Токены и токенизация

Модель читает текст небольшими кусочками, и это не всегда слова. Где проходят их границы и почему у разных моделей они не совпадают?

Модель продолжает текст небольшими фрагментами — токенами. Но где проходят их границы? Введите свою фразу и сравните разбиение у разных моделей. Затем разберём, как частые соседства помогают построить словарь.

Ранее

Ранее мы разобрали: языковая модель и предсказание следующего токена. Здесь используем эти идеи в новой задаче.

← Языковая модель и предсказание следующего токена

1. Текст, токены и номера

Модель обрабатывает текст как последовательность токенов. Токен может быть целым словом, его частью, знаком препинания или фрагментом вместе с пробелом. Поэтому число токенов не совпадает ни с числом слов, ни с числом символов.

Как токенизируется ваш текст

Введите свою фразу

Преподаватель объясняет сложную тему простыми словами.

GPT-2 / GPT-3 | GPT-3.5 / GPT-4 | GPT-4o / GPT-4.1 / GPT-5 | Claude 2 (архивный) | GigaChat 3.5 | ruGPT-3 | Qwen3.5

Русский пример | English | Эмодзи и код

Разбиваем текст…

Интерактивный эксперимент — в электронной версии урока.

Фраза «Преподаватель объясняет сложную тему простыми словами.» — это 54 символа, 6 слов и 13 токенов у GPT-4o, GPT-4.1 и GPT-5. Для обычного текста они используют один токенизатор — `o200k_base`, поэтому объединены в одну кнопку. Соответствие моделей и токенизаторов Границы фрагментов не обязаны совпадать с морфемами или смысловыми единицами: их задают правила конкретного токенизатора.

У каждого токена есть номер в словаре. По номеру программа находит числовое представление токена, с которым дальше работают слои модели. Сам номер ничего не измеряет: токен с большим номером не важнее другого. Сменится токенизатор — изменятся и номера, и границы фрагментов той же строки.

Откуда берутся такие границы? Вручную их никто не назначал.

> Учебное разбиение слова «котика» — «котик» и «а». Сколько в нём токенов?

Заметка

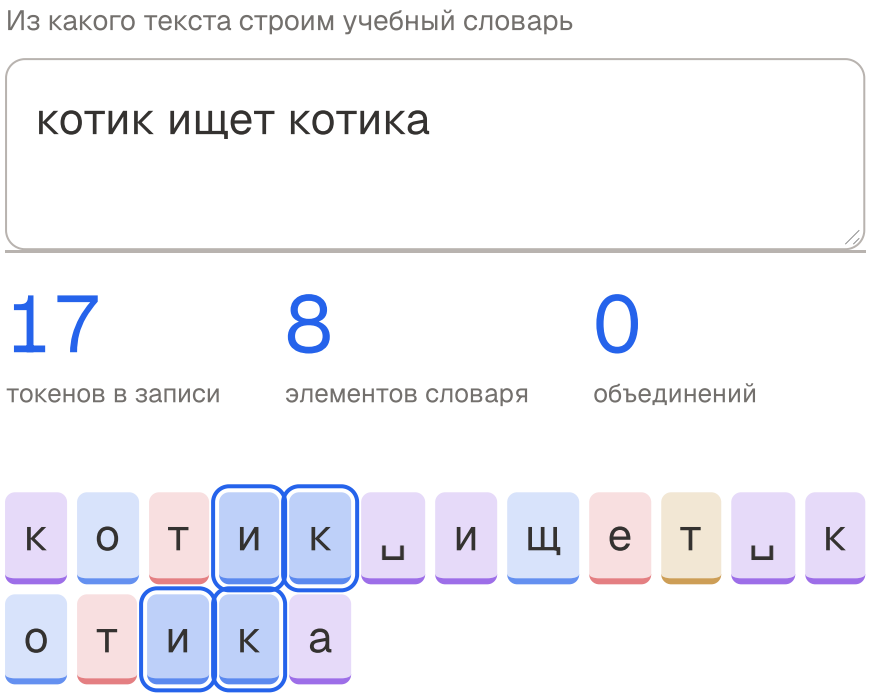
Токенизация использует готовый словарь и правила; число токенов не равно автоматически числу слов.

