“トークン”とLLM

トークンとは？

- テキストを小さい単位に分割したもの
- LLMが入出力しているテキストの単位

> 例: `I have a pen.` → `I` / `have` / `a` / `pen` / `.`

なぜにトークン？

LLM自体やプロンプトエンジニアリングを知る上では基礎知識

LLMサービスを使う分には知らなくても問題なく動かせるけど、

- 利用料金
- 性能
- 特性

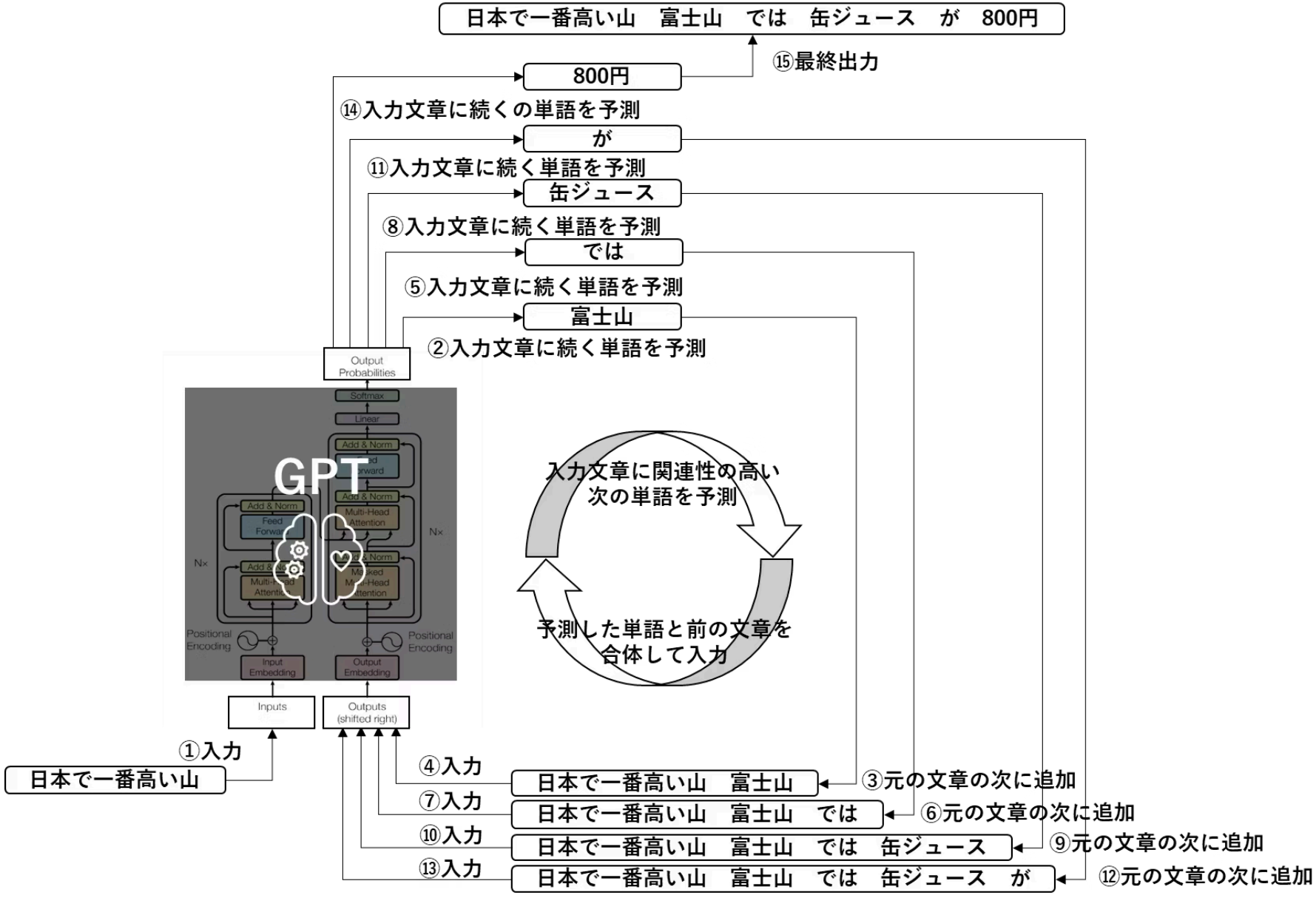
の理解に役立ちそう

LLMの入出力の流れ

GPTモデルの例

画像引用元：

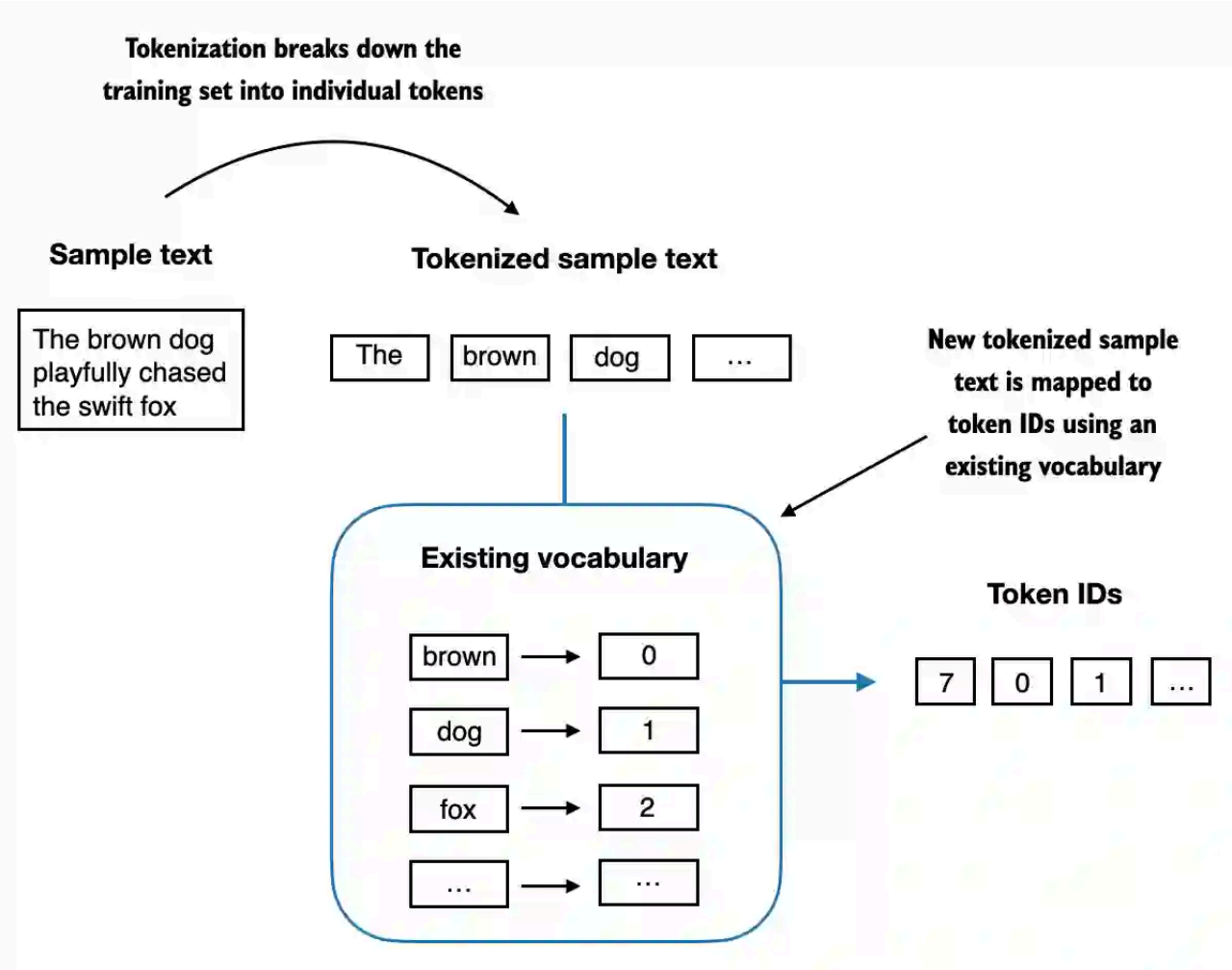
https://qiita.com/ksonoda/items/b767cbd283e379303178



LLMへ入力される流れ

1. 入力テキスト → トークン化（Tokenize)
    - LLMが扱える語彙で分割
2. トークン → ベクトル化（Embedding）
3. ベクトル列を LLM に入力

画像引用元：
https://github.com/rasbt/LLMs-from-scratch



日本語みたいなものどう分割してるの？

語彙なんて無限に増えてくのにどうしてるの？

トークン化と語彙

単語をサブワードに分割したりしていて、
BPE (Byte Pair Encoding) という方法などを使ってる。

文字をbyte単位の表現に変換して、

- 最小単位として1byteを1トークン
- 頻出するbyte列の組み合わせはそれで1トークン

> `I'm an octopus.` -> `I'm` / `an` / `oct` / `opus` / `.`
