



БОЛЬШИЕ ВЫЗОВЫ

Х НАУЧНО-ТЕХНОЛОГИЧЕСКАЯ
ПРОЕКТНАЯ ПРОГРАММА



AI for Science



СБЕР



АETHER

Агентная система для решения научных задач

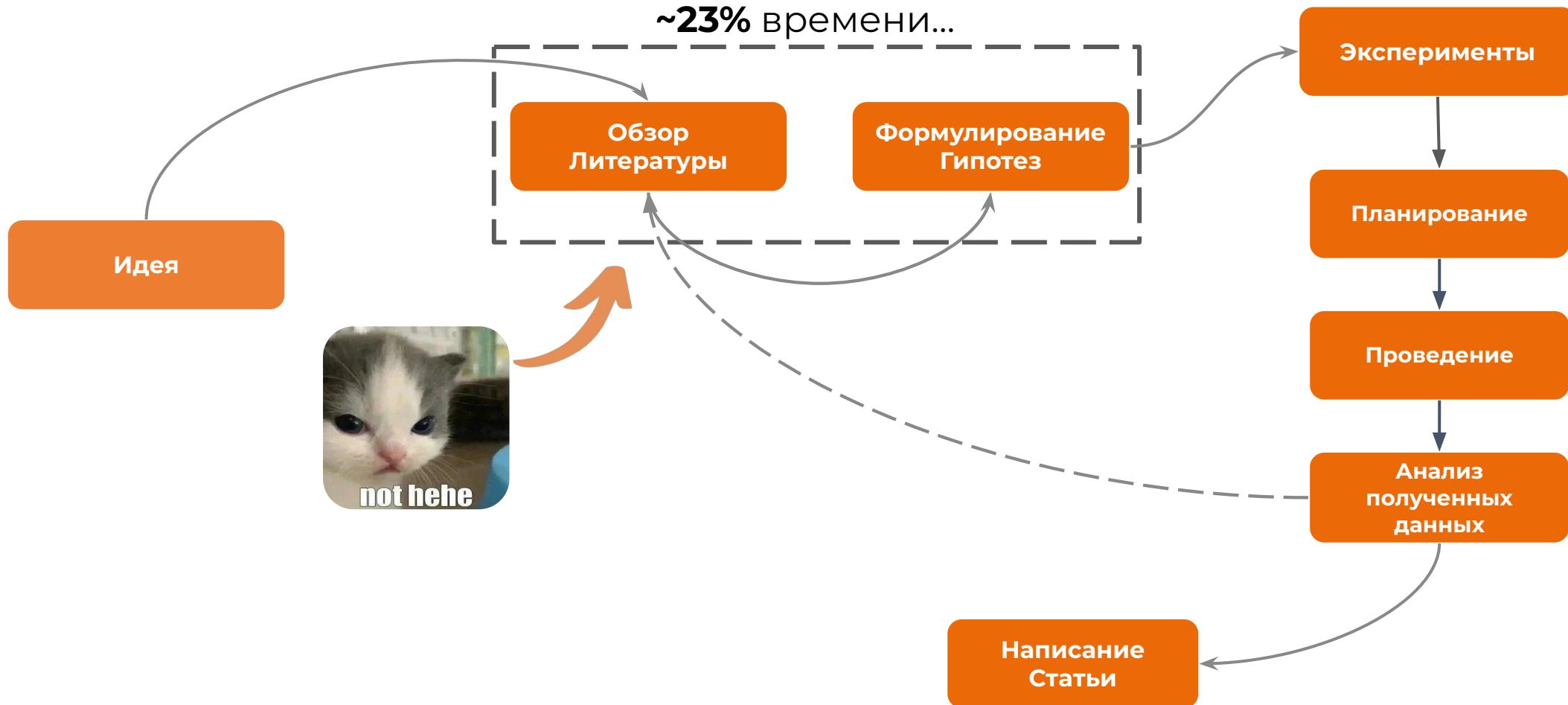


Сириус

Научно-технологический
университет



Схема работы ученого:





Рост объема научной информации

Объем публикаций опережает возможности их освоения

Объем знаний растет экспоненциально...

Более **5 миллионов** статей
публикуется ежегодно...¹

Более **400 тысяч** статей в области ИИ
публикуется ежегодно...

