3.3. Управление галлюцинациями и стилем (ICL/Few-shot)

Задача

При работе с большими языковыми моделями важно не только понимать их ограниченность, но и уметь управлять качеством выдачи: снижать количество галлюцинаций (выдумок) и добиваться нужного стиля и формата. Для этого используются техники In-Context Learning (ICL) и few-shot/multishot prompting — подача несколькими примерами в одном запросе. Правильно подобранные примеры позволяют модели лучше понять задачу, следовать желаемому формату и минимизировать фантазии.

Использование примеров (multishot prompting)

Документация Anthropic по prompt engineering подчёркивает, что примеры — «секретное оружие» для управления поведением модели. Несколько наблюдений:

- Примеры повышают точность и последовательность. По данным руководства, примеры уменьшают неправильное толкование инструкций, enforce ровную структуру и стиль и повышают способность модели справляться со сложными задачами[1].
- Рекомендовано включать 3–5 разнообразных, релевантных примеров. Чем больше хорошо проработанных примеров, тем лучше модель понимает ваш формат и контекст. В подсказке прямо говорится: «Power up your prompts: include 3–5 diverse, relevant examples»[1].
- Эффективные примеры должны быть:
    - релевантными — соответствовать реальным случаям применения;
    - разнообразными — покрывать разные нюансы, чтобы модель не подхватывала случайные паттерны;
    - чёткими — выделены в явные блоки (напр. с помощью тегов `<example>`), чтобы структура была однозначна[1].
- Просите модель генерировать или оценивать примеры. Руководство советует попросить модель оценить примеры на предмет релевантности и разнообразия либо сгенерировать дополнительные примеры на основе ваших заготовок — это помогает довести набор до оптимального.

На конкретном примере анализа отзывов показано, что без примеров Claude выдаёт длинный и путаный ответ, а с примерами — следует указанным категориям и формату. Это иллюстрирует, что примеры уменьшают галлюцинации и делают ответ структурированным.

Курс Anthropic по интерактивному prompt-инжинирингу

Открытый курс Anthropic «Interactive Prompt Engineering Tutorial» обучает основам создания эффективных подсказок. В нём девять глав, среди которых есть отдельные главы по использованию примеров и избежанию галлюцинаций. После прохождения курса слушатель:

- Осваивает базовую структуру хорошей подсказки, изучает распространённые ошибки и «80/20 техники» их устранения;
- Учится формулировать чёткие инструкции, назначать роли модели, отделять данные от инструкций, использовать системные промпты и управлять форматом вывода;
- Получает практические задания и «Example Playground» для экспериментов с изменением подсказок[2];
- Изучает главы «Using Examples» (few-shot prompting) и «Avoiding Hallucinations», где рассматриваются стратегии уменьшения домыслов. В этих главах рекомендуют давать модели достаточный контекст, формулировать ожидаемый стиль и быть готовым отвергать ответ, если модель не уверена.

Курс подчёркивает важность интерактивной практики: пользователю предлагают самостоятельно модифицировать примеры и наблюдать, как изменяется ответ модели. Такой подход помогает выработать навык выбора примеров, охватывающих разные типы входных данных и требуемых форматов.

Подходы к снижению галлюцинаций

Примеры — лишь один из инструментов. Ещё один важный подход — retrieval-augmented generation (RAG). Согласно исследованию, RAG комбинирует сильные стороны предобученных моделей и retrieval-моделей, интегрируя актуальные документы в процесс генерации. Авторы статьи отмечают, что RAG уменьшает число галлюцинаций и повышает качество ответов, особенно в специализированных доменах. В типичном RAG-конвейере выполняют классификацию запроса, поиск документов, ранжирование, упаковку и суммаризацию, а выбор оптимальных методов для каждого этапа позволяет сбалансировать производительность и скорость[3]. Исследование рекомендует комбинировать классический поиск (BM25), нейронные эмбеддинги и гибридные методы, а также использовать мультимодальный поиск для вопросов с изображениями[3].

Практические советы

- **Подготавливайте 3–5 примеров** с желаемым стилем и структурой. В примере для анализа отзывов каждое поле (категория, тональность, приоритет) было явно продемонстрировано: это избавило модель от излишних объяснений и ошибок.
- **Сделайте примеры разнообразными.** Включите разные типы входных данных и ожидаемых результатов, чтобы модель научилась распознавать варианты и не зацикливалась на шаблонах.
- **Используйте теги или явные разделители** (`<example>` или `---`) для разделения примеров и не смешивайте инструкции с данными: это позволяет модели легко выделить структуру.

- **Проверяйте ответы.** Если задача требует фактов, используйте RAG или заранее подготовленный контекст (документы, код). Это уменьшит вероятность галлюцинаций, поскольку модель будет опираться на предоставленную информацию, а не придумывать факты[3].
- **Обучайте модель говорить «не знаю».** В интерактивном курсе рекомендуют поощрять модель к признанию отсутствия информации, если контекст не даёт чёткого ответа. Это снижает риск домыслов.
- **Оценивайте качество.** Используйте как автоматические метрики, так и человеческую оценку для регулярного контроля стиля и достоверности. При необходимости корректируйте набор примеров.

Вывод

Управление галлюцинациями и стилем требует активного участия разработчика. Few-shot/multishot prompting с несколькими продуманными примерами — эффективный способ направить генерацию и минимизировать выдумки. Примеры должны быть релевантными, разнообразными и чётко структурированными[1]. Дополняя их retrieval-техниками, можно обеспечить модели актуальной информацией и ещё больше снизить вероятность галлюцинаций. Таким образом, комбинация ICL, строгого форматирования и RAG формирует надёжный инструментарий для получения предсказуемых и корректных ответов.

Источники

[1] Источник: docs.claude.com
[2] Источник: raw.githubusercontent.com
[3] Источник: arxiv.org