#### **Introduction**

In this project, we explored sentiment analysis using a Naive Bayes classifier, comparing two tokenization methods: Word-Based Tokenization and Byte Pair Encoding (BPE) Tokenization. The task involved classifying tweets in the Twi language from the AfriSenti dataset as either positive or negative. This report details our approach, key challenges, findings, and a comparison of the two tokenization methods.

#### **Approach**

1. **Data Preprocessing**
   * **Dataset Selection**: We focused on the Twi language from the AfriSenti dataset, filtering out neutral tweets to build a binary classifier.
   * **Text Cleaning**: URLs were replaced with [URL], numbers with [NUM], and extraneous spaces were removed.
   * **Tokenization**:
     + **Word-Based Tokenization**: This method tokenized the text into words by splitting based on whitespace.
     + **BPE Tokenization**: BPE iteratively merged the most frequent pairs of characters or sequences to form subword units, aiming to handle out-of-vocabulary words more effectively.
2. **Vocabulary Construction**
   * A vocabulary was built for both tokenization methods by processing the training set, forming the foundation for feature extraction.
3. **Feature Extraction**
   * **Count-Based Vectorization**: We used CountVectorizer to convert tokenized text into numerical feature vectors, which were then used to train the Naive Bayes model.
4. **Model Training**
   * Separate Naive Bayes classifiers were trained on the word-based and BPE tokenized text, with Laplace smoothing applied to manage unseen words.
5. **Prediction and Evaluation**
   * A prediction function was developed for each tokenization method to classify new sentences.
   * The models were evaluated on the development set using precision, recall, F1-score, and accuracy.

#### **Challenges**

* **Prediction Function Errors**: Early issues arose due to mismatches between the preprocessing steps and the tokenization methods. These were resolved by standardizing the input processing and ensuring consistency throughout the code.
* **Handling Non-Alphanumeric Characters in BPE**: Initial BPE implementation struggled with non-alphanumeric characters like emojis, which required adjustments in the text cleaning process.
* **CountVectorizer Compatibility**: The vectorizer initially encountered difficulties with the BPE tokenized text due to non-standard tokens, which were mitigated by adjusting the minimum token length and other vectorizer parameters.

#### **Code Structure**

* **Data Loading**: The dataset was loaded into dataframes, focusing on the Twi language.
* **Preprocessing**: Text cleaning and tokenization were performed using both word-based and BPE methods.
* **Vocabulary Construction**: Vocabularies were built from the training data.
* **Feature Extraction**: CountVectorizer was used to transform tokenized text into feature vectors.
* **Model Training**: The Naive Bayes classifier was trained on the extracted features.
* **Prediction**: Functions were implemented to predict sentiment from new sentences.
* **Evaluation**: The models were evaluated, and performance metrics were compared between the two tokenization methods.

#### **Comparison of Tokenization Methods**

* **Word-Based Tokenization**:
  + **Negative Class**: Precision: 0.73, Recall: 0.76, F1-Score: 0.74
  + **Positive Class**: Precision: 0.80, Recall: 0.78, F1-Score: 0.79
  + **Overall Accuracy**: 0.77
* **BPE Tokenization**:
  + **Negative Class**: Precision: 0.59, Recall: 0.46, F1-Score: 0.52
  + **Positive Class**: Precision: 0.63, Recall: 0.75, F1-Score: 0.69
  + **Overall Accuracy**: 0.62

#### **Strengths and Weaknesses of the Systems**

**Strengths:**

* **Word-Based Tokenization**:
  + **Context Preservation**: By maintaining full words, this method preserves the integrity of the language, making it easier for the model to understand the context.
  + **Simplicity**: The straightforward nature of word-based tokenization makes it easy to implement and interpret, especially in datasets with clear word boundaries.
  + **High Performance**: The method demonstrated strong performance metrics across all classes, indicating its effectiveness in sentiment classification tasks.
* **BPE Tokenization**:
  + **Handling of Rare Words**: BPE’s ability to break down words into subword units allows it to handle rare or out-of-vocabulary words, which can be beneficial in more diverse datasets.
  + **Flexibility**: BPE is particularly useful in multilingual contexts or when dealing with highly inflected languages, where it can reduce the vocabulary size and capture morphological variations.

**Weaknesses:**

* **Word-Based Tokenization**:
  + **Out-of-Vocabulary Words**: This method struggles with words that were not present in the training set, potentially leading to lower performance in scenarios involving rare or newly coined words.
  + **Larger Vocabulary Size**: The reliance on complete words results in a larger vocabulary, which can increase computational complexity and memory requirements.
* **BPE Tokenization**:
  + **Loss of Context**: By breaking words into subword units, BPE may lose some of the contextual meaning, which can dilute the effectiveness of the model, as seen in the reduced performance metrics.
  + **Complexity**: The process of iteratively merging subwords introduces additional complexity, which can complicate both the implementation and interpretation of the results.

#### **Discussion**

The word-based tokenization method outperformed BPE across all metrics, including precision, recall, F1-scores, and overall accuracy. Several factors contributed to this:

* **Context Preservation**: Word-based tokenization preserves complete words, which likely enabled the model to better capture contextual relationships in the text.
* **Data Complexity**: BPE may have introduced unnecessary complexity by breaking words into subword units, potentially diluting the context and effectiveness of feature representation.
* **Handling of Non-Standard Tokens**: Issues with non-standard tokens like emojis in BPE likely added noise, impacting the model's performance.

#### **Conclusion**

This project successfully implemented and compared two tokenization methods for sentiment analysis in Twi. The word-based tokenization method proved to be more effective, suggesting it may be preferable for similar tasks, particularly when dealing with datasets with clear word boundaries. However, BPE tokenization may still be valuable in more complex linguistic scenarios or with larger datasets that can benefit from its ability to handle out-of-vocabulary words.