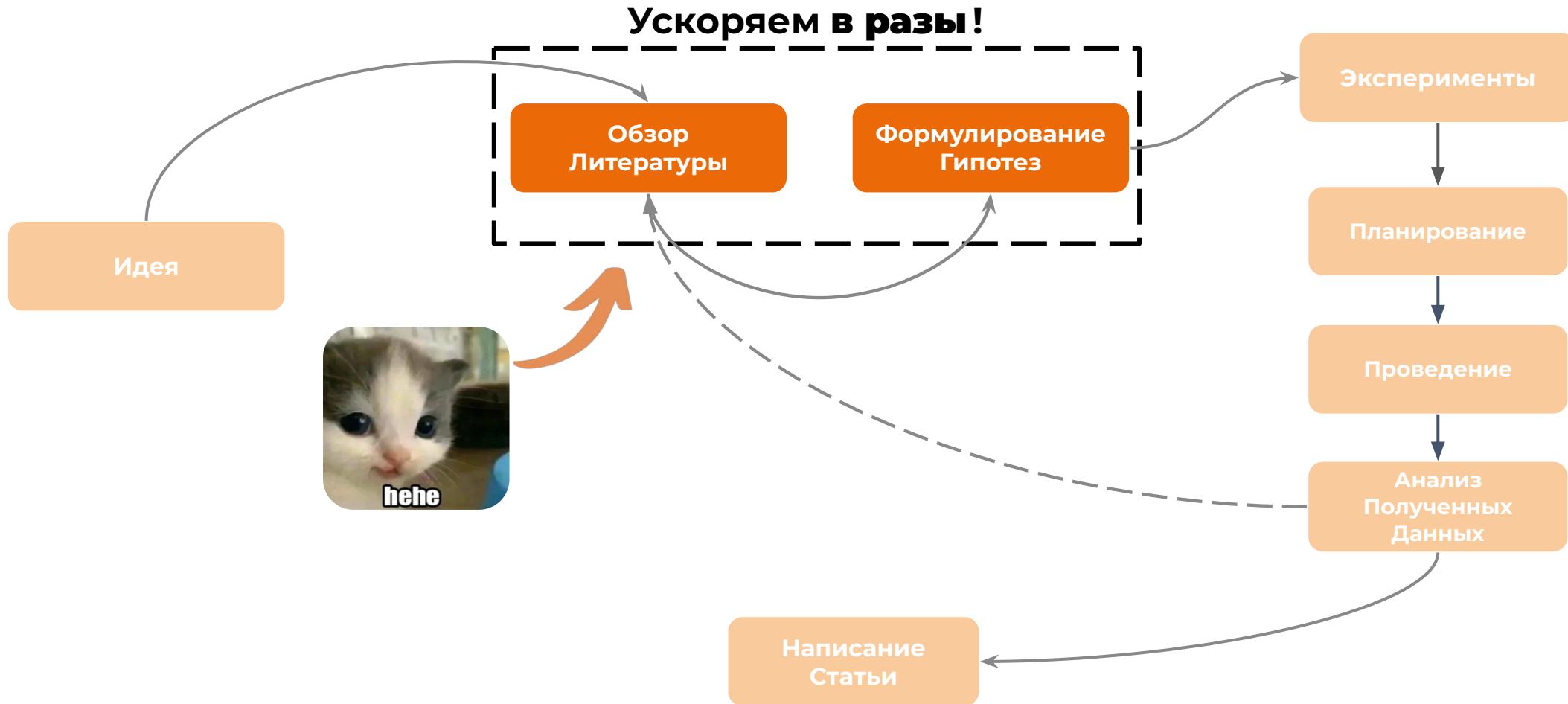
...а скорость прорывов замедляется.
За последние 50 лет
«разрушительность» (disruptiveness)
научных работ и патентов упала более
чем на 90%

Ученые тратят ~23% рабочего времени на поиск актуальной и полезной информации,
что снижает эффективность исследований.²

1. <https://blog dblp.org/2020/03/24/dblp-computer-science-bibliography-surpasses-5-million-publications/>
2. <https://blog scielo.org/es/2014/04/03/habitos-de-lectura-de-la-literatura-cientifica-entre-los-investigadores/>



Обновленная схема работы ученого:





Что хотят видеть пользователи?

Пользовательский запрос

Область и проблема, которой занимается ученый.



Что выдает система

- Научные гипотезы, проверенные критикой
- Обоснования: источники, выводы, аргументы

Пример запроса:

“Как улучшить объяснимость мультимодальных моделей?”

Пример ответа:

“Интерпретируемость мультимодальных моделей может расти, если визуальный ввод предварительно агрегирован в семантические кластеры”



Для кого мы это делаем?



Исследовательские группы (R&D Labs)



Проблема: риск упустить прорывные идеи из смежных областей.



Наше решение: AI, который находит **неочевидные, междисциплинарные связи.**



R&D отделы в компаниях

Проблема: высокая стоимость неудачных R&D проектов.

Наше решение: быстрая и дешевая проверка десятков идей перед вложением миллионов.



Анализ ключевых проектов в области AI-агентов

Проект	Проверка гипотезы	Готовый UI	Доступность в России	Возможность корректировки человеком	Мультидоменность
 Наш Проект	✗	✓	✓	✓	✓
 Google Co-Scientist	✗	✗	✗	✓	✓
 Zochi	✗	✓	✗	✓	✓
 AI-Descartes	✗	✗	✗	✗	✗
 A-Lab	✓	✗	✗	✗	✗
 ChemCrow	✗	✗	✓	✗	✗
 Gemini DeepResearch	✗	✓	✗	✗	✓



Большие данные, искусственный интеллект,
автоматизированные системы и безопасность

Наша цель

Создать MVP AI-ассистента, который **автоматизирует** анализ литературы и позволяет исследователям получать **качественные научные гипотезы**.



