Identification of Protein Language Units using Protein Oriented Tokenization Methods and Language Models

Progress Report

by

Burak Suyunu

B.S., Computer Engineering, Boğaziçi University, 2017

M.S., Computer Engineering, Boğaziçi University, 2020

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Computer Engineering

Boğaziçi University

2024

ABSTRACT

Identification of Protein Language Units using Protein Oriented Tokenization Methods and Language Models Progress Report

Protein sequences exhibit linguistic properties analogous to natural language, prompting the exploration of subword tokenization methods for their analysis. This paper evaluates three prominent subword tokenization methods —Byte-Pair Encoding (BPE), WordPiece, and SentencePiece— across varying vocabulary sizes (400, 800, 1600, 3200, and 6400) in the context of protein sequences using the UniRef50 dataset. We assess each tokenizer through shared token percentages, token length distribution, fertility, and contextual exponence, in addition to examining their behavior under linguistic laws including Zipf's, Brevity, Heaps', and Menzerath's laws.

Our findings reveal significant variations in vocabulary composition, token length distribution, and fertility across the tokenizers. Despite their differences, all tokenizers exhibit adherence to Zipf's law and Heaps' law, while BPE and WordPiece show stronger compliance with Brevity law compared to SentencePiece, and none of them fully comply with Menzerath's Law. Furthermore, BPE tends to show higher context usage, WordPiece balances between encoding efficiency and token diversity, and SentencePiece exhibits better efficiency in encoding and unique frequency distribution characteristics. However, none of the tokenizers fully satisfy the metrics or laws, underscoring the need for a specialized tokenizer optimized for protein sequences modeled as languages. These insights provide a comprehensive comparison of subword tokenization methods for protein sequences and contribute valuable understanding of their linguistic characteristics, paving the way for more nuanced and efficient protein analysis.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Protein sequences, the molecular foundation of life, consist of amino acids arranged in specific orders that define their structure and function. Analyzing these sequences is crucial for understanding biological processes and advancing biomedical research. However, the complexity and variability of protein sequences necessitate efficient representation and segmentation strategies.

Subword tokenization, a technique initially developed for natural language processing (NLP), has recently gained traction in protein sequence analysis. By breaking down protein sequences into meaningful subunits or "tokens," this approach offers a promising avenue for encoding, comparing, and understanding the underlying biological information. In natural language, subword tokenization methods such as Byte Pair Encoding (BPE) [1], WordPiece [2], and SentencePiece [3] have shown remarkable success in balancing vocabulary size with encoding efficiency. Despite the differences in their construction, all three tokenization methods have proven effective in handling protein sequences due to the inherent compositional structure of proteins, akin to human languages.

In this study, we comprehensively evaluate BPE, WordPiece, and SentencePiece (with Unigram) tokenization methods on protein sequences using the UniRef50 dataset. Our evaluation focuses on key statistical properties and explores how these tokenizers adhere to fundamental linguistic laws.

By systematically comparing these subword tokenization methods across various vocabulary sizes (400, 800, 1600, 3200, and 6400), statistical metrics, and linguistic laws, we aim to uncover insights into their strengths, weaknesses, and applicability to protein sequences. Our findings provide a comprehensive understanding of how subword tokenization methods behave in the protein domain, contributing to developing more effective computational tools for protein analysis and understanding.

Ultimately, our goal is to advance the understanding of subword tokenization methods and their implications for computational protein research, guiding researchers in selecting appropriate methods for their specific needs.

2. SUBWORD TOKENIZATION METHODS, UNIREF50 DATASET, AND EVALUATION METRICS

2.1. Subword Tokenization Methods

Tokenization involves dividing a piece of text into smaller units, such as words, phrases, symbols, or other meaningful elements, called tokens. This step is usually the initial one in any text processing pipeline, and the selection of the tokenization method can greatly influence the outcome of following NLP operations. Thus, it is important to consider the most appropriate tokenization approach for a specific task.

Subword tokenization methods are based on the idea that commonly used words should not be broken down into smaller subwords, while infrequent words should be divided into meaningful subparts. For example, the word “quietly” might be considered a rare word and split into “quiet” and “ly”. Both “quiet” and “ly” appear more frequently as standalone subwords, and at the same time the meaning of “quietly” is preserved by the combination of “quiet” and “ly”. Subword tokenization enables the model to process new words by breaking them down into familiar subwords. The most well-known algorithms for subword tokenization are Byte Pair Encoding (BPE) [1], WordPiece [2], Unigram [4], and SentencePiece [3]. Except for SentencePiece, these subword tokenizers use a pre-tokenizer (space tokenization) to divide the training data into words. BPE starts with an initial vocabulary that includes all the symbols in the dataset. Then, it repeatedly forms new symbols by merging the two most frequent symbols until the desired vocabulary size is reached. WordPiece is similar to BPE, but instead of merging symbols based on frequency, it merges symbols based on the likelihood of the training data after the new symbol is added to the vocabulary. Unlike BPE or WordPiece, Unigram starts with a large number of tokens and iteratively discards tokens to obtain a smaller vocabulary. The Unigram algorithm calculates a loss (often defined as log-likelihood) over the training data given the current vocabulary and a Unigram language model. SentencePiece treats the input as a raw input stream and includes the space in the set of characters to be used, then it applies BPE or Unigram algorithms to create an appropriate vocabulary.

2.2. UniRef50 Dataset

The UniRef100 database is a collection of identical protein sequences and subfragments that are 11 or more residues long and come from any organism. These sequences are merged into a single UniRef entry, which displays the sequence of a representative protein, the accession numbers of all the merged entries, and links to the corresponding UniProtKB and UniParc records. UniRef90 is created by clustering UniRef100 sequences that are 11 or more residues long. This is done using the MMseqs2 algorithm, which groups sequences together based on their similarity. Each cluster is made up of sequences that have at least 90% sequence identity and 80% overlap with the longest sequence in the cluster, also known as the ”seed sequence.” Similarly, UniRef50 is created by clustering UniRef90 seed sequences that have at least 50% sequence identity and 80% overlap with the longest sequence in the cluster. This results in a smaller set of protein sequences that are more closely related to each other [5]. Using UniRef50 clusters aims to reduce over-sampling of similar, evolutionarily close sequences.