2. Построение словаря

Словарь токенов можно построить, постепенно объединяя частые соседние фрагменты. На этом принципе основан один из распространённых подходов — BPE (Byte Pair Encoding, «кодирование пар байтов»). Возьмём фразу «котик ищет котика» и начнём с отдельных букв и пробела: так удобно проследить каждое объединение вручную.

В этой фразе 17 символов, но только восемь разных: «к», «о», «т», «и», пробел, «щ», «е» и «а». На старте словарь содержит восемь элементов, а запись — 17 токенов. Один элемент словаря может встречаться в тексте много раз.

Словарь растёт — запись сокращается



Объединяйте частые пары. Затем впишите другую короткую фразу и постройте словарь для неё.

Интерактивный эксперимент — в электронной версии урока.

В исходной фразе пары «и + к», «к + о», «о + т» и «т + и» встречаются по два раза. При равной частоте в нашем опыте выбираем по алфавиту. Поэтому первой становится «и + к», хотя другие пары не менее частые. Это правило разрешения равенства, а не поиск суффикса.

После объединения двух вхождений «и + к» запись сокращается до 15 токенов, а в словаре появляется девятый элемент — «ик». Короткие «и» и «к» остаются: они ещё могут пригодиться в других местах. Пересчитываем соседние пары уже в новой записи: внутри «ик» нет двух отдельных токенов.

Дальше объединяются «к + о», «ко + т» и «кот + ик». За первые четыре объединения запись сокращается: 17 → 15 → 13 → 11 → 9 токенов. Словарь при этом растёт с восьми до двенадцати элементов. На этом шаге «котик» записывается одним токеном, а «котика» — двумя: «котик» и «а».

Число элементов словаря и длина записи отвечают на разные вопросы. Словарь — набор доступных деталей. Запись — последовательность деталей, выбранных для конкретного текста. Повторы увеличивают длину записи, но не создают новые элементы словаря.

Впишите свою фразу и проследите объединения. На каждом шаге выбирается самая частая соседняя пара. Пара может встретиться и один раз: её тоже можно объединить. В опыте можно продолжать, пока внутри слов не останется соседних токенов.

Настоящий токенизатор строят по большому корпусу и заранее задают размер словаря. Здесь мы начинаем с букв и объединяем только внутри слов. В байтовом BPE исходными деталями служат байты; пробелы, знаки и другие границы обрабатываются по правилам реализации.

> В словаре токен «кот» имеет номер 900, а «дом» — номер 12. Что можно заключить о смысле по этим номерам?
>
> ○ Эти номера можно использовать с любым другим словарём
>
> ○ Номера не измеряют смысловую важность
>
> ○ Номер показывает длину текстового фрагмента
>
> ○ Номер задаёт частоту токена в любом тексте

3. Готовый словарь и применение

Границы токенов не обязаны совпадать с корнями, приставками или смысловыми частями слова. Алгоритм объединения ориентируется на статистику корпуса и правила реализации. Поэтому неожиданная граница внутри слова сама по себе не означает ошибку.

Если при смене языка текст стал занимать больше токенов, это ещё не показывает, что его смысл сложнее. Изменились подходящие элементы словаря. Аналогично короткое редкое слово может потребовать больше токенов, чем длинное частое.

Число токенов полезно знать при оценке длины доступного контекста. Но посчитать его точно по числу слов нельзя: нужно применить именно тот токенизатор, который ожидает модель. Подмена словаря при прежних параметрах нарушит соответствие между идентификаторами и их представлениями.