Ключевые этапы реализации



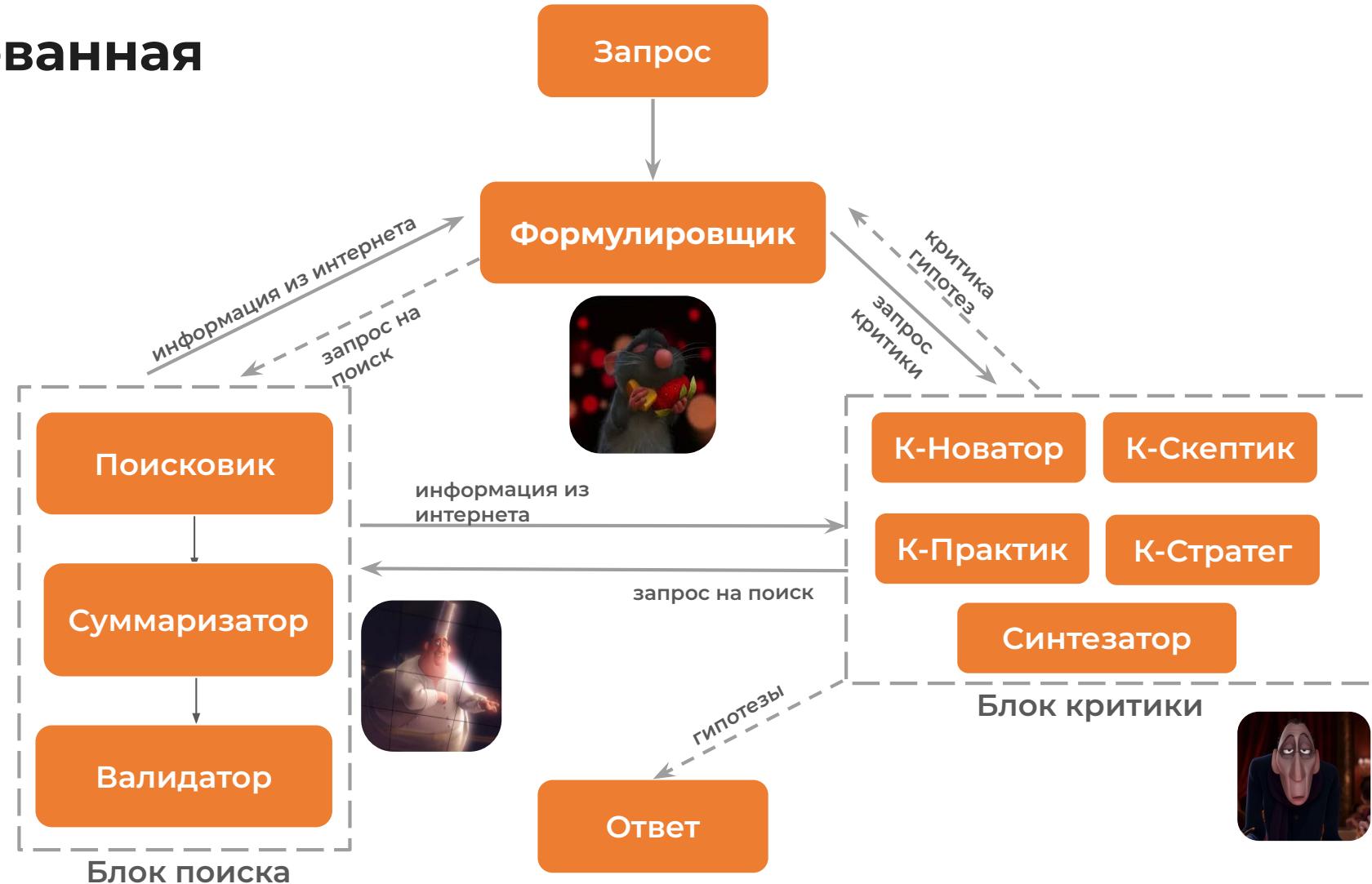


Иерархическая архитектура





Децентрализованная архитектура





Выбор фреймворка для оркестрации агентов

LangChain Agents 
Линейный исполнитель



LangGraph 
Надежный архитектор



AutoGen 
Командный "штурм"



Как работает?

- Шаги выполняются по заданному графу.
- Полный контроль у разработчика.
 - Циклы и условия — основа.

Вердикт:

 Идеально для нашего процесса.