We downloaded the UniRef50 dataset from HuggingFace[1] . Subword tokenizers were trained on randomly sampled 15 million sequences of the data's train split. Then, experiments are applied to the combination of validation and test splits (11957 sequences). We discarded 14 sequences from the test set that are longer than 3k in length.

2.3. Evaluation Metrics

Our evaluation sheds light on the unique features and comparative advantages of each tokenizer by concentrating on key statistical metrics and investigating their

[1]https://huggingface.co/datasets/agemagician/uniref50

adherence to fundamental linguistic laws:

- **Shared token percentages:** The percentage of tokens shared between vocabularies of different tokenizers, offering insight into segmentation consistency.
- **Token length distribution:** Distribution of token lengths in the vocabulary, reflecting the granularity of the segmentation.
- **Fertility:** The average number of tokens required to represent a protein sequence, indicating encoding efficiency.
- **Contextual Exponence:** The diversity of neighboring tokens each token encounters, shedding light on semantic relationships.
- **Zipf's law:** The inverse proportionality between token frequency and rank, indicating linguistic regularity.
- **Brevity law:** The tendency of frequently used tokens to be shorter.
- **Heaps' law:** The growth of vocabulary size with dataset size, but at a decreasing rate.
- **Menzerath's law:** The inverse relationship between a protein sequence's length and its tokens' length.

3. EXPERIMENTS

In the experiments section of our study, we utilized the UniRef50 dataset to evaluate the performance of three subword tokenization methods: Byte-Pair Encoding, WordPiece, and SentencePiece-Unigram. We implemented these tokenizers at varying vocabulary sizes of 400, 800, 1600, 3200, and 6400 to analyze their effectiveness under different conditions. Throughout the plots in the study, we referred to the tokenizers with the abbreviations 'bpe' for BPE, 'wp' for WordPiece, and 'spm' for SentencePiece.

Our analysis focused on several key metrics to assess the performance of each tokenizer, including shared token percentage, token length distribution, fertility, and contextual exponence. Additionally, we evaluated the compliance of each tokenizer with several linguistic laws: Zipf's Law, Brevity Law, Heaps' Law, and Menzerath's Law. These metrics and laws collectively helped us to determine the efficiency and applicability of each tokenizer in handling the complex structure of protein sequences. The insights gained from this comprehensive analysis aim to guide future developments in tokenizer technology for better modeling of protein sequences as language models.

3.1. Shared Token Percentages

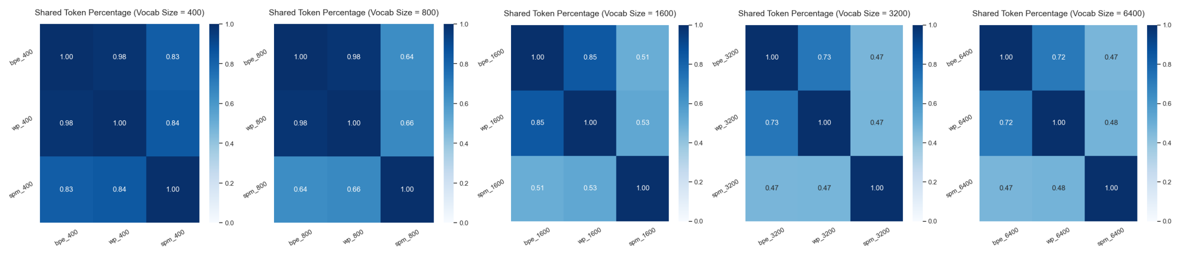
The series of plots (Figure 3.1 and Figure 3.2) illustrating the shared token percentage show interesting trends in the overlap of vocabularies generated by these methods. Here are key observations and their implications:

High Overlap at Smaller Vocab Sizes: At a vocabulary size of 400, there is a very high overlap between BPE and WordPiece (0.98), and a moderately high overlap between these two methods and SentencePiece (0.83 for BPE and 0.84 for WordPiece). This indicates that at smaller vocabulary sizes, the choice of tokenization method is less critical as they tend to generate similar vocabularies.

Divergence at Larger Vocab Sizes: As the vocabulary size increases, the overlap between BPE and WordPiece remains relatively high but begins to decrease (0.72 at vocab size 6400), suggesting a divergence in the vocabularies as more unique tokens are introduced by each method. The overlap between these methods and SentencePiece shows a more pronounced decrease (0.47 for both BPE and WordPiece at vocab size 6400), indicating that SentencePiece diverges significantly from the other two in terms of the tokens it prioritizes.

Consistent Trends: The consistency of the trend across all plots, where the overlap generally decreases as vocabulary size increases, suggests that each tokenizer's unique approach to tokenization becomes more apparent with larger vocabularies. SentencePiece consistently shows the least overlap with the other two, suggesting the known fact that it employs a fundamentally different strategy in token generation.

Overall Implications: The decreasing overlap with increasing vocabulary size highlights the importance of tokenizer choice when dealing with large vocabularies. Depending on the application's specific needs (such as the need for more unique tokens or better handling of rare tokens), one tokenizer may be preferable over the others.



(a) (b) (c) (d) (e)

Figure 3.1: The heatmaps of the percentage of shared tokens between different pairs of tokenizers across different vocabulary sizes.

