Извлечение, резюмирование, планирование: продвижение многопроходного ответного взаимодействия с помощью итеративного подхода

Дата: 2025-01-29 00:00:00

Ссылка на исследование: https://arxiv.org/pdf/2407.13101

Рейтинг: 65
Адаптивность: 70

Ключевые выводы:

Исследование представляет новый метод итеративного RAG (Retrieval-Augmented Generation) под названием ReSP (Retrieve, Summarize, Plan) для улучшения многоэтапного вопросно-ответного взаимодействия с LLM. Основная цель - решить две ключевые проблемы существующих итеративных RAG-методов: перегрузку контекстом из-за многократного поиска и избыточное планирование из-за отсутствия записи траектории поиска. Метод ReSP значительно превзошел существующие подходы на стандартных бенчмарках, показав улучшение F1-оценки на 4.1-4.4 пункта по сравнению с лучшими существующими методами.

Объяснение метода:

Исследование предлагает ценный итеративный подход к многоходовым вопросам с двойной суммаризацией. Концепции разбиения сложных вопросов на подвопросы, отслеживания глобального и локального контекста, а также методы эффективной компрессии информации могут быть адаптированы обычными пользователями, хотя и потребуют некоторой модификации для применения в стандартном чате.

Ключевые аспекты исследования: 1. **Итеративный подход ReSP (Retrieve, Summarize, Plan)** - метод, включающий двойную функцию суммаризации для многоходовых вопросно-ответных систем. Решает проблемы перегрузки контекста и избыточного планирования в итеративных RAG-системах.

Двухфункциональный суммаризатор - компрессирует информацию, ориентируясь одновременно на общий вопрос и текущий подвопрос, создавая две очереди памяти: глобальную (для основного вопроса) и локальную (для подвопросов).

Механизм итеративного процесса - система оценивает достаточность информации после каждого цикла поиска и либо формирует новый подвопрос, либо генерирует финальный ответ, избегая повторного запроса по уже обработанным

подвопросам.

Устойчивость к длине контекста - метод демонстрирует стабильную производительность независимо от объема извлекаемых документов благодаря эффективной компрессии информации.

Модульная архитектура - система состоит из четырех основных компонентов (Reasoner, Retriever, Summarizer, Generator), позволяющих гибко настраивать процесс вопросно-ответного взаимодействия.

Дополнение: Для работы метода, описанного в исследовании, в полном объеме действительно требуется специальная настройка и доступ к API для создания модульной архитектуры. Однако ключевые концепции и подходы можно адаптировать для использования в стандартном чате без дообучения или API.

Концепции, применимые в стандартном чате:

Итеративный подход к сложным вопросам Пользователь может самостоятельно разбить сложный вопрос на подвопросы Пример: "Сначала найдем информацию о X, затем о Y, и наконец о связи между X и Y"

Двойная суммаризация

Пользователь может запросить модель суммировать информацию двумя способами: "Суммируй эту информацию относительно моего общего вопроса [общий вопрос]" "Суммируй эту информацию относительно конкретного аспекта [подвопрос]"

Отслеживание прогресса Пользователь может вести "дневник исследования", прося модель записывать: "Запиши, что мы уже выяснили по общему вопросу [вопрос]" "Запиши, какие подвопросы мы уже исследовали и их результаты"

Оценка достаточности информации Перед формированием финального ответа: "На основе всей собранной информации, достаточно ли у нас данных для ответа на исходный вопрос [вопрос]? Если нет, какой еще подвопрос нам следует исследовать?"

Компрессия контекста При работе с большими объемами информации: "Суммируй эту информацию, сохраняя только ключевые факты, необходимые для ответа на вопрос [вопрос]"

Ожидаемые результаты от применения этих концепций:

Повышение точности ответов на сложные вопросы, требующие интеграции информации из разных источников **Снижение когнитивной нагрузки** на пользователя при работе со сложными многоступенчатыми задачами **Более структурированный подход к решению проблем**, где пользователь может отслеживать прогресс и не терять фокус **Эффективная работа с большими объемами информации** благодаря методам компрессии **Избегание повторений и**

циклических рассуждений благодаря отслеживанию уже исследованных подвопросов Хотя ручная реализация этих концепций требует больше усилий от пользователя по сравнению с автоматизированной системой, описанной в исследовании, они могут значительно улучшить качество взаимодействия со стандартным чат-интерфейсом LLM при решении сложных задач.

