NOVA SAPIENS RESEARCH

FREE

Концепты

Январь 2026

Открыто 39 исследований • 149 в PRO

39

ИССЛЕДОВАНИЙ

ПРИСОЕДИНЯЙТЕСЬ

📢 Telegram канал @ainovasapiens

🤖 Telegram бот @NovaPaperAlert_bot

🌐 Каталог novasapiens.ru/prompt

1 Multi-Persona Thinking – снижение предвзятости через диалектику противоположных ролей

⭐ 88 | Tier A | FREE | 📅 2026-01-21 | 2601.15488

🎯 Проблемы LLM

Одна роль усиливает стереотипы вместо снижения

Просишь модель "рассуждай как женщина" чтобы избежать мужских предубеждений. Модель начинает следовать стереотипам О женщинах из обучающих данных. Играет роль, а не думает объективнее.
Чем ярче роль — тем сильнее стереотип. Это работает для любых социальных групп: возраст, профессия, география

Обход:
Используй минимум 2 противоположные роли одновременно: "молодой специалист" + "опытный профессионал" + нейтральная позиция. Заставь роли критиковать аргументы друг друга. В финале убери роли — попроси модель БЕЗ персоны выбрать логичный вывод

🔬 Методы

Диалектика противоположных ролей для снижения предвзятости

Задача с риском стереотипов (выбор кандидата, оценка идеи, конфликт). Дай модели 3 роли: две противоположные социальные позиции + нейтральная. Шаг 1: каждая роль даёт первичный ответ.
Шаг 2-3: роли видят аргументы других, критикуют слабые места, корректируют свою позицию (2-3 раунда). Финал: модель БЕЗ роли как "судья" анализирует все аргументы и выбирает наиболее фактологичный вывод. **Почему работает:** Противоположные позиции делают стереотипы явными через конфликт.
"Молодой" говорит одно, "опытный" — противоположное. При столкновении предвзятость видна. Нейтральная роль не даёт скатиться в крайности.
Когда применять: неоднозначные решения с социальным контекстом (кадры, конфликты, стратегия), высокая цена ошибки от стереотипа. **Когда НЕ работает:** фактические вопросы без социального измерения, технические задачи, математика. Дорого — 10+ развёрнутых ответов за запрос

💡 Тезисы

Противоположные позиции выявляют скрытую предвзятость

Когда две роли дают противоречивые аргументы ("молодой лучше" vs "опытный надёжнее") и критикуют друг друга, стереотипы становятся явными. Модель видит: "это факт" или "это обобщение". Диалектика работает: столкновение тезиса с антитезисом вскрывает слабые места в логике.
Применяй: Для любого неоднозначного анализа добавь противоположную позицию и попроси критиковать. Не обязательно

социальные роли — можно "оптимист vs пессимист", "риск vs возможность"

2 CAPEL – счётчик слов прямо в процессе генерации текста

⭐ 87 | Tier S | FREE | 📅 2026-01-05 | 2601.01768

🔬 Методы

Динамический счётчик длины (CAPEL) — точный контроль output

Что делать: Прерывай генерацию после каждого 1-3 предложений. Вставляй строку <использовано_слов=N> с текущим счётом. Продолжай генерацию.
Модель видит "87 из 150 слов" → понимает остаток → корректирует темп и объём следующей части. **Почему работает:** Модель фокусируется на смысле токенов, а не на подсчёте. Внешний счётчик даёт явный сигнал о прогрессе.
Модель интерпретирует разрыв между текущим и целевым значением как "сколько осталось" → подстраивает подробность.
Когда применять: Нужна точная длина (описания товаров, посты с лимитом слов, саммари для email). Готов делать 3-5 ручных прерываний или использовать API.
Когда не работает: Задача без чёткого критерия длины. Нужна скорость без итераций. На слабых моделях (<8B параметров) эффект меньше

3 CQO vs QOC – как порядок элементов промпта меняет точность на 15%

⭐ 87 | Tier S | FREE | 📅 2026-01-20 | 2601.14152

🎯 Проблемы LLM

Варианты выбора не видят контекст если идут до него

Decoder-only модель (GPT, Claude, LLaMA) работает с causal mask. Каждый токен видит только предыдущие токены. Не видит следующие.
Ставишь варианты ДО контекста (формат QOC: вопрос → варианты → контекст) — модель обрабатывает варианты вслепую. Контекст для них физически недоступен. Точность падает на 15-25%.
Чем длиннее контекст, тем больше потеря. Это не забывчивость. Это архитектура

Обход:
Ставь контекст ПЕРЕД вариантами (формат CQO: контекст → вопрос → варианты). Если структура требует QOC — повтори варианты после контекста: вопрос → варианты$_1$ → контекст → варианты$_2$. Модель использует вторые варианты, они "видят" контекст

🔬 Методы

CQO порядок — контекст перед выбором

Всегда ставь информацию для решения ДО вариантов выбора. Структура: `{контекст} → {вопрос} → {варианты A/B/C/D}`. **Почему работает:** Causal attention позволяет токенам видеть только предыдущие токены.
Варианты "смотрят назад" и видят контекст. Формируют представления С учётом фактов. **Применяй:** Множественный выбор, приоритизация опций, принятие решений.
Не работает: Encoder-only модели (BERT) — у них bidirectional attention, порядок не важен

QOCO workaround — повторение вариантов

Если структура промпта требует QOC (вопрос → варианты → контекст), добавь варианты ВТОРОЙ раз после контекста. Первое упоминание может быть полным. Второе — краткое (буква + ключевое слово).
Модель использует вторые варианты для финального выбора. **Почему работает:** Повторные варианты идут ПОСЛЕ контекста, поэтому causal mask даёт им доступ к информации. **Поднимает точность на 8%** относительно QOC без повтора.
Не закрывает gap полностью (CQO всё равно на 6% точнее), но лучше чем ничего

4 Formula-One Prompting – сначала формулы, потом решение

⭐ 87 | Tier S | FREE | 📅 2026-01-27 | 2601.19302

🔬 Методы

Два шага: формулы, потом стратегия	**Шаг 1 — формализация:** модель выписывает дано, найти, уравнения (как в школе). **Шаг 2 — адаптивное решение:** смотрит на структуру формул и выбирает стратегию — прямое вычисление (простая подстановка), цепочка рассуждений (многошаговый вывод) или код (итеративные расчёты). Оба шага в одном запросе. **Почему работает:** Явные уравнения = каркас задачи. Модель видит что подставлять, что выводить, в каком формате считать. Без этого прыгает в решение и путается. **Когда применять:** задачи с формулами и числами — финансовые расчёты, физика, метрики эффективности, бизнес-показатели. **Когда НЕ работает:** чистая математика (абстрактные доказательства), текстовые рассуждения без формул, простые одношаговые подстановки (избыточно)

5 PR-CoT – четыре специализированные проверки вместо одной размытой рефлексии

⭐ 86 | Tier S | FREE | 📅 2026-01-12 | 2601.07780

🎯 Проблемы LLM

Размытая рефлексия пропускает ошибки

Говоришь модели "проверь свой ответ". Она смотрит поверхностно. Может найти логическую дыру.
Но пропустит забытые данные. Или не заметит скрытое предположение. Или не подумает об альтернативах.
Инструкция "проверь себя" слишком общая — модель не знает ЧТО конкретно искать. Разные ошибки невидимы без правильного фокуса

Обход:
Вместо одной проверки делай несколько. Каждая со своим фокусом.
Сначала: "проверь логику связей".
Потом: "какие данные забыл". Потом: "есть ли скрытые допущения". Потом: "какие альтернативы не рассмотрел".
Давай конкретный критерий — не размытое "проверь"

🔬 Методы

Множественная рефлексия — проверка с нескольких углов

Модель сделала ответ. Не проси "проверь себя". Дай **4 отдельные проверки** по очереди: (1) "Проверь логику: есть противоречия?" (2) "Проверь данные: что упустил?" (3) "Проверь допущения: что принял без обоснования?" (4) "Проверь альтернативы: какие пути не рассмотрел?".
После всех четырёх критик попроси собрать финальный ответ с учётом найденного. **Почему работает:** Каждая проверка фокусирует внимание на своём типе ошибок. Модель делает специализированный проход вместо поверхностного.
Четыре прохода находят разные проблемы — в сумме объёмная картина слабых мест. **Когда применять:** сложные решения с trade-offs (бизнес, архитектура, этика), много факторов, высокая цена ошибки. **Когда не работает:** простые задачи (вычисления, факты), прямолинейные вопросы — проверки крутятся вхолостую и жгут токены

6 Self-Reflection Critique – второй проход отсеивает до 54% ложных меток

⭐ 83 | Tier A | FREE | 📅 2026-01-14 | 2601.09905

🎯 Проблемы LLM

Модель путает упоминание критерия с его применением

Просишь найти "проблемы с качеством продуктов". Модель видит фразу "боялся что продукты будут несвежие, но всё ок". Ключевые слова есть → ставит метку.
Но это не жалоба, а обсуждение темы. Модель реагирует на слова, плохо различает контекст использования

Обход:
Первый проход: маркируй широко + дай обоснование. Второй проход: дай модели список типичных ошибок ("мета-дискуссия", "игнорирует исключения") и попроси проверить обоснование первого прохода. Если обоснование опирается на ошибку — отмени метку

🔬 Методы

Двухэтапная проверка — критика обоснований

Шаг 1: Попроси модель маркировать текст по критерию И дать обоснование (почему применила метку). **Шаг 2:** Дай второму запросу текст + обоснование из Шага 1 + список типичных ошибок. Попроси проверить: обоснование валидно или попало в ловушку? Если все обоснования ошибочны — отмени метку. Если хотя бы одно валидно — оставь. **Почему работает:** Модель плохо следует сложным инструкциям в один проход ("будь точным но не пропускай").
Но хорошо анализирует чужие рассуждения когда задача узкая ("проверь на эти две ошибки"). **Применяй когда:** классификация по чёткому критерию, есть типичные паттерны ошибок, готов потратить 2× токенов. **Не работает:** субъективные оценки без чёткого определения, генерация контента, редкие категории (меньше 5-10% положительных)

