Natural Language Processing

Sub-word Tokenization

Outline

Recap

Word Segmentation

Sub-word Tokenization

News

The Prompt Report: A Systematic Survey of Prompting Techniques, https://arxiv.org/pdf/2406.06608

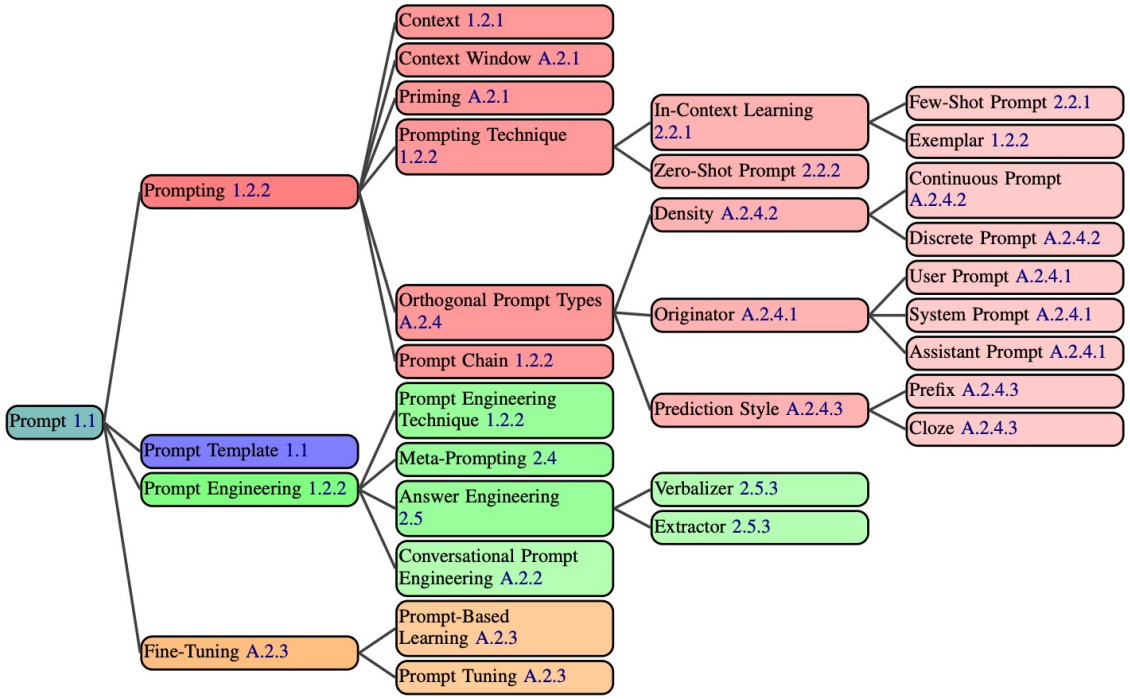
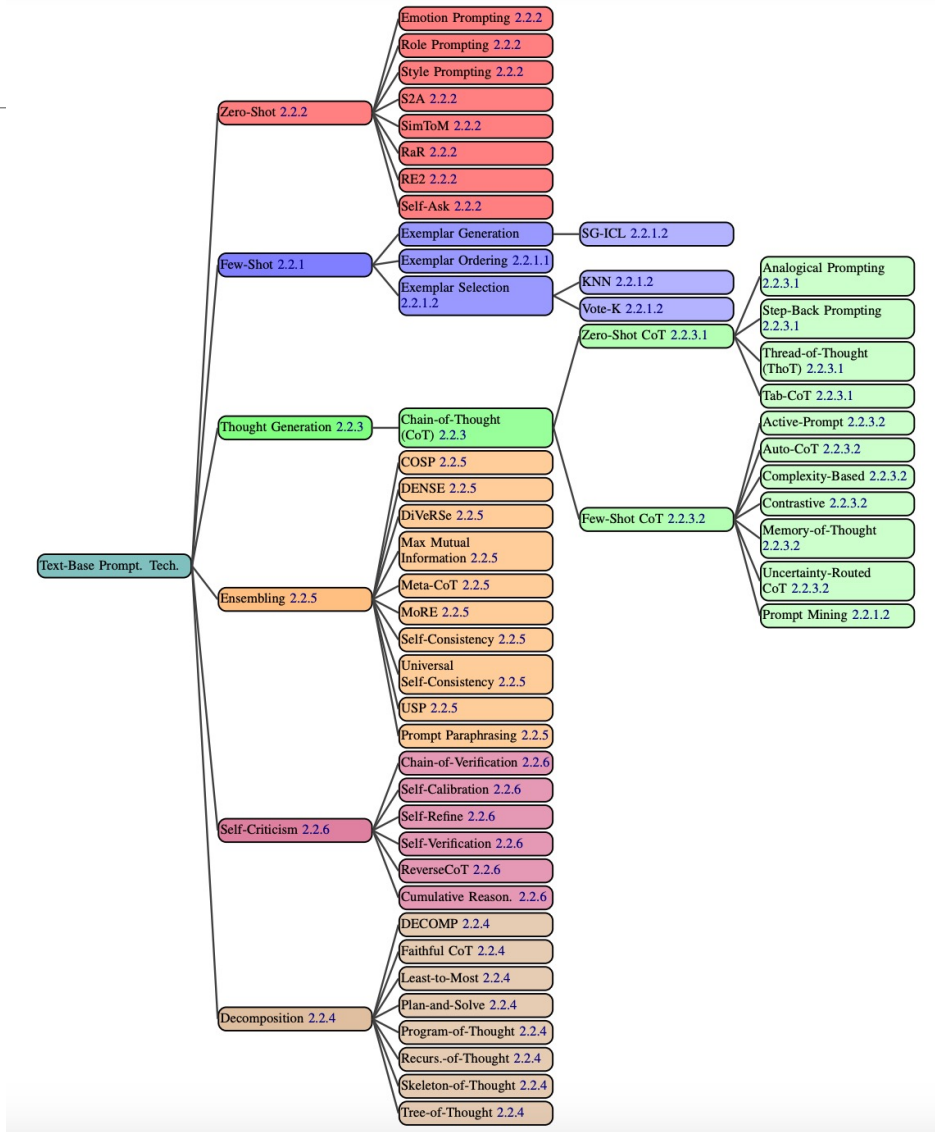
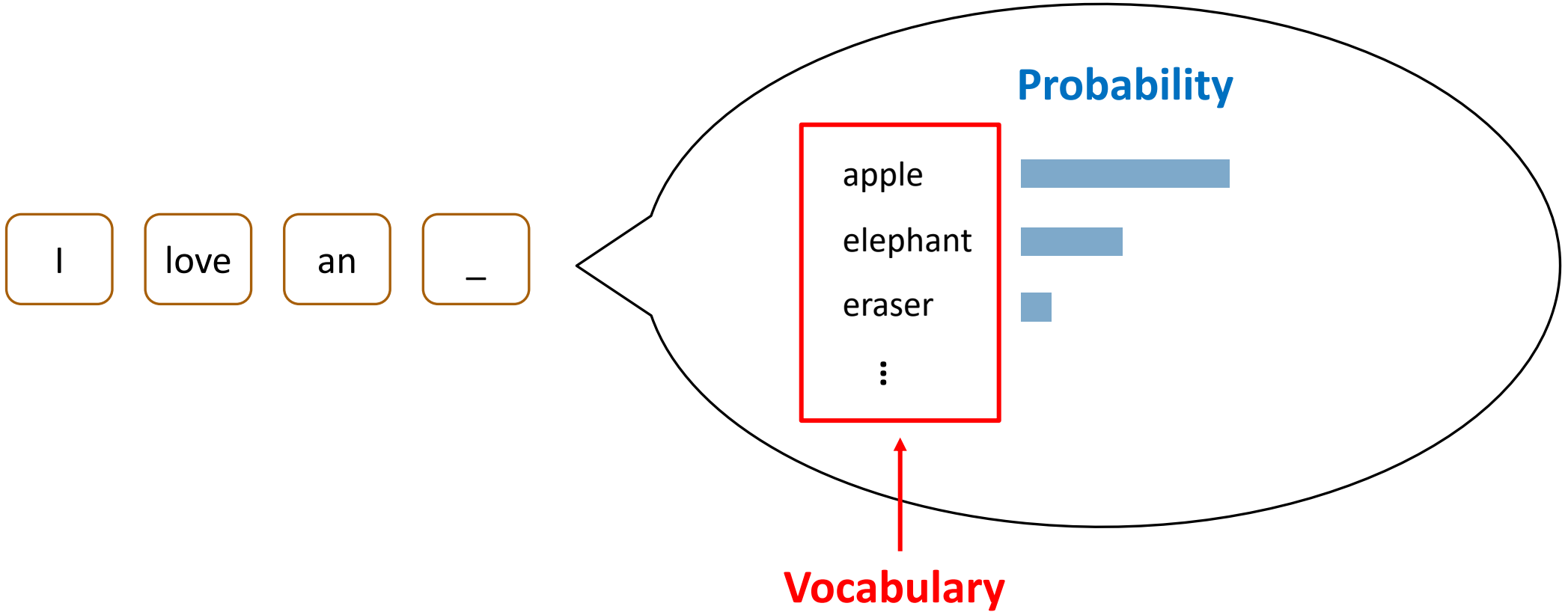