Анализ практической применимости: 1. **Итеративный подход ReSP** - **Прямая применимость**: Пользователи могут адаптировать принцип итеративного поиска для сложных вопросов, разбивая их на подвопросы и постепенно собирая информацию, что применимо даже в стандартных интерфейсах LLM. - **Концептуальная ценность**: Понимание итеративного подхода к сложным вопросам помогает пользователям осознать, что не все вопросы можно решить одним запросом. - **Потенциал для адаптации**: Пошаговый процесс рассуждения может быть реализован пользователями вручную через последовательные запросы к LLM.

Двухфункциональный суммаризатор Прямая применимость: Пользователи могут запрашивать модель суммировать информацию двумя способами: для основного вопроса и для конкретного аспекта проблемы. **Концептуальная ценность**: Высокая - демонстрирует важность сохранения как общей цели, так и промежуточных шагов при работе со сложными вопросами. **Потенциал для адаптации**: Пользователи могут создать собственные "очереди памяти", сохраняя в отдельных документах или чатах информацию по общему вопросу и подвопросам.

Механизм итеративного процесса

Прямая применимость: Пользователи могут применять логику "задать вопрос → получить ответ → оценить достаточность → задать уточняющий вопрос" в обычном диалоге с LLM. **Концептуальная ценность**: Помогает понять, как эффективно определять, когда информации достаточно для полного ответа. **Потенциал для адаптации**: Можно адаптировать для любого взаимодействия с LLM, где требуется многоэтапное решение проблемы.

Устойчивость к длине контекста

Прямая применимость: Пользователи могут применять технику суммаризации для сжатия больших объемов информации без потери ключевых данных. **Концептуальная ценность**: Демонстрирует, что сжатие контекста более эффективно, чем простое добавление всей информации в запрос. **Потенциал для адаптации**: Применимо к любым сценариям, где пользователь работает с большими объемами информации.

Модульная архитектура

Прямая применимость: Ограниченная, так как требует технических знаний для полной реализации. **Концептуальная ценность**: Помогает понять, что сложные задачи могут быть разбиты на отдельные функциональные компоненты. **Потенциал для адаптации**: Пользователи могут мысленно разделить процесс на этапы (поиск, суммаризация, планирование, ответ) и применять их последовательно.

Prompt:

Использование знаний из исследования ReSP в промптах для GPT ## Ключевые аспекты исследования для промптов

Исследование ReSP (Retrieve, Summarize, Plan) предлагает эффективный итеративный подход к многоэтапным запросам, который можно адаптировать для промптов GPT через:

Двойную суммаризацию — отслеживание как общего прогресса, так и текущих подзадач **Эффективное планирование** — сокращение избыточных итераций **Управление контекстом** — предотвращение перегрузки контекстом ## Пример промпта на основе ReSP

[======] # Запрос с использованием методологии ReSP

Я хочу, чтобы ты действовал как исследовательский помощник, применяя метод ReSP (Retrieve, Summarize, Plan) для ответа на мой сложный вопрос: [СЛОЖНЫЙ МНОГОЭТАПНЫЙ ВОПРОС].

Следуй этому процессу:

ПЛАНИРОВАНИЕ: Разбей мой вопрос на необходимые подвопросы, которые нужно решить последовательно.

ИТЕРАТИВНЫЙ ПРОЦЕСС: Для каждого подвопроса:

Укажи, какую информацию нужно найти Предоставь ответ на подвопрос Создай две суммаризации: ГЛОБАЛЬНАЯ ПАМЯТЬ: Краткое резюме того, как этот ответ помогает решить основной вопрос ЛОКАЛЬНАЯ ПАМЯТЬ: Ключевые выводы для текущего подвопроса

ФИНАЛЬНЫЙ ОТВЕТ: После решения всех подвопросов:

Составь окончательный ответ на основной вопрос Предоставь краткое обоснование, как подвопросы привели к этому ответу Пожалуйста, ограничься максимум 3 итерациями для эффективности. [======]

Как это работает

Данный промпт адаптирует ключевые принципы ReSP для работы с GPT:

• Структурированное разбиение задачи — подобно компоненту Reasoner в ReSP

• Двойная суммаризация — имитирует работу Summarizer, создавая глобальную память (для основного вопроса) и локальную память (для текущего подвопроса)

• Ограничение итераций — исследование показало, что 3 итерации оптимальны

(среднее число итераций ReSP составляло 1.24-1.60)

• Предотвращение повторений — благодаря суммаризации предыдущих шагов избегается повторное запрашивание той же информации

Этот подход позволяет эффективно решать сложные многоэтапные задачи, сохраняя прозрачность процесса рассуждения и экономно используя контекстное окно модели.