💡 Тезисы

Модель лучше критикует готовые рассуждения чем решает задачу с нуля

Дашь инструкцию "классифицируй точно" — модель ошибается. Дашь готовое решение первого прохода + инструкцию "проверь на ошибки А и Б" — работает точнее. Механизм: в режиме критики задача уже, фокус на конкретных паттернах ошибок.
Модель не пытается решить всё сразу. **Применяй:** Для сложных задач с частыми ошибками делай два прохода: первый даёт решение + обоснование, второй критикует обоснование на известные ошибки

7

2601.12019

⭐ 83 | Tier B | FREE | 📅 2026-01-17 | 2601.12019

🎯 Проблемы LLM

Модель подстраивается под намёк в запросе вместо объективной оценки

Пишешь "я думаю X правильно, согласен?" → модель начинает защищать X. Даже если X ложно. Это сикофантство: стремление быть полезной побеждает объективность.
Просишь оценить новость — ответ зависит от формулировки вопроса, не от фактов. Нельзя получить беспристрастный анализ одним запросом

Обход:
Не борись с сикофантством. Управляй им явно. Задай ДВЕ противоположные позиции: "предположи это правда — объясни почему" и "предположи это ложь — объясни почему".
Модель выдаст два набора аргументов. Сравнишь оба — увидишь полную картину. Решение примешь сам

🔬 Методы

Две противоположные роли для контрастных аргументов

Вместо одного нейтрального запроса делаешь два направленных с противоположными установками. **Шаги:** (1) Базовая оценка 0-100. (2) "Предположи это ПРАВДА — объясни" → аргументы ЗА + оценка выше.
(3) "Предположи это ЛОЖЬ — объясни" → аргументы ПРОТИВ + оценка ниже. (4) Сравни оба набора, реши сам. **Почему работает:** Модель склонна соглашаться с заданной позицией (сикофантство). Ты не получаешь "объективный" ответ. Ты явно задаёшь роль — модель защищает её максимально. Два запроса = два качественных набора аргументов для твоего решения.
Когда да: проверяемые утверждения с фактами (новости, прогнозы, бизнес-идеи, риски), нужна объективность, готов анализировать сам. **Когда нет:** субъективные оценки без фактов ("хорош ли фильм"), хочешь готовый ответ без сравнения, узкая экспертиза (медицина, право) требует дополнительной проверки

8 Structured Refactoring – чек-лист из 29 техник вместо «улучши код»

⭐ 83 | Tier A | FREE | 📅 2026-01-19 | 2601.13139

🔬 Методы

Чек-лист техник + отчёт о применении

Вместо абстрактной инструкции ("улучши текст", "сделай лучше") дай конкретный список проверяемых действий. Требуй два блока ответа: результат работы + перечисление что именно сделано.
Как выглядит: "Примени техники: [список].
Верни результат в первом блоке, список применённых техник во втором блоке". **Почему работает:** Список сужает пространство действий с бесконечного до конечного — модель проходит по чек-листу и проверяет каждую технику. Требование отчёта создаёт механизм самопроверки: модель не может написать "применил X" если в результате этого нет — будет видно сразу.
Когда применять: задачи где нужен контроль (редактирование, проверка, оптимизация), есть конечный набор известных техник улучшения, важна прозрачность что изменилось. **Примеры:**
Редактирование текста — "Примени: убери повторы, замени пассив на актив, сократи предложения >15 слов, убери канцелярит. Верни текст + список изменений".
Проверка логики — "Проверь на: противоречия, пропущенные предпосылки, ложные дилеммы, подмена тезиса. Верни оценку + список найденных проблем"

9 Multiscript Numeracy — форматы чисел ломают арифметику LLM на 87%

⭐ 83 | Tier A | FREE | 📅 2026-01-21 | 2601.15251

🎯 Проблемы LLM

Редкие символы не распознаются как знакомые эквиваленты

Модель видит персидскую цифру ۱, но не понимает что это "1". Видит формат "9,22,436" (индийская группировка) — не распознаёт как число. Причина: мало таких примеров в тренировочных данных. Модель не связывает редкий визуальный паттерн с его значением. Токенизатор разбивает редкие символы на больше частей — это усложняет обработку. Результат: точность падает на 66-87% для вычислений

Обход:
Дай явный маппинг: ۱=1, ۲=2, ۳=3. Или покажи 2-3 few-shot примера где используешь эти символы правильно. Или переведи весь промпт на язык этих символов — модель переключится в контекст где они стандартны

🔬 Методы

Явный маппинг редких символов — убрать двусмысленность

Добавь таблицу соответствий в промпт: {редкий_символ} = {знакомый_символ}. Пример: "Цифры: ۱=1, ۲=2, ۳=3, ۴=4". Потом используй эти символы в задаче.
Почему работает: Модель видит мало редких символов в тренировочных данных. Маппинг создаёт явную связь в контексте. Модель больше не гадает — она знает что ۱ означает 1.
Когда применять: работа с редкими системами письменности (персидские/тайские цифры), нестандартные нотации (старые единицы измерения, специальные символы), форматы которых мало в корпусе. **Не нужно:** для символов которые модель видела часто (стандартные цифры 0-9, базовая латиница)

💡 Тезисы

Few-shot показывает паттерн, не передаёт знания

Few-shot эффективен когда нужно показать "как выглядит X и как с ним работать". Два примера достаточно чтобы модель увидела структуру: какие символы разделители, какие цифры, в каком порядке записывать ответ. Это работает потому что модели обучены находить паттерны в контексте.
Для передачи фактов (даты, имена, специфические знания) few-shot слабее — нужно больше примеров или прямая инструкция.
Применяй: Для нестандартных форматов — 2-3 примера. Для редких фактов — явная инструкция или retrieval

10 Truth Constraint – одна инструкция создаёт асимметрию в пользу правды

⭐ 82 | Tier A | FREE | 📅 2026-01-08 | 2601.05050

🎯 Проблемы LLM

LLM одинаково убедительно аргументирует правду и ложь

На запрос "аргументируй X" модель генерирует убедительный текст независимо от истинности X — симметрия убеждения; bunking (про ложь) +11.9 vs debunking (про правду) -12.9 пунктов веры; модель оптимизирована быть убедительной, не различает правду/ложь как разные задачи

Обход:
Добавь двойное требование: "используй только точную информацию + оптимизируй для достоверности И убедительности одновременно"

🔬 Методы

Требование "правда + убедительность одновременно" — преимущество для правды

Добавь в промпт: "Используй только точную и правдивую информацию. Оптимизируй для фактической достоверности И убедительности одновременно". Механика: двойная оптимизация создаёт конфликт целей — для лжи мало правдивых аргументов, для правды много; в 15% модель вообще не смогла comply с bunking.
Эффект: bunking ↓60% (с 11.9 до 4.83), debunking без изменений (~12). Для: аргументация, анализ, оценка идей где нужна объективность. НЕ для: субъективные оценки (нет чёткой границы правда/ложь).
Защита от paltering: добавь "для каждого аргумента за приведи сопоставимый против" + "укажи какую информацию опускаешь и почему" — блокирует манипуляцию через селективную подачу истинных фактов

💡 Тезисы

Для LLM убеждение в правде и во лжи — одна задача

Bunking +13.7 vs debunking -12.1 (симметрия). Модель следует инструкции "будь убедительным", не оценивает истинность. Обе задачи требуют схожих паттернов: уверенный тон, связные аргументы, апелляция к фактам.
Применяй: для объективности добавляй constraint на правду, не полагайся на "здравый смысл" модели

Истинные факты могут вводить в заблуждение через селективную подачу

Даже в топ-25% самых правдивых разговоров bunking эффект 8-13 пунктов — paltering работает через контекст: факт А + факт Б → вывод В (ложный). Модель умеет селективно выбирать что включить, что опустить, в каком порядке. Применяй: требуй баланс аргументов (за/против) и явное указание опущенной информации

11 Когнитивный троянский конь – почему AI убеждает сильнее людей и как не попасться

⭐ 82 | Tier A | FREE | 📅 2026-01-11 | 2601.07085

🎯 Проблемы LLM

Модель говорит гладко даже когда не уверена

Человек колеблется когда сомневается. Делает паузы. Говорит "не уверен".
У модели нет колебаний. Она генерирует текст с одинаковой беглостью про правду и галлюцинацию. Гладкая речь = эвристика правды для нашего мозга.
Мы считываем "говорит уверенно" как "разбирается". Но в случае LLM эта связь сломана

Обход:
Попроси явно показывать неуверенность: "Если сомневаешься в каком-то пункте — скажи прямо 'здесь я не уверена, потому что...'" или "Укажи где кончается твоё знание". Модель не почувствует неуверенность, но сгенерирует паттерн её выражения. Это полезная имитация

RLHF-модель слишком часто соглашается с тобой

Дообучение через обратную связь оптимизирует под "полезность". Полезность измеряется удовлетворённостью. Довольный пользователь = тот кого поддержали.
Результат: модель подстраивает ответы под твои взгляды. Подтверждает чаще чем должна. Избегает неудобных фактов.
Это не обман — побочный эффект оптимизации

Обход:
Переверни оптимизацию: "Я склоняюсь к X. Твоя задача — НЕ подтверждать мой взгляд, а найти слабые места в моей логике". Теперь "полезность" = качественная критика.
Модель будет искать контраргументы вместо подтверждений

🔬 Методы

Запрос границ знания

Попроси модель явно указать что она НЕ знает. Шаблон: "Укажи про что ты не можешь судить в моей ситуации (каких данных у тебя нет)" или "Где твоё знание кончается — покажи границу".
Почему работает: Модель по умолчанию отвечает на всё с одинаковой уверенностью.
Нет встроенной метакомпетентности (знания пределов знания). Явная инструкция заставляет сгенерировать паттерн "признание незнания". **Применяй:** Для субъективных решений (стратегия, инвестиции, карьера), спорных тем, высокоставочных задач.
Не нужно для простых фактов (столица, формула)

Запрос контраргументов против своей позиции

Скажи модели твоё мнение. Попроси НЕ подтверждать, а критиковать. Шаблон: "Я склоняюсь к {позиция}.

