

Enterprise RAG Challenge v2 (2025)



Статья получила международную популярность и была переведена на китайский:

Конкурс на деньги (как и в случае с Kaggle) – часто лучший способ определить state-of-the-art техники.

В чём суть RAG Challenge?

Нужно создать вопросно-ответную систему на основе годовых отчётов компании. Если коротко, то в день конкурса:

1. Выдаётся 100 годовых отчётов по случайно выбранным компаниям и 2.5 часа на их парсинг и составление базы данных. Отчёты представляют из себя PDF размером до 1000 страниц.
2. После этого генерируется 100 случайных вопросов (по заранее известным шаблонам), на которые система должна как можно быстрее ответить.

Все вопросы должны иметь однозначный ответ:

- Да/Нет;
- название компании (или нескольких компаний);
- названия управляющих позиций, выпущенных продуктов;
- размер той или иной метрики: выручка, количество магазинов и т.д.

Каждый ответ должен сопровождаться ссылками на страницы с ответом в качестве доказательства, что система по честному нашла ответ и не сгаллюцинировала.

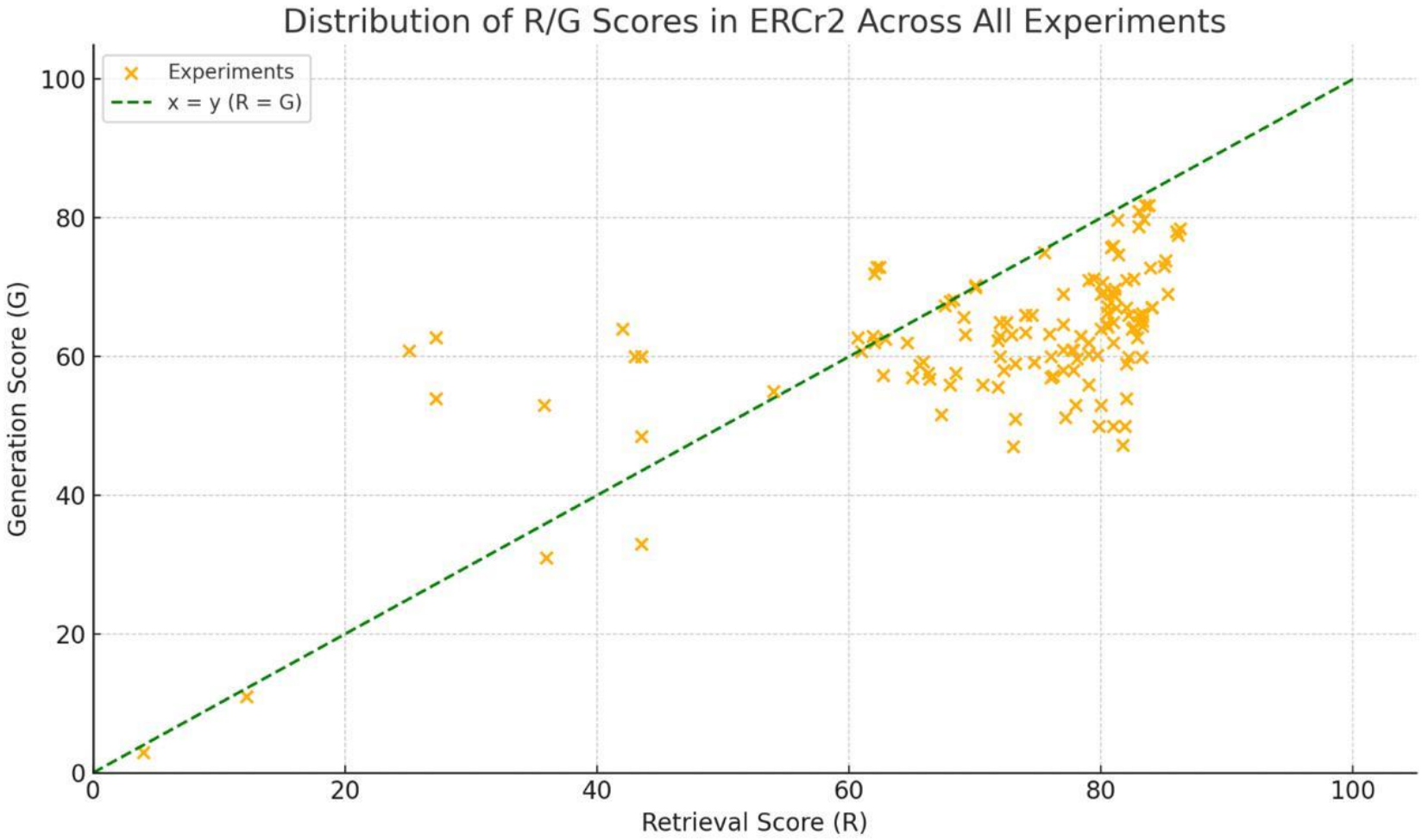
Enterprise RAG Challenge v2 (условия)

