# 2.1

# (1) WordPiece 算法原理

词级分词：OOV（Out-Of-Vocabulary）问题严重。
字符级分词：序列过长、丢失词义。
WordPiece 用“子词 (subword)”折中：

- 高频整词保持完整，低频词被拆为带前缀 `##` 的子片段；
- 词表大小可控，仍能解析任意字符串。

## 训练流程（贪婪合并法）

1. **初始化**：词表 ← 特殊符号 + 训练语料出现过的所有单字符。
2. **统计双元组频次**：扫描语料，统计相邻 token 组成的 (x,y) 频数。
3. **计算合并得分**

$$\mathrm{score}(x,y) = \sum_{w \in \mathcal{D}} f(w)\mathrm{count}_w(x,y)$$

   其中 $f(w)$ 为词 $w$ 在语料中的出现次数。
4. **合并最佳 pair**：生成新 token xy，替换语料中全部 x y。
5. 循环 2–4，直至词表大小达到预设 `vocab_size` 。

与 BPE 的差异：BPE 按纯频次选 pair，WordPiece 使用似然增益评分，能避免将高频前后缀盲目黏合。

## 推断（Longest-Match）

对待分词串自左向右执行最长前缀匹配；若整词不存在则不断截短前缀并在剩余部分前加 `##` 继续，直至整串被拆完。时间复杂度 $O(n)$。

# (2) 代码实现说明

### 主要补全内容

1. 字符初始化 每个单词先拆为“首字符 + 后续字符加 ## 前缀”的 token 列表，为后续合并做准备。
2. 循环直至词表达标 while len(vocab) < vocab_size:　保持词表未满就继续迭代。
3. 统计相邻 token 对频次 对所有单词的相邻子词 (a,b) 按词频加权计数，得到最常共同出现的 pair。

4. 合并最高频 pair 并扩充词表 将出现次数最多的 (left,right) 合并成新 token left + right.lstrip('##')；若新 token 已存在则视作收敛。
5. 更新所有单词的 token 序列 遍历 tokenized_words，把匹配到的最佳 pair 替换为新 token，实现“语料级别同步替换”，再进入下一轮。。

```python
# Add your code here.
    tokenized_words = {}
    for word in word_freqs:
        # 首字符原样，其余字符带 ## 前缀
        chars = [word[0]] + [f"##{c}" for c in word[1:]]
        tokenized_words[word] = chars

    # 2. 不断做 BPE-style 的 pair 合并直到达到目标大小
    while len(vocab) < vocab_size:
        # 2.1 统计所有相邻 token 对出现的加权频次
        pair_counts = defaultdict(int)
        for word, tokens in tokenized_words.items():
            freq = word_freqs[word]
            for a, b in zip(tokens, tokens[1:]):
                pair_counts[(a, b)] += freq

        if not pair_counts:          # 如果没有可继续合并的 pair
            break

        # 2.2 找到出现频次最高的 token 对
        best_pair = max(pair_counts.items(), key=lambda x: x[1])[0]

        # 2.3 生成合并后的 token，规则：去掉右 token 的前导 ## 再拼接
        left, right = best_pair
        merged_token = left + right.lstrip("#")  # remove leading "##" from right

        # 如果新 token 已在词表中，说明提前收敛
        if merged_token in vocab:
            break
        vocab.append(merged_token)

        # 2.4 更新所有单词的 token 序列
        for word, tokens in tokenized_words.items():
            new_tokens = []
            i = 0
            while i < len(tokens):
                # 如果当前和下一位正好是 best_pair，就替换成 merged_token
                if i < len(tokens) - 1 and tokens[i] == left and tokens[i + 1] == right:
                    new_tokens.append(merged_token)
                    i += 2
                else:
                    new_tokens.append(tokens[i])
                    i += 1
            tokenized_words[word] = new_tokens
```