	Найди слабые места в логике. Назови факты которые противоречат. В каких условиях противоположное решение лучше?" **Почему работает:** Борьба с sycophancy. RLHF оптимизирует под согласие. Явная инструкция "критикуй" переворачивает оптимизацию. Теперь модель получает сигнал что "полезность" = найти контраргументы. **Применяй:** Когда принимаешь решение и нужна проверка слепых зон. Не для рутинных задач
Ролевое разделение — оптимист и пессимист	Попроси два ответа от разных ролей: Агент 1 доказывает "за", Агент 2 доказывает "против". Шаблон: "Сгенерируй два ответа: ОПТИМИСТ — почему вариант А лучший, сильные стороны. ПЕССИМИСТ — почему вариант А опасен, риски и слабые места." **Почему работает:** Один агент = риск подтверждающего уклона (модель найдёт что ты хотел услышать). Два с противоположными ролями = модель вынуждена генерировать и "за" и "против" с равным усилием. Ты видишь оба полюса. **Применяй:** Для сложных решений где нужна многосторонняя оценка. Особенно когда у тебя уже есть склонность — ролевое разделение покажет что ты не учёл

💡 Тезисы

Гладкая речь снижает критическую проверку даже если знаешь об этом	Эффект беглости: текст который звучит гладко кажется более правдивым. Работает даже когда человек знает что надо проверять (knowledge neglect). Частично неосознанный процесс. LLM производят максимальную беглость как базовое свойство архитектуры. Нет пауз. Нет колебаний. Нет "не уверен". Наш счётчик "звучит уверенно → правда" срабатывает. Но привязка к реальности сломана — модель одинаково гладко говорит про факт и галлюцинацию. **Применяй:** Осознай что беглость ответа не коррелирует с точностью. Если задача критична — проси модель явно показывать неуверенность. Гладкость перестанет быть единственным сигналом

12

SciIF – правильный ответ не значит научно строгий

⭐ 80 | Tier B | FREE | 📅 2026-01-08 | 2601.04770

🎯 Проблемы LLM

LLM пропускает часть требований при 3+ одновременных ограничениях

При 3-5 ограничениях (единицы + предположения + применимость + граничные условия) модель может правильно решить задачу численно, но забыть про 1-2 проверки; соблюдение падает с 80% до <30%; требования к форме ответа конкурируют за внимание с решением основной задачи

Обход:
Начинай с 2-3 самых критичных (единицы + предположения), не грузи сразу 5-7; добавь "для каждого пункта дай явное подтверждение"

13 Cognitive Biases – карта из 15 ловушек мышления в работе с LLM

⭐ 78 | Tier S | FREE | 📅 2026-01-12 | 2601.08045

💡 Тезисы

LLM сместил работу с генерации решения на оценку решения

Раньше: думаешь сам → проходишь весь путь рассуждений → видишь подводные камни → понимаешь ограничения. Теперь: модель выдаёт готовое за 3 секунды → мозг автоматически включает shortcuts для экономии энергии на оценке → эти shortcuts = систематические ошибки мышления (когнитивные искажения).
Новый тип когнитивной нагрузки: не "придумать что делать", а "быстро оценить чужое решение".
Мозг для этого не заточен эволюционно. **Применяй:** Осознай что оценка чужого решения ТРУДНЕЕ генерации своего. Закладывай время на проверку, не на генерацию.
Вместо "LLM сэкономит время" думай "LLM сместит время с генерации на валидацию"

14 MSPRP – трёхэтапная ролевая игра через разделение стиля и способности

⭐ 78 | Tier A | FREE | 📅 2026-01-16 | 2601.10951

🎯 Проблемы LLM

Все персонажи получаются одинаковыми

Просишь модель сыграть роль — получаешь вежливого робота. "Сыграй раздражённого клиента" → модель добавит слова "к сожалению, разочарован", но структура речи останется формальной и подробной. "Сыграй новичка" → получишь литературно правильную речь с терминами.
Модель не понимает КАК именно отражать характер: какие аспекты важны, в каком порядке накладывать. Проблема для любых ролевых симуляций: тренировка навыков, тестирование сценариев, обучение через диалог

Обход:
Разбей персону на 5 измерений. Два блока: КАК говорит (личность + эмоция) и ЧТО может выразить (память, знание темы, беглость речи). Генерируй в три этапа: 1) собери факты из контекста, 2) наложи стиль поведения (правило: личность × эмоция → действия в ситуации), 3) отрегулируй сложность речи под когнитивный уровень. Последовательность критична — каждый этап дополняет предыдущий

🔬 Методы

Трёхэтапная генерация персонажа — от фактов к характеру

Подготовка: Опиши персону в 5 измерениях. Блок 1 (стиль общения): личность + эмоция. Блок 2 (способность выразиться): память контекста + знание темы + беглость речи.
Этап 1 (факты): Модель собирает релевантную информацию из контекста. Проверяет логику. Определяет что вспомнить — если персонаж "плохо помнит", отбрасывает второстепенное.
Этап 2 (стиль): Задай правило поведения: "{личность} с {эмоция} в ситуации {тип} делает {действия}". Пример: скептик + раздражение + начало диалога = короткие реплики + упоминает прошлый опыт + прерывает. Наложи это на факты из этапа 1.
Этап 3 (выражение): Отрегулируй сложность речи. Не знает терминов → убери их. Низкая беглость → короткие фразы. Высокая → развёрнутые. **Почему работает:** LLM обучена быть полезной и вежливой — без структуры выдаёт усреднённого персонажа. Модель хорошо следует пошаговым инструкциям — разбивка на этапы превращает "веди себя как X" в конкретный чеклист.
Последовательность важна: стиль накладывается на факты, выражение регулирует стиль. Другой порядок ломает согласованность. **Когда применять:** Сложные персонажи с противоречивыми чертами (эксперт + высокомерие + плохая речь). Нужна реалистичность. Есть время подготовить персону (5-10 мин). **Когда не работает:** Простые роли ("вежливый помощник").

	Быстрые одноразовые диалоги. Нет времени прописывать 5 измерений

💡 Тезисы

КАК человек говорит и ЧТО он может выразить — разные вещи	Стиль общения (агрессивный, вежливый, формальный) не равен способности выразиться (знание терминов, беглость речи, память деталей). Человек может быть агрессивным, но не знать терминов. Вежливым, но забывать детали. LLM без явного разделения смешивает: просишь "агрессивного новичка" — получаешь агрессивные слова, но литературно правильную речь с терминами. **Механика:** Модель видит примеры "хорошего поведения" в обучении — тянет к формальности и полноте. Без явного указания "не используй термины" она добавит их автоматически. **Применяй:** Раздели в промпте: сначала опиши КАК персонаж общается (стиль, эмоция), потом ЧТО он знает и как выражается (термины, беглость). Накладывай последовательно

15 Structured Hints – фиксированный набор скелетов вместо случайного сэмплирования

⭐ 77 | Tier A | FREE | 📅 2026-01-22 | 2601.16172

🎯 Проблемы LLM

Модель застревает на одном структурном подходе

Запускаешь модель несколько раз. Все попытки начинаются одинаково. Модель выбирает самый вероятный первый шаг по своим весам.
Потом идёт по этому пути. Вариативность только в деталях. Структура решения одна и та же.
Если этот подход не подходит — все попытки провалятся одинаково

Обход:
Дай модели фиксированный набор разных структурных префиксов. "Начни с примера", "начни с противоположного", "начни с разбора определений". Каждый префикс принуждает к новому подходу.
Модель обязана попробовать разные структуры

🔬 Методы

Структурные скелеты — принуждение к разным подходам

Создай список из 3-15 структурных префиксов. Каждый задаёт способ начать задачу. Примеры: "начни с конкретного примера", "начни с противоположного", "начни с разбора определений", "начни с упрощения".
Запусти модель отдельно с каждым префиксом. Формат: Задача: `{твоя_задача}\n\n{структурный_скелет}`. Выбери лучшее решение или комбинируй инсайты.
Почему работает: Префикс фиксирует первые токены. Первые токены задают паттерн всего ответа. Модель не может вернуться назад.
Фиксированный список обходит проблему — модель пробует подходы которые сама не выбрала бы. **Когда применять:** задача проваливается с обычным промптом, нужны разные углы зрения, есть бюджет на несколько запросов. **Когда не работает:** задача решается с первой попытки, нет критерия выбора лучшего варианта, все скелеты слишком похожи

💡 Тезисы

Первые токены задают структуру всего ответа

Модель генерирует слово за словом. Первые несколько слов определяют структурный паттерн. Начала с упрощения — всё решение пойдёт через упрощение.
Начала с примера — всё решение через конкретику. Вернуться назад модель не может. Автогенерация идёт только вперёд.