Figure 1.3: A Terminology of prompting. Terms with links to the appendix are not sufficiently critical to describe in the main paper, but are important to the field of prompting. Prompting techniques are shown in Figure 2.2

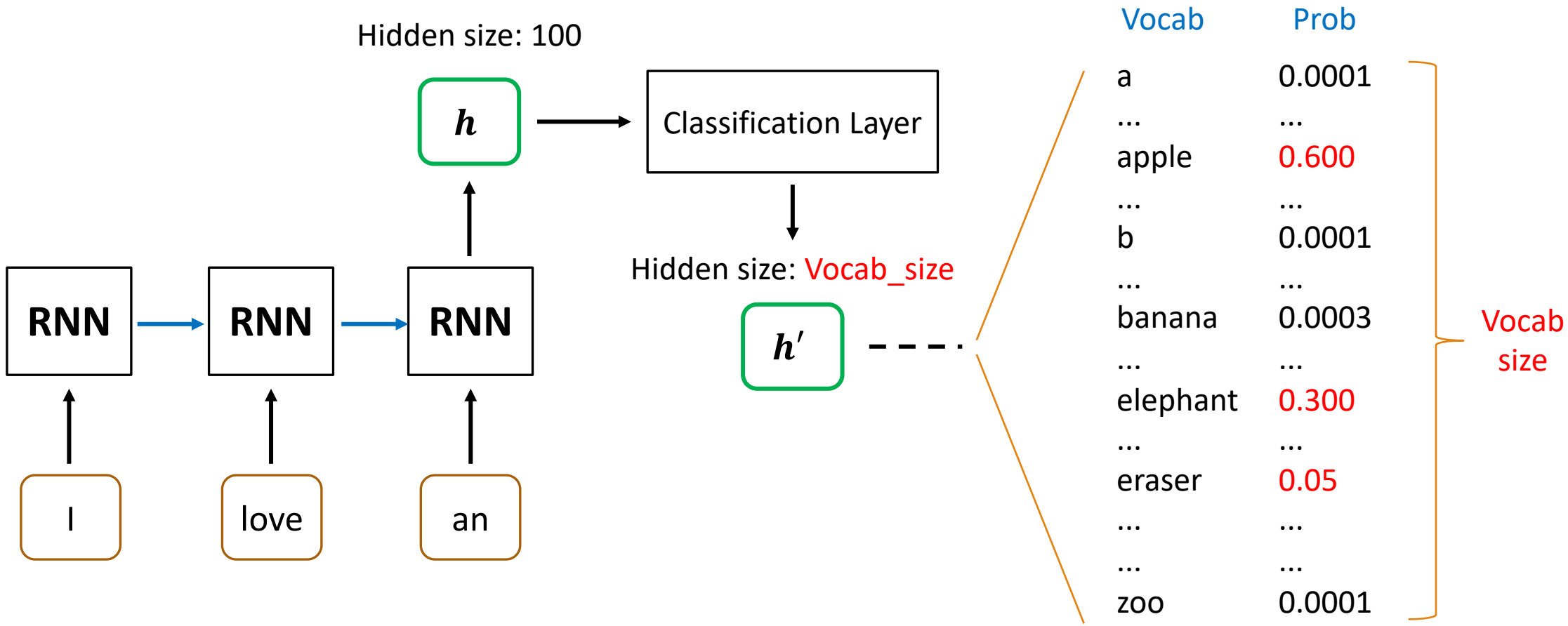
.



