Assignment 1 (Part 1):

Natural Language Processing

Uzair Ahmad

Byte Pair Encoding (BPE) Implementation and Evaluation on NLTK Dataset

Objective: The primary goal of this assignment is to implement the Byte Pair Encoding (BPE) algorithm for tokenization and assess its performance using a NLTK dataset, with a focus on books. Additionally, you will create a reference tokenization using NLTK's punkt tokenizer for comparative analysis.

Tasks:

1. Implement BPE Algorithm (4 marks):

- Develop a Python implementation of the Byte Pair Encoding (BPE) algorithm, covering key steps such as learning byte pair merges and encoding/decoding using the learned merge operations.

2. Train on NLTK Dataset (3 marks):

- Utilize NLTK's Gutenberg Corpus, selecting books like "austen-emma.txt," "blake-poems.txt" and "shakespeare-hamlet.txt" for training the BPE algorithm. Create a vocabulary based on the training.

3. Test on NLTK Dataset (3 marks):

- Evaluate the BPE algorithm on a separate set of books from the NLTK Gutenberg Corpus, such as "Frankenstein," "Dracula," and "The Adventures of Sherlock Holmes." Measure tokenization accuracy, coverage, and other relevant metrics.

```
import nltk

# Download the NLTK Gutenberg Corpus
nltk.download('gutenberg')

# Import the Gutenberg Corpus module
from nltk.corpus import gutenberg

# Get the list of available books in the Gutenberg Corpus
book_list = gutenberg.fileids()

# Print the list of available books
```

```python
print("Available Books:")
for book in book_list:
    print(book)

# Load a specific book (e.g., "shakespeare-hamlet.txt")
selected_book = gutenberg.raw('shakespeare-hamlet.txt')

# Display the first 500 characters of the selected book
print("\nSample Text from 'shakespeare-hamlet.txt':")
print(selected_book[:500])

```

4. Create Reference Tokenization (2 marks):

- Use NLTK's punkt tokenizer to create a reference tokenization for the test dataset. Save the tokenized results in a structured format for later comparison.

5. Compare with Standard Tokenization (2 marks):

- Implement a baseline tokenization using NLTK's default method (e.g., word_tokenize) on the test dataset. Compare the BPE algorithm's performance with the standard tokenization in terms of accuracy, coverage, and other relevant metrics.
- When measuring tokenization accuracy, coverage, and other relevant metrics, you can use various formulas depending on your specific goals and requirements. Here are some common metrics:
  1. **Tokenization Accuracy:**

     Formula:

     $$\text{Accuracy} = \frac{\text{Number of Correctly Tokenized Tokens}}{\text{Total Number of Tokens}} \times 100$$

     Explanation:

     - Count the number of tokens that were correctly tokenized by your algorithm.
     - Divide this by the total number of tokens in the ground truth (reference tokenization).
     - Multiply by 100 to express the result as a percentage.
  2. **Tokenization Coverage:**

     Formula:

     $$\text{Coverage} = \frac{\text{Number of Unique Tokens Covered}}{\text{Total Number of Unique Tokens in the Ground Truth}} \times 100$$

     Explanation:

     - Identify the unique tokens covered by your algorithm.
     - Divide this by the total number of unique tokens in the ground truth.
     - Multiply by 100 to express the result as a percentage.
  3. **Precision, Recall, and F1-Score:**

     Formulas:

     $$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

     $$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

     $$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Explanation:

- True Positives: Number of correctly identified tokens.
- False Positives: Number of tokens identified by your algorithm but not present in the ground truth.
- False Negatives: Number of tokens in the ground truth but not identified by your algorithm.
- Precision measures the accuracy of the positive predictions.
- Recall measures the ability to capture all the relevant instances.
- F1-Score is the harmonic mean of precision and recall, providing a balanced metric.

4. **Jaccard Similarity:**

   Formula:

   $$\text{Jaccard Similarity} = \frac{\text{Intersection of Predicted and Ground Truth Tokens}}{\text{Union of Predicted and Ground Truth Tokens}}$$

   Explanation:

   - Identify the common tokens between the predicted and ground truth sets.
   - Divide this by the total unique tokens in both sets.

6. Visualizations (2 marks):

- Provide visualizations of the BPE algorithm's learning process, illustrating the evolution of the vocabulary and the frequency of byte pair merges. Compare the vocabulary before and after training.

7. Report and Discussion (3 marks):

- Prepare a detailed report documenting the implementation, experimental setup, and results.
- Discuss the strengths and weaknesses of BPE, and compare it with standard tokenization methods.
- Address any challenges encountered during implementation and suggest potential improvements.

Submission Guidelines:

- Submit the Python code for the BPE implementation, including a README file with instructions and dependencies.
- Include a folder containing the selected training and test books from the NLTK Gutenberg Corpus.
- Submit a report document (PDF) with detailed explanations, visualizations, and comparative analysis.
- Include the reference tokenization file obtained using NLTK's punkt tokenizer.