NPTEL ONLINE CERTIFICATION COURSES

DEEP LEARNING FOR NATURAL LANGUAGE PROCESSING

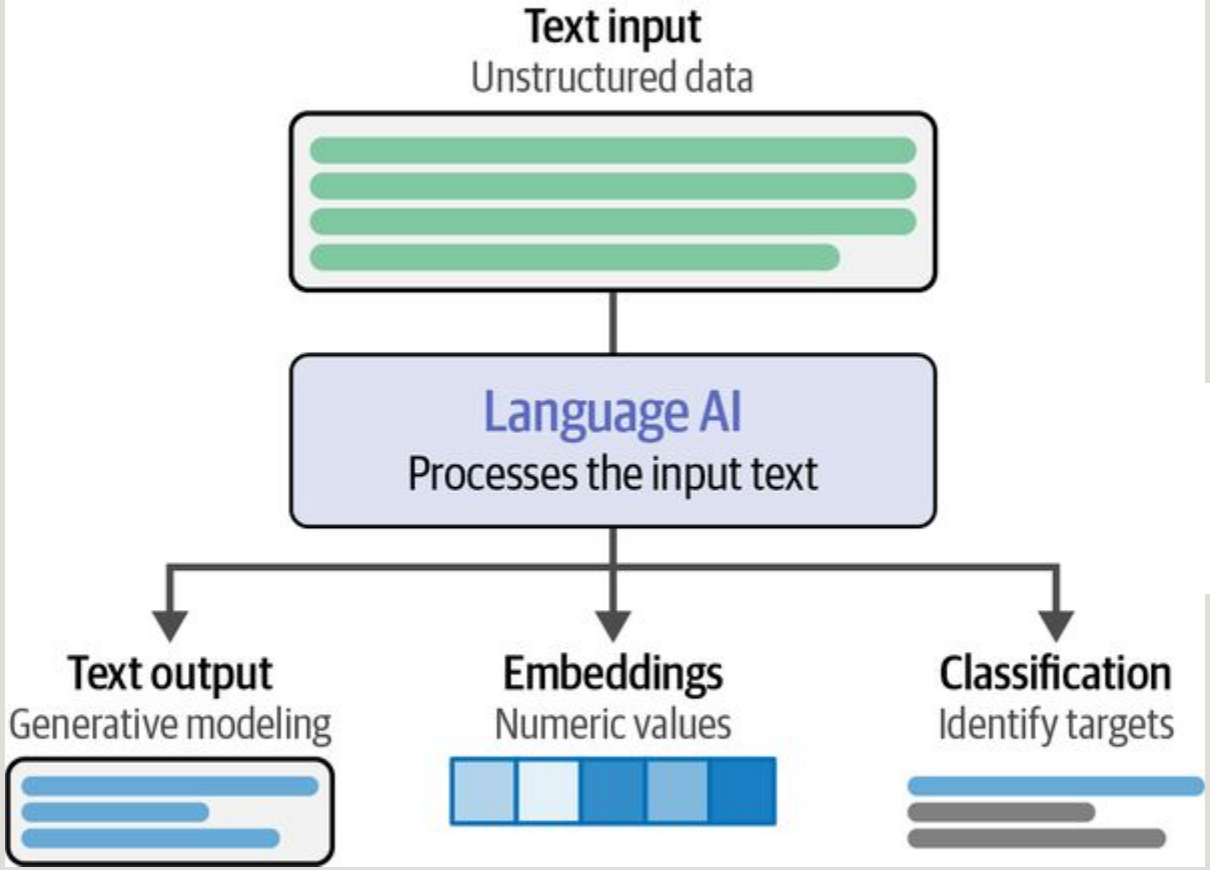
Lecture 02 : Text Processing Basics, Tokenization

PROF.PAWAN GOYAL
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

CONCEPTS COVERED

- Processing Text Input
- Whitespace Tokenizer
- Byte-Pair Encoding

Processing Text Input



For any NLP application, the input text needs to be processed first.

The first step in processing text is *tokenization*.

Input text: students opened their books

Input token IDs: 11 298 34 567

Source: Alammar, J., & Grootendorst, M. (2024). Hands-On Large Language Models. O'Reilly.

