3.2. Ключевой урок: «LLM — не редактор»

Задача

Многие задачи, которые кажутся тривиальными для человека — например, точная замена текста или поиск конкретного шаблона — оказываются проблематичными для больших языковых моделей. Причина в том, что эти модели *не «исполняют программу»*, а генерируют последовательность токенов, предсказывая следующее слово на основе вероятностной модели. Эта генеративная природа означает, что модель склонна к перестройке текста, пропускам и «галлюцинациям» (выдуманным фактам, отсутствующим в исходном контексте). В результате LLM-ы плохо справляются с детерминированными задачами, где требуется строгий алгоритм поиска и замены.

Почему LLM не подходит для детерминированных операций

Согласно обзору Лилиан Вэн о галлюцинациях в LLM, «галлюцинация» — это генерация несоответствующего, вымышленного или бессвязного содержимого; она возникает, когда модель выдаёт факт, не подкреплённый контекстом или знаниями[1]. Автор выделяет два типа галлюцинации: *внутриконтекстную* (выход должен соответствовать предоставленному источнику) и *внешнюю* (выход должен быть правдив и проверяем на уровне обучающих данных). Проблемы возникают уже на этапе предобучения: огромные корпусные данные содержат устаревшие или ошибочные сведения, которые модель запоминает, что приводит к ошибкам[1]. Дополнительное дообучение на небольших выборках также повышает склонность к галлюцинациям[1]. Эти факты подтверждают: LLM генерирует ответы на основе вероятностных закономерностей, а не следуя жёстким правилам.

Документация OpenAI по *best-практикам оценки* подчёркивает, что *генеративный ИИ является переменным*: модели могут выдавать разные результаты на один и тот же запрос, что делает традиционные методы тестирования непригодными[2]. Для оценки таких систем требуются специальные метрики и регулярная проверка качества. Эта изменчивость — следствие того, что модели не выполняют детерминированный «скрипт», а лишь предсказывают вероятные токены.

Наконец, мнение одного из сооснователей OpenAI Андрея Карпаты — разработчики переоценивают возможности агенто-ориентированных ИИ. На подкасте он отметил, что современные агентные системы «не работают» должным образом, поскольку недостаточно разумны, не способны работать с компьютером, не имеют долговременной памяти, и для их полноценной работоспособности потребуется около десяти лет[3]. Это ещё раз подчёркивает ограниченность текущих LLM-ов в качестве универсального инструмента, особенно для жёстко сформулированных задач.

Что происходит внутри модели

LLM формирует ответ, *последовательно предсказывая токен* и корректируя его в соответствии с вероятностным распределением. Эта цепочка решений приводит к тому, что при

попытке выполнить «поиск-замену» она скорее *создаст новый текст* с похожей смысловой конструкцией, чем просто заменит строку. Например, если попросить модель заменить переменную `x` на `y` в коде, она может переписать функции, переименовать другие символы и даже удалить участки, которые считает неуместными. Такие «творческие» преобразования полезны при рефакторинге, но опасны там, где требуется точность. Галлюцинация же проявляется тогда, когда модель додумывает факты, которых нет в исходных данных или мире, что делает её непригодной для автоматизированного поиска-замены.

Советы для разработчиков

- **Для детерминированных задач используйте программные инструменты.** Если нужно заменить строку, изменить синтаксис или проанализировать структуру кода, надёжнее применить регулярные выражения, инструменты разбора синтаксического дерева (AST) или написать Python-скрипт. Такие методы гарантируют точность и повторяемость результата.
- **Для недетерминированных или творческих задач ограничивайте модель.** При генерации нового текста важно жёстко задавать формат (например, в виде шаблона или JSON-схемы), использовать несколько примеров (few-shot) и добавлять внешние проверки, чтобы минимизировать галлюцинации. Регулярные оценки качества — обязательное условие, учитывая переменность выхода LLM[2].
- **Понимайте границы технологии.** Не стоит ожидать от современных LLM способности выполнять скрипты поиска-замены. Даже разработчики индустрии отмечают, что агентные системы далеки от реального применения и ещё долго будут развиваться[3]. Помните, что LLM — это инструмент для *генерации* идей, а не для выполнения точных операций.

Вывод

Ключевой урок для разработчиков: **LLM — не текстовый редактор и не интерпретатор кода**. Она создаёт текст, исходя из вероятностной модели, и поэтому склонна к «творческим» изменениям и галлюцинациям. Для точных, детерминированных действий над кодом и текстом лучше использовать классические инструменты (regex, AST, статический анализ). Если же задачи требуют генеративных возможностей, ограничивайте модель строгими рамками, подбирайте качественные примеры и регулярно оценивайте её поведение, понимая, что она может ошибаться и выдавать разные результаты при одинаковых запросах.

Источники

[1] Источник: lilianweng.github.io
[2] Источник: platform.openai.com
[3] Источник: businessinsider.com