

Tokenization 的近期技術

補充 Jurafsky & Martin Chap. 2

Week 3 · 補充 A

我們先來看看 J&M 的 tokenization 章節/slides

這份補充要補什麼

J&M ch. 2 已有的	這裡補上的
BPE 的學習與編碼演算法	WordPiece 與 Unigram LM：另外兩套邏輯完全不同的做法
「跑一次學到的 merges」	學習 $\neq$ 推論：同一個詞表可以有不同切法，貪婪不等於最佳
展示 GPT-2 的 pre-tokenizer 正則式	那個正則式本身就是一份語言學承諾清單
SuperBPE 一句帶過	跨空白合併，以及這對中文意味著什麼
「未來會講神經模型」	不用 tokenizer 的路線：byte-level、動態切分、學習式 chunking
詞表大小是給定的	詞表大小是超參數，而且有 scaling law
—	詞表的病態：訓練不充分的 token、機率偏誤、數字與字母計數
—	評估指標，以及它們跟下游表現的關係有多弱

一、三個演算法

課本給了一個，實務上有三個

BPE (Sennrich et al. 2016)

- 由下而上，合併最高頻的相鄰對
- 目標：壓縮

WordPiece (Schuster & Nakajima 2012; BERT)

- 由下而上，合併最能提升語料似然的對

$$\text{score}(a, b) = \frac{P(ab)}{P(a)P(b)}$$

Unigram LM (Kudo 2018)

- 由上而下，先給一個很大的候選集，用 EM 估計每個單元的機率
- 反覆丟掉「拿掉之後似然損失最小」的單元，直到縮到目標大小

三者的關鍵差異

	BPE	WordPiece	Unigram LM
方向	由下而上合併	由下而上合併	由上而下剪枝
判準	相鄰對頻率	似然增益 (≈ 互資訊)	語料似然損失
模型	無	隱含	顯式： $P(\mathbf{x}) = \prod_i p(x_i)$
切分	依 merge 順序，唯一	最長匹配，唯一	Viterbi，且可取樣
歧義	沒有這個概念	沒有這個概念	內建：一個字串有切分的機率分布

WordPiece 的判準其實就是互資訊。結構主義者用互資訊找語素邊界，已經用了六十年；差別只在於是誰在算。

SentencePiece 不是演算法

A recurring confusion worth heading off. SentencePiece is an *implementation* (Kudo & Richardson 2018) that can run either BPE or Unigram LM. Its contribution is engineering, not algorithmic:

- 直接吃原始文字，不預設有空白分詞
- 把空白當成一個普通字元 ，因此可逆
- 對沒有空白的語言（中文、日文、泰文）特別重要

承認單位不唯一：subword regularization

Subword regularization (Kudo 2018)：訓練時從切分分布中取樣，同一個句子每個 epoch 切法不同。

BPE-dropout (Provilkov et al. 2020) $\triangle$ ：把同樣的想法移植到 BPE—以機率 p 隨機跳過某些 merge。

效果是資料增強，以及對拼寫變異的穩健性。但語言學上的意涵更大：

這兩個方法等於承認：單位不是唯一的，
而讓模型看到多種切分反而更好。

字本位／詞本位之爭的一個工程版答案：兩個都要，隨機給。

二、學習 $\neq$ 推論

「用 tokenizer」其實是兩個決定

課本把 tokenizer 當成一件事。它是兩件：

1. 學一套詞表（merge 表 / 單元集合）
2. 用那套詞表切一個新字串

BPE 的編碼器依學到的順序套用 merge。那是貪婪的，不保證能得到「這個詞表所能給出的最短切分」。

實測：同一個詞表，貪婪不是最佳

`cl100k_base`，同一份詞表，比較貪婪編碼與最短路徑切分：

DT0bcrypt	greedy=3	['DT', '0b', 'crypt']	optimal=2
ysqlibecause	greedy=3	['ysql', 'ib', 'ecause']	optimal=2
intptromez	greedy=4	['int', 'pt', 'rome', 'z']	optimal=2
charAtdims	greedy=4	['char', 'A', 'td', 'ims']	optimal=2

這些字串是我把詞表裡的兩個單元串起來造的，不是自然文本，所以不要拿去宣稱「BPE 平均浪費多少 token」。它只證明一件事：編碼器是一個獨立的設計選擇。

這條線上的研究

- PathPiece (Schmidt et al. 2024) ⚠️：把切分當成最短路徑問題，直接找單元數最少的切分。
- "Greed is all you need?" (Uzan et al. 2024) ⚠️：系統比較推論時的各種切分策略。

一個反直覺的結論值得注意：最短切分不必然帶來最好的下游表現。如果壓縮就是目的，那答案很單純；既然不是，那 tokenizer 的目標函數到底是什麼，目前沒有共識。

三、pre-tokenization 比較是一種語言學承諾

BPE 永遠不跨越 pre-token 邊界

所以那個正則式就是理論本身。逐項讀 `cl100k_base` 的：

```
r""""(?:'s|'t|'re|'ve|'m|'ll|'d)
| [^\r\n\p{L}\p{N}]?\p{L}+
|\p{N}{1,3}
| ?[^\s\p{L}\p{N}]+[\r\n]*
|\s*[\r\n]+|\s+(?!\\S)|\s+""""
```