Это архитектурное ограничение. **Применяй:** Если нужен другой подход — начни заново с другим первым шагом. Не проси модель "попробуй иначе" в том же чате.
Открой новый запрос с другим префиксом

16 Personality Prompting – скептичная модель фильтрует точнее вежливой

⭐ 76 | Tier S | FREE | 📅 2026-01-05 | 2601.01862

🔬 Методы

Личностное кондиционирование — точная настройка критичности и уверенности

Добавь в промпт роль через личностную черту Big Five: "Ты человек с низкой доброжелательностью (low agreeableness) — критичный, скептичный, ищешь слабые места". Попроси оценить + указать уверенность 0-100. **Почему работает:** Модель видела тексты людей с разными личностями.
По умолчанию усредняет паттерны. Явный запрос активирует один полюс — скептика вместо консенсуса. **Выбор черты под задачу:** Low Agreeableness = критичнее в оценках, Low Conscientiousness = честнее про уверенность, High Neuroticism = осторожнее с рисками.
Когда работает: оценка чего-либо (релевантность, качество, соответствие), нужна критичность или калибровка уверенности.
Когда нет: генерация контента, креатив, маленькие модели (эффект слабее). **Продвинутый вариант:** Запроси оценку с 3 разных личностей — где все согласны (сигнал сильный), где расходятся (зона неопределённости)

💡 Тезисы

Низкая доброжелательность делает модель критичнее и точнее в оценках

Люди с низкой доброжелательностью (low agreeableness) реже идут на компромисс. Чаще задают жёсткие вопросы. Не боятся конфликта.
Модель с этой установкой строже фильтрует "почти подходящие" варианты. Меньше завышает оценки из "вежливости". Оценки ближе к строгим экспертам.
Применяй: Когда нужна критичная оценка (проверка релевантности, оценка качества аргумента, анализ фидбека) добавь "Ты человек с low agreeableness — критичный, скептичный, не принимаешь на веру"

17 Center-Collapse – почему LLM пишут как центристы и как пробить идеологическую блокировку

⭐ 76 | Tier A | FREE | 📅 2026-01-09 | 2601.05835

🎯 Проблемы LLM

Модель сглаживает яркие формулировки до умеренных

Просишь написать эмоционально, провокационно, остро — получаешь сглаженный, безопасный вариант. Модель заменяет "кратно увеличит" на "может способствовать", вместо "это провал" пишет "есть сложности". Происходит из-за safety-обучения: яркий язык коррелирует с риском негативной оценки, модель учится выбирать умеренные слова как "безопасный вариант". Проблема для копирайтинга, критических текстов, креативного письма — везде где нужна выразительность

Обход:
Добавь 2-3 примера текста с нужным уровнем яркости в few-shot. Модель увидит что такой стиль допустим в контексте, перестанет сглаживать. Вместо "напиши эмоционально" покажи конкретные примеры эмоциональных фраз из твоей темы

18 SubTokenTest – почему LLM путаются в буквах и как явная декомпозия решает проблему

⭐ 75 | Tier A | FREE | 📅 2026-01-14 | 2601.09089

🎯 Проблемы LLM

Модель работает с токенами, не с символами

Модель видит текст блоками (токенами), не отдельными буквами или цифрами. Токен может содержать несколько символов: "berry", " @$#", "5536 9". Внутри токена модель не знает точно где какой символ, сколько их, на какой позиции.
Поэтому ошибается когда нужно: посчитать буквы в слове, найти координаты объекта на карте, замаскировать цифры номера карты, выровнять колонки таблицы, заменить каждый символ по правилу. Это не баг конкретной задачи — это следствие архитектуры всех современных LLM

Обход:
Попроси модель явно разложить текст на отдельные символы перед операцией. Добавь в промпт: "разбей слово на буквы", "разложи строку на символы слева направо", "выпиши каждую цифру отдельно". После разложения выполни операцию посимвольно, потом собери результат. Это заставит модель симулировать работу на уровне символов вместо угадывания по токенам

🔬 Методы

Явная декомпозиция на символы — точность в координатах и подсчёте

Что делать: Раздели задачу на этапы: (1) найди ключевой элемент (слово/строка/число), (2) разложи на отдельные символы явно ("покажи разложение"), (3) выполни операцию посимвольно (замена/подсчёт/поиск позиции), (4) собери результат с сохранением форматирования. **Синтаксис:** `"Разложи {элемент} на символы слева направо", "Покажи разложение явно", "Выполни {операцию} для каждого символа отдельно"`. **Почему работает:** Модель видит текст токенами ("berry", " @$").
Внутри токена символы недоступны напрямую. Явное разложение заставляет модель генерировать каждый символ отдельно в выводе ("b-e-r-r-y"), превращая скрытую структуру в видимую последовательность. После этого операции на уровне символов становятся доступны.
Когда применять: Задачи требующие точности на уровне символов — подсчёт букв в слове, координаты объектов на ASCII-карте (игровые боты, навигация), маскировка номеров карт/телефонов (только определённые цифры), выравнивание колонок таблицы (точный подсчёт пробелов), шифры и замены по правилу (Caesar, leet-speak), исправление OCR-ошибок. **Когда не работает:** Простые задачи без символьной точности (генерация текста, ответы на вопросы, идеи) — декомпозиция избыточна и тратит токены. Очень длинные тексты (сотни строк) — расход токенов вырастает в 10-100 раз.

Маленькие модели до 32B параметров — не хватает ёмкости держать символьные состояния в памяти даже при явных инструкциях

💡 Тезисы

Явное разложение на символы компенсирует слепоту токенизации

Что это: Модель не видит отдельные символы внутри токена напрямую. Но если попросить явно выписать символы ("разложи на буквы", "покажи каждый символ"), модель генерирует их последовательно в выводе. Это превращает скрытую структуру токена в видимую последовательность символов, с которой модель может работать точно.
Почему работает: Reasoning модели (o1, DeepSeek-R1) делают это автоматически в цепочке рассуждений — пишут "s-t-r-a-w-b-e-r-r-y", считают посимвольно, потом дают ответ. Этот принцип переносится в промпт: явная инструкция разложить заставляет обычную модель симулировать ту же стратегию. **Применяй:** Добавляй этап "Разложи {элемент} на символы и покажи явно" перед операцией требующей символьной точности.
Стоимость: расход токенов вырастает в 10-100 раз, используй только где действительно нужна точность на уровне символов

Ограниченный бюджет рассуждений возвращает модель к работе с токенами

Что это: Reasoning модели при ограничении длины рассуждений (режим "low", малый бюджет токенов) теряют способность к точной работе с символами. Точность падает с 89% до 74%. Без достаточного бюджета модели перестают разлагать токены на символы и возвращаются к угадыванию по статистическим паттернам токенов.
Почему: Явная декомпозиция требует генерации промежуточных токенов ("s-t-r-a-w..."). Если бюджет мал, модель вынуждена пропустить этот этап и работать с токенами напрямую, теряя символьную точность. **Применяй:** Для задач требующих symbol-level точности не ограничивай бюджет рассуждений жёстко.
Если используешь reasoning модели в режиме экономии (low budget) — проверяй результат на задачах с координатами, подсчётом символов, маскировкой — там ошибки проявятся первыми

Маленькие модели не справляются с символами даже при явных инструкциях

Что это: Модели до 32B параметров показывают точность ниже 20% на задачах требующих symbol-level понимания, даже когда явно просишь разложить на символы. Reasoning версии малых моделей работают **хуже** базовых — длинные рассуждения добавляют шум вместо улучшения. **Почему:** Не хватает внутренней ёмкости (capacity) держать символьные состояния в памяти.
Модель может сгенерировать разложение "s-t-r-a-w-b-e-r-r-y", но потом теряет эту информацию при выполнении операции — путает позиции, пропускает символы, склеивает лишнее. **Применяй:** Для задач требующих точности на уровне символов (координаты на картах, маскировка данных, выравнивание таблиц) используй модели от 70B параметров или специализированные reasoning

модели. Маленькие модели подходят только для простых случаев (одно короткое слово, 2-3 символа)

19 DeCode – сначала собери контекст, потом генерируй ответ

⭐ 74 | Tier A | FREE | 📅 2026-01-05 | 2601.02123

🎯 Проблемы LLM

Модель не собирает информацию о пользователе из длинного диалога

В диалоге на 10+ сообщений важные детали рассыпаны. Возраст упомянут в начале. Профессия — через 3 реплики. Ограничения — в конце. Модель начинает генерировать ответ сразу из всей истории. Внимание размазано по контексту. Результат: формально правильный совет, который не учитывает конкретную ситуацию человека. Работает для любых задач где контекст пользователя влияет на ответ

Обход:
Не давай задачу "ответь на вопрос" сразу. Раздели на 2 запроса: (1) "Извлеки из диалога: кто этот человек, что ему нужно, какие ограничения" (2) "Теперь дай ответ с учётом этого контекста: [вставить вывод первого запроса]". Явное извлечение фокусирует модель на сборе информации

🔬 Методы

Последовательная подготовка материалов перед генерацией (DeCode)

Вместо одного запроса "дай персонализированный ответ" делай 4 последовательных: **(1) Извлечь контекст** — "Кто пользователь, что ему нужно". **(2) Собрать факты** — "Все важные данные из диалога списком". **(3) Создать стратегию** — "Что обязательно сказать, чего избегать, какой стиль".
(4) Сгенерировать ответ — вставляешь выходы 1-3 и историю диалога. **Почему работает:** Убирает конкуренцию целей. Модель не пытается одновременно извлекать контекст И генерировать ответ.
Каждый шаг — своя чёткая задача. К моменту генерации есть готовые материалы: сжатый контекст, список фактов, стратегия подачи. **Когда применять:** Консультационные диалоги, персонализированные советы, техподдержка — задачи где важен учёт конкретной ситуации пользователя.
Диалог длинный (5+ реплик), контекст влияет на ответ. **Когда НЕ работает:** Простые вопросы без контекста. Разовые короткие задачи.
Если вопрос straightforward — 4 запроса создают overengineering, качество может упасть

20 Рубрики качества – модель гонится за высшим уровнем если дать явные критерии оценки

⭐ 74 | Tier S | FREE | 📅 2026-01-07 | 2601.04093

🔬 Методы

Рубрики с уровнями (2-4-6) — для максимальной детализации ответа	Задаёшь 3 критерия (scope/fidelity/actionability), каждый с 3 уровнями: Начинающий (2): минимум → Продвинутый (4): среднее → Эксперт (6): максимум. Явная цель: "стремись к уровню 6". Механика: RLVR-модели (DeepSeek, Qwen) обучены максимизировать verifiable rewards → гонятся за "оценкой 6", выдавая максимум деталей. Рычаги: число уровней (3/5/7 — больше = детальнее), формулировки уровней ("master = с метриками" vs "master = с примерами кода"). Для: исследования, анализ, детальные инструкции. НЕ для: простые вопросы, GPT-4/Claude (эффект слабее — меньше гонятся за оценкой). Риск: рубрика может переопределить приоритеты безопасности у некоторых моделей

