tejasgokhale.com

CMSC 475/675 Neural Networks

Lecture 11

Addendum:

Tokenization

UMBC

Tokenization

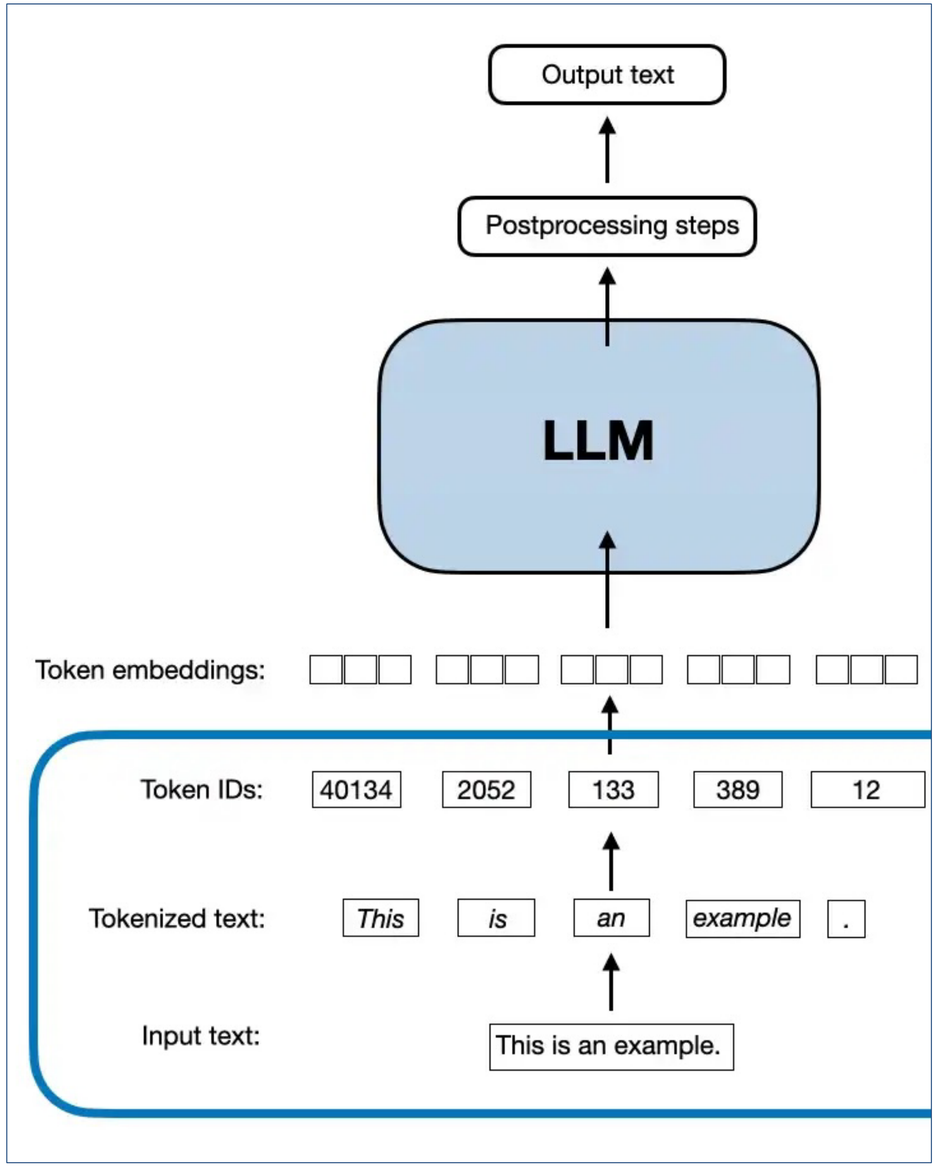
(tokens are units of language)