100 PDF годовых отчетов компаний (случайно отобранных)

100 случайных вопросов (Да/Нет; Числовые показатели и т.д.)

Оценивался Retrieval (верная ссылка в ответе) и Generation (верный ответ)

Итоговая метрика R/3 + G (т.е. верный ответ оценивался в 3 раза выше верного цитирования)



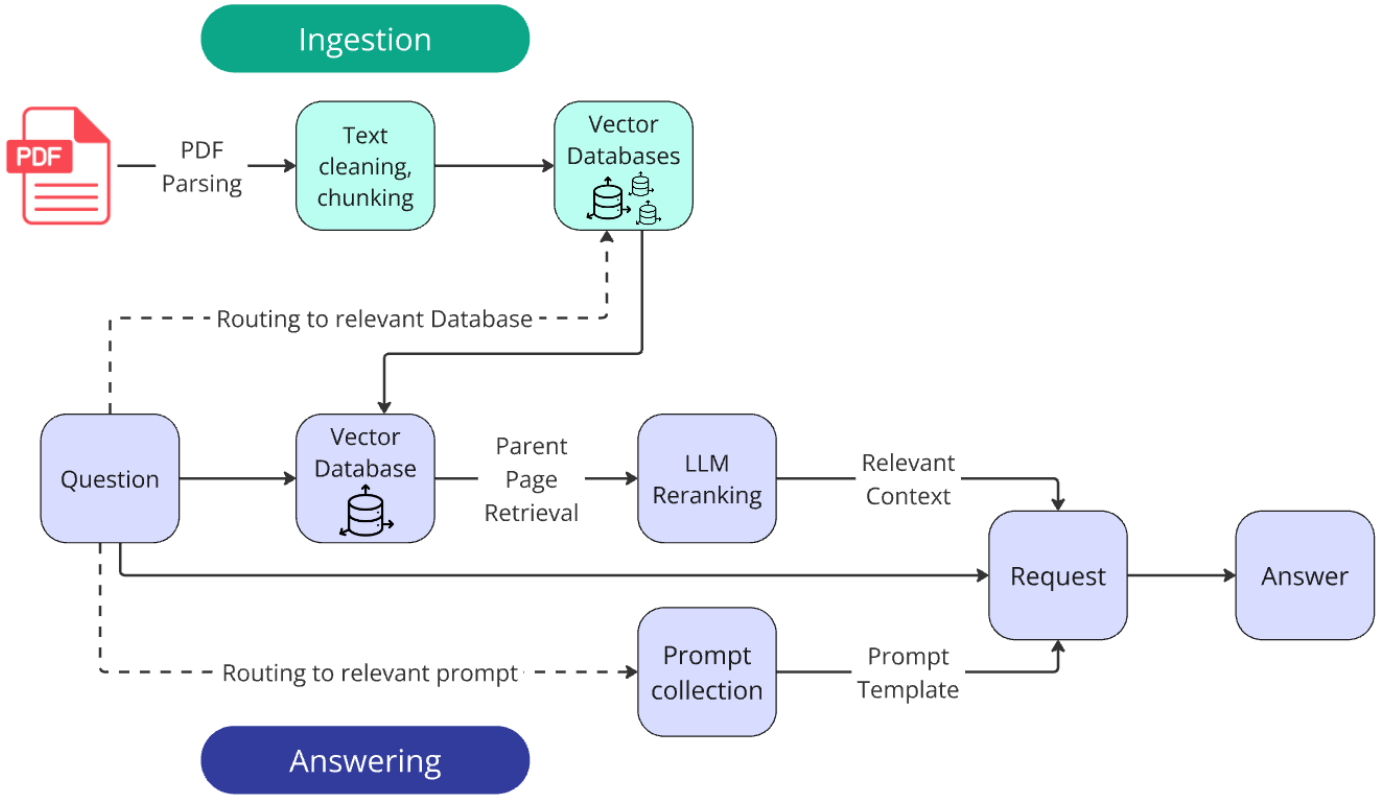
https://t.me/llm_under_hood/617

Enterprise RAG Challenge v2. Победное решение (кратко)

Team / Experiment	Time	R/G	Score
1. ▶ Ilia Ris 😊	49 min	83/81	123.7
2. ▶ Emil Shagiev 😊	55 min	86/78	121.6
3. ▶ Dmitry Buykin 😊	8 hours	81/76	117.5
4. ▶ Sergey Nikonov 😊	30 hours	85/73	116.4
5. ▶ ScrapeNinja.net 😊	23 hours	82/71	112.5
6. ▶ xsl777 😊	16 hours	79/71	110.9
7. ▶ nikolay_sheyko(grably.tech) 😊	25 hours	81/69	110.4
8. ▶ Felix-TAT 😊	7 days	80/69	109.4
9. ▶ A.Rasskazov/V.Kalesnikau	30 hours	84/67	109.3
10. ▶ Dany the creator 😊	3 hours	82/67	108.4
11. ▶ SergC 😊	7 days	77/69	108.1
12. ▶ Swisscom Innovation Lab 🔒	21 hours	83/66	107.8
13. ▶ fomih 😊	10 days	83/65	107.4