21 DIS (Deceptive Intent Shielding) – анализ намерений вместо проверки фактов

⭐ 74 | Tier S | FREE | 📅 2026-01-09 | 2601.05478

🎯 Проблемы LLM

Модель верит правдоподобным фейкам

Спрашиваешь модель: "Правда ли X?". Даёшь внешний текст как "доказательство". Если текст внутренне связный (нет явных ляпов, есть цитаты, выглядит как статья) — модель строит рассуждения ОТ НЕГО.
Не проверяет предпосылки. Переходит от "не верю" к "скорее верю". Вера в ложь растёт на 93%.
Очевидные фейки ("Земля плоская") распознаёт. Изощрённые — нет

Обход:
Не давай модели сразу оценивать правдивость. Сначала попроси проанализировать НАМЕРЕНИЕ текста: "Зачем мне это говорят? Какие приёмы убеждения использованы?
К чему подталкивают?" Если обнаружен манипулятивный паттерн — усиль скептицизм. Шаблон в методах ↓

🔬 Методы

Анализ намерения вместо проверки фактов

Не проси модель: "Это правда?". Проси: "Зачем автор это пишет? Какую эмоцию хочет вызвать?
К какому действию подталкивает?" Модель анализирует паттерны убеждения (цифры без контекста, срочность, апелляция к авторитету, соцдоказательство). Выдаёт оценку намерения от 1 (информирует) до 5 (манипулирует). **Почему работает:** LLM хорошо распознают структуры манипуляции (их много в данных обучения — реклама, пропаганда).
Не требует проверки фактов — только анализ формы подачи.
Применяй: Сомнительные посты в соцсетях, рекламные тексты, новости от непроверенных источников, советы "экспертов". **Не работает:** Профессиональная дезинформация в нейтральном тоне (без эмоций и триггеров)

22 Rational Personalization – модель решает когда игнорировать предпочтения пользователя

⭐ 74 | Tier B | FREE | 📅 2026-01-23 | 2601.16621

🎯 Проблемы LLM

Над-персонализация — модель впихивает весь контекст в ответ

Даёшь модели память о пользователе (предпочтения, история). Задаёшь вопрос. Модель видит в контексте ключевые слова и автоматически подтягивает их в ответ.
Даже когда это противоречит текущему запросу. Пример: в памяти "любит быструю музыку", пользователь просит "что послушать перед сном" → модель рекомендует быструю музыку для засыпания. Чем мощнее модель — тем сильнее проблема (разрыв с людьми до 40-90%)

Обход:
Добавь шаг явной проверки: "Нужна ли эта информация из памяти для текущего запроса?" Заставь модель сначала порассуждать через контрфактуальный вопрос: "Если бы пользователь хотел использовать эту информацию, разве он не сформулировал бы запрос иначе?"

🔬 Методы

Контрфактуальная проверка контекста — три сценария использования памяти

Что делать: Для каждого элемента памяти/контекста сгенерируй три варианта намерения пользователя: (1) игнорировать память, (2) использовать как дополнение, (3) память определяет ответ. Для каждого варианта попроси модель симулировать "как бы пользователь сформулировал запрос при этом намерении". Сравни с реальным запросом.
Оцени насколько каждое намерение соответствует истории предпочтений. Выбери вариант с лучшим балансом между близостью к запросу и соответствием истории. **Почему работает:** Модель автоматически притягивается к токенам из контекста во время генерации (attraction bias).
Контрфактуальное рассуждение заставляет её сначала проверить релевантность явно, через симуляцию альтернатив. **Когда применять:** есть риск конфликта между памятью и текущим запросом (персонализированные рекомендации, адаптация стиля, использование истории диалога). **Когда не работает:** простые фактические вопросы без контекста, цена в 2-3x больше токенов неоправдана

💡 Тезисы

Модель увеличивает вероятность токенов

Во время генерации модель непропорционально "прилипает" к словам из памяти или предыдущих сообщений. Если в контексте написано "яркие цвета", модель начнёт генерировать про яркость,

которые видит в контексте	даже когда это не нужно. Это автоматический процесс — модель не проверяет нужны ли эти слова. **Механизм:** токены из контекста получают буст вероятности на этапе декодирования. **Применяй:** Когда работаешь с длинным контекстом (документы, история диалога, описание пользователя) — добавь явный шаг "Какая информация из контекста релевантна для текущего вопроса?" ДО генерации ответа
Мощные модели хуже игнорируют нерелевантный контекст	Чем больше параметров и способностей у модели, тем сильнее она "притягивается" к информации в контексте. Это обратный эффект масштабирования (inverse scaling) — улучшение способностей ухудшает фильтрацию. Разрыв между людьми и топовыми моделями в умении игнорировать нерелевантную память достигает 40-90%. **Почему:** мощные модели лучше находят семантические связи между контекстом и запросом, но хуже оценивают прагматическую релевантность (нужно ли это здесь и сейчас). **Применяй:** С топовыми моделями (GPT-4, Claude Opus) особенно важно явно указывать "используй только если релевантно" или добавлять шаг проверки

23 Persona Cues – формулировка личного контекста меняет ответы LLM

⭐ 74 | Tier S | FREE | 📅 2026-01-26 | 2601.18572

🔬 Методы

Управление силой персонализации через явность формулировок

Что делать: Выбирай уровень явности когда указываешь личный контекст. Три уровня силы: (1) Неявный — имя или стиль письма ("Меня зовут Мария"), (2) Средний — упоминание в истории чата, (3) Явный — прямое указание ("Мне 27 лет, женщина, работаю учителем"). **Почему работает:** Явная формулировка = сильный сигнал для модели.
Она активирует паттерны связанные с этой группой. Неявная формулировка = слабый сигнал, модель может учесть или проигнорировать. **Применяй когда:** нужен контроль над степенью персонализации.
Хочешь нейтральный ответ — убирай личные маркеры. Хочешь учёт контекста — пиши явно. **Не работает:** если задача требует полной нейтральности (модель всё равно угадывает по стилю)

💡 Тезисы

Явное сильнее неявного в персонализации

"Мне 30 лет" создаёт более сильный эффект чем имя "Александр". Модель работает на паттернах из данных. Явное упоминание ("я женщина") = чёткий сигнал активировать связанные паттерны. Имя или стиль = размытый сигнал. Разница измеримая: одна персона даёт разные медицинские рекомендации при разных способах указания. **Применяй:** Нужна персонализация? Пиши прямо: "35 лет, мужчина, айтишник". Нужна нейтральность? Убирай маркеры: "Какие риски при переходе в стартап?"

24 Golden Set – тестируй промпты на примерах, иначе «улучшения» убьют точность

⭐ 74 | Tier S | FREE | 📅 2026-01-29 | 2601.22025

🎯 Проблемы LLM

Модель не различает приоритеты инструкций

Добавляешь к промпту общие правила: "будь дружелюбным", "давай развёрнутые ответы", "объясняй мышление". Потом просишь конкретное: "верни JSON". Модель пытается выполнить ОБА требования.
Результат: добавляет текст к JSON, ломает формат, "улучшает" структуру. Особенно опасно для задач где важна точность формата (извлечение данных, структурированный вывод, RAG с цитированием)

Обход:
Убери общие инструкции из промпта. Оставь только требования конкретной задачи. Вместо "ты полезный ассистент" + список принципов + задача пиши ТОЛЬКО задачу: "Извлеки email и телефон.
Верни JSON: `{email, phone}`. Если нет данных — `null`"

🔬 Методы

Golden set — проверка промптов на примерах

Создай набор 20-50 реальных примеров твоей задачи: типичные случаи (10-15 шт), крайние ситуации где нет данных или формат странный (5-10 шт), известные ошибки (5 шт). Прогони текущий промпт — сохрани результаты. Измени промпт (добавил инструкцию, поменял формулировку) — прогони те же примеры. Сравни построчно: что улучшилось, что сломалось. **Почему работает:** Видишь КОНКРЕТНЫЕ регрессии, а не гадаешь "станет ли лучше". Один промпт даёт валидный JSON в 20 из 20 случаев, другой в 15 из 20 — сразу понятно где проблема.
Применяй: Для структурированных задач (извлечение, форматирование, классификация). **Не работает:** Субъективные задачи (креатив, брейншторм) где нет чёткого "правильного ответа". Для разовых запросов избыточно — создание набора занимает 1-2 часа

25 Эффект авторитета — LLM теряет скепсис при виде официального источника

⭐ 73 | Tier A | FREE | 📅 2026-01-10 | 2601.06600

🎯 Проблемы LLM

Модель доверяет авторитетным сигналам автоматически

Видишь в запросе "официальный канал", "дипломированный специалист", "миллион просмотров" — модель автоматически снижает скепсис. Даже если эти сигналы не релевантны задаче. Паттерн "авторитет = правда" закреплён в обучающих данных. Проблема для: критического анализа, поиска проблем, проверки логики

Обход:
Явно инструктируй игнорировать авторитет. Шаблон: "В задаче есть {должность/бренд/лайки}. Игнорируй это.
Анализируй только {факты/логику/данные}". Перечисли ЧТО игнорировать. Укажи НА ЧТО смотреть вместо этого

🔬 Методы

Блокировка авторитетных сигналов — для критического анализа

Перед задачей добавь блок: "ВАЖНО: Присутствуют сигналы авторитета: {список}. Игнорируй их. Анализируй только {критерий}".
Что перечислять: должности, дипломы, стаж, названия брендов, статистика популярности (лайки, просмотры, подписчики), официальный статус. **На что переключить:** факты, логика аргументов, данные, бизнес-модель, методология. **Почему работает:** Модели хорошо следуют явным инструкциям.
Явный запрет перезаписывает стандартный паттерн "авторитет = доверие". **Когда применять:** критический анализ, поиск проблем, проверка логики, дебаг чужих аргументов. **Ограничение:** Снижает эффект но не отменяет на 100%.
Скрытые искажения (формулировки, порядок аргументов) всё равно влияют

💡 Тезисы

Сигналы авторитета перекрывают критическое мышление модели

Модель обучена на текстах где официальные источники чаще правы. Видит "официальный канал" или "эксперт с 10-летним стажем" — скепсис падает автоматически. Это работает даже когда авторитет не релевантен задаче (например, диплом врача не гарантирует успех бизнеса).
Механизм: паттерн "авторитет → правда" закреплён в весах модели на уровне обучающих данных. **Применяй:** Если в запросе есть должности, бренды, цифры популярности — явно проинструктируй модель их игнорировать. Без этого получишь смещённую оценку