← 手寫的附著詞清單：形態學

← 一個「詞」，可帶前導符號

← 數字最多三位一組：關於數詞的承諾

這不是工程細節。這是有人坐下來，決定了英語的附著詞有哪些、數字該怎麼分組、以及什麼算一個詞。

數字那一行的後果

$\text{p}\{N\}\{1,3\}$ 讓 1234567 被切成 123|456|7 —— 由左而右分組。但直式加法是由右而左進位的。

Llama-3 改成由右而左分組（1|234|567），據說因此改善了算術表現。⚠

可以接著追的方向：tokenization 與算術能力的關係、以及「strawberry 有幾個 r」這類字母計數問題——模型看不到字母，它看到的是 str|aw|berry。

這是一個清楚的例子：表徵的粒度決定了哪些任務是困難的。

$\backslash p\{L\}+$ 對中文做了什麼

英文：pre-token = 空白分隔的詞，BPE 只在詞內作用，詞界被保證。

中文： $\backslash p\{L\}+$ 回傳整串連續漢字，常常是整整一個子句。BPE 可以在裡面任意合併，詞界不存在，也不被保證。

中文從第一步就沒有 pre-tokenization 這道保險。這件事在 2025 年變得有趣起來。

SuperBPE：英語終於遇到中文的問題

Liu et al. (2025) ⚠ 讓 BPE 進行第二階段、跨越空白的合併，學出 *superwords* (`by the way` 、 `in order to`)。結果是：同樣詞表大小下序列更短、下游更好。

語言學上的意涵比數字有趣：

- 英語的空白 pre-tokenization 一直被當成「免費的正確性」。
SuperBPE 說那其實是一個限制，拿掉會更好。
- 中文從來沒有這個限制。中文一直都是 SuperBPE。

還沒解答的問題：中文長期「沒有詞界保護」，究竟是缺陷還是優勢？
目前沒有清楚的答案，這是個好的期末題目。

四、不要 tokenizer 的路線

目前的版圖

系統	單位	關鍵想法
ByT5 (Xue et al. 2022)	UTF-8 byte	直接吃位元組，序列長但無詞表偏誤
CANINE (Clark et al. 2022)	character	下採樣 + 局部注意力
Charformer / GBST (Tay et al. 2022) ⚠	軟性子詞塊	用可微分模組學出分塊權重
MegaByte (Yu et al. 2023) ⚠	固定長度 patch	階層式：patch 內 + patch 間
MambaByte (2024) ⚠	byte	用 SSM 抵銷長序列成本
BLT (Pagnoni et al. 2024/25) ⚠	動態 patch	邊界由下一位元組的熵決定
H-Net (Hwang et al. 2025) ⚠	學出來的 chunk	端到端可微分的階層切分

共同動機：詞表是一個固定的、外生的、不可微分的東西，而深度學習其他部分早就不是了。

BLT 的切分規則

用一個小的位元組級模型估計 $H(b_t \mid b_{<t})$ ，在熵升高的位置切開：

- 可預測的區段 → 併成一個大 patch → 省算力
- 不可預測的位置 → 開新 patch → 花算力

這樣一來，計算量會自動流到資訊密度高的地方。the 幾乎不花錢，一個沒見過的專名很貴。

這個規則，語言學家在 1955 年就寫過

Harris (1955) 的 *successor variety* :

在一個語素內部，下一個音位的選項很少；
到了邊界，選項突然變多。邊界 = 後繼變異度的峰值。

Saffran, Aslin & Newport (1996)：八個月大的嬰兒用 transitional probability 的低谷切分連續語流。

BLT 的熵準則，與這兩者是同一個直覺的不同寫法。

但有一個真正的差別

Harris / Saffran 在語料上一次算完，產出一份型（type）層次的單位清單。同一個字串永遠同一種切法

BLT / H-Net 在推論時逐位元組決定 產出符（token）層次的切分。同一個字串在不同語境切法不同，這個差別不是小事。

上下文敏感的切分 正是構式語法一直要求、而 BPE 給不出來的性質。補充 B 會回到這一點。

五、詞表是超參數

詞表大小有 scaling law

Tao et al. (2024) ⚠️：在固定計算預算下，最適詞表大小隨模型規模成長，而目前多數模型的詞表偏小。

語言學上的意涵：

「要不要把 越走越快 收進詞表」不是語言學判斷，
而是算力預算的函數。

詞彙化程度在這裡被重新定義成一個最佳化參數。

相關的工程決策

- **Tokenizer choice ablations** (Ali et al. 2024) ⚠️：
在其他條件相同下換 tokenizer，下游差異有多大。
- **詞表擴充**：Chinese-LLaMA 等為中文補詞表的工作，
在原詞表上補中文單元再續訓。
- **Zero-shot tokenizer transfer** (Minixhofer et al. 2024) ⚠️：
訓練一個 hypernetwork，把模型移植到新詞表而不需重訓。