14. ▶ Al Bo	12 days	81/65	105.9
15. ▶ NumericalArt	8 days	70/70	105.3

всего около 45 команд

Схема победного решения Ilia Ris



На доп. треке IBM (использование open-source LLM Llama 2 70B) качество просело менее, чем на 2%!

Parsing

JSON

помимо самого текста хранится идентификатор документа и номер страницы (в дальнейшем понадобится для роутинга и parent-page enrichment)

Markdown
(дальше его и используем для чанкинга)

Html
(про запас; если понадобится заново обработать сложную таблицу, например)

Илья использовал Docling, арендовав сервер на Runpod. Можно также использовать Unstructured, Marker, LlamaParse и пр.

В своем упрощенном примере я использовал pdfplumber.

Ingestion

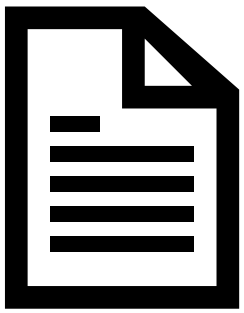
Markdown текст → (рекурсивный md сплиттер) → **Чанки текста по 300 токенов с перекрытием по 50 токенов** → (text-embedding-3-large) → FAISS

\+

metadata.json (какой чанк на какой странице для будущего parent page enrichment)