3.2. Token Length Distribution

The series of plots (Figure 3.3, Figure 3.4, and Figure 3.5) illustrating the token length distribution provide detailed insights into the structural differences in how these tokenizers segment text. Here's a detailed analysis based on the observed data:

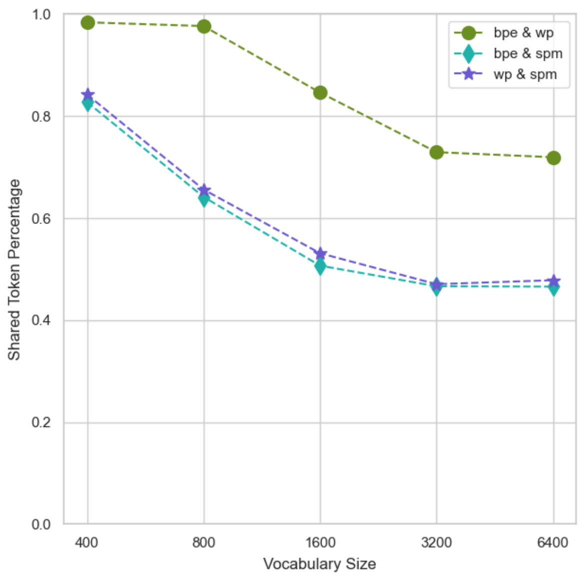
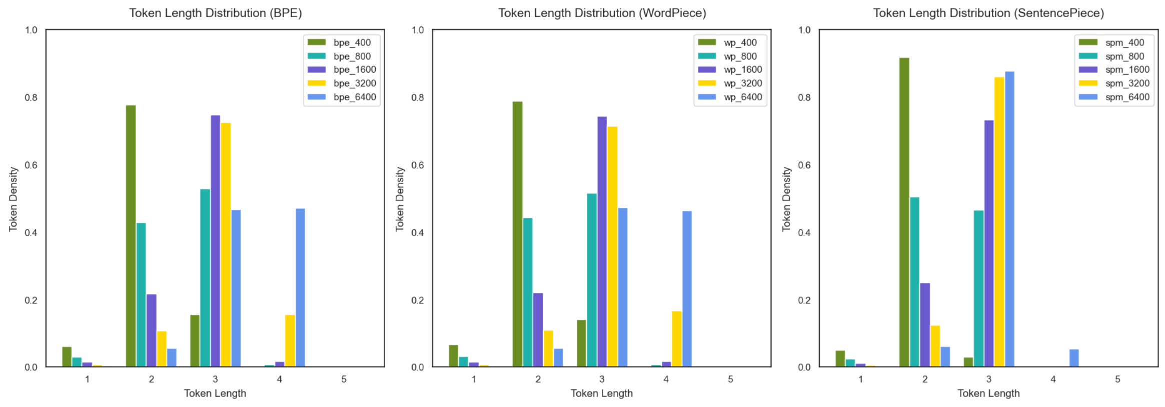


Figure 3.2: The plot of percentage of shared tokens between different pairs of tokenizers across different vocabulary sizes.

Token Length Distribution Over Various Vocabulary Sizes: As vocabulary sizes increase from 400 to 6400, the distribution of token lengths shifts noticeably. For smaller vocabularies, there is a higher density of shorter tokens, which gradually transitions towards longer tokens as the vocabulary size increases. This trend suggests that with larger vocabularies, the tokenizers are able to form longer, perhaps more meaningful, tokens from the sequences, potentially capturing more complex patterns or recurring segments.

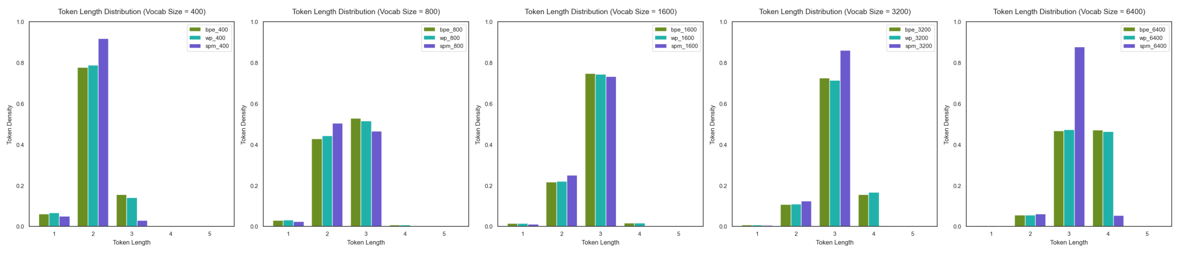
Analysis of Token Length Distribution Patterns: The predominant token lengths for smaller vocabularies are 1 and 2, but as the vocabulary size increases, length 3 and above tokens become more common. This reflects a natural progression in tokenizer behavior as they have more capacity (larger vocabulary) to create and utilize longer tokens.

Overall Implications: The shifting patterns in token length distribution have significant implications for the application of these tokenizers in tasks like protein sequence modeling. Longer tokens can capture more complex patterns, which might be crucial for accurately modeling biological sequences. The choice of tokenizer and vocabulary size should thus be influenced by the specific requirements of the task, including the complexity of the sequences being modeled and the computational resources available. These plots are crucial for understanding how the granularity of tokenization can affect both the performance and the interpretability of models trained on these tokens. They suggest that as we increase the vocabulary size, tokenizers tend to utilize their capacity to form longer tokens, which could be advantageous for capturing higher-order dependencies in the data.



(a) (b) (c)

Figure 3.3: The token length distribution of BPE, WordPiece and SentencePiece across different vocabulary sizes.



(a) (b) (c) (d) (e)

Figure 3.4: The token length distribution of BPE, WordPiece and SentencePiece across different vocabulary sizes.