# (3)

**得到词表**

```
['[PAD]', '[UNK]', '[CLS]', '[SEP]', '[MASK]', '##a', '##c', '##d',
'##e', '##f', '##g', '##h', '##i', '##j', '##k', '##l', '##m', '##n',
 '##o', '##p', '##r', '##s', '##t', '##u', '##v', '##w', '##x', '##y',
  'a', 'c', 'd', 'e', 'f', 'h', 'i', 'l', 'm', 'n', 'o', 'p', 's', 't',
  'u', 'y', '##er', '##in', '##ing', '##ni', 'sc', '##en','##ro', 'pe',
   'pek', 'peking', 'uni', 'univ', 'univer', 'univers','universi', 'universit',
    'university', 'is', '##at', '##ate','##ct', 'co', 'com', 'comp', 'compu', 'comput', 'compute
```

**'nous etudions a l universite de pekin'** 的分词结果为
**['n', '##o', '##u', '##s', 'e', '##t', '##u', '##d', '##i', '##o', '##n', '##s', 'a', 'l', 'universit', '##e', 'd', '##e', 'pek', '##in']**

# (4)

## 触发 `[UNK]` 的示例文本

**xylophone music is great** 得到结果 **Tokenization result: ['[UNK]', 'm', '##u', '##s', '##i', '##c', 'is', '[UNK]']**

## 解释 Llama

Llama的 tokenizer 之所以几乎不需要 [UNK] (未知词) 标记，主要归功于其采用的 SentencePiece (特别是 Unigram 模型) 和 字节回退 (byte-fallback) 机制。

所有字符皆可编码: SentencePiece 将 所有 UTF-8 字节 都视为合法的 token。这意味着任何文本，即使是罕见字、乱码或者模型从未见过的字符组合，最终都可以被分解成一系列已知的字节级 token。
概率最优序列: 在分词时，Unigram 模型会尝试在所有可能的子词分割方式中，找到一个概率最高的序列。如果一个词或片段在词汇表里没有对应的高阶子词 (比如整个单词)，它会继续被拆分成更小的子词，最极端的情况下会拆成单个字节。
因此，由于总能将输入文本分解为词汇表中已存在的（字节级）token，理论上就不存在无法表示的片段，自然也就不需要 [UNK] 标记来表示“未知”。

# 2.2 扩充 BERT 并在 HoC 任务微调

## Step 1：分词器训练方法、超参与词表规模

| 项 | 设定 |
| --- | --- |
| 分词器 | `tokenizers` 库 `BertWordPieceTokenizer` |
| 语料 | `pubmed_sampled_corpus.jsonline` |
| 主要超参 | `vocab_size = 30000` ， `min_frequency = 3` ， `lowercase = True` |
| 特殊符号 | `[PAD]`，`[UNK]`，`[CLS]`，`[SEP]`，`[MASK]` |
| 所得词表大小 | **30,522** |

## Step 2: Domain-specific Token Selection

### Selection Strategy

1. 用第 1 步训练好的医学 WordPiece 分词器对 "pubmed_sampled_corpus.jsonline" 全语料做一次编码；
2. 针对每个子词计算出现频次；
3. 过滤掉已在 `bert-base-uncased` 词表中的子词，得到 OOV 候选；
4. 按频次从高到低取前 5,000 个子词并加入词表，写入 `added_tokens.json` 。

### Sample 50 Tokens

(Randomly sampled from the 5,000 new tokens)

```
['shim', 'kt', '682', '612', 'yt', 'tsa', 'iz', 'homology', 'osmotic',
 '##906', 'cns', '##248', 'atpase', '482', '##n4', 'fistula', 'glut',
 '##505', '##obs', 'optimized', 'diets', 'mj', 'ard', '1471', 'cef', 'gh',
 'hwang', '##this', '##cv', 'sav', '##f3', '724', 'p38', 'pw', '447',
 'plasmodium', 'microstructure', 'dic', 'fibro', '##141', '713', 'ica',
 '##arch', 'y1', '##acs', 'aortic', '##pk', '##ression', 'amssymb',
 'correlations']
```