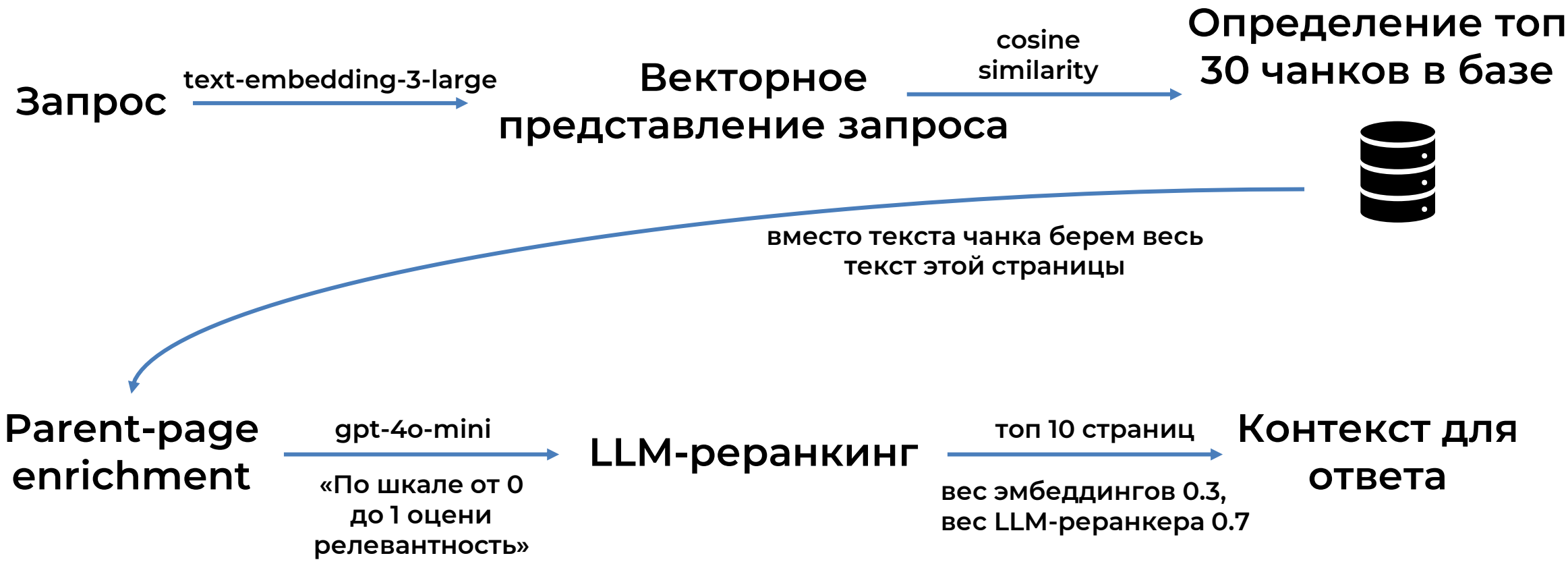
Для каждого отчета создавалась своя векторная база данных

Как работает рекурсивный md сплиттер:



страница → (режем по пустым строкам (/n/n)) → Абзац 1 (пустая строка) Абзац 2 (пустая строка) … Абзац n → (Если чанки все еще больше 300 токенов, режем по строкам, потом по словам, потом по символам) → Чанки не более 300 токенов