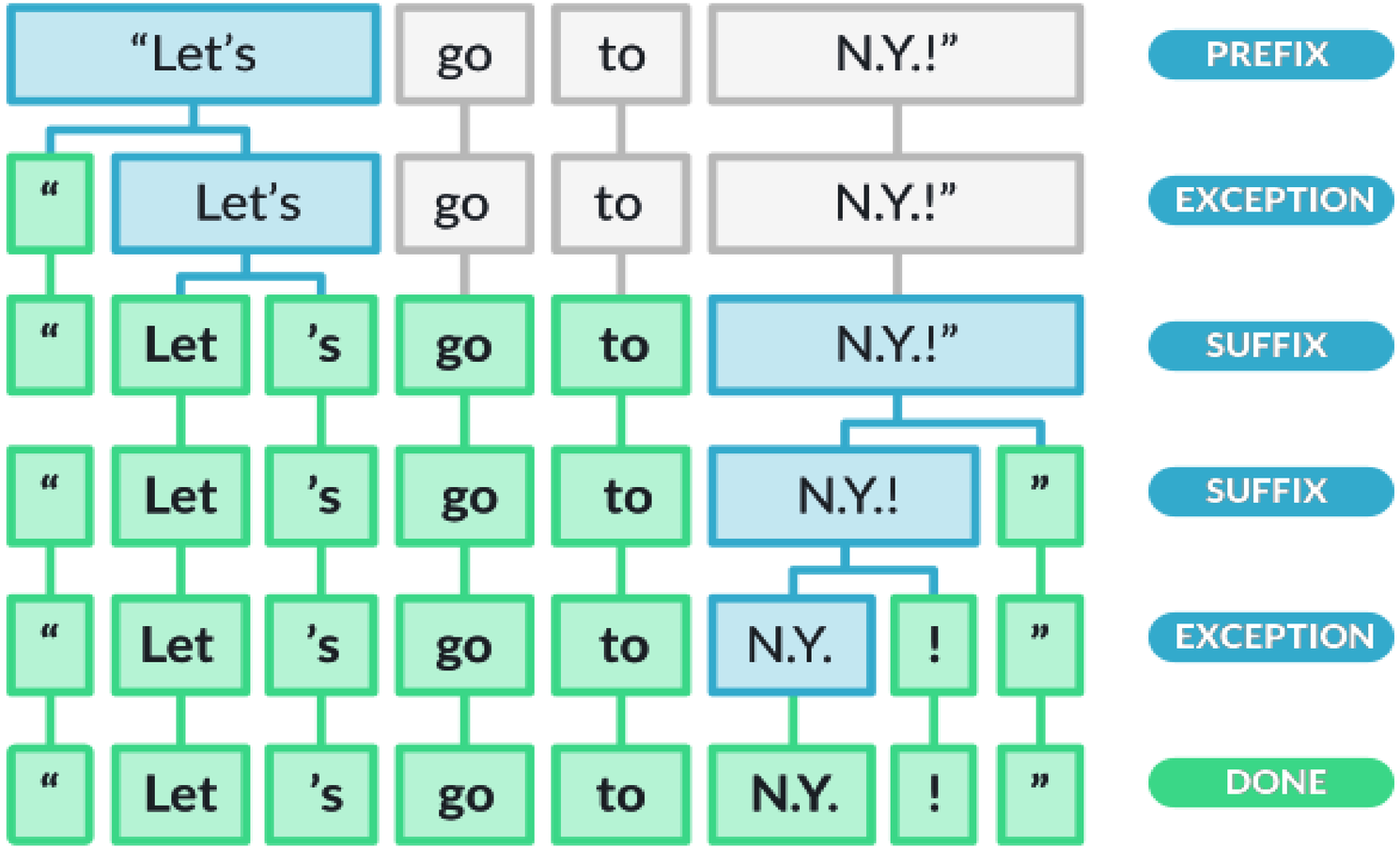
(think “atoms”)

Revisiting Words ... Tokenization

- Text broken into smaller units = tokenization
  - Tokens are discrete text units
  - Different ways to do tokenization (words, morphemes, characters)
- Typical preprocessing step for NLP tasks
- Why tokenize?
  - Text encoding is necessary – better to encode smaller units than large chunks
  - Deal with whitespaces
  - Building up “meaning” in a bottom-up way

Tokenization Example





Tokenization Variants

Paragraph Tokenization

- Break documents into paragraphs
- Rarely used

Sentence Tokenization

- Break documents into sentences

Word Tokenization

- Break documents into words
- Very common

Sub-Word Tokenization

- Break documents into morphemes
- Very common

Character Tokenization

- Break documents into individual characters

Word Tokenization

- Very popular – applied as a preprocessing step in many NLP tasks
- Considers dictionary words and delimiters as “tokens” in the vocab

```
What is the tallest building

➔ “What”, “is”, “the”, “tallest”, “building”, “?”
```

Sub-Word (Morpheme) Tokenization

- Fine-grained than word tokenization
  - Breaks text into words
  - Further breaks words into smaller units based on root, prefix, suffix, etc
  - Based on linguistic rules
- Important for “flective” languages
  - Words have many forms with prefixes and suffixes that change the meaning of the word

```
What is the tallest building

➔ “What”, “is”, “the”, “tall”, “est”, “build”, “ing”, “?”
```

Byte-Pair Encoding (BPE) Tokenization

- Initially developed for compressing text (Gage 1994)
- Modified for use in Large Language Models (GPT etc.)
- It's an iterative algorithm

Byte-Pair Encoding (BPE) Tokenization

- Initial Steps:
  - Begin by creating a unique set of words in corpus
  - Set a desired token vocabulary size: D
    - Example: let's say the corpus has 5 words: "hug", "pug", "pun", "bun", "hugs"
    - The "base vocab" is the set of all characters: *b, g, h, n, p, s, u*

- "Merge"
  - Add new tokens until desired vocab size is reached by learning "merges"
  - At any step during tokenization, search for the most frequent pair of consecutive tokens and merge it to create a new token
    - *Merges* are rules to merge two elements of the existing vocabulary to create a new token
    - The first *merge* creates tokens with 2 characters -- as algo proceeds longer tokens are created

BPE Example

CORPUS: “hug”, “pug”, “pun”, “bun”, “hugs”
Base vocab: *b, g, h, n, p, s, u*

https://huggingface.co/learn/llm-course/en/chapter6/5

- Get Frequencies:

```
("h" "u" "g", 10), ("p" "u" "g", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "u" "g" "s", 5)
```

- Pair (“u”, “g”) is the most frequent (appears 20 times in total)
  - Merge: (“u”, “g”) → “ug”

```
Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug"]
Frequency: ("h" "ug", 10), ("p" "ug", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "ug" "s", 5)
```

- Pair (“u”, “n”) is the most frequent (appears 16 times). Merge to get “un”

```
Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug", "un"]
Corpus: ("h" "ug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("h" "ug" "s", 5)
```

- Pair (“h”, “ug”) is the most frequent …. Merge to get “hug”

```
Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug", "un", "hug"]
Corpus: ("hug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("hug" "s", 5)
```

Back to NN for LM