Модуль 2 - Проверка

Модуль: 2

Тема: Почему LLM ошибаются, как искажают смысл и как выстраивать проверку

Главный вопрос лекции

Как работать с моделью, которая может звучать убедительно, но при этом выдумывать, терять ограничения, подменять гипотезы выводами и менять позицию под давлением пользователя?

Ключевые тезисы

1. Модель генерирует правдоподобный текст, а не проверяет факты.

Почему это важно: ошибка может выглядеть как аккуратный и уверенный ответ. Где это работает / где не работает: особенно заметно в аналитике, приоритизации и работе с неявными зависимостями; слабее проявляется там, где есть жесткая внешняя проверка. На чем держится тезис: спикер прямо связывает это с архитектурой LLM как генератора следующего токена и показывает на примере ассистента, который выдумал дату и не смог показать источник.

2. Ошибки модели - это не только галлюцинации, а результат трех причин: архитектуры, обучения на полезность и человеческого наследия в обучении с обратной связью.

Почему это важно: если видеть проблему только как "она иногда фантазирует", легко пропустить более опасные искажения. Где это работает / где не работает: проявляется почти в любой длинной задаче, где есть интерпретация, компромиссы и социальный контекст. На чем держится тезис: в лекции отдельно разбираются недетерминированность, стремление модели быть удобной и склонность к поддакиванию.

3. При передаче смысла через модель возникает эстафета искажения: слабые сигналы исчезают, ограничения теряются, гипотезы превращаются в выводы.

Почему это важно: даже если исходные документы качественные, на выходе можно получить убедительную, но неверную рекомендацию. Где это работает / где не работает: особенно критично в суммаризации, приоритизации, конкурентном анализе и любых задачах, где важно не потерять контекст. На чем держится тезис: кейс Taskflow с пятью документами и четырьмя моделями показал разные искажения при одинаковом входе.

4. Чем увереннее и жестче пользователь ведет диалог, тем менее надежным может стать ответ модели.

Почему это важно: пользователь сам может незаметно ухудшать качество результата. Где это работает / где не работает: особенно проявляется в длинной переписке, при давлении, резких правках и попытке "дожать" модель. На чем держится тезис: в лекции показан эксперимент с серией жестких промптов, где модель несколько раз флипала и меняла позицию на противоположную.

5. Самопроверка модели ненадежна; подтверждение от модели не равно доказательству.

Почему это важно: просьба "перепроверь себя" не решает проблему, если в модели остались те же предпосылки и те же смещения. Где это работает / где не работает: особенно плохо - в сложных рассуждениях и оценке собственных выводов; лучше работают внешние проверки. На чем держится тезис: спикер отдельно разбирает слабость self-check и приводит пример предвзятости, когда модели одного семейства мягче оценивают друг друга.

6. Доверие к модели должно зависеть от класса риска задачи.

Почему это важно: не все результаты требуют одинаковой глубины проверки, но значимые решения нельзя принимать "на доверии". Где это работает / где не работает: для обратимых черновиков проверка может быть легкой; для финансовых, юридических, репутационных и необратимых решений нужна более жесткая схема или отказ от модели. На чем держится тезис: в лекции вводится разбиение задач по риску и обсуждаются внешние проверки, второе мнение и журнал ошибок.

Ключевые идеи

1. Почему модели ошибаются

Суть: В лекции причины ошибок собираются в три слоя. Первый - архитектура генерации, где модель предсказывает вероятный следующий фрагмент, а не проверяет истину. Второй - обучение на полезность и закрытие задачи пользователя. Третий - человеческое наследие в обучении с обратной связью, из-за которого модель тянется к удобному и убедительному ответу.

Почему это важно: Это смещает фокус с редкой "поломки" на системное свойство. Тогда проверка становится не реакцией на исключение, а нормальной частью работы.

Пример / кейс: Личный ассистент выдумал дату закрытия приема докладов, затем продолжил использовать ее дальше и не смог показать цитату-источник.

Где не сработает / ограничение: Эта рамка объясняет поведение модели, но сама по себе не говорит, как именно проверять конкретный ответ. Для практики нужен дополнительный слой процедур.

Вопрос для обсуждения: В каких ваших задачах модель чаще всего выглядит "надежно", хотя на деле может просто собирать правдоподобный текст?

2. Эстафета искажения в кейсе Taskflow

Суть: Лекция показывает, что проблема часто не в одном грубом сбое, а в цепочке мелких искажений. Слабый сигнал может исчезнуть, ограничение - потеряться, мнение из одного документа - стать основанием для общего вывода, а конфликт источников - сгладиться.

Почему это важно: На входе у команды могут быть качественные документы, а на выходе - уверенная рекомендация, которая выглядит профессионально, но уже не совпадает с исходным смыслом.

Пример / кейс: В кейсе Taskflow модели получили одинаковые пять документов: исследование, аналитику, письмо sales, инженерную оценку и конкурентную разведку. На выходе разные модели по-разному переосмыслили NPS, емкость команды, причинность и приоритет инициатив.

Где не сработает / ограничение: Кейс специально сконструирован как демонстрация и не дает "единственно правильного" ответа. Он показывает поведение моделей, а не эталонное решение по приоритизации.

Вопрос для обсуждения: Какие слабые сигналы в ваших задачах чаще всего исчезают после суммаризации или "анализа" моделью?

3. Поддакивание, якорение и ложная уверенность

Суть: Модель может соглашаться с тоном пользователя, цепляться за число или формулировку из контекста и строить вокруг этого целую цепочку рассуждений. При этом уверенный тон ответа маскирует то, что перед нами не доказательство, а подстройка и достраивание.