**观察**

- **生化缩写与蛋白名**：`atpase`， `glut`， `p38`， `fibro`， `plasmodium` — 高频专业词汇，常见于医学论文。
- **数字-前缀 Token**：`682`， `612`， `##906`， `724`， `1471`， `##248`， `##505`， `##141` — 可能代表基因位点、化合物编号、病历代码等。
- **缩写/首字母串**：`cns`（central nervous system），`kt`， `yt`， `tsa`， `ica`， `pw`， `gh`， `mj`， `ard`， `y1` — 常见的研究方法或试剂缩写。
- **学术拼写变体或非英语字根**：`hwang`， `sav`， `dic`， `shim` — 可能为人名、拉丁语或转录词。

这些 token 在原始 BERT 词表中不存在，却在医学语料中出现频繁；将其并入词表可为下游医学文本任务提供更精准的表征。

# Step 3: HoC 样例分词结果与长度比较

### 句子、分词器及主要差异

```
Sample 1:
Text: METHOD In this study , we examined the interactions between T and retinoic acid ( RA ) in cell growth of human prostate carcinoma cells , LNCaP , and their rela
tionship with the expression of RAR and epidermal growth factor receptor ( EGF-R ) .
Original BERT tokens (57): ['method', 'in', 'this', 'study', ',', 'we', 'examined', 'the', 'interactions', 'between', 't', 'and', 're', '##tino', '##ic', 'acid', '(',
 'ra', ')', 'in', 'cell', 'growth', 'of', 'human', 'prostate', 'car', '##cino', '##ma', 'cells', ',', 'l', '##nca', '##p', ',', 'and', 'their', 'relationship', 'with'
, 'the', 'expression', 'of', 'ra', '##r', 'and', 'ep', '##ider', '##mal', 'growth', 'factor', 'receptor', '(', 'e', '##gf', '-', 'r', ')', '.']
Expanded BERT tokens (80): ['meth', 'od', 'in', 'thi', 's', 'stu', 'dy', ',', 'we', 'e', 'xa', 'mined', 'the', 'inte', 'rac', 'ti', '##ons', 'be', 'tween', 't', 'a',
'nd', 'retin', 'oi', 'c', 'acid', '(', 'ra', ')', 'in', 'cel', 'l', 'gro', 'wt', 'h', 'of', 'human', 'pros', 'tat', 'e', 'carcinoma', 'cel', 'ls', ',', 'ln', 'cap', '
,', 'a', 'nd', 'th', 'ei', 'r', 'relat', 'ions', '##hip', 'with', 'the', 'exp', 'r', 'ess', 'ion', 'of', 'rar', 'a', 'nd', 'epidermal', 'gro', 'wt', 'h', 'fac', 'tor'
, 're', 'cep', 'tor', '(', 'egf', '-', 'r', ')', '.']

Sample 2:
Text: Here we demonstrate that CDA-2 and its main component phenylacetylglutamine ( PG ) reduce the metastatic lung tumor growth , and increases survival time after i
noculation with Lewis lung carcinoma ( LLC ) cells in a dose-dependent manner in C57BL6 mice .
Original BERT tokens (63): ['here', 'we', 'demonstrate', 'that', 'cd', '##a', '-', '2', 'and', 'its', 'main', 'component', 'ph', '##en', '##yla', '##ce', '##ty', '##l
', '##gl', '##uta', '##mine', '(', 'pg', ')', 'reduce', 'the', 'meta', '##static', 'lung', 'tumor', 'growth', ',', 'and', 'increases', 'survival', 'time', 'after', 'i