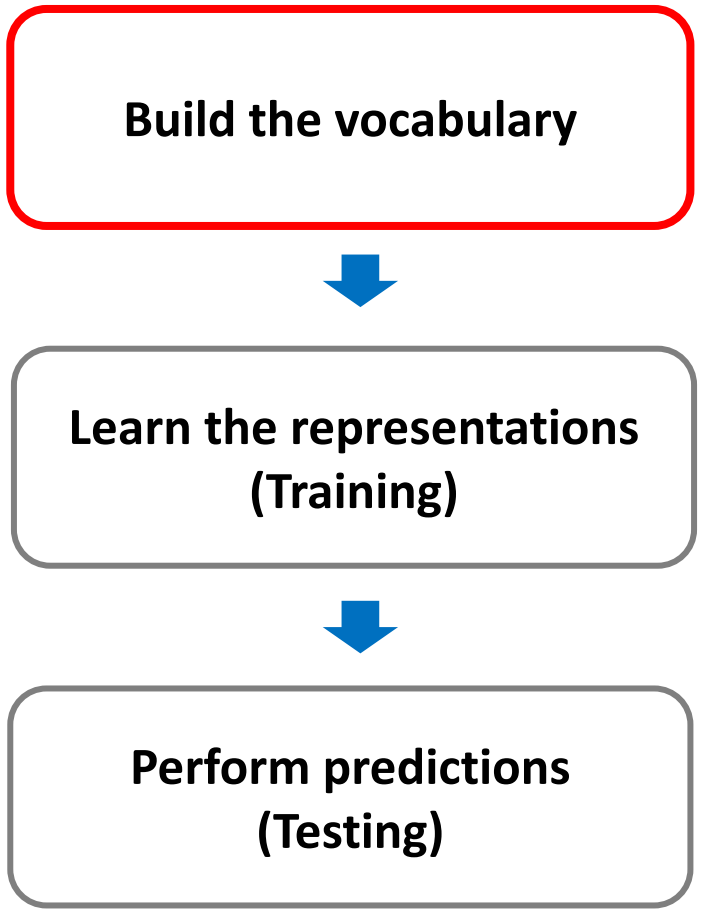
Recap: Word Representations



Recap: Word Representations (Details)



Basic Pipeline of Natural Language Processing



Size(vocabulary)
BERT: 30522
GPT-2 / GPT-3: 50257
T5: 32,128

How to build the vocabulary?

- Segment words based on delimiters (e.g., white spaces).
- Then we can collect the words from training corpora.

I love apples. I like apples and pineapples.