Tokenization: How many words in a sentence?

they lay back on the San Francisco grass and looked at the stars and their

Type: an element of the vocabulary.

Token: an instance of that type in running text.

How many?

- 15 tokens
- 13 types

Source: Speech and Language Processing, 3rd Ed.

How many words in a corpus?

N = number of tokens

V = vocabulary = set of types, **|*V*|** is size of vocabulary

Heaps Law = Herdan's Law = $|V| = kN^{\beta}$ where often $.67 < \beta < .75$

i.e., vocabulary size grows with > square root of the number of word tokens

	Tokens = N	Types = |V|
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
COCA	440 million	2 million
Google N-grams	1 trillion	13+ million

Source: Speech and Language Processing, 3rd Ed.

Corpora: Where do the words come from?

Words don't appear out of nowhere!

A text is produced by

- a specific writer(s),
- at a specific time,
- in a specific variety,
- of a specific language,
- for a specific function.

Source: Speech and Language Processing, 3rd Ed.

Corpora vary along dimensions like

- **Language**: 7097 languages in the world
- **Variety**, like African American Language varieties.

  Twitter posts might include forms like "*iont" (I don't)*
- **Code switching**, e.g., Spanish/English, Hindi/English:

  S/E: Por primera vez veo a @username actually being hateful! It was beautiful:)

  [For the first time I get to see @username actually being hateful! it was beautiful:)]

  H/E: dost tha or rahega ... dont worry

  [“he was and will remain a friend ... don’t worry ”]
- **Genre:** newswire, fiction, scientific articles, Wikipedia
- **Author Demographics**: writer's age, gender, ethnicity

Source: Speech and Language Processing, 3rd Ed.

Whitespace tokenization

Tokens are implied to be *words*

Example:

Input text: students opened their books

Input token IDs: 11 298 34 567

Whitespace tokenizer issues

- *conjunctions:* isn't ⇒ is, n't
- *hyphenated phrases:* prize-winning ⇒ prize, -, winning
- *punctuation:* great movie! ⇒ great, movie, !

(Word tokenizers require lots of specialized rules about how to handle specific inputs)

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

What if a new (or infrequent) word appears?

Out-of-vocabulary (OOV): Words that were seen very rarely during training or not even at all

Closed-vocabulary models: Unable to produce word forms unseen in training data

<UNK> tokens:

- Historically rare word types were replaced with a new word type UNK (unknown) at training time
- At test time, any token that was not part of the model’s vocabulary could then be replaced by UNK
- But you should not generate UNK when generating text
- UNKs don’t give features for novel words that maybe useful anchors of meaning
- In languages other than English, in particular those with more productive morphology, removing rare words is infeasible

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Limitations of <UNK>

We lose lots of information about texts with a lot of rare words / entities

The chapel is sometimes referred to as "Hen Gapel Lligwy" ("hen" being the Welsh word for "old" and "capel" meaning "chapel").

The chapel is sometimes referred to as " Hen <unk> <unk> " (" hen " being the Welsh word for " old " and "<unk> " meaning " chapel ").

Source: https://people.cs.umass.edu/~miyyer/cs685

Maximal Decomposition into Characters

But deciding what counts as a word in Chinese is complex. For example, consider the following sentence:

(2.4) 姚明进入总决赛
"Yao Ming reaches the finals"

As Chen et al. (2017) point out, this could be treated as 3 words ('Chinese Treebank' segmentation):