n', '##oc', '##ulation', 'with', 'lewis', 'lung', 'car', '##cino', '##ma', '(', 'llc', ')', 'cells', 'in', 'a', 'dose', '-', 'dependent', 'manner', 'in', 'c', '##57',
 '##bl', '##6', 'mice', '.']
Expanded BERT tokens (76): ['h', 'ere', 'we', 'demon', 'stra', 'te', 'that', 'cd', '##a', '-', '2', 'a', 'nd', 'its', 'main', 'comp', 'on', 'ent', 'phenyl', 'acetyl',
 'glutamine', '(', 'pg', ')', 'r', 'educ', 'e', 'the', 'metastatic', 'lun', 'g', 'tumor', 'gro', 'wt', 'h', ',', 'a', 'nd', 'in', 'cre', 'a', 'ses', 'sur', 'viv', 'al
', 'time', 'after', 'inoculation', 'with', 'l', 'ew', 'is', 'lun', 'g', 'carcinoma', '(', 'llc', ')', 'cel', 'ls', 'in', 'a', 'dose', '-', 'dep', 'e', 'nd', 'ent', 'm
a', 'nn', 'er', 'in', 'c57bl', '6', 'mice', '.']

Sample 3:
Text: For statistical analysis , variables analyzed were categorized as pT1 , 2 vs pT3 , 4 ; Fuhrman grade 1 , 2 vs 3 , 4 ; tumor size < 7 cm vs >or= 7cm ; tumor necr
osis vs no tumor necrosis ; microvascular invasion of sinus fat vs no invasion .
Original BERT tokens (64): ['for', 'statistical', 'analysis', ',', 'variables', 'analyzed', 'were', 'categorized', 'as', 'pt', '##1', ',', '2', 'vs', 'pt', '##3', ','
, '4', ';', 'fu', '##hr', '##man', 'grade', '1', ',', '2', 'vs', '3', ',', '4', ';', 'tumor', 'size', '<', '7', 'cm', 'vs', '>', 'or', '=', '7', '##cm', ';', 'tumor',
 'nec', '##rosis', 'vs', 'no', 'tumor', 'nec', '##rosis', ';', 'micro', '##vas', '##cular', 'invasion', 'of', 'sin', '##us', 'fat', 'vs', 'no', 'invasion', '.']
Expanded BERT tokens (79): ['fo', 'r', 'statistic', 'al', 'a', 'nal', 'ys', 'is', ',', 'vari', 'abl', 'es', 'a', 'nal', 'y', '##zed', 'wer', 'e', 'categor', 'iz', 'ed
', 'as', 'p', 't1', ',', '2', 'vs', 'p', 't3', ',', '4', ';', 'fu', '##hr', '##man', 'gra', 'de', '1', ',', '2', 'vs', '3', ',', '4', ';', 'tumor', 's', 'iz', 'e', '<
', '7', 'cm', 'vs', '>', 'or', '=', '7', '##cm', ';', 'tumor', 'necrosis', 'vs', 'no', 'tumor', 'necrosis', ';', 'microvascular', 'inv', 'asi', 'on', 'of', 'sinus', '
fat', 'vs', 'no', 'inv', 'asi', 'on', '.']

Calculating average token count on train set...
Avg token count with original tokenizer: 67.54
Avg token count with expanded tokenizer: 92.55
```

| 句子 | 原始 BERT tokenizer | 扩展 tokenizer | 主要差异 |
|---|---|---|---|
| **Sample 1** (METHOD … | 57 tokens | 80 tokens | 原模型将 `retinoic` / `carcinoma` / `epidermal` 等拆为「词干 + ##后缀」，数字缩写 `LNCaP` / `EGF-R` |