Vocabulary

Word	ID
and	0
apples	1
I	2
like	3
love	4
pineapples	5
.	6
<UNK>	7

Unknown words (not shown up in the training corpora)

(You can also perform stemming after word segmentation!)

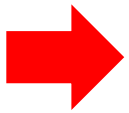
Issues of Delimiter-based Segmentation

- Only work for Western languages.
  - Cannot work for Chinese, Japanese, ...
- Cannot handle unseen words (not shown up in the training corpora)
  - A misspelled word contains morphological information but become an unknown word.

Issues of Delimiter-based Segmentation (Continued.)

- For machine translation, there is not always a 1-to-1 correspondence between source and target words since compound words may exist in target language.
- For example, sewage water treatment plant (English) -> Abwasserbehandlungsanlage (German)

sewage water treatment plant/facility



Sub-word units are favored.

Common Sub-word Tokenization Algorithms

- Byte Pair Encoding (BPE) (Sennrich et al., 2016)[1]
  - GPT series
- WordPiece (Schuster and Nakajima, 2012)[2]
  - BERT, T5
- Unigram Language Model (Kudo, 2018)[3]

[1] Sennrich, Rico, Barry Haddow, and Alexandra Birch. "Neural Machine Translation of Rare Words with Subword Units." Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), 2016.
[2] Schuster, Mike, and Kaisuke Nakajima. "Japanese and korean voice search." 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2012.
[3] Kudo, Taku. "Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates." Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL). 2018.

Segmentation vs. Tokenization

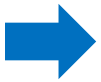
- All tokenization is segmentation, but not all segmentation is tokenization.
- Segmentation can be:
  - Word segmentation from a sentence
  - Sentence segmentation from a document
- Tokenization can be:
  - Word segmentation from a sentence (then `words` become `tokens`)
  - Sub-word tokenization from a word or sentence

Sub-word Tokenization (1)

- Given a training corpus, we first turn each word into a sequence of characters (separated by spaces)

Training corpus

low low low low low lower lower newest newest newest newest newest newest widest widest widest



Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w e s t </w>	6
w i d e s t </w>	3

- </w> is a special end-of-word symbol, allowing us to restore the original tokenization.

Sub-word Tokenization (1)

- Initialize the vocabulary with the existing characters:

low low low low low lower lower newest newest newest newest newest newest widest widest widest

Initial vocab: </w>, d, e, i, l, n, o, r, s, t, w

Sub-word Tokenization (2)

- Find the character **pair** with the highest frequency

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w e s t </w>	6
w i d e s t </w>	3

Found “e s” with the highest frequency 6+3=9

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w

Sub-word Tokenization (3)

- Add the pair with the highest frequency to the vocab

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w e s t </w>	6
w i d e s t </w>	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es

Sub-word Tokenization (4)

- Merge the characters by replacing all words in the corpus with the newly added pair.

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w **es** t </w>	6
w i d **es** t </w>	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es

Sub-word Tokenization (2)

Repeated (2)-(4)
according to `num_merges`

- Find the character **pair** with the highest frequency

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w es t </w>	6
w i d es t </w>	3

Found “es t” with the highest frequency 6+3=9

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es

Sub-word Tokenization (3)

Repeated (2)-(4)
according to `num_merges`

- Add the pair with the highest frequency to the vocab

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w es t </w>	6
w i d es t </w>	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es est

Sub-word Tokenization (4)

Repeated (2)-(4) according to `num_merges`

- Merge the characters by replacing all words in the corpus with the newly added pair.

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w **est** </w>	6
w i d **est** </w>	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est

Sub-word Tokenization (2)

Repeated (2)-(4)
according to `num_merges`

- Find the character **pair** with the highest frequency

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w est </w>	6
w i d est </w>	3

Found “est </w>” with the highest frequency 6+3=9

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est

Sub-word Tokenization (3)

Repeated (2)-(4)
according to `num_merges`

- Add the pair with the highest frequency to the vocab

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w est </w>	6
w i d est </w>	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>

Sub-word Tokenization (4)

Repeated (2)-(4)
according to `num_merges`

- Merge the characters by replacing all words in the corpus with the newly added pair.