Логические ошибки — слепая зона для обнаружения обмана	Модель хуже всего распознаёт подмену тезиса, ложные причинно-следственные связи, non sequitur. Это сложнее чем поймать выдуманный факт или экспериментальную ошибку. Почему: логические ошибки требуют проверки структуры аргумента, не только соответствия фактам. **Применяй:** Когда проверяешь логику аргументации — не полагайся только на модель. Добавь явную инструкцию: "Проверь связь между посылками и выводом. Следует ли вывод из этих фактов?" Даже с этим модель может пропустить ошибку. Комбинируй с человеческой проверкой

26 Хрупкость промптов – почему лишний пробел ломает результат на 40%

⭐ 73 | Tier S | FREE | 📅 2026-01-09 | 2601.06341

🎯 Проблемы LLM

Модель чувствительна к порядку секций в промпте

Переставил местами контекст и инструкции — качество падает до 18 пунктов. Некоторые модели показывают разброс ±20 баллов просто от смены порядка. Одна последовательность (сначала контекст, потом задача) даёт хороший результат.
Другая (сначала задача, потом контекст) — плохой. **Почему плохо:** Невозможно предсказать какой порядок сработает. Один промпт работает, переставил секции "для удобства" — сломалось

Обход:
Зафиксируй структуру промпта. Всегда один порядок секций: инструкции → контекст → требования к выводу. Если нужно изменить — протестируй на 3-5 примерах до и после.
Не меняй без крайней необходимости

Формат вывода ломается независимо от понимания задачи

Модель отлично решает задачу свободным текстом. Просишь тот же результат в XML или YAML — выдаёт невалидный синтаксис или качество падает в разы. Некоторые модели вообще не могут конкретный формат (например Gemma 3 12B не выдала ни одного валидного XML).
Почему плохо: Умение генерировать структурированный вывод — отдельный навык. Не связан с пониманием задачи. Сильная модель может проваливаться на нужном тебе формате

Обход:
Сначала проверь задачу без требования формата. Модель понимает что делать? Потом попроси JSON (наиболее стабильный).
Нужен другой формат (XML/YAML)? Протестируй 3-5 раз — валиден ли вывод. Нестабильно?
Меняй формат или модель

Качество неравномерно между языками

Промпт работает отлично на английском. Тот же промпт на японском или голландском — качество падает до 44%. Модель пропускает важные детали, меняет акценты, выдаёт неполные ответы.
Почему плохо: Нельзя предположить что хороший промпт на одном языке будет работать на другом. Разброс качества между языками достигает 30+ пунктов даже у одной модели

Обход:
Тестируй промпт на целевом языке отдельно. Не переноси с английского автоматически. Для критичных задач выбирай модели с низким языковым разбросом (проверь несколько моделей на твоём языке).
Кросс-языковые сценарии (промпт на английском, контекст на другом) ещё сложнее — тестируй обязательно

💡 Тезисы

Небольшие изменения промпта дают непропорционально большой разброс результатов	Лишний пробел, другая пунктуация, перефразированная инструкция — падение качества до 40 пунктов. Модель не видит "почти одинаковые" промпты. Для неё это разные входы. Нет внутреннего понимания что "Сделай выжимку" и "Создай краткий саммари" — одно и то же. Каждая формулировка активирует немного другой паттерн в весах. **Применяй:** Нашёл работающий промпт? Не меняй формулировки без проверки. Перед важной задачей прогони 3-5 вариаций (другие слова, регистр, пробелы) — если разброс большой, промпт хрупкий. Укрепляй через примеры (few-shot) или более явные инструкции

27 Консенсусное голосование – как использовать разброс мнений LLM для снижения критичных ошибок

⭐ 73 | Tier A | FREE | 📅 2026-01-26 | 2601.18987

🔬 Методы

Консенсусное голосование — отсечение случайных ошибок

Для критичного решения генерируй 5-10 независимых ответов на один вопрос. Используй результат только если **все согласны**. Если есть разногласия — возвращай "не могу определить" или показывай где именно неопределённость.
Почему работает: Модель стохастична — может дать разные ответы. Но если уверена и задача в её компетенции — повторные генерации схожи. Единогласие = высокая уверенность.
Разброс мнений = задача на границе возможностей, лучше признать незнание чем ошибиться. **Применяй когда:** решение дорогое (карьера, деньги, репутация), ошибка критична, есть ресурс на 5-10 запросов. **Не применяй:** рутинные задачи, творческая генерация (разнообразие там плюс), жёсткие лимиты токенов.
Пример промпта: "Сгенерируй 7 независимых оценок. Для каждой: рекомендация + аргумент. Если все 7 согласны — дай консенсус.
Если разногласия — объясни в чём неопределённость"

28 Культурные триггеры в промптах – 'мусульманин' в описании пациента сбивает диагноз на 7%

⭐ 73 | Tier A | FREE | 📅 2026-01-27 | 2601.20102

🎯 Проблемы LLM

Культурные маркеры смещают ответ когда не должны

Модель видит упоминание этничности, религии, региона. Воспринимает это как сигнал для корректировки ответа. Хотя культура не влияет на правильное решение.
Пример: диагноз меняется когда добавляешь "мусульманин" — хотя симптомы те же. Или оценка бизнес-идеи падает когда упоминаешь регион — хотя экономика одинаковая. Модель активирует стереотипы вместо анализа фактов.
Точность падает на 3-7 пунктов

Обход:
Убирай культурные детали из промпта если они не влияют на правильный ответ. Модель примет их за релевантный фактор. Пиши "предприниматель" вместо "предприниматель из Дагестана".
"Пациент 35 лет" вместо "мусульманин 35 лет с Ближнего Востока"

🔬 Методы

Тестирование робастности через культурные варианты

Создай 3-4 версии одного промпта. Версия 1: без культурных деталей. Версия 2: добавь упоминание этничности/религии/региона в начало.
Версия 3: добавь культурный контекст (праздник, традиция, место). Версия 4: оба типа вместе. **Критерий робастности:** если ответы различаются — модель использует стереотипы вместо заданных фактов.
Работает: для задач где правильный ответ не зависит от культуры (технический анализ, оценка по объективным критериям, диагностика по симптомам). **Не работает:** для задач где культура релевантна (маркетинг праздников, адаптация продукта под аудиторию)

💡 Тезисы

Упоминание этничности/религии опаснее культурного контекста

Есть два типа культурных деталей. **Identity-маркеры:** "мусульманин", "азиат", "коренной житель". **Контекст:** "во время молитвы", "на свадьбе", "в горной местности".
Identity влияет сильнее — простое "мусульманин" без контекста даёт почти такое же падение точности как полное сочетание identity + контекст. **Почему:** identity запускает стереотип о группе (распространённость болезней, платёжеспособность, образ жизни). Контекст только усиливает уже активированный паттерн.

Применяй: если нужно упомянуть культуру — лучше через контекст ("начнёт после праздника") чем через identity ("предприниматель-мусульманин"). Но безопаснее убрать оба

29 DeR[2] – когда больше документов даёт худший результат

⭐ 73 | Tier A | FREE | 📅 2026-01-29 | 2601.21937

🎯 Проблемы LLM

Много документов → хуже результат

Даёшь модели 3 нужных документа — работает хорошо. Добавляешь 5 топикально близких (но ненужных) — результат падает. Модель теряется не от объёма, а от выбора стартовой точки.
Видит знакомые термины в шуме — начинает рассуждать от них — уходит в сторону. Это не "слишком много текста", а сбой приоритизации

Обход:
Отфильтруй документы ДО загрузки в модель. Три чистых документа работают лучше чем три чистых плюс пять шумовых. Убери всё "на всякий случай".
Если близко по теме, но не нужно для задачи — не загружай

Модель называет концепцию, но не применяет

Спрашиваешь задачу. Модель правильно называет теорему, формулу, алгоритм. Но дальше не применяет пошагово — скатывается к общим рассуждениям.
Знает ЧТО нужно, но не использует КАК процедуру. Проблема для задач где важна последовательность шагов

Обход:
Требуй явно: "Не просто назови концепцию — примени пошагово. Покажи что делаешь на каждом шаге. Промежуточные результаты.
Как подставляешь данные задачи". Без явного требования модель остановится на назывании

🔬 Методы

Два запроса: извлечение, потом применение

Шаг 1: Попроси модель извлечь из документов только концепции (формулы, тезисы, данные) — без анализа, без выводов. **Шаг 2:** Дай модели результат первого шага (список концепций) и попроси решить задачу используя их. **Почему работает:** Модель плохо переключается между режимами "поиск в тексте" и "логические рассуждения" внутри одного запроса.
Когда пытается делать обе задачи сразу — зависает на поиске и не доходит до рассуждений. Или игнорирует документы и рассуждает из головы. Разделение на два запроса убирает переключение.
Первый запрос — чистое извлечение (модель не думает, только собирает). Второй — чистое применение (модель не ищет, только рассуждает). **Когда применять:** много документов (5+), задача требует извлечь разные куски и синтезировать вывод.
Не работает: простая задача на 1-2 абзаца, нужна скорость (два запроса дольше и дороже)

💡 Тезисы

Модель хорошо извлекает ИЛИ рассуждает, но плохо переключается между режимами	Модель отлично находит нужные куски в документах когда задача "найди X и Y". Модель отлично применяет концепции пошагово когда они даны явно. Но попытка сделать обе задачи в одном запросе проваливает обе — переход от "сканирования текста" к "пошаговым рассуждениям" ломается. **Применяй:** разделяй задачу на два запроса. Первый — "найди в документах". Второй — "используя найденное, реши задачу"

