EDAN20 - Assignment 3
Tokenization using Byte-Pair Encoding and a Unigram Language Model

Hugo Mattsson
hu5174ma-s

October 2024

1 Objectives and dataset

1.1 Objectives

The objectives of the assignment were to explore how to tokenize a text, not using words, but instead focusing on character and subword tokens. The two techniques explored were BPE and a unigram language model

1.2 Dataset

The corpus consisted of the smallest novel written by Selma Lagerlöf: *En herrgårdssägen.*

2 Method and program structure

2.1 Byte-Pair Encoding

2.1.1 Background

BPE is an iterative method which, given a corpus, returns a vocabulary of a wanted size comprised of subwords from the corpus. The goal of BPE is to find the most frequent ”building blocks” from a text. The starting point of the vocabulary will be all characters present in the text, these will then be merged into pairs and the most frequent token combination will be added to the vocabulary. This merge operation will also be performed on the corpus, and the cycle starts anew, trying to find the next most frequent token combinations until the vocabulary reaches the wanted size.

2.1.2 Implementation

My implementation of BPE looks as follows:

1. Start with a wanted vocabulary size and a string that has already been split into individual characters
2. Get the initial vocabulary(i.e. all present characters)
3. Construct all possible pairs of current vocabulary tokens
4. Pick the most frequent token pair from 3 that exist in the string
5. Add this new token to vocabulary and to a list of performed merge operations
6. Merge all occurrences of the new token pair in the string
7. Repeat from step 3 until vocabulary size is met
8. Return the found vocabulary and the list of token merge operations

The found merge operations are later used to tokenize the corpus or any other string of interest.

2.2 Unigram

2.2.1 Background

The Unigram language model achieves the same goal as BPE: construct a vocabulary of a given size, but starts in the opposite end. Unigram starts with a "big seed vocabulary", and prunes it to the desired size. Acquiring a seed vocabulary can be done using BPE or, as Kudo suggests, use the union of all characters and the most frequent substrings in the corpus.

The log probabilities of the individual subwords are computed, which can then be used to compute the loss for each subword. The loss represents how likely the removal of a subword is to improve the total likelihood of the vocabulary. Then the best candidate is removed from the vocabulary, the model is refitted and the cycle continues until the correct size vocabulary is reached.

In order to tokenize a text using the unigram language model we'll use the vocabulary probabilities in order to recursively segment our text and only keep the candidate with the highest probability.

2.2.2 Implementation

The big seed vocabulary was constructed using our implementation of BPE. These tokens were used to iteratively and recursively improve the segmentation of the corpus and fine-tune the subword probabilities for the unigram model. Finally the loss was computed and one subword pruned to improve the model.

3 Results

3.1 BPE

The new tokens found by the BPE algorithm were the same as those found by Sentencepiece.

```
_s, de, _h, en, an, tt, ar, _v, _f, _a, om,
on, ll, _de, _m, ör, _o, ch, _b, ade, _k, _t,
ig, _att, er, ng, _och, st, _d, _hon, _g, _i,
et, _e, _l, _var, är, ck, _för, _H, _n, _han,
_p, or, na, än, _en, _det, _u, fv
```

3.2 Unigram LM

After removing the candidate `'_o'` from the vocabulary, the final probability for the corpus improved from `-494583.24282243924` to `-494534.0921189297`

4 Conclusion

The objectives of this assignment, to create a subword tokenization program, was accomplished. Two methods were tried: BPE and Unigram Language Model.