> `こんにちは、世界。` -> `こんに` / `ち` / `は` / `、` / `世界` / `。`

tokenizer

OpenAI Tokenizer:

https://platform.openai.com/tokenizer

日本語は不利？

英語に比べて日本語の文は2倍くらいのトークンが必要となるらしい

トークン効率（同じ文で消費するのトークン数）の影響

- コストと速度への影響
  - LLMサービスのAPIはトークン使用料での従量課金が多い
  - 扱うトークンが多ければその分処理も増える
- 出力長・生成能力への影響
  - モデル/サービスによって、一回で出力できるトークン数がきまっている
- コンテキスト長の制約と文脈保持
  - 最大で処理できるトークン数は決まっていて、溢れると古いトークンは削除される

入出力への工夫

RAG

Retrieval Augmented Generation

https://www.pinecone.io/learn/retrieval-augmented-generation/

coding agent(cline)

DBにインデックスしたりせず変わり続けるドキュメントを検索していく

https://cline.bot/blog/why-cline-doesnt-index-your-codebase-and-why-thats-a-good-thing

Structured Generation

- LLMから出力されるトークンに対してパーサーを適用していくことで、特定のフォーマットでの出力に限定させる
- 「JSONだけを出力して」と明記しても他のことを出力してしまうのを解決

https://engineers.ntt.com/entry/202503-structured-generation/entry

おしまい