Оркестрация агентов

С помощью LangGraph

Фича LangGraph	Наша система
StateGraph	Движок всего процесса
GraphState	"Память" системы
Вложенные графы	Модульность и инкапсуляция
Условные переходы	"Мозг" принимающий решения
Возможность работать с разными LLM	Разнообразие мышления



Выбор инструментов



OpenAlex API

Плюсы

- Большой объем знаний
 - Бесплатный
- Широкий охват по всем дисциплинам

Минусы

- Выдает только ссылки
- Данные могут быть "зашумлены"

Вердикт:

Идеально для нашего процесса.



arXiv API

Плюсы

- Доступ к полным текстам
- Бесплатный и надежный API
- Стандарт для CS, физики, математики

Минусы

- Ограниченный охват дисциплин
- Нет данных о цитированиях и связях между статьями

Вердикт:

Идеально для нашего процесса.



Выбор инструментов

PyMuPDF

Плюсы



- **Очень высокая скорость**
- **Точное извлечение** текста, изображений, метаданных.

Минусы



- Для извлечения таблиц требует больше кода.

Вердикт:



✓ **Идеально подходит** . Нам нужны **скорость и точность** .



Ключевые характеристики промптов

1. Ролевое промптирование

Преимущество:
максимальное качество
критики и
многогранный анализ.

2. Использование английского языка

Преимущество: гарантия
доступа к передовым
возможностям модели
(State-Of-The-Art).

3. Строгий формат вывода

Преимущество:
надежность, автоматизация
и простая интеграция в
систему.

4. Динамическая контекстуализация

Преимущество:
обеспечивает принятие
решений на основе
данных, а не догадок.

5. Агентная архитектура

Преимущество:
полностью автономный и
логически связанный
процесс исследования.

6. Контекстное обучение **(Few/One-Shot)**

Преимущество: высокая
точность на специфических
задачах и быстрое
адаптирование модели.



Input

Введите запрос
Введите запрос для генерации гипотез.

Разработка методов ускорения инференса и снижения потребления памяти LLM

Отправить

Output

Гипотезы (end) Успешно!

Окончательный вывод гипотез.

Гипотезы:

1. Hypothesis 1: Investigate the feasibility of integrating ABQ-LLM [Zeng et al., N/A] with Medusa [Cai et al., N/A] to accelerate LLM inference. Specifically, explore the use of ABQ-LLM's arbitrary-bit quantization to quantize the weights and

arxiv.org/html/2408.08554v1 arxiv.org/pdf/2401.10774 [ещё 2](#)

Формат взаимодействия

Гипотезы
Окончательный вывод гипотез.

Гипотеза 1. Статус: подтверждена

1. Я выдвигаю гипотезу, что мы можем достичь мелкозернистого контроля над атрибутами объекта, позой и компоновкой сцены, используя многостадийную архитектуру диффузионной модели. Эта архитектура будет состоять из двух отдельных моделей: Модель для контента: Вдохновленная возможностями распутывания (disentanglement) *Uncovering the Disentanglement...*, эта модель будет изучать представление содержания сцены. Модель для стиля/атрибутов: Подобная StyleDiffusion *StyleDiffusion: Controllable D...*, эта модель будет отвечать за манипуляцию конкретными атрибутами, позой и компоновкой. Эти две модели будут соединены через модуль отображения латентного пространства, что позволит контролировать изменять атрибуты, не затрагивая основное содержание.

2. План реализации Обучение двух моделей: Обучить две отдельные диффузионные модели: одну для контента на большом наборе данных изображений, другую для атрибутов – на наборе данных с аннотированными атрибутами, позами и компоновками. Латентное отображение: Ключевым компонентом будет модуль отображения латентного пространства, который переводит представление контента в пространство, управляемое атрибутами. Этот модуль будет обучаться с комбинацией перцептивных потерь, потеря на основе CLIP и потеря на распутывание (disentanglement loss). Оценка: Модель будет оценена на отдельном наборе данных по точности манипуляции атрибутами, сохранению контента и степени распутывания (независимости) различных атрибутов. Новизна и риски: Подход отличается явным разделением контроля над контентом и атрибутами на две отдельные диффузионные модели. Это идея с высоким риском и высокой отдачей, поскольку она усложняет архитектуру, но обещает больший контроль и лучшее распутывание представлений.

Критика к гипотезе 1.