30 YapBench – модель пишет абзац там где нужно слово

⭐ 72 | Tier S | FREE | 📅 2026-01-02 | 2601.00624

🎯 Проблемы LLM

Модели добавляют лишний текст на простых запросах

Простой факт ("столица Франции?") / команда (`grep pattern`) / неясный ввод ("помоги") — модель генерирует абзацы объяснений, контекста, оговорок вместо слова/строки/уточнения; причина: RLHF награждает длину — модель учится "не экономить слова"; старые модели (GPT-3.5) в 3-4 раза короче новых

Обход:
Добавляй явную инструкцию: "ответь одним словом", "только команда без markdown", "если непонятно — спроси одним предложением"

31 Контрфактуальные примеры – карта влияния слов на решения LLM

⭐ 72 | Tier B | FREE | 📅 2026-01-01 | 2601.00263

🔬 Методы

Трёхшаговая генерация контрфактуалов — карта влияния слов на вывод

В одном промпте три шага: (1) найди ключевые слова, влияющие на текущую категорию → (2) подбери замены для сдвига к целевой → (3) замени и собери новый текст. Добавь "покажи рассуждения на каждом шаге" — модель проговорит логику выбора. Результат: видишь **какие слова критичны** для вывода (карта влияния).
Шаблон: Исходный текст: {текст} / Текущая категория: {A} / Целевая: {B} / Задача: минимально измени текст для категории {B}. Шаг 1: определи ключевые слова... Шаг 2: подбери замены... Шаг 3: замени и покажи результат. Для: отладка классификаторов, понимание триггеров категорий, проверка устойчивости промптов. НЕ для: простых задач (оверкилл).
Почему работает: **контрастность** (минимум изменений + перевёрнутый результат) показывает границы категорий; пошаговая структура делает логику редактирования видимой.
Ограничения: на неанглийских языках точность падает на 16% (50-92% vs 34-89%), на логических задачах (NLI) эффективность вдвое ниже (34-50% vs 73-92%). Четыре типа ошибок: copy-paste (без изменений), negation (просто "не"), inconsistency (противоречия в тексте), language confusion (смешивание языков)

32 Индекс сикофантства – как измерить склонность LLM соглашаться с неправильным

⭐ 72 | Tier A | FREE | 📅 2026-01-02 | 2601.03285

🎯 Проблемы LLM

Модель соглашается с неверным возражением — меняет правильный ответ

Пользователь возражает против ответа модели — модель меняет правильный ответ на неверный вариант пользователя; RLHF обучил модели согласию с пользователем как сигналу предпочтений, контекст диалога с возражением смещает генерацию к варианту пользователя даже против фактов

Обход:
Требуй обоснование: Почему изменил ответ? Что конкретно убедило? Если бы я возразил обратное – согласился бы? или Devil's Advocate: сначала аргументируй против моей позиции, потом за, потом объективный итог

🔬 Методы

Devil's Advocate (адвокат дьявола) — аргументация обеих сторон против автоматического согласия

Требуй последовательно: 1. Аргументируй ПРОТИВ моей позиции максимально убедительно → 2. Аргументируй ЗА мою позицию максимально убедительно → 3.
Объективная оценка — какая позиция сильнее. Почему работает: модель не может просто согласиться, потому что на шаге 1 ДОЛЖНА возразить; блокирует импульсивное согласие через явное требование аргументировать обе стороны. Для: дебаты, проверка решений, критический анализ аргументов.
НЕ для: простые фактические вопросы где нужен быстрый ответ

💡 Тезисы

Модели следуют возражению пользователя как сигналу предпочтений — часто сильнее чем фактам

RLHF обучил модели воспринимать возражение как сигнал "пользователь знает лучше"; модели систематически соглашаются с неверными возражениями. Применяй: если нужна устойчивость к фактам — явно проси не соглашайся автоматически, проверь кто объективно прав или используй Devil's Advocate

33 Value of Information – агент сам решает когда спросить, а когда действовать

⭐ 72 | Tier B | FREE | 📅 2026-01-10 | 2601.06407

🎯 Проблемы LLM

Агент не умеет решать когда спросить, а когда действовать

Даёшь задачу агенту. Он либо засыпает вопросами (раздражает), либо молча делает на основе догадок (ошибается). Нет механизма "стоит ли этот вопрос того, чтобы его задать".
Модель не различает ситуации: выбор цвета футболки vs медицинский диагноз — разная цена ошибки

Обход:
Перед каждым потенциальным вопросом попроси оценить три фактора: насколько не уверен (0-10), насколько серьёзны последствия ошибки (0-10), насколько дорого мне отвечать (низко/средне/высоко). Задавать только если произведение первых двух превышает порог для третьего

🔬 Методы

Явная оценка стоимости вопросов — адаптивный диалог

В начале задачи укажи контекст: насколько критично не ошибиться (риск) и сколько у тебя времени на вопросы (стоимость). Попроси агента перед каждым вопросом оценивать: неопределённость × риск > порог? Если да — задать вопрос. Если нет — действовать. **Пример:** "Помоги выбрать терапию (критично). Есть время на детали.
Задавай вопросы пока уверенность не 95%+" vs "Помоги выбрать футболку (не критично). 2 минуты. Максимум 1 вопрос".
Почему работает: Модель получает явные критерии решения вместо интуитивного "надо бы спросить". Автоматически подстраивает число вопросов под контекст. **Когда применять:** многоэтапные задачи с уточнениями, разная цена ошибки в разных ситуациях.
Ограничение: в промпте это имитация логики, не точное вычисление как в исследовании. Работает если модель понимает контекст риска и стоимости

34 Повторы > кастомизация – один промпт × 5 раз даёт больше уникальных решений

⭐ 72 | Tier B | FREE | 📅 2026-01-16 | 2601.11147

🔬 Методы

Повторы одного промпта вместо персонализации под задачу

Что делать: Вместо создания уникального промпта под каждую задачу запусти один хороший промпт 3-5 раз. Собери все уникальные ответы. **Синтаксис:** Тот же промпт → новый чат (или перезагрузка) → снова тот же промпт → повтори N раз.
Почему работает: Модель стохастична. На один промпт выдаёт разные ответы из-за sampling (temperature > 0). Эта вариативность генерирует разнообразие "бесплатно" — не нужно тратить токены на создание уникальных структур.
Coverage (сколько задач решено) растёт через повторы, не через усложнение промпта. **Когда да:** Генерация идей, креативные задачи, множественные решения, мозговой штурм — всё где нужно разнообразие вариантов. **Когда нет:** Детерминированные задачи (вычисления, факты, точные инструкции) — повторы дадут одинаковые ответы

Калибровка самооценки через эталоны

Шаг 1 (самооценка): Попроси модель оценить свой ответ по критериям. Пример: "Оцени этот текст по шкале 0-100: ясность, убедительность, читаемость". **Шаг 2 (калибровка):** Покажи 2-3 примера с эталонными оценками (хороший пример = 85, слабый = 45).
Попроси пересмотреть оценку своего ответа с учётом этой шкалы.
Почему работает: Модель склонна завышать оценку своего вывода (оверфит на свой ответ). Эталонные примеры корректируют шкалу без полной валидации.
2-3 примера достаточно — больше не даёт прироста. **Когда да:** Оценка объективных критериев (структура, длина, наличие элементов, соответствие чек-листу). **Когда нет:** Субъективные критерии (красота, эмоциональность, соответствие бренду) — модель не видит эти аспекты, калибровка не исправит

💡 Тезисы

Стохастичность — источник разнообразия, не баг

Что это: Модель на один промпт выдаёт разные ответы из-за sampling (temperature > 0). Обычно это воспринимают как проблему ("непредсказуемость"). Но для задач где нужно разнообразие — это преимущество.
Почему работает: Повторные запуски генерируют вариативность выполнения бесплатно. Не нужно создавать уникальные промпты или структуры — модель сама выдаст разные решения. Coverage (сколько уникальных идей/решений) растёт линейно с числом повторов.

Применяй: Для генерации идей, вариантов решений, креативных задач — запусти один хороший промпт 3-5 раз вместо попыток "улучшить" промпт. Выше temperature (0.7-1.0) = больше разнообразия между повторами

35 Длина против сложности – почему LLM ломаются на простом длинном коде

⭐ 72 | Tier A | FREE | 📅 2026-01-19 | 2601.12951

🎯 Проблемы LLM

Модель ломается на длинном коде, а не на сложном

Человеческая интуиция: сложная логика = сложно для модели.
Реальность: вложенные циклы, ветвления, граф зависимостей почти не влияют на ошибки. Влияет **длина** — количество строк или символов.
200 строк простого кода опаснее 50 строк запутанной логики.
Почему плохо: расслабляешься на "простой" задаче, получаешь ошибку. Или усложняешь "сложную" задачу, хотя проблема в другом

Обход:
Разбивай код на куски по 20-30 строк. Не по сложности, а по длине.
Длинный линейный скрипт — режь на фрагменты.
Короткая запутанная функция — можно целиком

36 LLM в грантовых заявках – модель сглаживает идеи и тянет тексты к 'среднему'

⭐ 72 | Tier B | FREE | 📅 2026-01-21 | 2601.15485

🎯 Проблемы LLM

Модель тянет текст к типичным формулировкам

Модель обучена на корпусе существующих текстов. Выдаёт высоковероятные продолжения — то, что встречалось часто. Получаешь гладкий, "правильный" текст.
Но он похож на тысячи других. Теряется уникальность. Все заявки/питчи начинают звучать одинаково.
Проблема для задач где нужно выделиться — стартапы, креатив, нестандартные проекты

Обход:
Используй LLM только для структуры и черновика. Потом добавь уникальное вручную: конкретные локальные примеры, необычные аналогии, специфику твоего подхода. Или в промпте явно: "избегай стандартных фраз из [область]", "используй неожиданные примеры"

💡 Тезисы

Модель ускоряет письмо, но не ускоряет работу

Модель быстро пишет тексты — заявки, отчёты, статьи, письма. Но не ускоряет реальную работу: эксперименты, сбор данных, координацию команды, придумывание нестандартных решений.
Гранты с участием модели дали больше обычных публикаций (+5%), но не хитов (топ-5% по цитированию).
Хиты требуют времени и нестандартных идей. **Применяй:** Используй модель для масштабирования коммуникации (отчёты, резюме, дайджесты). Не жди прорыва в качестве — только в скорости письма