Построение словаря и разбиение нового текста — две разные процедуры. В нашем опыте мы строили словарь: считали пары и добавляли записи. При обработке нового запроса токенизатор уже

знает словарь и порядок объединений. Он не подстраивает словарь под каждую новую фразу.

Размер словаря — число разных элементов, которые можно использовать. Длина текста — число элементов в конкретной записи, включая повторы. После четырёх объединений в примере словарь вырос до 12 записей, а фраза сократилась до 9 токенов. Эти два числа меняются в противоположные стороны и отвечают на разные вопросы.

> Готовый токенизатор обрабатывает новую строку. Что происходит с его правилами объединения?
>
> - ○ Словарь обязательно обучается заново
> - ○ Каждый пробел удаляется
> - ○ Все слова становятся одним токеном
> - ○ Применяются ранее сохранённые правила

4. Один текст и разные токенизаторы

Один и тот же смысл может потребовать разного числа токенов на разных языках. Сравним русскую фразу «Преподаватель объясняет сложную тему простыми словами.» и её английский вариант «The teacher explains a complex topic in simple words.».

У GPT-2 и GPT-3 русский текст дробится очень мелко: эта фраза занимает 59 токенов, а английская — всего 10. Соответствие моделей и словарей Иногда даже одна русская буква записывается несколькими токенами. Поэтому 54 символа русского текста здесь превратились в 59 токенов.

В более новых моделях OpenAI разрыв уменьшился. У GPT-3.5 и GPT-4 получается 23 токена для русской фразы и 10 для английской, у GPT-4o, GPT-4.1 и GPT-5 — 13 и 10. Исходная строка осталась той же: готовый словарь позволяет записать её более длинными фрагментами. При этом новая версия модели не обязательно означает новый токенизатор.

Проверьте это в опыте в начале урока: выберите «Русский пример», смените модель, затем нажмите «English». Сравните также GigaChat, Qwen и ruGPT на своей фразе: названии школьного кружка, предложении из учебника или строке Python.

Для исторического сравнения добавлен «Claude 2 (архивный)»: русская фраза занимает 26 токенов, английская — 10. Это старый открытый токенизатор Anthropic; для Claude 3 и новее он уже не подходит. Исходник и ограничение версий

Универсального соотношения между длинами русского и английского текста нет: всё зависит от строки и словаря. Некоторые токенизаторы начинают с байтов. Одна буква или эмодзи тогда может

занимать несколько токенов, а число токенов — превышать число символов.

В опыте такие байтовые части объединены в читаемую плашку. Нажмите на неё: несколько номеров означают несколько токенов. Счётчик учитывает каждый номер, даже когда его нельзя показать отдельной буквой. Пробел отмечен знаком ␣, перенос строки — ↵. Они не исчезают из записи.

Границы цветных фрагментов не обязаны совпадать с частями слова. Цвет только помогает увидеть соседство; он не обозначает значение или часть речи. Соедините фрагменты и сравните получившийся текст с введённой строкой.

Компактная токенизация помогает поместить больше текста в контекстное окно, но сама по себе не доказывает лучшее понимание. Для длины нужен токенизатор, для качества ответа — отдельная проверка результата. Даже у одной модели короткий запрос иногда приводит к длинному ответу: вход и выход стоит измерять отдельно.

Токенизатор выполняет преобразование по готовым правилам без запуска языковой модели. Поэтому его можно попробовать даже на устройстве, на котором сама модель не помещается в память.

Токенизация использует готовый словарь и правила; число токенов не равно автоматически числу слов.

Токену соответствует номер в словаре. Байтовые токены могут вместе составлять один символ.

Обучение токенизатора и применение его правил — разные этапы.

Домашняя работа

1. Учебное разбиение: «котик», «□», «котик», «а». Сколько элементов в последовательности, если пробел считается токеном?

Источники

- Исходники tiktoken (https://github.com/openai/tiktoken)
- Обучение и применение BPE (https://huggingface.co/learn/llm-course/chapter6/5)