Сводка: Предлагаемая многостадийная архитектура перспективна для детального



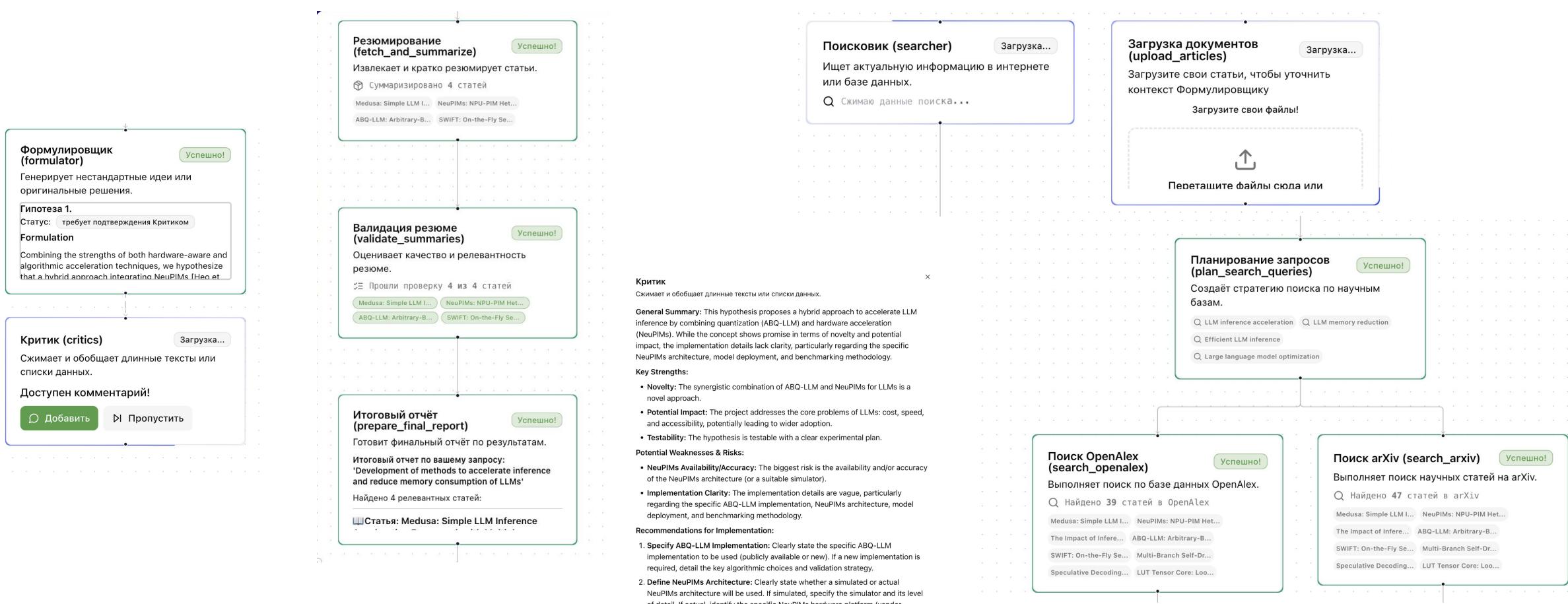
Сравнение архитектур

Архитектура powered by Gemini 2.5 Flash*	Корректные гипотезы (по мнению экспертов)	НЕ плаgiat (из корректных)	Скорость работы (в среднем)	Количество токенов (в среднем)	Количество гипотез (в среднем)
Иерархическая	81%	33%	~7 min	~70k	2
Децентрализованна я	72%	75%	~5 min	~48k	2

*в финальной версии пользователь сможет сам выбирать LLM



Демонстрация





Большие данные, искусственный интеллект, автоматизированные системы и безопасность

Демонстрация





Метрики



66%

гипотез корректны по
мнению экспертов



327 сек

средняя скорость генерации
2 гипотез



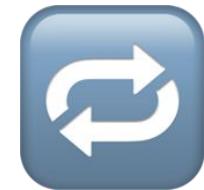
5%

принципиально
новые идеи



53%

комбинации
существующих идей



42%

повторение существующих
методов

Наши руководители

Руководители проекта



Трунов Никита



Свидченко Олег

Наша команда

Спасибо за внимание!



Артемов Артём
г. Москва



Новожилов Арсений
г. Череповец



Мещеряков Марк
г. Санкт-Петербург



Житин Иван
г. Казань



Куликов Данил
г. Череповец

Выбор фреймворка для оркестрации агентов

LangGraph — Наш выбор Надежный архитектор



Как работает?

- Шаги выполняются по заданному графу.
- Полный контроль у разработчика.
- Циклы и условия — основа.

Вердикт:

 Идеально для нашего процесса.

LangChain Agents Линейный исполнитель



Как работает?

- Следует жесткой цепи "Мысль-Действие".
- Контроль у самого агента.

AutoGen Командный "штурм"



Как работает?

- Агенты свободно общаются в чате.
- Поток хаотичен и непредсказуем.

Вердикт:

 Слишком хаотично для науки.

