Документация по LLM Endpoint

Этот документ описывает параметры, которые можно передать в теле запроса к LLM (Large Language Model) Endpoint. Каждый параметр контролирует определенные аспекты генерации ответа. Обращаемся в **`/generate`** эндпоинт.

Параметры запроса (поместите их в тело запроса)

`- prompt: (string | array<string>).` **`Required`**
Входная строка или массив строк, которые будут использоваться как начальный контекст для генерации ответа.
Example: `"Once upon a time..."`

`- system_prompt: (string | null)`
Дополнительное системное сообщение для настройки модели. Предоставляет общий контекст для генерации.
Example: `"You are a helpful assistant"`

`- temperature: (number | null)`
Параметр, регулирующий уровень случайности в генерации. Низкие значения делают результат более предсказуемым, высокие — более разнообразным.
Example: `0.42`

`- apply_chat_template: (boolean | null)`
Применяет шаблон чата к входному тексту перед генерацией ответа. Подробнее можно узнать здесь.
Example: `true`

`- stop: (string | array<string> | null)`
Строка или массив строк, которые останавливают генерацию при их появлении. Эти стоп-токены не включаются в окончательный ответ.
Example: `"</Stop>"`

`- include_stop_str_in_output: (boolean | null)`
Указывает, включать ли стоп-токены в вывод модели.
Example: `true`

`- max_tokens: (integer | null)`
Максимальное количество токенов, которое можно сгенерировать. Ограничивает длину вывода.
Example: `100`

- `min_tokens: (integer | null)`
Минимальное количество токенов, которые модель должна сгенерировать.
Example: `10`

- `n: integer`
Количество вариантов ответов, которые модель должна сгенерировать.
Example: `4`

- `best_of: (integer | null)`
Количество ответов, которые будут сгенерированы, после чего модель выберет n лучших.
Example: `4`

- `frequency_penalty: (number | null)`
Параметр, штрафующий за частое повторение токенов в ответе. Значения больше 0 снижают вероятность повторений.
Example: `0.1`

- `presence_penalty: (number | null)`
Штрафует за наличие новых токенов. Значения больше 0 поощряют новые токены; меньше 0 — повтор.
Example: `0.0`

- `repetition_penalty: (number | null)`
Контролирует повторение токенов. Значения больше 1 снижают вероятность повторений, меньше 1 — увеличивают.
Example: `1.1`

- `length_penalty: (number | null)`
Управляет длиной вывода. Значения больше 1 сокращают ответ, меньше 1 — удлиняют.
Example: `1.1`

- `seed: (integer | null)`
Случайное значение для воспроизводимости генерации.
Example: `42`

- `stream: (boolean | null)`
Если установлено в true, ответ будет возвращаться по частям (потоково).
Example: `false`

- `top_p: (number | null)`

Ограничивает выбор следующего токена в зависимости от вероятностного распределения. Значение в пределах (0, 1]. Установите 1, чтобы учитывать все токены.
Example: `0.9`

- `min_p: (number | null)`
Величина, представляющая минимальную вероятность того, что токен будет рассмотрен, относительно вероятности наиболее вероятного токена. Должна находиться в диапазоне [0, 1]. Установите значение 0, чтобы отключить эту функцию.
Example: `0.1`

- `top_k: (integer | null)`
Параметр, который ограничивает выбор следующего токена только теми, которые находятся в указанной части распределения вероятностей. Установите значение -1, чтобы учитывать все токены.
Example: `1000`

- `stop_token_ids: (array<integer> | null)`
Идентификаторы токенов, которые останавливают генерацию при их появлении.
Example: `[0, 101]`

- `ignore_eos: (boolean | null)`
Игнорировать ли токен конца последовательности (EOS) и продолжать генерацию после его появления.
Example: `false`

- `skip_special_tokens: (boolean | null)`
Пропускать ли специальные токены в сгенерированном тексте. Например токен начала предложения.
Example: `false`

- `spaces_between_special_tokens: (boolean | null)`
Добавлять ли пробелы между специальными токенами в выводе.
Example: `true`

Ограниченная генерация

Модель использует библиотеку для ограниченной генерации. Для получения подробной документации по библиотеке вы можете посетить этот веб-сайт: Outlines

- `choice: (array<string> | null)`
Массив строк, из которых модель может выбрать один для генерации ответа.

Example: [“cat”, “dog”]

- schema: (string | object | object | null)
JSON-схема, которая определяет структуру данных, которую модель должна сгенерировать.
Example:
“{
  "title": "User",
  "type": "object",
  "properties": {
    "name": {"type": "string"},
    "last_name": {"type": "string"},
    "id": {"type": "integer"}
  }
}”

- regex: (string | null)
Регулярное выражение, которому должен соответствовать сгенерированный текст.
Example: "^[a-zA-Z0-9]+$"

- substring: (string | null)
Строка для поиска в сгенерированном тексте. Модель будет проверять наличие этой подстроки в выводе. Подробнее можно узнать здесь
Example: "наши участники хакатона - самые лучшие <3"

Пример запроса к LLM

```
{
  "prompt": "Please provide only a json with the name of the biggest company in the world",
  "stop": "}",
  "max_tokens": 100,
  "temperature": 0.42
}
```