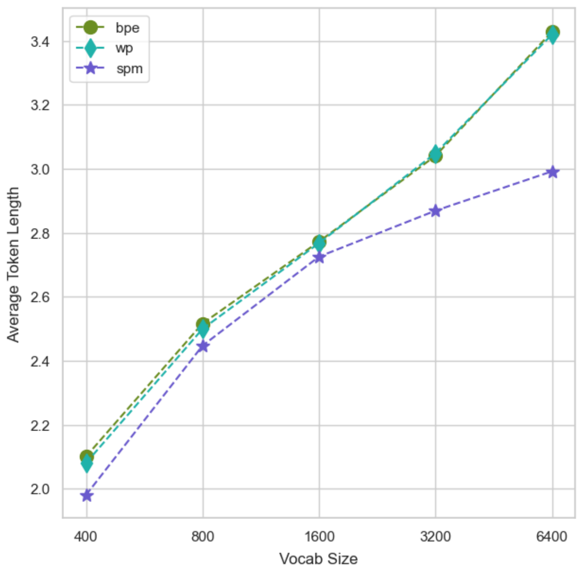


Figure 3.5: The plot of average token lengths of BPE, WordPiece and SentencePiece across different vocabulary sizes.

3.3. Fertility

Fertility is the average number of tokens required to represent a protein sequence, indicating encoding efficiency. Here are the observations from the Figure 3.6:

Decreasing Fertility Across All Methods: For all tokenization methods, fertility decreases as vocabulary size increases. This indicates that with larger vocabularies, fewer tokens are required to represent a protein sequence, suggesting an increase in encoding efficiency.

Comparison of Tokenization Methods: BPE starts with the highest fertility at a vocabulary size of 400, meaning it initially uses more tokens to represent sequences compared to the other methods. However, its fertility decreases sharply as the vocabulary size increases, becoming more efficient at larger vocabularies. WordPiece shows a similar trend to BPE but begins with slightly lower fertility at smaller vocabulary sizes. Its decrease in fertility is also steady, indicating a consistent improvement in encoding efficiency as vocabulary size grows. SentencePiece demonstrates the lowest fertility across almost all vocabulary sizes, indicating it generally requires fewer tokens to represent sequences. This suggests that SentencePiece might be the most efficient in encoding sequences from the outset, particularly at larger vocabulary sizes. Even though SentencePiece has the shortest average token length, it is interesting to see that SentencePiece has the best fertility score.

Overall Implications: The trend of decreasing fertility with increasing vocabulary size is expected and desirable in tokenization, as it suggests that tokenizers are able to represent more information with fewer tokens as they have more token options to choose from. The notable efficiency of SentencePiece, especially at larger vocabularies, may make it particularly suitable for applications where model size and speed are crucial, as it efficiently packs more information into fewer tokens. The choice between BPE, WordPiece, and SentencePiece should consider the specific requirements of the task, such as the balance between model complexity, training time, and runtime efficiency.

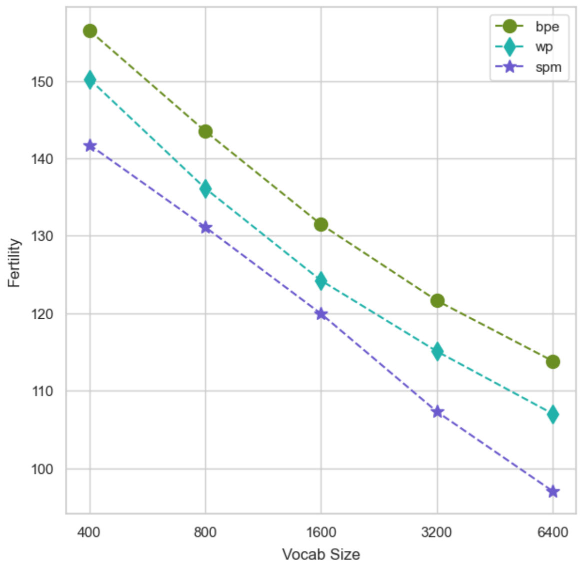


Figure 3.6: The plot for fertility of BPE, WordPiece and SentencePiece across different vocabulary sizes.

3.4. Contextual Exponence

Contextual expoenence is the diversity of neighboring tokens each token encounters in a 5-width window, shedding light on semantic relationships. The series of plots in Figure 3.7 provides a detailed view of how diverse the token interactions are. Here's a detailed analysis:

Increased Vocabulary Size Leads to Greater Diversity: As the vocabulary size increases, the number of distinct neighbors each token encounters generally increases. This suggests that larger vocabularies allow tokens to be used in more varied contexts, possibly enhancing the semantic richness captured by the models.

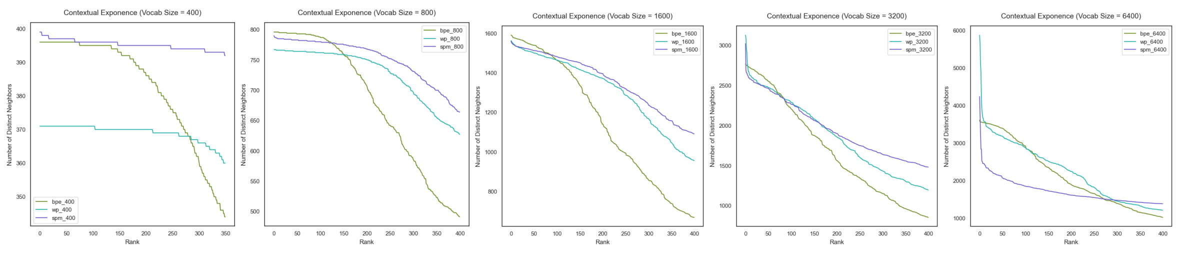
Initial Peak and Decline: Each plot shows an initial peak in the number of distinct neighbors, followed by a sharp decline as rank increases. This pattern indicates that some tokens are very versatile and appear in many contexts, while others are more specialized.