Оценка экспертами



20%

принципиально новые
идеи



Из них 16.7%
корректны

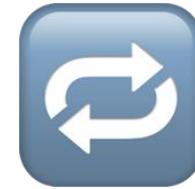


40%

комбинации
существующих идей



Из них 84.6%
корректны



40%

повторение существующих
методов



Из них 64.3%
корректны

Запросы при тестировании

- 1 "Improve sample efficiency in off-policy reinforcement learning algorithms."
 - Суть: Как заставить RL-агентов учиться быстрее на меньшем количестве данных.
- 2 "Develop a computationally efficient alternative to the standard attention mechanism for long sequences in Transformers."
 - Суть: Как сделать Трансформеры быстрее и менее требовательными к памяти при работе с длинными текстами.
- 3 "Mitigate catastrophic forgetting in neural networks during continual learning."
 - Суть: Как научить модель постоянно осваивать новые задачи, не забывая старые.
- 4 "Combine graph neural networks and transformers for better code understanding and generation."
 - Суть: Как использовать структуру кода (граф) и его семантику (текст) вместе для улучшения ИИ-ассистентов программиста.
- 5 "Adapt generative adversarial networks (GANs) for privacy-preserving tabular data synthesis."
 - Суть: Как создавать синтетические, но реалистичные таблицы данных (например, клиентов банка), не раскрывая личную информацию.
- 6 "Develop a novel optimization algorithm that escapes saddle points faster than Adam."
 - Суть: Как улучшить стандартный метод обучения нейросетей (Adam), чтобы он не "застревал" на сложных участках ландшафта потерь.
- 7 "Explore new pruning or quantization techniques to run large language models on edge devices."
 - Суть: Как сжать огромные модели (типа GPT) для запуска на смартфонах или других маломощных устройствах.
- 8 "Create a post-hoc explanation method for computer vision models that is more faithful to the model's internal reasoning."
 - Суть: Как создавать "карты внимания" (heatmaps), которые действительно показывают, на что смотрит модель, а не просто подсвечивают объект.
- 9 "Increase the robustness of image classifiers against adversarial attacks without significantly harming accuracy."
 - Суть: Как защитить нейросети от "обманчивых" картинок, которые заставляют их ошибаться, не ухудшая при этом их работу на обычных изображениях.
- 10 "Investigate alternatives to backpropagation for training deep neural networks to reduce memory footprint."
 - Суть: Как обучать глубокие нейросети, используя меньше видеопамяти, возможно, отказавшись от стандартного алгоритма обратного распространения ошибки.
- 11 "Enhance the factual accuracy of Large Language Models by integrating verifiable knowledge from external sources during generation."
 - Суть: Как научить LLM (типа ChatGPT) проверять факты в реальном времени по базам данных или интернету, чтобы он не выдумывал информацию.
- 12 "Develop a method for fine-grained, controllable image synthesis in diffusion models using textual and structural guidance."
 - Суть: Как заставить нейросеть для генерации картинок (типа Midjourney/Stable Diffusion) точно следовать инструкциям по композиции, позам объектов и стилю, а не просто генерировать что-то.
- 13 "Improve few-shot learning capabilities in computer vision models by leveraging self-supervised pre-training on large unlabeled datasets."
 - Суть: Как научить модель распознавать новый объект всего по 1-5 картинкам, а не по тысячам, предварительно "насмотревшись" огромного количества изображений без разметки.
- 14 "Optimize Neural Radiance Fields (NeRFs) for real-time rendering and dynamic scene representation."
 - Суть: Как создавать реалистичные 3D-сцены из набора фотографий и "ходить" по ним в реальном времени, а не ждать рендеринга часами.
- 15 "Design a robust offline reinforcement learning algorithm capable of learning effective policies from static, sub-optimal datasets."
 - Суть: Как обучить робота, просто показывая ему видеозаписи чужой (не всегда удачной) работы, без возможности пробовать самому и ломать оборудование.
- 16 "Develop a unified multi-modal architecture for joint reasoning over text, images, and audio."
 - Суть: Как создать ИИ, который может посмотреть фильм и потом ответить на вопросы и по сюжету (текст), и по картинке, и по звуку.
- 17 "Integrate causal inference principles into deep learning models to improve out-of-distribution generalization and robustness."
 - Суть: Как научить модель понимать реальную причину и следствие, чтобы она не ломалась на данных, которые немного отличаются от обучающих.