(2.5) 姚明 进入 总决赛
YaoMing reaches finals

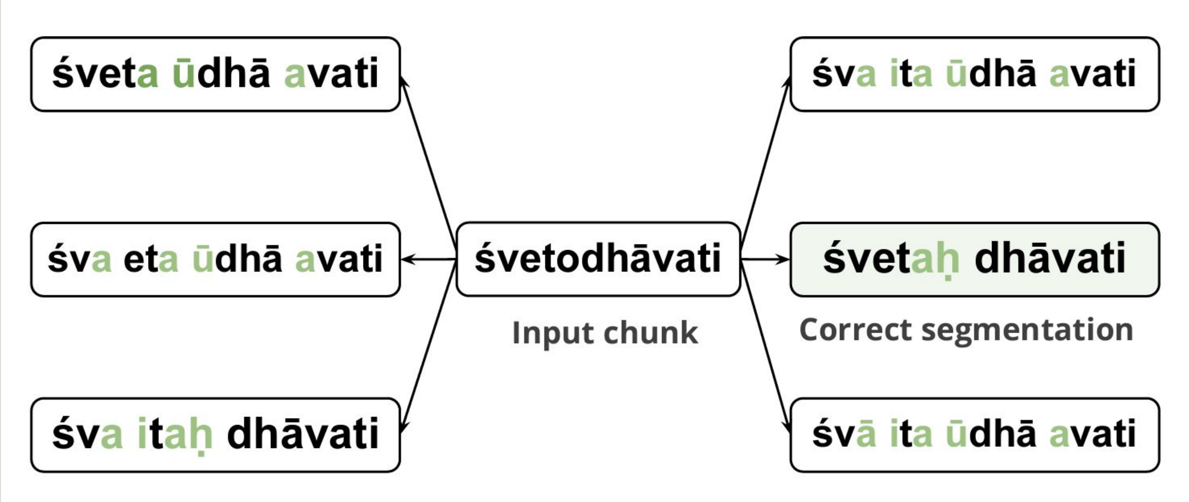
or as 5 words ('Peking University' segmentation):

(2.6) 姚 明 进入 总 决赛
Yao Ming reaches overall finals

Finally, it is possible in Chinese simply to ignore words altogether and use characters as the basic elements, treating the sentence as a series of 7 characters:

(2.7) 姚 明 进 入 总 决 赛
Yao Ming enter enter overall decision game

In fact, for most Chinese NLP tasks it turns out to work better to take characters rather than words as input, since characters are at a reasonable semantic level for most applications, and since most word standards, by contrast, result in a huge vocabulary with large numbers of very rare words (Li et al., 2019).



Challenges due to *sandhi* phenomena for Sanskrit Word Segmentation

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Preprocessing / Text normalization

- **Lemmatization:** determining that two words have the same root, despite their surface differences
  - sang, sung, and sings are forms of sing
- **Stemming**: strip suffixes from the end of the word
- **Sentence segmentation:** Breaking up a text into individual sentences
- **Stopword removal:** Remove commonly used words in a language
  - a, the, is, are
- **Casing:** Lowercase all words or not

With pretrained language models, besides casing, we do none of the other steps

After text normalization, most tokenizers are **irreversible**

we cannot recover the raw text definitively from the tokenized output

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

A redefinition of the notion of tokenization

Due to:

- Scientific results: The impact of sub-word segmentation on machine translation performance in 2016
- Technical requirements: A fixed-size vocabulary for neural language models

…in current NLP, the notion of token and tokenization changed

"Tokenization" is now the task of segmenting a sentence into non-typographically (and non-linguistically) motivated units, which are often smaller than classical tokens, and therefore often called **sub-words**

Typographic units (the "old" tokens) are now often called **"pre-tokens",** and

what used to be called "tokenization" is therefore called **"pre-tokenization"**

- https://github.com/huggingface/tokenizers/tree/main/tokenizers/src/pre_tokenizers

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Subwords are expected to be meaningful units

Subwords can be arbitrary substrings…

…but subwords can be meaning-bearing units like the morphemes -est or -er

- A **morpheme** is the smallest meaning-bearing unit of a language
  - “unlikeliest” has the morphemes {un-, likely, -est}
- **Morphology** is the study of the way words are built up from morphemes
- **Word forms** are the variations of a word that express different grammatical categories (tense, case, number, gender, etc) and thus help convey the specific meaning and function of the word in a sentence

Unseen word like lower can thus be represented by

some sequence of known subword units, such as {low, er}

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE)

Main idea: Use data to automatically tell what the tokens should be

Token learner

Raw train corpus ⇒ Vocabulary (a set of tokens)

Token segmenter

Raw sentences ⇒ Tokens in the vocabulary

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70
[coined by Gage et al., 1994; adapted to the task of word segmentation by **Sennrich et al., 2016**; see Gallé (2019) for more]

Byte-Pair-Encoding (BPE) – **Token learner**

Raw train corpus ⇒ Vocabulary (a set of tokens)