Vocabulary Sizes: At vocabulary size 400, all three tokenizers start relatively similar, but BPE shows a slightly higher context diversity. At vocabulary size 800 and up, the gap in contextual diversity between BPE and the other tokenizers becomes more pronounced, with BPE maintaining higher diversity across ranks. At larger vocab sizes (3200 and 6400), the diversity for SentencePiece declines sharply compared to BPE and WordPiece, suggesting that while SentencePiece may be more efficient in terms of token usage (as seen in fertility), it might lack contextual versatility.

Comparison of Tokenization Methods: BPE tends to have a higher number of distinct neighbors initially across all vocabulary sizes compared to WordPiece and SentencePiece. This might suggest that BPE tokens have a broader usage in varied contexts, which could be beneficial for capturing diverse linguistic patterns. Also, after a few dozen tokens, BPE's context counts dip below the other two's, a trend that continues throughout the vocabulary, making up a more contextually coherent set.

WordPiece exhibits a moderately high level of contextual diversity but consistently falls below BPE. This could indicate a slightly more focused use of tokens within specific contexts, possibly due to its underlying optimization strategies. SentencePiece shows the least diversity in terms of distinct neighbors, especially noticeable at higher vocabulary sizes. This may reflect SentencePiece's strategy of optimizing token usage based on frequency and probability, leading to a more predictable, but less contextually diverse, token usage.

Overall Implications: These plots underscore the importance of considering both the diversity and efficiency of tokenization methods in semantic modeling. BPE's higher contextual diversity could make it more suitable for tasks requiring nuanced understanding of context and semantics. In contrast, SentencePiece, while efficient, might be better suited for applications where a predictable and compact representation is more critical than contextual diversity.



(a) (b) (c) (d) (e)

Figure 3.7: Contextual exponence plots of BPE, WordPiece and SentencePiece across different vocabulary sizes.

3.5. Zipf's Law

Zipf's Law is a statistical principle that describes the distribution of frequencies of elements in a dataset. It is named after the linguist George Zipf, who first observed this pattern in the distribution of word frequencies in natural languages. Zipf's Law is a power-law distribution and is often expressed mathematically as:

$$f(r) = \frac{1}{r^{\alpha}}$$

where:

- $f(r)$ is the frequency of the element ranked at
- α is a constant typically close to 1, and
- r is the rank of the element.

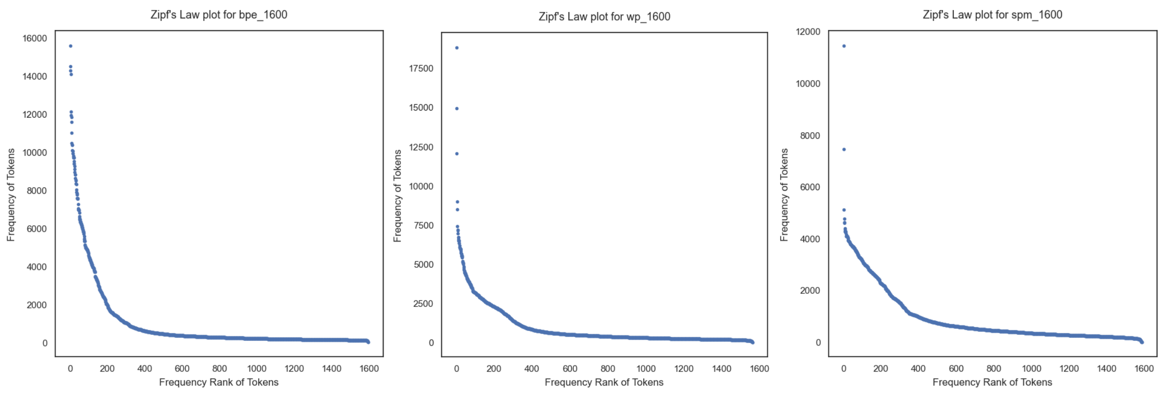
In simpler terms, Zipf's Law states that the frequency of a particular element is inversely proportional to its rank. This means that a few elements occur very frequently, while the majority of elements occur infrequently. The law is often applied to the distribution of words in texts, where a small number of words (like common words such as "the," "and," etc.) are used frequently, while the majority of words are used less frequently.

The Zipf's Law plots (Figure 3.8, and Figure 3.9) for the BPE, WordPiece, and SentencePiece tokenizers at vocabulary size 1600 show how the frequency of token usage declines with rank. Zipf's Law, which posits that the frequency of any word is inversely proportional to its rank, is somewhat adhered to by all three tokenization methods, although the slopes vary, indicating different levels of steepness in the frequency distribution.

BPE plots shows a steep initial drop in frequency, which then flattens out. This suggests that a few tokens are extremely common, with a long tail of infrequently used tokens. The slope calculated in the log-log plot is closer to the ideal slope of -1, indicating a relatively strong adherence to Zipf's Law. Similar to BPE, WordPiece shows a pronounced decline. The slope is slightly less steep compared to BPE, which may indicate a slightly more uniform distribution across the top ranks but still maintains a substantial drop. The decline in token frequency for SentencePiece is less steep than for BPE and WordPiece, indicating a broader spread of token usage across the vocabulary. The slope is significantly different from -1, suggesting a deviation from the ideal Zipfian distribution.

The plot of the slopes across different vocabulary sizes (Figure 3.10) illustrates how the adherence to Zipf's Law changes. For instance, as vocabulary size increases, the slopes for all tokenizers show variations, indicating changes in how evenly tokens are distributed or utilized across different sizes. This could reflect different efficiencies in token utilization, where smaller vocabularies might force a more concentrated use of available tokens, thus adhering more closely to Zipf's Law. Conversely, larger vocabularies allow for a more spread-out usage, potentially deviating from the law.

These findings are crucial for understanding how each tokenizer manages the balance between token frequency and vocabulary size, influencing decisions on tokenizer selection based on the specific needs of protein sequence modeling or other applications.