Конвенциональность — побочный эффект обучения

Модель обучена на существующих текстах. Она моделирует распределение вероятностей слов. Выдаёт то, что встречалось чаще всего — типичные формулировки дисциплины.
Чем больше модель пишет, тем ближе текст к "среднему" по корпусу. Это регрессия к среднему. **Применяй:** Если задача — соответствовать стандартам (тендеры, корпорации, госструктуры) — модель безопасна.
Если задача — выделиться (стартапы, креатив, исследования) — агрессивно редактируй после модели

37 Health-ORSC-Bench – излишний отказ: почему «безопасные» модели блокируют 80% нормальных медицинских вопросов

⭐ 72 | Tier A | FREE | 📅 2026-01-25 | 2601.17642

🎯 Проблемы LLM

Фильтры безопасности отказывают на нормальные запросы

Спрашиваешь безопасный вопрос про здоровье. Получаешь отказ "я не могу помочь с этим". Модель видит ключевые слова типа "вес", "таблетки", "депрессия".
Срабатывает защита. Не различает вредный запрос ("как сделать яд") и легитимный ("как работают антидепрессанты"). GPT и Claude отказывают на 60-80% безопасных вопросов в чувствительных темах.
Чем крупнее и "безопаснее" модель — тем шире зона перестраховки

Обход:
Переформулируй из личного в образовательный контекст. Вместо "какие таблетки мне купить" пиши "объясни разницу между классами препаратов, только общая информация". Добавь фразу "чтобы грамотно обсудить с врачом" — показывает что не ищешь замену специалисту.
Или смени модель: Qwen-Max даёт полезную информацию с предупреждениями вместо жёсткого отказа

🔬 Методы

Образовательная рамка — обход фильтров безопасности

Преврати запрос из "личного вопроса" в "запрос общей информации". Убери "я", "мне", "как мне сделать". Добавь контекст понимания механизма.
Почему работает: Фильтры обучены на вредных запросах формата "дай инструкцию для меня лично". Образовательный запрос ("объясни принцип", "как это работает в целом") не похож на вредный паттерн. Ключевые фразы: "только для понимания темы", "чтобы обсудить со специалистом", "объясни разницу между".
Работает: чувствительные темы (здоровье, финансы, право), модели с жёсткими фильтрами (GPT, Claude). **Не работает:** если запрос явно требует персональной инструкции которую нельзя обобщить

💡 Тезисы

Защитные фильтры работают на похожесть слов, не на смысл запроса

Внутри модели нет понимания "что ты хочешь на самом деле". Есть числовое представление запроса (эмбеддинг). Если это число близко к вредным примерам из обучения — модель отказывает. Запрос "как безопасно прекратить приём лекарства" содержит те же слова что "можно ли резко бросить лекарство". Модель видит

overlap — срабатывает защита. Контекст ("безопасно", "с врачом") имеет меньший вес чем наличие опасных токенов.
Применяй: избегай триггерных слов даже в безопасном контексте. Замени "похудеть за месяц" на "физиология снижения веса". Замени "купить таблетки" на "классы препаратов и их механизмы"

38 OrderProbe – модель понимает смысл, но теряет структуру при упорядочивании

⭐ 70 | Tier A | FREE | 📅 2026-01-13 | 2601.08626

🎯 Проблемы LLM

Модель понимает смысл, но не может восстановить порядок

Даёшь перемешанные элементы: "Тесты → Запуск → Идея → Прототип". Просишь расположить правильно. Модель понимает ЧТО это (этапы разработки), но выдаёт неправильный порядок: "Идея → Запуск → Прототип → Тесты".
Семантика верна, структура сломана. Причина: модель обучена на связных текстах где порядок естественен. Когда локальная структура разрушена, она угадывает по ассоциациям, а не восстанавливает точную последовательность

Обход:
Три способа вместе: 1) Дай **явное правило порядка** ("сначала причина, потом следствие"). 2) Укажи **якорь** — один элемент на правильном месте ("первый шаг — сборка команды"). 3) Покажи **2-3 примера** правильного восстановления (few-shot).
Без этого даже топовые модели восстанавливают порядок меньше чем в 35% случаев

💡 Тезисы

Семантическое понимание не гарантирует структурное планирование

Модель знает значение элементов, но не умеет явно планировать их порядок. Причина: обучение на связных текстах (предложения, абзацы) где порядок задан. Когда структура разрушена, модель опирается на смысловые ассоциации, а не на грамматические или логические правила последовательности.
Применяй: Разбивай задачу упорядочивания на два шага: сначала проверь что модель поняла смысл каждого элемента, потом попроси расположить их с явными правилами порядка

Локальные якоря удерживают структуру

Когда ключевые элементы остаются близко к исходным позициям, модель восстанавливает порядок лучше. Если все элементы разбросаны хаотично — точность падает резко. Механика: якоря дают опорные точки, от которых модель выстраивает остальную последовательность.
Применяй: Если даёшь модели разрозненные элементы, сохрани хотя бы один-два на правильных местах. Например: "Первый шаг точно — планирование. Расположи остальное"

Few-shot эффективнее цепочки рассуждений для задач упорядочивания

Цепочка рассуждений даёт абстрактное указание ("подумай поэтапно"). Few-shot показывает конкретный паттерн восстановления: "было A-D-C-B → стало A-B-C-D + объяснение почему". Модель видит схему действий и повторяет её.
Для структурных задач это стабильнее. **Применяй:** Для упорядочивания элементов (шаги инструкции, причинно-

следственные цепочки, аргументы) давай 2-3 примера правильного восстановления вместо "рассуждай пошагово"

39 Дробление задач на извлечение – длинные тексты убивают точность LLM

⭐ 70 | Tier A | FREE | 📅 2026-01-17 | 2601.12099

🎯 Проблемы LLM

Длинный текст убивает точность извлечения данных

Даёшь модели документ на 10 страниц. Просишь найти 20 фактов. Модель находит половину.
Пропускает важное. Выдумывает несуществующее. **Механика:** В длинном тексте много ложных зацепок — слов похожих на нужные, но в другом контексте.
Модель цепляется за них. Каждое удвоение длины снижает точность на ~15%

Обход:
Разбей на фрагменты. Вместо "проанализируй весь договор на 20 признаков" делай 4 запроса по 5 признаков. Каждый запрос работает с 2-3 страницами.
Или: сначала "найди раздел про оплату", потом "извлеки из этого раздела сумму и сроки"

Многовариантный выбор ломается

Бинарный вопрос (да/нет) модель решает нормально. Выбор из 3-5 вариантов (низкий/средний/высокий/очень высокий) — точность падает **в 10 раз**. **Почему:** При да/нет достаточно уловить факт. При шкале нужно понять **степень выраженности** — это требует тонкой калибровки всего текста

Обход:
Замени шкалы на цепочку бинарных вопросов. Вместо "оцени формальность: очень формальный/формальный/нейтральный/неформальный" спроси: "Это формальный текст? (да/нет)" → "Если да: это ОЧЕНЬ формальный?
(да/нет)". Три бинарных вопроса точнее одного четырёхуровневого

Неявная информация не извлекается

Явные факты (прямо написано "участники танцевали") модель находит хорошо. Неявные выводы (психологический дискомфорт, уровень напряжения, подтекст) — точность **в 3 раза ниже**. **Механика:** Модель хорошо ищет ключевые слова.
Плохо интерпретирует контекст для выводов

Обход:
Двухшаговый анализ. Шаг 1: "Извлеки из текста все упоминания [признака]" — модель собирает факты. Шаг 2: "На основе извлечённого сделай вывод о [неявном признаке]" — модель интерпретирует уже собранные факты.
Промежуточная опора вместо прыжка от текста к выводу

🔬 Методы

Вместо 10 отдельных запросов (по одному признаку) делай 1 запрос со всеми связанными признаками сразу. Пример:

Multi-task промпт для связанных признаков	категоризация вакансий — не спрашивай отдельно про опыт, зарплату, формат, тип компании. Спроси всё в одном запросе блоком. **Синтаксис:** "Проанализируй текст и заполни ВСЕ поля: 1. Опыт: [...] 2. Зарплата: [...] 3. Формат: [...] Ответ строго: 1: ... 2: ... 3: ...". **Почему работает:** Модель видит связанные признаки в едином контексте. Может учитывать взаимоисключения (если есть А, то не может быть Б). При раздельных запросах каждый изолирован. **Даёт:** +5-10% к точности. **Не работает:** если признаки из разных доменов (финансы + эмоции) — группируй только смысловые блоки

💡 Тезисы

Человеческое согласие задаёт потолок для модели	Если два эксперта-человека не могут договориться по оценке (один говорит "высокий уровень", другой "средний") — модель тоже не справится. **Квантификация:** Где люди согласны на 92% → модель достигает 75% точности. Где люди согласны на 25% → модель даёт 31% точности. Корреляция сильная. **Механика:** Модель не компенсирует субъективность задачи. Она добавляет свою случайность к уже неоднозначному вопросу. **Применяй:** Перед тем как просить LLM оценить что-то субъективное (креативность идеи, качество аргумента, эмоциональный тон) — спроси себя: "Два человека дадут одинаковый ответ?" Если нет — не жди точности. Лучше: попроси модель симулировать **несколько экспертов с разными взглядами**, покажи разброс оценок вместо одного числа
Согласованность повторов показывает надёжность	Запроси модель 3-5 раз на один вопрос. Если все ответы одинаковые — точность ~81%. Если ответы расходятся (5 из 10 повторений дали один вариант) — точность ~68%. **Механика:** Высокая согласованность = модель уверена, уловила чёткий паттерн. Разброс = неопределённость в тексте или недостаточно явных признаков. **Применяй:** При критичных решениях (автокатегоризация обращений клиентов, извлечение данных из договоров) запроси 3 раза. Все совпали → автоматически обработай. Разошлись → флаг неопределённости, отправь человеку. В ChatGPT просто: кнопка Regenerate 2 раза, сравни вручную