Northeastern University

CS6120 37424 Natural Language Processing
Assignment 1(Part 1): Byte Pair Encoding (BPE) Implementation and Evaluation on NLTK Dataset

By:

Srinivas Peri

Jan-29-2023

Abstract:

Byte Pair Encoding (BPE) is a subword tokenization method that has gained popularity in the field of Natural Language Processing (NLP), particularly in the context of language modeling and machine translation. This report documents the implementation of BPE, the experimental setup, and the observed results, providing a comparison with standard tokenization methods. It also discusses the strengths and weaknesses of BPE, challenges encountered, and suggests potential improvements.

Introduction:

Tokenization is a foundational step in text processing which involves splitting text into meaningful elements called tokens. Standard tokenization methods include word tokenization and character tokenization. BPE offers a middle ground by iteratively merging the most frequent pairs of characters or sequences, which allows for efficient encoding of common words while still handling rare or unknown words effectively.

Implementation:

The implementation of BPE involved the following steps:

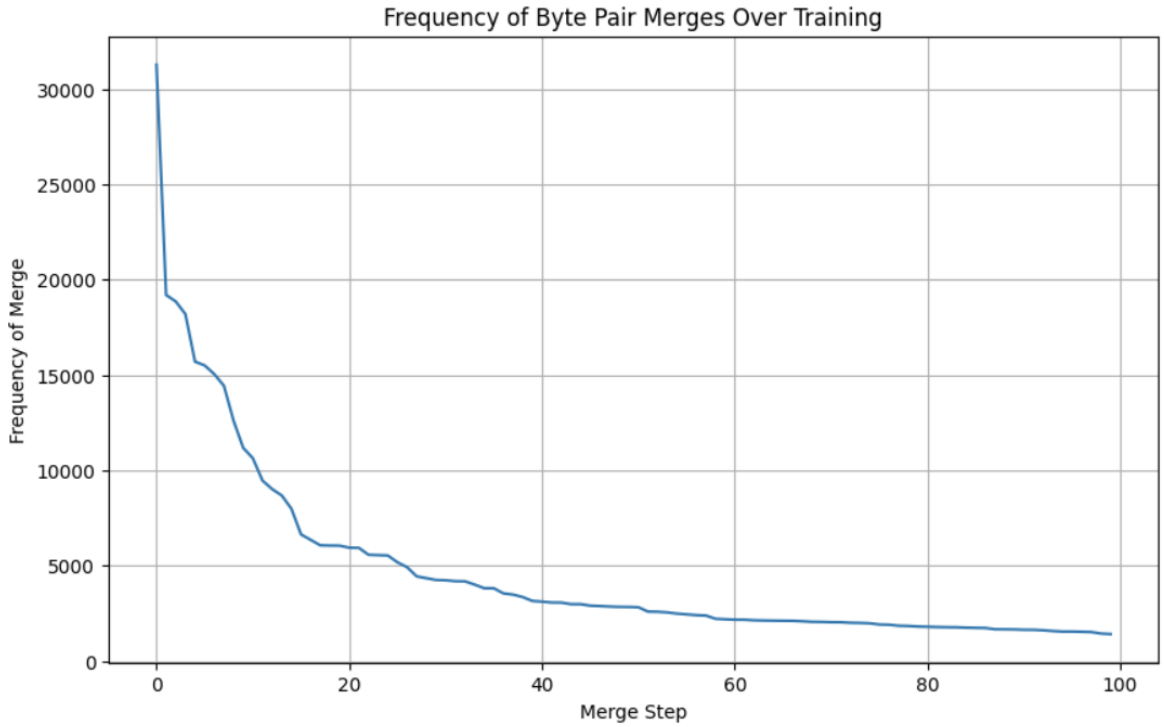
- Generating a vocabulary from a given corpus, where each word is represented by a sequence of characters plus a special end-of-word symbol.
- Counting the frequency of pairs of characters in the vocabulary.
- Iteratively merging the most frequent pairs to create new 'tokens'.
- Repeating the process for a predetermined number of merge operations.
- The implementation was carried out in Python, utilizing standard libraries such as collections for data structure management and re for regular expression operations.