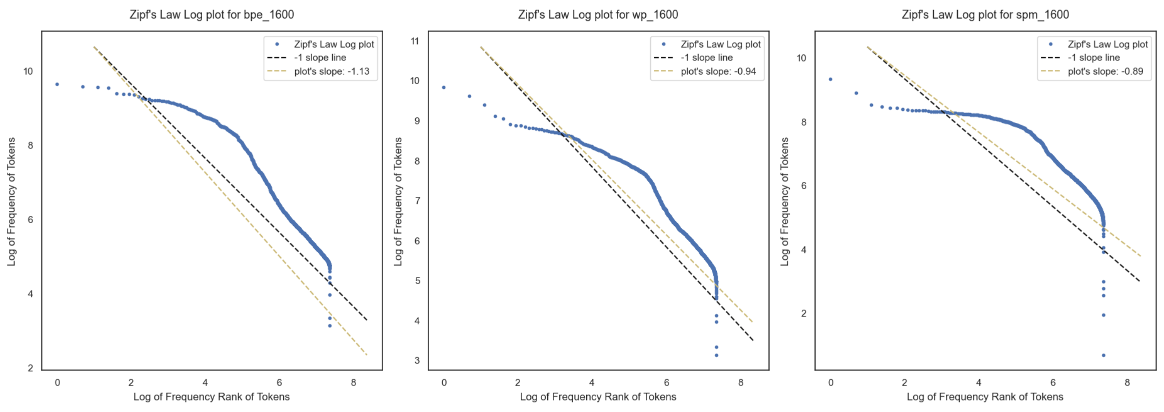
(a) (b) (c)

Figure 3.8: Zipf's law plots of BPE, WordPiece and SentencePiece for vocabulary size of 1600.

3.6. Brevity Law

Brevity Law (or Zipf's Law of Abbreviation) is a concept related to the distribution of abbreviations or acronyms in a language. It is an extension of Zipf's Law, which describes the distribution of frequencies for elements in a dataset, such as words in a natural language.

In the context of abbreviations, Zipf's Law of Abbreviation suggests that a small number of abbreviations are used very frequently, while a large number of abbreviations



(a) (b) (c)

Figure 3.9: Zipf's law plots (log of frequency rank of tokens vs log of frequency of tokens) of BPE, WordPiece and SentencePiece for vocabulary size of 1600.

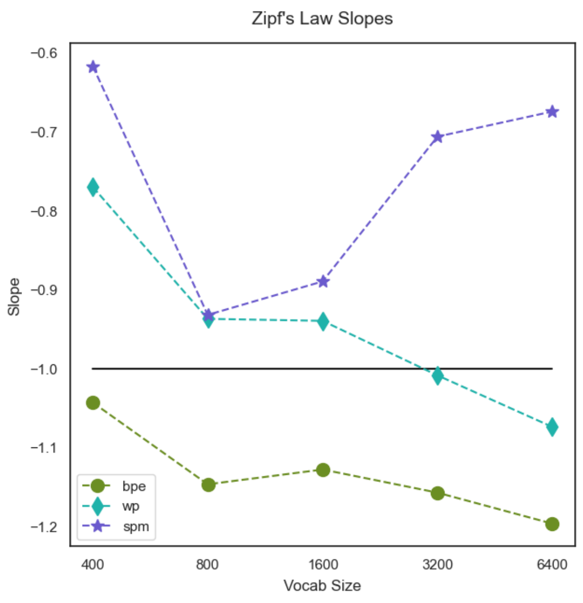


Figure 3.10: The plot for Zipf's law slopes of BPE, WordPiece and SentencePiece across different vocabulary sizes.

are used infrequently. In other words, Brevity Law qualitatively states that the more frequently a word is used, the shorter that word tends to be, and vice versa; the less frequently a word is used, the longer it tends to be.

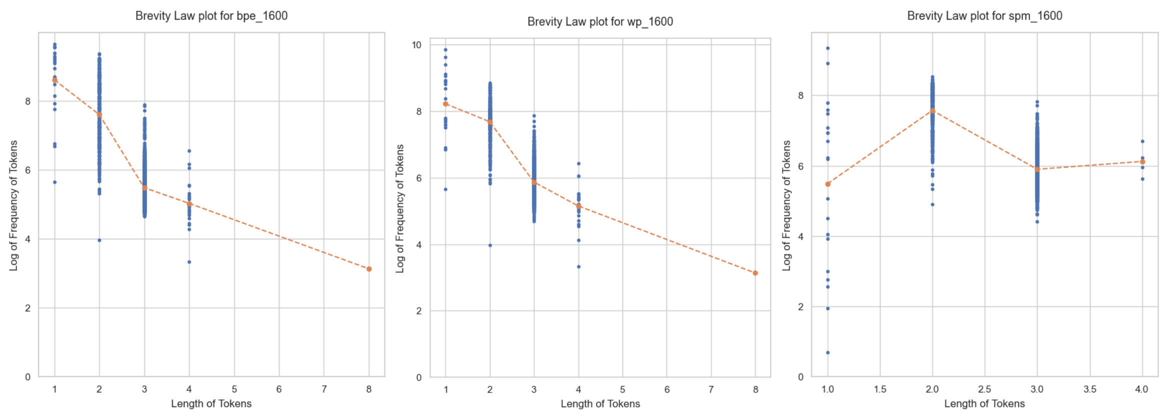
The plots (Figure 3.11, Figure 3.12, and Figure 3.13) for the Brevity Law showcase the relationship between the token length and frequency of tokens. Brevity Law, which posits that shorter words are used more frequently, is generally adhered to across all tokenizers and vocabulary sizes. However, as vocabulary size increases, the maximum log frequency of tokens begins at a higher point but decreases more steeply, indicating that tokens become less frequent as they lengthen.