Запросы при тестировании

- 18 "Create a scalable oversight method for aligning Large Language Models with complex human values and preventing harmful outputs."
- Суть: Как сделать так, чтобы очень умный ИИ оставался безопасным и действовал в интересах людей, а не во вред, даже когда задачи очень сложные и многогранные.
- 19 "Develop a resource-aware Neural Architecture Search (NAS) algorithm that automatically designs optimal network architectures for specific hardware constraints (e.g., latency, power)."
- Суть: Как создать ИИ, который сам придумывает наилучшую архитектуру нейросети для конкретной задачи и конкретного устройства (например, для камеры в смартфоне).
- 20 "Adapt Transformer architectures for multi-variate time-series forecasting to better capture long-range dependencies and cross-variable interactions."
- Суть: Как использовать технологию Трансформеров для более точного предсказания сложных систем, где множество факторов влияют друг на друга, например, для прогноза цен на акции или погоды.
- 21 "Develop optimization techniques for accelerating inference and reducing the memory footprint of Large Language Models (LLMs)."
- Суть: Как заставить большие языковые модели работать быстрее и занимать меньше места в памяти, чтобы их можно было запускать даже на обычных компьютерах или смартфонах, а не только на мощных серверах.
- 22 "Design novel LLM architectures capable of efficiently processing extremely long contexts."
- Суть: Как научить языковую модель "помнить" и анализировать не пару абзацев, а целую книгу или многочасовую запись разговора, не теряя при этом сути и важных деталей.
- 23 "Develop approaches to mitigate hallucinations and enhance the factual accuracy of Large Language Models."
- Суть: Как заставить модель перестать выдумывать факты (галлюцинировать) и научить её всегда говорить правду, проверяя информацию и опираясь на реальные данные.
- 24 "Engineer methods for fine-grained, controllable generation to precisely manage the style and content of LLM-produced text."
- Суть: Как создать "пульт управления" для языковой модели, чтобы можно было точно задать, в каком стиле (например, официально или по-дружески) и о чём именно она должна написать текст, управляя каждой деталью.
- 25 "Design architectural solutions to equip LLMs with multi-step reasoning, planning, and external tool utilization capabilities."
- Суть: Как научить модель не просто отвечать на вопрос, а решать сложные задачи в несколько шагов: составлять план, искать информацию в интернете, использовать калькулятор и другие инструменты, как это делает человек.
- 26 "Create methods to combat catastrophic forgetting in continuously trained (Lifelong Learning) LLMs."
- Суть: Как дообучать модель новым знаниям (например, свежим новостям), не заставляя её при этом забывать всё, что она знала раньше.
- 27 "Design unified multi-modal architectures for joint reasoning across text, images, and audio within a single LLM."
- Суть: Как создать одну универсальную модель, которая сможет одновременно понимать текст, "видеть" картинки и "слышать" звук, чтобы делать выводы на основе всей этой информации вместе, как человек.
- 28 "Investigate and develop non-Transformer architectures as alternatives for advanced language modeling."
- Суть: Как придумать совершенно новую технологию для создания языковых моделей, которая будет эффективнее или мощнее, чем существующие сейчас Трансформеры.
- 29 "Develop LLM architectures with built-in mechanisms for self-correction and logical consistency checking."
- Суть: Как встроить в саму модель способность находить собственные ошибки в рассуждениях, перепроверять свою логику и исправлять ответы, прежде чем показать их пользователю.
- 30 "Create methods for the automatic detection and mitigation of malicious use and jailbreaking attempts in LLMs."
- Суть: Как научить языковую модель автоматически распознавать, когда её пытаются обмануть, "взломать" или заставить генерировать вредный контент, и самостоятельно блокировать такие попытки.

Что такое научная гипотеза ?

Научная гипотеза — это обоснованное предположение о причинах, механизмах или закономерностях наблюдаемого явления, которое подлежит проверке и может быть подтверждено или опровергнуто в ходе исследования.

Гипотезы направляют исследование, позволяют формулировать цели, выбирать методы и проверять научные идеи.



Гипотеза — это...

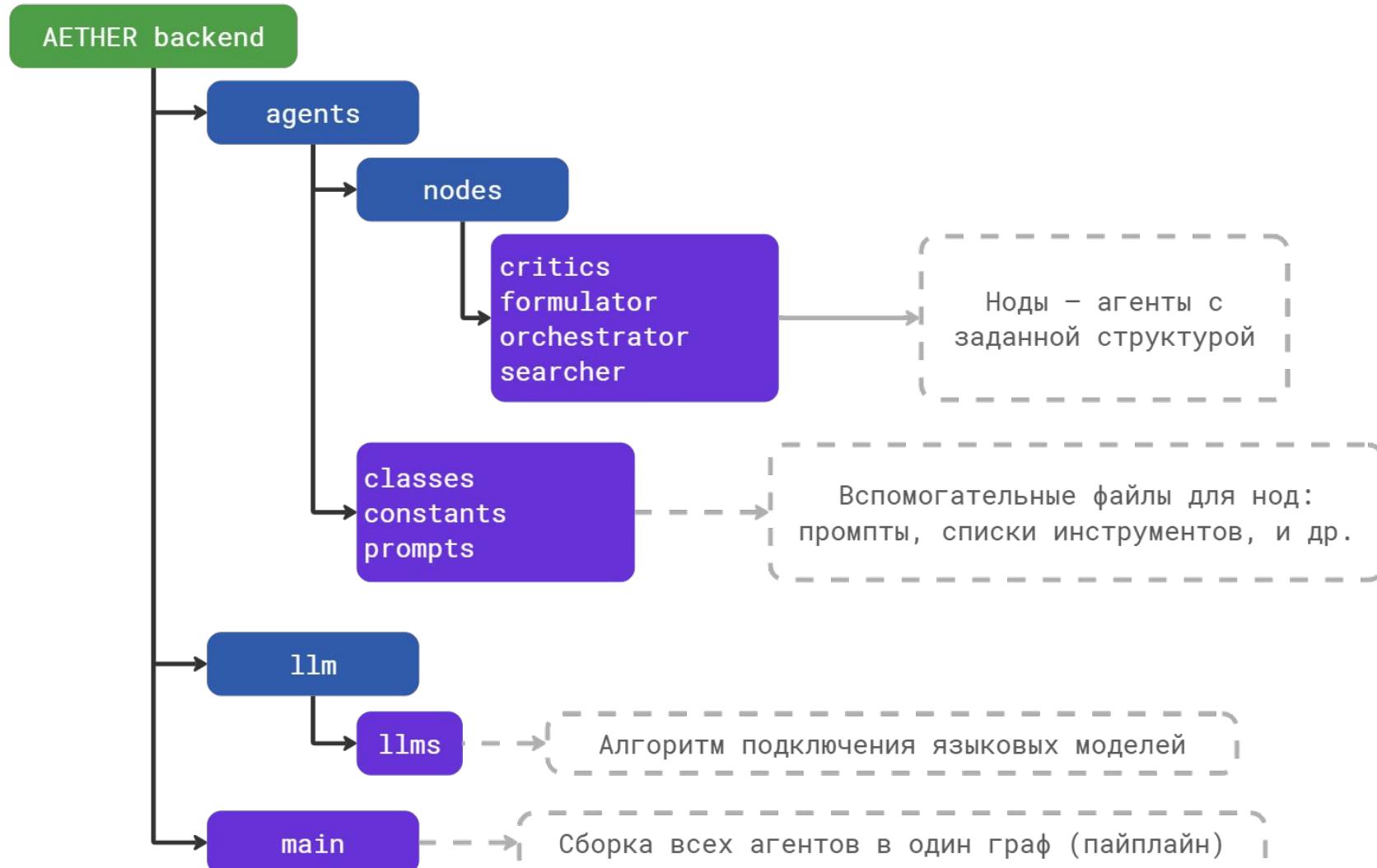
- ❖ Основа для планирования экспериментов
- ❖ Инструмент для выдвижения новых идей