- Pre-tokenize the corpus in words & append a special end-of-word symbol _ to each word
- Initialize vocabulary with the set of all individual characters
- Choose 2 tokens that are most frequently adjacent (“A”, “B”)
  - Respect word boundaries
- Add a new merged symbol (“AB”) to the vocabulary
- Change the occurrence of the 2 selected tokens with the new merged token in the corpus
- Continues doing this until k merges are done

All k new symbols and initial characters are the final vocabulary

What’s k? Open research question

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) – Example

corpus		**vocabulary**
5	l o w _	_, d, e, i, l, n, o, r, s, t, w
2	l o w e s t _	
6	n e w e r _	
3	w i d e r _	
2	n e w _	

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) – Example

corpus		vocabulary
5	l o w _	_, d, e, i, l, n, o, r, s, t, w, $\boxed{\text{er}}$
2	l o w e s t _	
6	n e w $\boxed{\text{er}}$ _	
3	w i d $\boxed{\text{er}}$ _	
2	n e w _	

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) – Example

corpus	
5	l o w _
2	l o w e s t _
6	n e w $\boxed{\text{er\_}}$
3	w i d $\boxed{\text{er\_}}$
2	n e w _

vocabulary

_, d, e, i, l, n, o, r, s, t, w, er, $\boxed{\text{er\_}}$

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) – Example

corpus		vocabulary
5	l o w _	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne
2	l o w e s t _	
6	ne w er_	
3	w i d er_	
2	ne w _	

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) – Example

merge	**current vocabulary**
(ne, w)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new
(l, o)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo
(lo, w)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low
(new, er_)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low, newer_
(low, _)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low, newer_, low_

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) – **Token segmenter**

Just runs on the test data the merges we have learned from the training data, greedily, in the order we learned them

First we segment each test sentence word into characters

Then we apply the first merge rule

- E.g., replace every instance of “e”, ”r” in the test corpus with “er”

Then the second merge rule

- E.g., replace every instance of “er”, “_” in the test corpus with “er_”

And so on

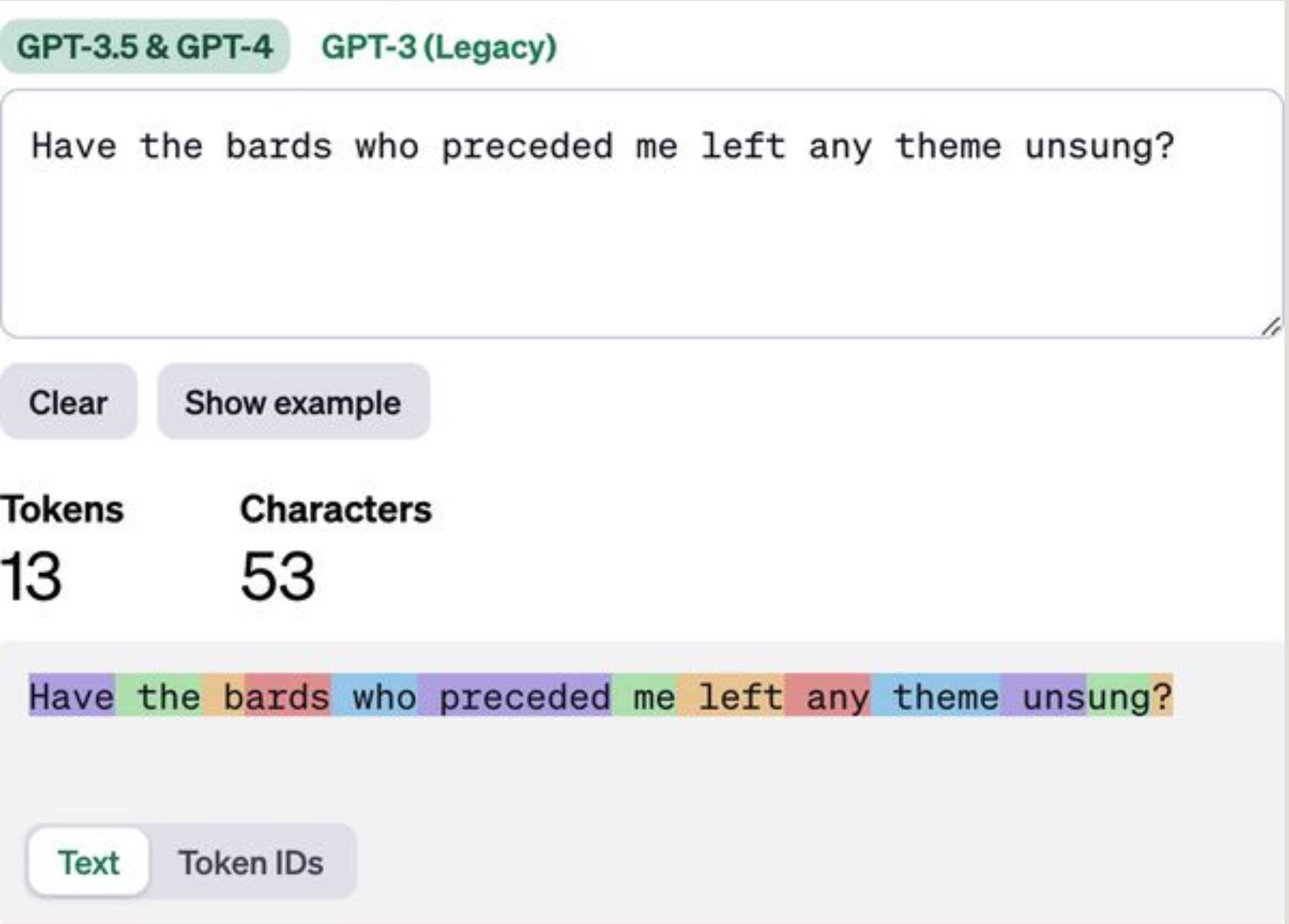
Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) Vocabulary

