Language Technology
http://cs.lth.se/edan20/
Chapter 7, Subword tokenization

Pierre Nugues

Pierre.Nugues@cs.lth.se
http://cs.lth.se/pierre_nugues/

September 7 and 11, 2023

Chapter 13, Subword tokenization

Subword Tokenization Techniques

In a multilingual context, it is often preferable to use a subword tokenization, where variants include:

- Byte-Pair Encoding (BPE)
- WordPiece
- SentencePiece

These techniques use the set of characters and merge them to form words until they have reached a predefined vocabulary size.
Merging is done by frequency or with an entropy criterion and the vocabulary consists then in the most frequent character sequences.
The tokenizer splits the text into subwords.

Chapter 13, Subword tokenization

Language Differences (Source: Xerox)

Language	# stems	# inflected forms		Lex. size (kb)
English	55,000	240,000		200–300
French	50,000	5,700,000		200–300
German	50,000	350,000	or	450
		infinite	(compounding)	
Japanese	130,000	200	suffixes	500
		20,000,000	word forms	500
Spanish	40,000	3,000,000		200–300

Byte-Pair Encoding (BPE)

Originally, BPE is a compression algorithm
Sennrich adapted the original BPE algorithm to build automatically a lexicon of subwords from a corpus.
These subwords consist of a single character, a sequence of characters, possibly a whole word, and the size of the lexicon is fixed in advance.

Chapter 13, Subword tokenization

BPE Algorithm

The main steps of the algorithm are:

1. Split the corpus into individual characters. These characters will be the initial subwords and will make up the start vocabulary:
2. Then:
    1. Extract the most frequent adjacent pair from the corpus;
    2. Merge the pair and add the corresponding subword to the vocabulary;
    3. Replace all the occurrences of the pair in the corpus with the new subword;
    4. Repeat this process until we have reached the desired vocabulary size.

To tokenize a text, the merge rules are applied in the same order
BPE can be character or byte-based
With bytes, by construction, there is no unknown word. The algorithm always falls back to a byte

Chapter 13, Subword tokenization

Pretokenization

BPE normally does not cross the whitespaces.
This means that we can speed up the learning process with a pretokenization of the corpus and a word count.
The simplest pretokenization uses the whitespaces as delimiters, where we match the words, the numbers, and the rest, excluding the whitespaces, with this regular expression:

```
pattern = r'\p{L}+|\p{N}+|[^\s\p{L}\p{N}]+'
```

Another one: `https://github.com/karpathy/minGPT/blob/master/mingpt/bpe.py`

Chapter 13, Subword tokenization

WordPiece

To be written

Chapter 13, Subword tokenization

Unigram

To be written

Chapter 13, Subword tokenization

Unigram

To be written

Chapter 13, Subword tokenization

SentencePiece

To be written

Chapter 13, Subword tokenization

Further Reading

See: https://huggingface.co/course/chapter6/