概述

BPE (Byte-Pair Encoding) 是一种高效的分词器，被广泛应用于 GPT 系统模型中。它基于字节级别的合并算法，具有高效、无损和支持多语言的特点。本文将重点介绍在已有词汇表和合并规则的前提下，如何将文本转换为 token，以及如何将 token 转换回文本。

初始化

BPE 分词器首先建立一个基础字节表，用于处理所有可能的输入。

1. **基础字节表**

   GPT-2 BPE 使用 256 个字节作为基础词汇表。为了处理 UTF-8 文本，它将 0-255 的字节值映射到 Unicode 码点。

   - 可打印的 ASCII 字符（如 `a`、`b`、`c`）映射到其自身对应的 Unicode 码点。
   - 不可打印或控制字符（如换行符 `\n`）以及非 ASCII 字节（用于表示多字节的 UTF-8 字符）则映射到从 256 开始的专用 Unicode 码点。

   这个映射过程是固定的，旨在确保任何字节序列都能被表示为唯一的 Unicode 字符序列，从而避免在处理过程中出现歧义。这个映射表是可逆的，可以轻松地将 Unicode 码点转换回原始字节。

2. **加载词汇表 (Vocab)**

   GPT-2 的词汇表包含 50257 个 token。它是一个从子词字符串到 token ID 的映射。

   - **子词**：词汇表中的键是子词字符串（例如 `"the"`，`"Ġthe"`）。
   - **Token ID**：值是对应的整数 ID（例如 `50256`）。

   在加载词汇表时，需要将每个子词字符串解析为一系列的**字节序列**。例如，子词 `"Ġthe"` 会被解析为字节 `[32, 116, 104, 101]`。**这里不需要将子词字符串映射到 Unicode 码点，因为分词过程本身是基于字节操作的。**

   通过加载词汇表，我们得到了两个核心映射：

   - `token_to_subword`：从 token ID 到其对应字节序列的映射。
   - `subword_to_token`：从字节序列到其对应 token ID 的映射。

3. **加载合并规则 (Merges)**

   合并规则文件（`merges.txt`）存储了 BPE 算法训练出的所有合并操作。每行包含一个字节对，代表了应该被合并的两个子词。

   - 例如，一行 `['t', 'h']` 表示字节 `t` 和 `h` 应该被合并。
   - GPT-2 的合并规则是有序的，**每一行都代表了一个优先级**。靠前的规则优先级更高，会先进行合并。

   这些规则被加载到一个字典中，键是**字节对**，值是合并的**优先级（rank）**。`rank` 越小，优先级越高。

Encode 过程

Encode 过程是将原始文本转换为 token ID 序列。

1. **预处理和分割**

   首先，对输入的文本应用正则表达式，将其分割成多个**文本片段**。这个步骤的目的是将文本分解为单词、标点符号、空格等，为后续的字节化和合并做准备。

2. **字节化和合并**

   接下来，对每个文本片段进行处理：

   - 将文本片段转换为其**UTF-8 字节序列**。例如，文本片段 `"the"` 变为字节序列 `[116, 104, 101]`。
   - **应用合并规则**：这是一个迭代过程。在每一步中，分词器会扫描当前的字节序列，**查找所有能够被合并的字节对**。
   - 它会根据加载的合并规则，找到**优先级最高（rank 最小）**的那个字节对。
   - 然后，将所有出现的这个最高优先级的字节对进行合并，生成一个新的子词。
   - 重复这个过程，直到没有可以合并的字节对为止。

   例如，字节序列 `[t, h, e]`，如果合并规则中 `(t, h)` 的优先级最高，则会合并为 `[th, e]`。如果 `(th, e)` 也可以合并，则继续合并为 `[the]`。

3. **Token ID 转换**

   当合并过程结束后，我们会得到一个由字节序列组成的列表，每个字节序列代表一个最终的子词。

   - 使用 `subword_to_token` 映射表，将每个最终的子词字节序列转换为其对应的 token ID。

4. **特殊 token 处理**

   GPT-2 的特殊 token `<|endoftext|>` 也是作为普通字符串处理的。如果它出现在输入文本中，预处理步骤会将其作为一个单独的片段。由于它本身就是一个完整的 token，它不会经过 BPE 合并过程，而是直接在 `subword_to_token` 映射中找到对应的 ID。

Decode 过程

解码过程相对简单：

1. **Token ID 到子词**：对输入的 token ID 序列，使用 `token_to_subword` 映射表，将每个 token ID 转换回其对应的**字节序列**。
2. **拼接**：将所有字节序列按顺序拼接在一起，形成一个完整的字节序列。
3. **字节到字符串**：将最终的字节序列通过 UTF-8 解码，转换回原始字符串。