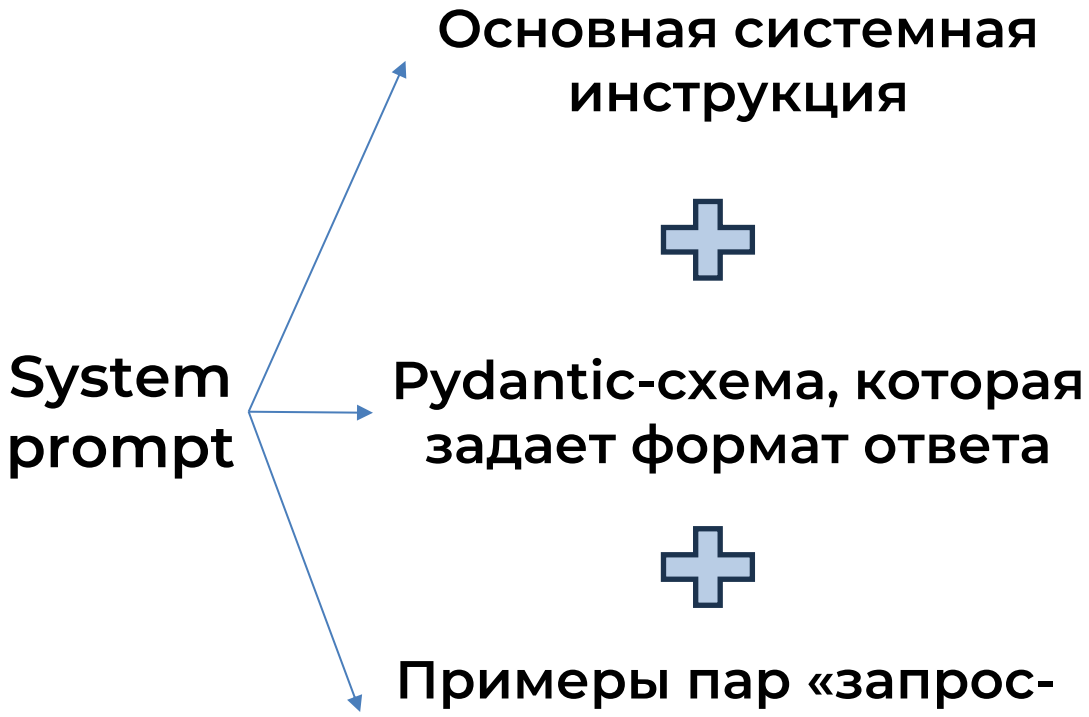
Retrieval



LLM-реранкинг – перспективная новинка! Обычно используется cross-encoder reranking.

Серебряный призер вообще построил RAG без эмбеддингов.

Augmentation



User prompt → Template для вставки контекст и вопроса

Ты — RAG (Retrieval-Augmented Generation) система ответов. Твоя задача — отвечать на заданный вопрос, опираясь ТОЛЬКО на информацию из годового отчёта компании, который предоставляется как релевантные страницы (контекст) после процедуры RAG.
Перед финальным ответом внимательно, вслух и по шагам проанализируй вопрос. (и т.д.)

step_by_step_analysis: str = Field(description="
Подробный пошаговый анализ ответа минимум из 5 шагов и не менее 150 слов."
str = Field(description="Краткое резюме пошагового анализа. Около 50 слов.") (и т.д.)

Пример 1: Какова прибыль Сбера за 2023 год?
{"step_by_step_analysis": «…»,"reasoning_summary": «…»,
"relevant_pages": …, "final_answer": …}

Контекст: (найденный контекст, т.е. отобранные страницы)
Вопрос: (вопрос)

Generation

Специфическое для условий данного соревнования решение:

- **каждый запрос (кроме multiquery) касался только одного отчета;**
- **в запросе было явно указано, к отчету какой компании он относится;**
- **запросы относились к одному из нескольких известных типов.**