Model	Tokenizer	Vocabulary Size
BERT base (uncased) [2018]	WordPiece	30,522
BERT base (cased) [2018]	WordPiece	28,996
GPT-2 [2019]	BPE	50,257
Flan-T5 [2022]	SentencePiece	32,100
GPT-4 [2023]	BPE	> 100,000
StarCoder2 [2024]	BPE	49,152
Llama2 [2023]	BPE	32,000

You can play with different tokenizers here: https://tiktokenizer.vercel.app/

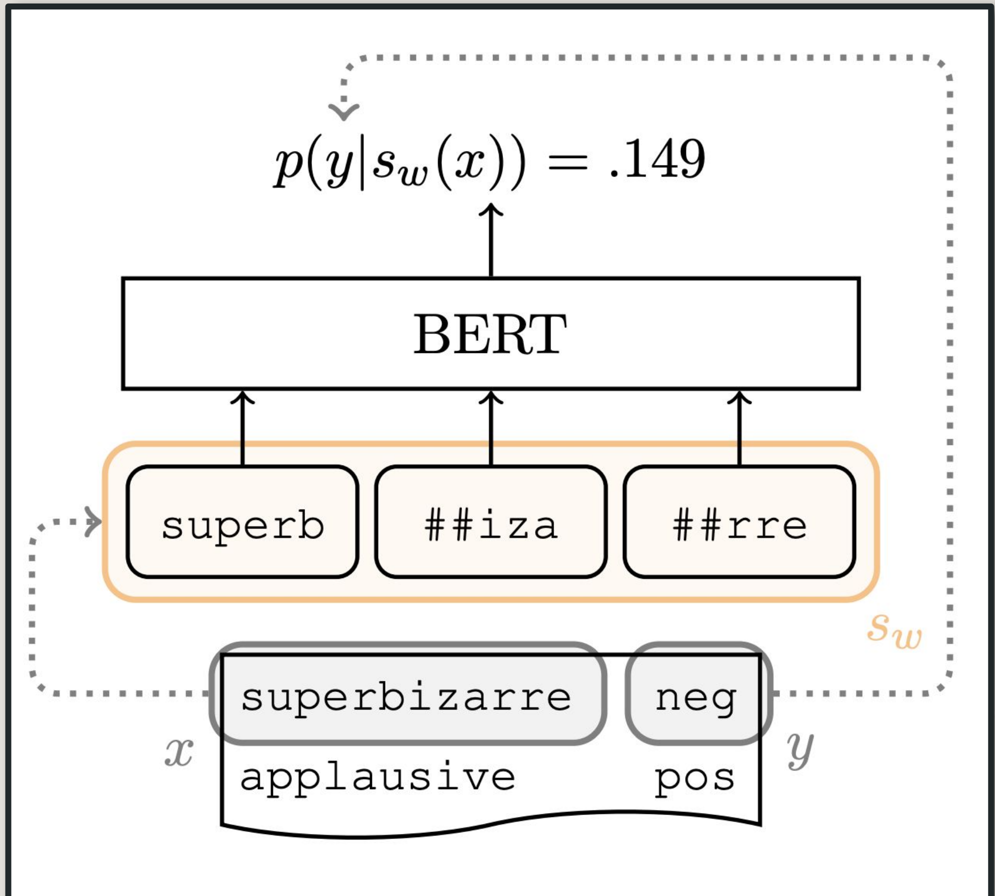
Subwords - Example



Source: Alammar, J., & Grootendorst, M. (2024). Hands-On Large Language Models. O'Reilly.

Byte-Pair-Encoding (BPE) Implications [Hofmann et al., 2021]

BERT thinks the sentiment of "superbizarre" is positive because its tokenization contains the token "superb"



(a) BERT (s_w)

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Byte-Pair-Encoding (BPE) Implications – **Do All languages cost the same?**

[Ahia et al., 2023]

Proprietary models, as GPT-4, are accessible only through **paid APIs**

API cost is measured by the number of tokens processed or generated

Subword tokenizers lead to disproportionate fragmentation rates for different languages and writing scripts