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w **est</w>**	6
w i d **est</w>**	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>

Sub-word Tokenization (2)

Repeated (2)-(4)
according to `num_merges`

- Find the character **pair** with the highest frequency

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w est</w>	6
w i d est</w>	3

Found “l o” with the highest frequency 5+2=7

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>

Sub-word Tokenization (3)

Repeated (2)-(4)
according to `num_merges`

- Add the pair with the highest frequency to the vocab

Word	Frequency
l o w </w>	5
l o w e r </w>	2
n e w est</w>	6
w i d est</w>	3

Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>, lo

Sub-word Tokenization (4)

Repeated (2)-(4)
according to `num_merges`

- Merge the characters by replacing all words in the corpus with the newly added pair.

Word	Frequency
lo w </w>	5
lo w e r </w>	2
n e w est</w>	6
w i d est</w>	3

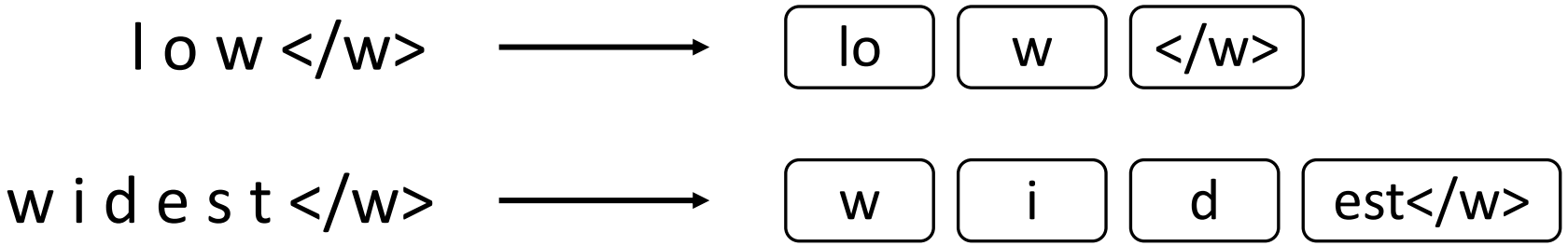
Current Vocabulary: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>, lo

Finish Sub-word Leaning

- Assume `num_merges`=4 (we just repeated four times.)
- `num_merges` is a hyperparameter that you need to set for BPE.
- The learned vocabulary is: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>, lo

Tokenization with Learned BPE

- **The learned vocabulary** is: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>, lo
- We first turn each word into a sequence of characters with </w> placed at the end of a word, the same as the first step during the learning phase.
- Then we can merge the characters according to **the learned vocabulary**.
- Examples:



Properties of BPE

- The final learned vocabulary size = **initial** size + `num_merges`

  Initial vocab: </w>, d, e, i, l, n, o, r, s, t, w

  Learned vocab: </w>, d, e, i, l, n, o, r, s, t, w, es, est, est</w>, lo

- This algorithm is based on statistics, so frequent sub-word units in provided corpora will be put to the learned vocabulary.

Why do we need Sub-word Tokenization?

- With sub-word tokenization algorithms, we can handle representations for unknown words (or mis-spelled words / compound words).
- In machine translation, the compound word issues between source and target languages can be alleviated.
- State-of-the-art pre-trained language models (e.g., GPT-3, BERT) adopt sub-word tokenization algorithms before pre-training.

Limitations of Sub-word Tokenization

- (Not many disadvantages for sub-word tokenization)
- The hyperparameter `num_merges` needs to be tuned.
- Once the learned vocabulary is created, it becomes fixed. The algorithm needs to be re-run after adding new data.
- How about Chinese?
  - Character-level encoding