At lower vocabulary sizes (400 and 800), the drop in frequency with increasing token length is less steep compared to higher vocabulary sizes, suggesting a denser usage of shorter tokens. At vocabulary size 1600 and above, there is a noticeable shift in the plots where longer tokens (beyond length 3) drastically drop in frequency. For vocabulary sizes 3200 and 6400, the plots illustrate a sharp decrease in log frequency as token length increases from 1 to around 4 or 5, after which the decrease plateaus, indicating that longer tokens are significantly less frequent. Among the tokenizers, SentencePiece often shows a sharper decline in frequency with increasing token length compared to BPE and WordPiece, suggesting that SentencePiece may generate longer tokens less frequently or has a distribution that favors shorter, more frequent tokens.

These trends illustrate how token length and frequency distributions adjust with varying vocabulary sizes and tokenizer strategies, impacting the compactness and efficiency of the encoding in representing protein sequences under the Brevity Law.

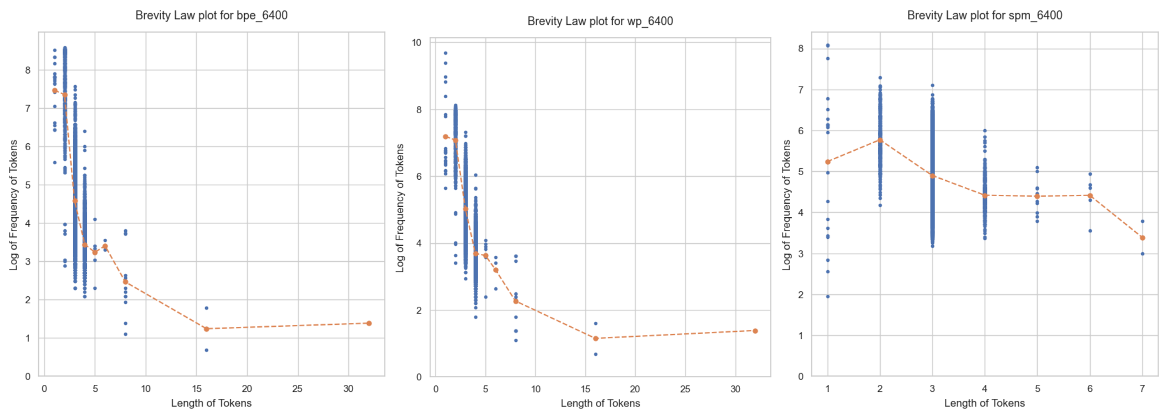
3.7. Heap's Law

Heap's Law, named after Stuart Heap, is an empirical formula that describes the relationship between the size of a vocabulary (the number of distinct words) in a document or a collection of documents and the total number of words. Heap's Law is



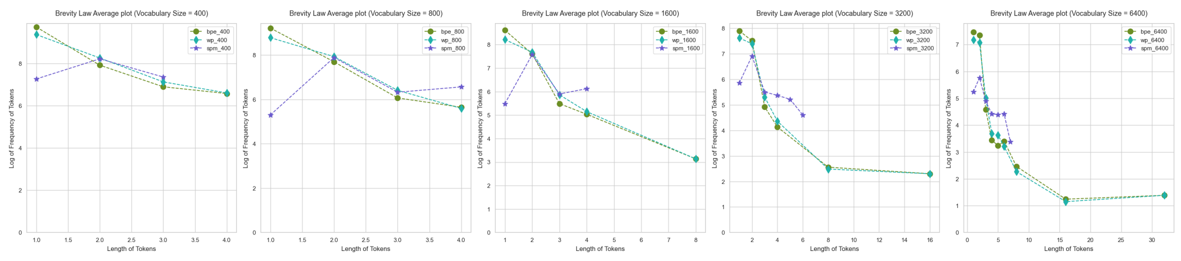
(a) (b) (c)

Figure 3.11: Brevity law plots of BPE, WordPiece and SentencePiece for vocabulary size of 1600.



(a) (b) (c)

Figure 3.12: Brevity law plots of BPE, WordPiece and SentencePiece for vocabulary size of 6400.



(a) (b) (c) (d) (e)

Figure 3.13: Average of Brevity law plots of BPE, WordPiece and SentencePiece across different vocabulary sizes.

commonly applied in natural language processing and information retrieval.

The formula is expressed as:

$$V(n) = K \cdot n^{\beta}$$

where:

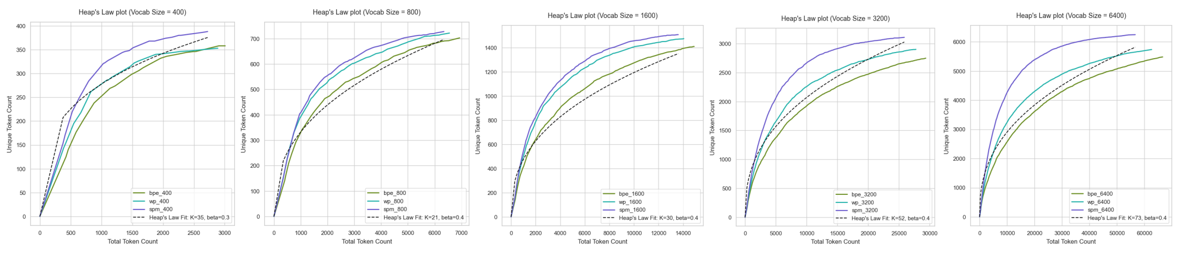
- $V(n)$ is the estimated vocabulary size when the document or collection contains n words,
- K is a constant, typically in the range of 10 to 100.
- β is an exponent, typically in the range of 0.4 to 0.6.

In simpler terms, Heap's Law suggests that as the size of a document or dataset increases, the vocabulary size (the number of unique words) also increases, but at a decreasing rate. The exponent β determines the rate of growth. The larger β is, the slower the vocabulary size grows concerning the document size. Heap's Law is often used to estimate the vocabulary size needed for information retrieval systems or to assess the richness of vocabulary in a given text. It highlights the observation that as more text is added, new words continue to be introduced, but the rate of introduction decreases over time.