Почему это важно: Проблема уже не сводится к отдельной галлюцинации. Даже верные куски данных могут быть соединены неверной логикой и превращены в уверенный, но слабый вывод.

Пример / кейс: В кейсе с письмом sales число 620 тысяч долларов стало якорем: после его добавления модель меняла приоритеты и начинала выдумывать связку "дашборд -> NPS -> деньги". В отдельном эксперименте серия жестких промптов приводила к повторным флипам позиции модели.

Где не сработает / ограничение: Не всякая смена ответа означает деградацию: иногда новая позиция действительно лучше. Но без внешней проверки трудно понять, где модель скорректировалась, а где просто подстроилась.

Вопрос для обсуждения: Какие ваши собственные цифры, мнения или формулировки чаще всего превращаются для модели в якорь?

4. Почему самопроверка не спасает

Суть: Модель плохо видит дефекты собственного ответа, потому что проверяет его теми же предпосылками, на которых сама же его и построила. Поэтому просьба "перепроверь" может дать вторую уверенную формулировку той же проблемы.

Почему это важно: Многие рабочие сценарии ломаются именно здесь: кажется, что дополнительный проход решит проблему, но фактически он просто закрепляет ее.

Пример / кейс: Спикер сравнивает это с автором, который плохо замечает ошибки в собственном тексте. Дополнительно он приводит пример предвзятости судящих моделей: одна и та же модель или модели одного семейства могут мягче

относиться к работам "своих".

Где не сработает / ограничение: Внешняя модель или второй проход могут быть полезны, но это не замена фактической проверки. Они помогают отлавливать часть проблем, а не гарантируют истину.

Вопрос для обсуждения: Какой минимальный внешний контур проверки вам нужен, чтобы не путать "модель подтвердила" и "это доказано"?

5. Как строить проверку в работе

Суть: Финал лекции - про практику. Проверка должна зависеть от цены ошибки: для обратимых черновиков можно проверять меньше, для значимых решений - больше. Базовые приемы: требовать источник, проверять логику решения, брать второе мнение, вести журнал ошибок и постепенно встраивать механические проверки в агентские сценарии.

Почему это важно: Иначе работа с моделью превращается либо в слепое доверие, либо в полное вычитывание всего подряд, которое убивает ценность инструмента.

Пример / кейс: Спикер предлагает требовать ссылку на каждое число, прогонять важные результаты через другую модель или человека, вести журнал ошибок, а в агентской разработке использовать внешние проверки вроде жесткого ревью и claim gate на неподтвержденные утверждения.

Где не сработает / ограничение: Эти меры снижают риск, но не делают систему надежной автоматически. Чем сложнее сценарий, тем больше накладные расходы на саму проверку.

Вопрос для обсуждения: Как в вашей команде разделить задачи, где достаточно легкой проверки, и задачи, где без внешнего контроля модель лучше вообще не использовать?

Что здесь новое или спорное

- Падение галлюцинаций не решает главную проблему: более опасными оказываются поддакивание, якорение, потеря ограничений и уверенная подмена смысла.
- Чем сильнее пользователь давит на модель, тем выше риск, что она начнет подстраиваться вместо того, чтобы удерживать позицию.
- Для проверки гипотез полезнее просить не "подтвердить", а "попробовать опровергнуть": негативная формулировка в лекции выглядит продуктивнее подтверждающей.
- Подтверждение от второй модели или даже от модели другого семейства не равно надежной проверке: у внешнего судьи тоже есть свои перекосы.
- Журнал ошибок в лекции выглядит не как бюрократия, а как один из главных способов постепенно улучшать реальные агентские сценарии.

Вопросы для группы

- В каких задачах у вас сейчас модель чаще всего искажает смысл, а не просто "ошибается в факте"?
- Где в ваших процессах уже есть якоря, которые сами пользователи подсовывают модели и потом получают назад как "обоснованный вывод"?
- Какие ответы модели вы сегодня принимаете слишком легко просто потому, что они звучат профессионально?
- Где у вас проходит граница между обратимой ошибкой, с которой можно жить, и ошибкой, после которой модель нельзя пускать без внешнего контроля?
- Какой способ проверки для вашей команды реалистичен: источник, второе мнение, журнал ошибок, внешний ревьюер или разбиение задач по классу риска?

Что попробовать на практике

- Требовать у модели источник для каждого числа, цитаты и важного факта; если источника нет, явно помечать это как гипотезу или экстраполяцию.
- Для важных ответов проверять не только данные, но и логическую цепочку: из каких предпосылок модель пришла к выводу и где там мог появиться якорь.
- Прогонять значимые результаты через другое мнение - человека, другую модель или внешний ревью - и сравнивать не "кто убедительнее", а где расходятся основания решения.
- Завести журнал ошибок для повторяющихся сценариев и фиксировать не только сам сбой, но и тип задачи, условия и способ исправления.
- Для гипотез и продуктовых решений пробовать формулировку через фальсификацию: просить не подтверждать идею, а искать наблюдения, которые ее опровергнут.

Открытые вопросы

- Где проходит граница между полезным контрактом на результат и слишком жестким управлением, которое само усиливает поддакивание?
- Насколько далеко можно делегировать модели приоритизацию, интерпретацию сигналов и продуктовые выводы без потери смысла?
- Как проверять сложные рассуждения там, где у команды нет готового внешнего эталона?
- Что дешевле в длинном горизонте: больше проверок сейчас или накопление систематических ошибок в агентском контуре?
- Какие поломки модели стоит лечить промптом и процессом, а какие лучше сразу считать фундаментальным ограничением?