這些都在回答同一個問題：****詞表可不可以事後更換？****如果可以，那「詞表即語言理論」的說法要打折；如果不行，那模型出廠時就把某種語言觀焊死了。

六、詞表的病理學

訓練不充分的 token

有些 token 進了詞表，卻幾乎沒在訓練資料中被好好看過。把它們餵進去，模型的行為會變得異常。早期在 GPT-2/3 上的 `SolidGoldMagikarp` 現象是最有名的例子；Land & Bartolo (2024) $\triangle$ 提出了系統性的偵測方法。

成因通常是：詞表訓練語料 $\neq$ 模型訓練語料。

詞表看過的垃圾，模型後來被清掉了，位置卻留著。

直接看詞表裡有什麼

`cl100k_base` 最長的純漢字 token：

生命周期函数 · 监听页面 · 不能为空 · 微软雅黑 · 删除成功 · 请输入验证码

`o200k_base` 最長的純漢字 token：

微信公众号天天中彩票 · 给主人留下些什么吧 · 无码一区二区三区 · 大发快三

前者是中文前端部落格，後者是博弈與成人網站的套版文字。詞表是訓練語料的化石紀錄，而且是可以直接讀的那種。

一個具體的後果

`o200k_base` 中，這些地名各自佔一個 token：

北京 (rank 19340)	香港 (23485)	上海 (53339)
台灣 (75626)	重慶 (6554)	廣州 (132426)

這些不是：

台灣 / 臺灣 (各 2 tokens)	重慶 (3 tokens)	廣州 (2 tokens)
高雄 · 新竹 · 台北 · 臺北	(皆 2 tokens)	

繁體寫法的地名幾乎一個都沒進詞表。這不需要統計也看得懂，而且它就在你每天用的模型裡。

機率也會被 tokenization 扭曲

同一個字串可以有多種切分，但模型只對它實際看到的那一種給機率。

於是 $P(\text{字串})$ 與 $\sum_{\text{切分}} P(\text{切分})$ 不相等。

後果：

- 比較不同字串的機率時會有系統性偏誤
- 受限解碼（constrained decoding）時，前綴被切錯會讓模型走進死路
→ token healing：退回到前綴邊界重新切
- 精確的位元組層次機率需要額外計算 (Phan et al. 2024) ⚠

對做語言學實驗的人來說，這一點特別要緊：

你用 LLM 算的 surprisal，是 token 的 surprisal，不是詞的。

七、怎麼評估一個 tokenizer

指標

指標	定義	出處 ⚠
fertility	tokens / 參照詞	Rust et al. (2021)
parity	$ t(s_A) / t(s_B) $ ，語義等價句對	Petrov et al. (2023)
compression	bytes / token	通用
Rényi efficiency	$H_\alpha / \log V $ ， $\alpha = 2.5$	Zouhar et al. (2023)
morphological alignment	與語素邊界的一致率	Batsuren et al. (2024)
cognitive plausibility	與人類詞彙處理資料的相關	Beinborn & Pinter (2023)

但要小心

這些是內在指標。它們與下游表現的相關，文獻裡的結果並不一致：

在某些設定下 Rényi efficiency 預測 BLEU 不錯，在別的設定下就不行。

原因不難想：壓縮率高不代表單位切得對，而「切得對」本身又依任務而定。

這與 Week 2 講斷詞標準時的處境完全一樣：

沒有一個獨立於任務的「正確切分」。

差別只在於，這次連標準都沒有人寫下來。

附帶一提：「token」已經不只是語言的事

影像：VQ-VAE / VQ-GAN 把影像量化成離散碼本索引。

語音：neural audio codec（EnCodec、SoundStream 類）把波形變成 token。

這樣一來，同一個 Transformer 可以同時吃文字、影像、語音 token。

這帶來一個術語上的問題：

「token」現在指的是「任何被離散化後餵進序列模型的東西」。

它已經不是語言學單位了。討論多模態模型時，語言學的直覺沒辦法直接套用。

小結

1. BPE 只是三種演算法之一，而且是唯一沒有歧義概念的那個。
2. 學詞表與用詞表是兩件事；貪婪編碼不是最佳編碼。
3. pre-tokenization 的正則式是真正的語言學承諾清單。
4. 動態切分正在把「單位」從型層次搬到符層次。
5. 詞表大小是超參數；詞表內容是語料的化石。
6. 內在指標不能替代下游評估，而下游評估又依任務而定。

讀什麼（請自行查核書目）

基礎：Sennrich et al. 2016 (BPE) · Kudo 2018 (Unigram LM、subword regularization)

· Kudo & Richardson 2018 (SentencePiece)

近期：Schmidt et al. 2024 (Tokenization Is More Than Compression)

· Liu et al. 2025 (SuperBPE) · Pagnoni et al. 2024/25 (BLT) · Tao et al. 2024 (vocab scaling)

評估：Rust et al. 2021 · Petrov et al. 2023 · Zouhar et al. 2023

· Beinborn & Pinter 2023

語言學背景：Harris 1955 · Saffran, Aslin & Newport 1996