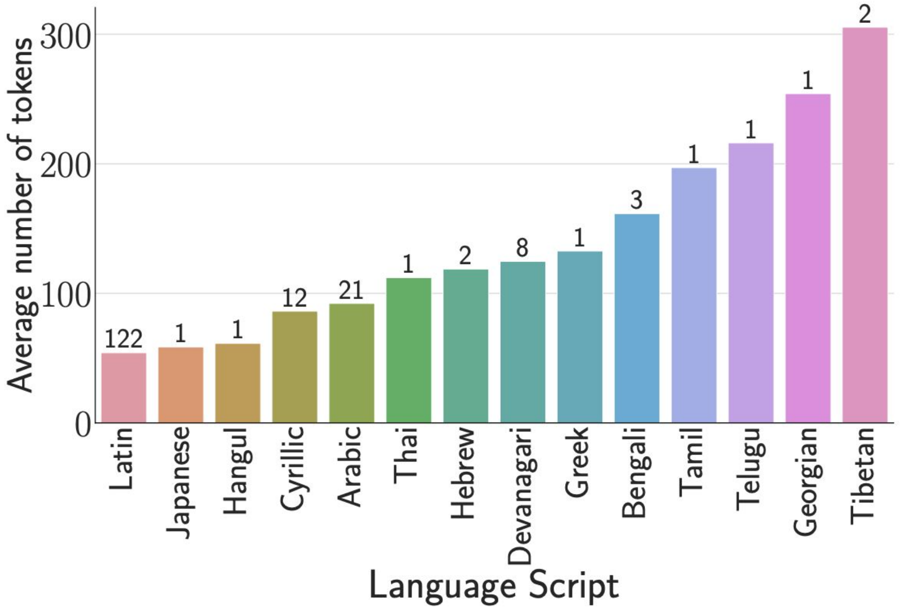


Figure 2: Average number of tokens by script after tokenizing the Flores dataset. The fragmentation rate is lower for Latin script languages and higher for other scripts. Number of languages per language group is indicated at the top of each bar.

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

Other subword encoding schemes

WordPiece (Schuster et al., ICASSP 2012): merge by likelihood as measured by language model, not by frequency

SentencePiece (Kudo et al., 2018): can do subword tokenization without pretokenization (good for languages that don't always separate words w/ spaces), although pretokenization usually improves performance

Source: https://utah-cs6340-nlp.notion.site/Natural-Language-Processing-bd1a2ca290fc44f69556908ad8d25c70

REFERENCES

Daniel Jurafsky and James H. Martin. 2024. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, 3rd edition. Online manuscript released August 20, 2024. https://web.stanford.edu/~jurafsky/slp3. [Chapter 2]

THANK YOU