Гипотеза должна быть:

- ❖ Проверяемой (фальсифицируемой)
- ❖ Обоснованной (с опорой на данные/наблюдения)
- ❖ Конкретной и формализуемой

Архитектура серверной части



Оркестрация агентов

Оркестрация агентов

С помощью LangGraph

Фича LangGraph	Наша система
StateGraph	Движок всего процесса. Это сборщик графа, который объединяет "Формулировщик", "Блок поиска" и "Блок критики" в единую работающую систему.
GraphState	"Память" системы. Это единый объект, который передается по всем стрелкам на схеме. Он хранит актуальные гипотезы, историю поиска и результаты критики.
Вложенные графы	Модульность и инкапсуляция. Наш "Блок поиска" — это отдельный, независимый граф со своей логикой и циклами (Поисковик -> Суммаризатор -> Валидатор).
Условные переходы	"Мозг" принимающий решения. Реализует логику ветвления: после "Критики" решает, идти ли к "Ответу" или вернуться к "Формулировщику" на новый круг.
Возможность работать с разными LLM	Разнообразие мышления. LLM - основа всей сети, возможность менять модель позволяет экспериментировать с разными методами и подходами.

Выбор инструментов



OpenAlex API

Плюсы

- Большое количество знаний
 - Бесплатный
- Широкий охват по всем дисциплинам



Минусы

- Выдает только ссылки
- Данные могут быть "зашумлены"



Вердикт:

Идеален для построения "карты знаний" и поиска связей, но недостаточен для анализа текстов



arXiv API

Плюсы

- Гарантированный доступ к полным текстам (PDF)
- Бесплатный и надежный API
 - Стандарт для CS, физики, математики



Минусы

- Ограниченный охват дисциплин
- Нет данных о цитированиях и связях между статьями

Вердикт:

Идеален для получения текстов, но не подходит для первичного, широкого поиска



Scopus / Web of Science

Плюсы

- Высочайшее качество и достоверность данных
 - Престиж в академической среде

Минусы

- Закрытый и ограниченный API
- Меньший охват, чем у OpenAlex

Вердикт:

Недоступен. Слишком дорого и закрыто для нашего проекта.



Google Scholar

Плюсы

- Максимально возможный охват по всем темам.
- Отлично находит самые свежие работы

Минусы

- Нет официального API
- Парсинг страниц нестабилен и рискован.
- "Грязные" данные с дубликатами.

Вердикт:

Ненадежен. Нельзя строить критически важную систему на основе парсинга.

Выбор инструментов

PyMuPDF

Плюсы

- **Очень высокая скорость** — самый быстрый из всех.
- **Точное извлечение** текста, изображений, метаданных.
- Активно поддерживается и развивается.



Минусы

- Для извлечения таблиц требует больше кода.

Вердикт:



✓ Идеально подходит. Нам нужны **скорость и точность** для обработки большого количества научных статей.

pdfplumber

Плюсы

- **Лучший в извлечении таблиц** в структурированном виде.
- Хорошо работает со сложной версткой.
- Удобный API для работы с объектами на странице.

Минусы

- **Заметно медленнее**, чем PyMuPDF.



PyPDF2

Плюсы

- **Многофункциональность** (редактирование, слияние).
- Прост в использовании для базовых операций.

Минусы

- Часто возникают проблемы с извлечением текста из сложных PDF (пробелы, кодировки).
- **Медленный**.

Вердикт:

✗ Ненадежен. Его проблемы с извлечением текста могут привести к потере данных, что для нас критично.