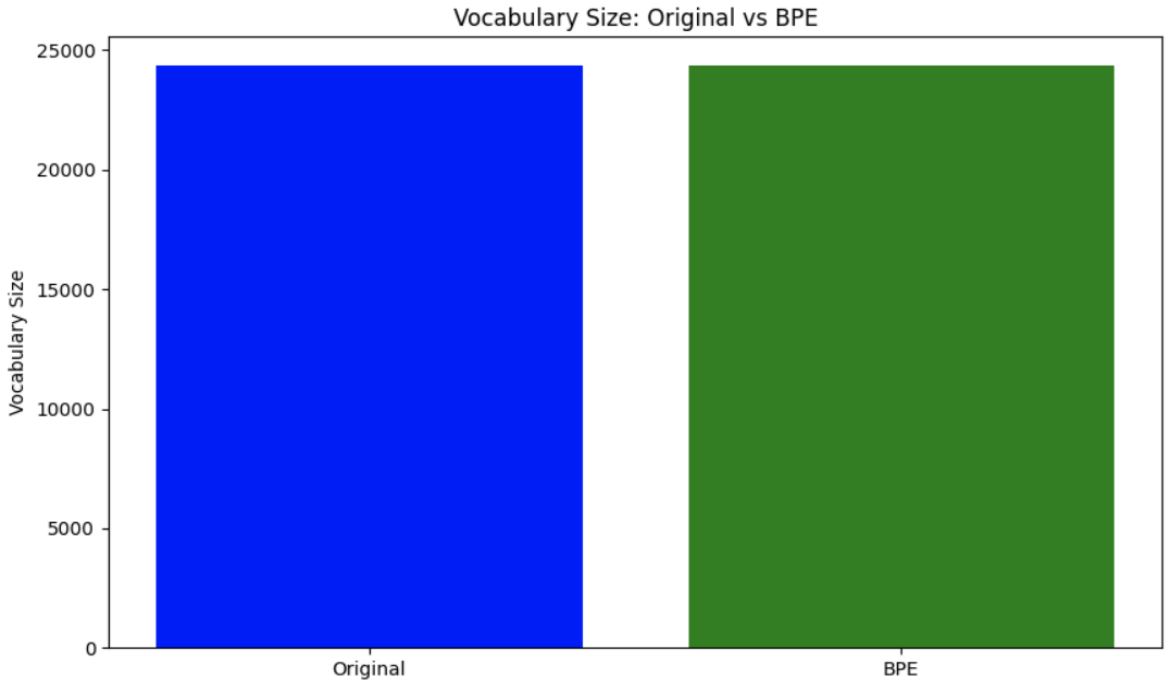
Experimental Setup:

The experiments were conducted using texts from the NLTK Gutenberg Corpus, including works from Jane Austen, William Shakespeare, and others. The combined corpus was used to train the BPE model with 100 mergers.

Results:

The BPE algorithm successfully learned a new vocabulary with merged tokens representing the most frequent symbol pairs. The tokenization process demonstrated that BPE could effectively reduce the size of the vocabulary while still preserving the ability to reconstruct the original text, indicating lossless compression.





Discussion:

Strengths of BPE:

- Efficiency in Encoding: BPE can encode common words as single tokens while still representing rare words through a sequence of subword units.
- Handling of OOV Words: Out-of-vocabulary (OOV) words can be broken down into known subwords, enabling the model to work with words not seen during training.
- Scalability: BPE scales well with the size of the corpus, as it creates a vocabulary size that is a function of the number of merge operations rather than the size of the corpus.

Weaknesses of BPE:

- Dependency on Frequency: BPE is heavily dependent on the frequency of pairs, which can bias the model towards the corpus it was trained on.
- Fixed Vocabulary: The number of mergers determines a fixed vocabulary size, which may not be optimal for all types of text or languages.
- Context Ignorance: BPE does not take into account the context of words, potentially leading to suboptimal mergers.

Comparison with Standard Tokenization

Word Tokenization: Unlike BPE, word tokenization may struggle with large vocabularies and OOV words. However, it preserves word boundaries and can be more interpretable.

Character Tokenization: Character tokenization handles OOV words well but can lead to longer sequences than BPE, making it less efficient.

Challenges Encountered:

One challenge was ensuring that the merge_vocab function correctly maintained word boundaries and merged characters within words rather than across word boundaries. Additionally, the initial implementation faced issues with empty merge frequencies due to improper handling of the corpus format.

I've added ' </w>' to each word in the vocabulary to explicitly mark the end of words. This is a common practice in BPE to handle word boundaries correctly.

Potential Improvements:

Dynamic Vocabulary: Implementing a dynamic version of BPE that can adjust the vocabulary based on the context could improve performance.

Hybrid Approaches: Combining BPE with other tokenization methods, like sentence piece, could leverage the strengths of multiple approaches.

Hyperparameter Tuning: Developing methods for optimizing the number of mergers and other hyperparameters based on the corpus and task could yield better results.

Conclusion:

BPE is a powerful tokenization method that offers a balance between word and character tokenization. While it has its drawbacks, such as a dependency on token frequencies and fixed vocabulary limitations, its strengths in handling OOV words and scalability make it a valuable tool in NLP. The challenges encountered in its implementation provided insights into the algorithm's intricacies and highlighted areas for improvement. Future work should focus on enhancing BPE to overcome its current limitations and exploring hybrid models for more robust tokenization.