Большие языковые модели

05.10.2024

Какие бывают LLM

ChatGPT

(SOTA, общего назначения)

Claude

(наука и математика)

LLAMA

(open source, общего назначения)

Saiga (Mistral)

(русский язык / можно запустить локально)

GigaChat*

(русский язык, общего назначения)

Gemini

(большое контекстное окно)

Суммаризация бооооольших документов

Одна из задач, с которой хорошо справляются генеративные модели – это **суммаризация** текстов. Однако если вы хотите обобщить текст, вам нужен **весь текст**, чтобы он **поместился в контекстное окно**, плюс prompt для его обобщения.

Ограничения контекстного окна

Это может оказаться сложным, если текст, который нужно обобщить, слишком длинный. Существуют LLM с очень большими контекстными окнами:

GPT-4 имеет вариант с контекстным окном в 32 тыс. токенов.

Claude 3 Opus имеет контекстное окно в 200 тыс. токенов.

Gemini 1.5 Pro может иметь контекстное окно до 1 миллиона токенов.

Нецелесообразность использования LLM с большим контекстным окном

Эти модели могут быть слишком **дорогими**.

Иногда моделям **трудно** использовать **всю информацию** в очень длинных промптах.

Ваш **текст может быть длиннее**, чем все доступные модели (да, даже с контекстными окнами 1M).

MapReduce суммаризация

К счастью, существует **техника**, позволяющая заставить LLM суммаризовать документ, длина которого превышает размер контекстного окна. Эта техника называется **MapReduce**.

Она основана на **разделении** текста на **множество** более **мелких** текстов, которые помещаются в контекстное окно, и последующей **суммаризации** каждой части **отдельно**.

MapReduce суммаризация. Шаг 1

Сначала длинный документ делится на фрагменты с помощью разделителя текста.

Если в тексте есть разделы, и **все** разделы **меньше** контекстного **окна**, мы можем разделить его на эти разделы.

Если в тексте **нет** четких **разделов** или разделы слишком **велики**, текст можно разделить на **одинаковые** по размеру **фрагменты** по знакам. В последнем случае возникает **проблема** разделения связанных предложений на разные куски.

Чтобы избежать этой проблемы, мы можем использовать перекрытие между фрагментами. При таком решении последние N символов одного куска будут повторяться как первые N символов следующего куска, поэтому контекст не будет потерян.

Map–Reduce Суммаризация. Шаг 2

Далее, все фрагменты суммаризируются отдельно с помощью LLM.

Не забудьте настроить **system prompt**, чтобы помочь модели понять, что это за документ и **как** правильно его **суммаризировать**.

Например, вы можете захотеть, чтобы саммари представляло собой список **основных моментов** текста, а можете захотеть, чтобы саммари состояло всего из **нескольких предложений**.

Map–Reduce Суммаризация. Шаг 3

Третий шаг **необязателен**. Если комбинация всех саммари не помещается в контекстное окно, мы не сможем запросить консолидированное саммари у LLM. Вместо этого нам нужно сделать комбинацию всех саммари меньше.

Мы **объединим** саммари в **группы**, которые помещаются в контекстное окно.

Затем мы напишем **prompt**, который **объединит** все саммари в **единое саммари** с ключевыми идеями.

После сокращения всех групп саммари, если получившиеся объединенные саммари все еще не помещаются в контекстное окно, этот процесс выполняется снова.

Map–Reduce Суммаризация. Шаг 4

Наконец, когда все саммари **помещаются** в контекстное окно, мы можем объединить или сократить их в **итоговую саммари**.

Мы напишем prompt, который объединит все саммари в одно итоговое саммари со всеми ключевыми идеями. Этот prompt обычно **такой же, как и в шаге 3**, поскольку идея в основном та же: объединить список саммари в одно короткое саммари.

Map–Reduce Суммаризация. Выводы

Этот метод может оказаться не менее дорогим, чем использование LLM с большим контекстным окном, особенно если ваши затраты приходятся на один токен (как это происходит во всех LLM MaaS). Используя этот метод, вы все равно будете **использовать все токены** в документе, **плюс** токены **промежуточных** саммари (в качестве выходных данных и затем в качестве входных данных для промежуточных prompt). Сначала необходимо изучить, какой метод будет более затратным для вашего случая использования.

Хотя лучшие модели будут создавать более качественные саммари, возможности саммари **небольших моделей** будут достаточно **хороши** для большинства случаев. Результат после многоуровневого саммари будет одинаково качественным, поэтому вы можете обойтись более дешевыми моделями, если решите использовать этот метод.

Комментарии (Михайлов А. А.) по презентациям

- - Во всех презентациях нет **четкой** архитектуры решения.
- - Есть хорошие презентации (красиво + информативно)