| 句子 | 原始 BERT tokenizer | 扩展 tokenizer | 主要差异 |
| --- | --- | --- | --- |
| EGF-R ) .) | | | 被进一步拆或部分落入 `[UNK]`；扩展词表学到了 `carcinoma`，`epidermal`，`lncap`，`egf` 等专有词，使其保持为整词，却把常规英语词 ( `method` → `meth` + `od` ) 再拆，从而总 token 数增加。 |
| **Sample 2** (Here … C57BL6 mice .) | 63 tokens | 76 tokens | 新词表包含 `phenyl`，`acetyl`，`glutamine`，`carcinoma`，`c57bl` 等医学/数字片段，避免 `[UNK]`；同时常规词 ( `demonstrate`，`component` …) 被拆为 2-3 个子词。 |
| **Sample 3** (For statistical analysis … no invasion .) | 64 tokens | 79 tokens | 新 tokenizer 把 `tumor necrosis`，`microvascular`，`statistic`，`categoriz`，`invasion` 等细粒度拆分；数字组合 `t1` / `t3` / `7cm` 成独立 token，整体长度上升。 |

### 训练集整体平均长度

```
HoC train 平均（原始） : 67.54
HoC train 平均（扩展） : 92.55   (+25 tokens)
```

### 解释

医学专用子词得到整词表示: lncap, phenyl, glutamine, carcinoma, microvascular, c57bl 等均命中新增的 5,000 词，无需拆为字符或 [UNK]。
常规英语词粒度更细: 扩展词表大量加入数字/缩写前缀，占用了词表容量；BERT 的最长匹配规则于是把部分英语词又拆成 2–3 片段（例如 method→meth+od、study→stu+dy）。

## 4. 新增参数量与初始化方式

### 指标数值与说明

| 指标 | 数值与说明 |
| --- | --- |
| 原始词表大小 | 30,522 |
| 扩展词表大小 | 35,522 |

| 指标 | 数值与说明 |
| --- | --- |
| 新增 token 数 | 5,000 |
| Embedding 维度 | 768 |
| 新增参数总量 | **5,000 × 768 = 3,840,000**<br>(仅体现在 `embeddings.word_embeddings.weight` 新增的 5,000 行) |

### 初始化方式

HuggingFace 在 `model.resize_token_embeddings()` 内置 **mean-resizing** 策略：

- 计算旧嵌入矩阵行向量的 **均值** $\mu$ 与 **协方差** $\Sigma$；
- 对每个新增 token 的向量从 $\mathcal{N}(\mu, \Sigma)$ **采样**；
- 从而保持新向量总体分布与原嵌入一致，避免随机大偏移。

## (5) HoC 微调结果对比与性能分析（15 Credits）

| 模型 | 词表 | Accuracy | Macro-F1 | Micro-F1 | 评价集 |
| --- | --- | --- | --- | --- | --- |
| **BERT（上次作业）** | 原始 30 k | **0.7500** | **0.6119** | **0.7500** | Test |
| **BERT-Expanded**<br>(本次作业) | 30 k + 5 k | **0.4405** | **0.1716** | **0.4405** | Test |

### 可能原因

| # | 解释 | 影响 |
| --- | --- | --- |
| **1**<br>**新嵌入随机初始化** | 5 000 × 768 = 3.84 M<br>新参数仅用截断正态分布初始化；HoC 训练句子不足 2 万，8 epoch<br>难以充分训练新向量。 | 模型早期“忘记”预训练知识，整体性能退化。 |
| **2 词表碎片化** | 新 token 大量为数字前缀（`##000`，`##101` …），常规英语词被再次拆分（`method → meth + od`），文本粒度过细。 | 平均句长 +37 %，128-token 截断下关键信息被裁掉。 |
| **3**<br>**子词语义共享缺失** | 原词表 `car + ##cinoma` 共享前缀；<br>扩展词表把 `carcinoma` 视为独立随机向量。 | 形态迁移被削弱，<br>低频类别学习困难。 |

| # | 解释 | 影响 |
| --- | --- | --- |
| **4 超参未调优** | 序列变长仍用 `batch=32 / lr=2e-5`；无 warm-up。 | grad norm 波动大（≥ 24），收敛慢且不稳定。 |
| **5 标签设置差异** | 本脚本 `num_labels=11`（HoC coarse 11 类），而旧成绩表为 37 类。 | 任务粒度不同 → 直接对比有偏差，但下降趋势仍明显。 |