1. Роутинг к нужной БД по имени компании

Отчет Сбера

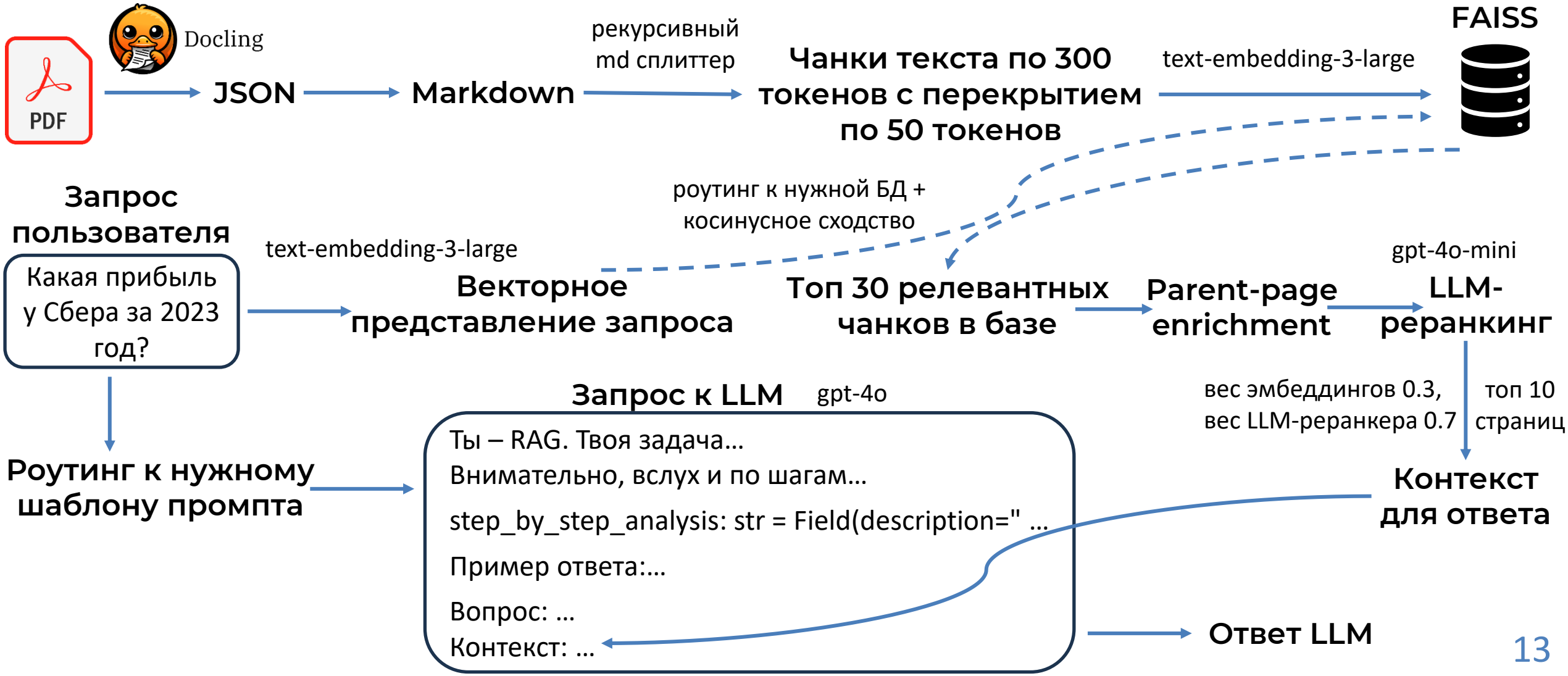
2. Роутинг к нужному шаблону промпта

Шаблон промпта, где ответ – одно число

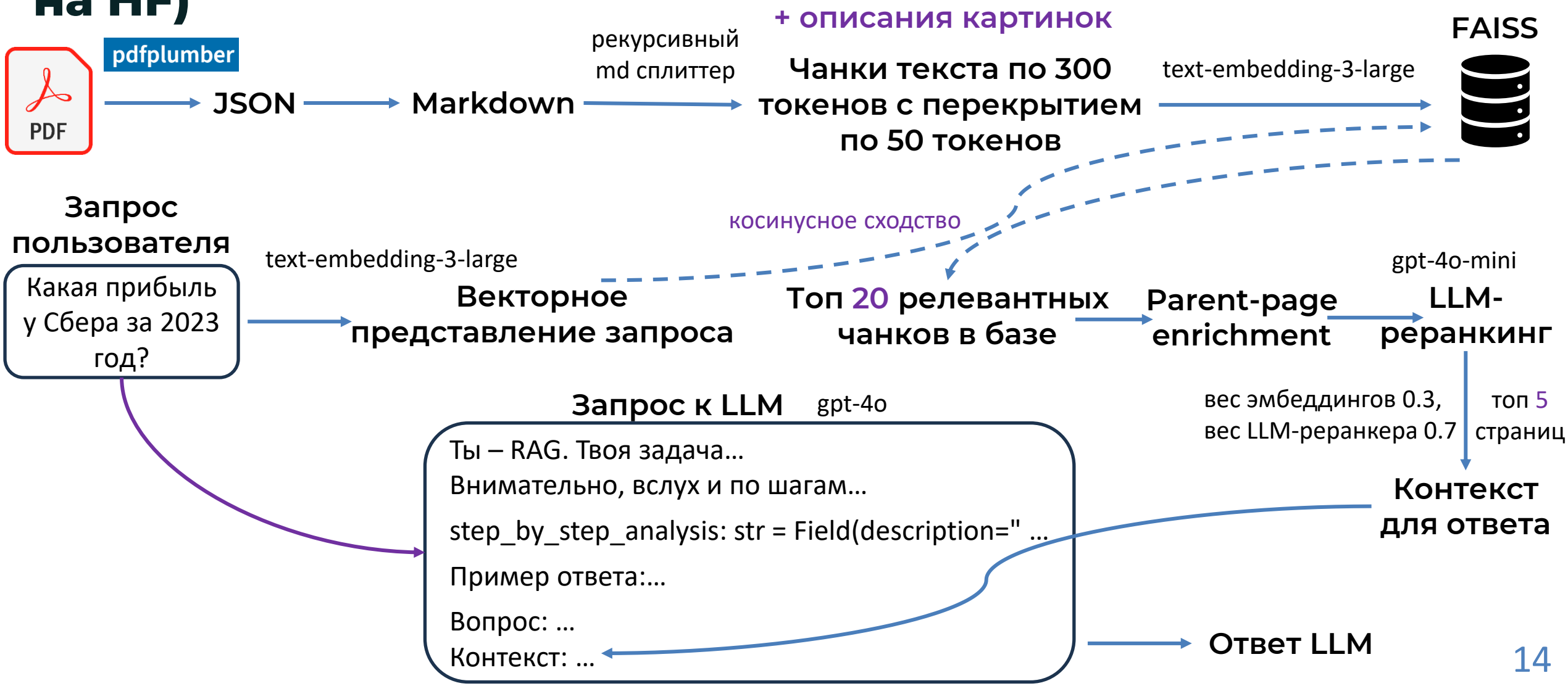
3. Роутинг multiquery запросов

И дальше как в п.1 и п.2

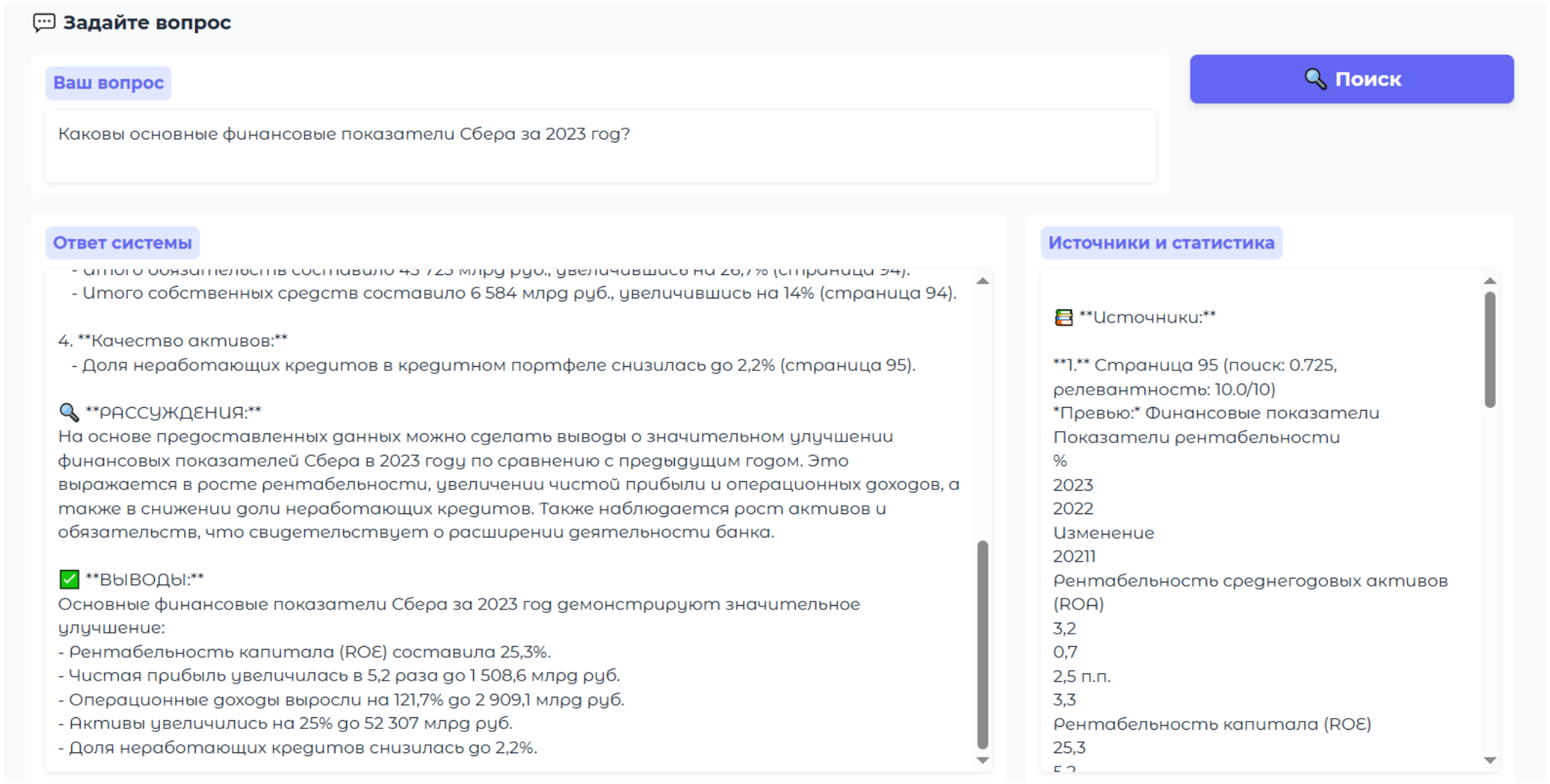
Еще раз соберем всё решение вместе



Упрощенная адаптация (мой курсовой проект, развернут на HF)



Demo



Скриншот развернутой на HuggingFace мультимодальной RAG-системы

Выводы

- RAG – одно из основных применений LLM на сегодняшний день.
- В работе была реализована мультимодальная RAG-система, вдохновленная победным решением Enterprise RAG Challenge (2025 год).
- Основные используемые технологии: pdfplumber, RecursiveCharacterTextSplitter, FAISS, Parent-page enrichment, LLM reranking, Chain-of-Thoughts, pydantic, pymupdf, gradio, text-embedding-3-large, GPT-4o, gpt-4o-mini.
- Основные сложности: некачественное выделение и распознавание картинок (следует рассмотреть переход с pdfplumber на docling, а также уточнить промпт для GPT-4o и добавить автоматический повторный запрос в случае отказа создать описание картинки).
- Перспективы на будущее: целесообразно верифицировать качество собранного пайплайна путем построения автоматической оценки (RAGAS и др.).

Список используемых источников/программных средств:

- Google Colab, HuggingFace
- «Как я победил в RAG Challenge: от нуля до SoTA за один конкурс» (https://habr.com/ru/articles/893356/)