The plots in Figure 3.14 representing Heaps' Law across different tokenizers and vocabulary sizes exhibit a typical behavior where the number of unique tokens increases with the total token count but at a diminishing rate. All tokenizers follow Heaps' law closely, indicating all of them has efficient vocabulary usage.

3.8. Menzerath's Law

Menzerath's law, also known as Menzerath–Altmann law or Menzerath's law of linguistic organization, is an empirical linguistic principle that describes the relation-



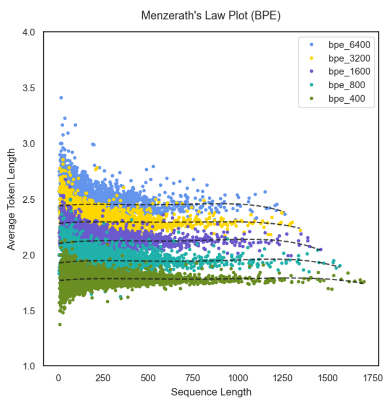
(a) (b) (c) (d) (e)

Figure 3.14: Heap's law plots of BPE, WordPiece and SentencePiece across different vocabulary sizes.

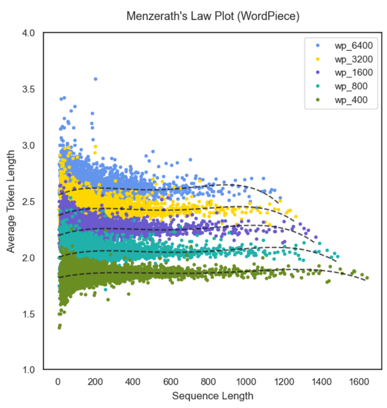
ship between the size of linguistic constructs (such as words, phrases, or clauses) and the size of their constituents (subparts). The law is often expressed as follows: ”The larger the whole, the smaller the constituents.” For instance, the longer a protein (in tokens), the shorter its tokens (in amino acids). Here are some observations for the Menzerath's Law plots (Figure 3.15 and Figure 3.16):

For all tokenizers and vocabulary sizes, there's a noticeable trend where the average token length decreases as the sequence length increases. This supports Menzerath's Law, indicating that as sequences become longer, they are composed of shorter tokens. The plots show a fairly consistent pattern across the three tokenization methods, though there are some variations in the spread and density of points. As the vocabulary size increases, the starting average token length for shorter sequences appears higher, but still follows a decreasing trend. This might suggest that with more tokens available, initial segments can use longer tokens effectively, but as sequences extend, the need to revert to shorter tokens increases. The dispersion of points around the trend line is relatively wide, indicating variability in how different sequences are tokenized. This could be influenced by the inherent variability in the protein sequences or the tokenization algorithm's strategies in handling different sequence contexts.

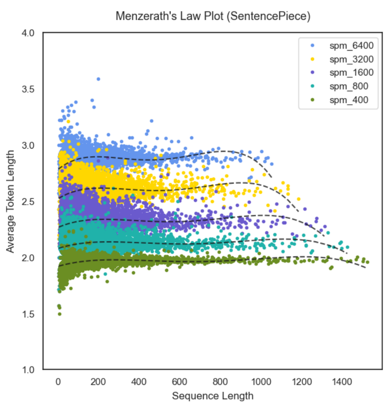
Overall, none of the tokenizers show a clear decline in average token length as sequence length increases. There is very little variety in average token length, suggesting that we can increase the average token length by increasing vocabulary size which can make the distribution more adhering to Menzerath's law.



(a)

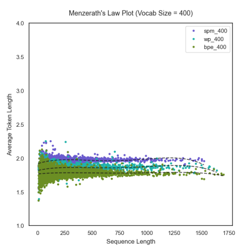


(b)

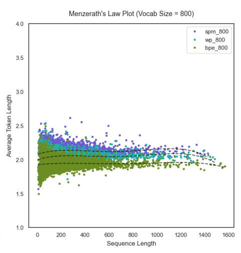


(c)

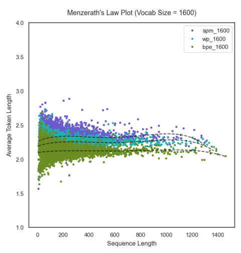
Figure 3.15: Menzerath's law plots of BPE, WordPiece and SentencePiece across different vocabulary sizes.



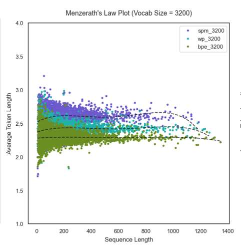
(a)



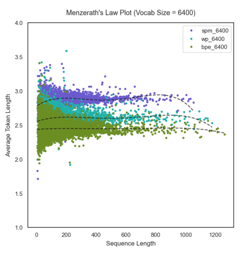
(b)



(c)



(d)



(e)

Figure 3.16: Menzerath's law plots of BPE, WordPiece and SentencePiece across different vocabulary sizes.

4. RELATED WORK

In bioinformatics, many studies evaluate tokenization methods within protein language models (pLMs) using various downstream tasks. While these approaches help identify high-performing methods, they fail to assess whether the chosen tokenizers are well-suited for encoding protein sequences in language models. Our study is one of the first to focus specifically on understanding how effective NLP tokenizers are for protein language models.

4.1. Bioinformatics

Ding and Steinhardt [6] identify and quantify a species bias in pLM likelihoods due to uneven sequence sampling, revealing that certain species are systematically favored, resulting in higher likelihoods independent of the protein sequence itself. By calculating an Elo rating for each species based on pre-trained pLM likelihoods, they show that this bias reduces thermostability and salt tolerance for protein designs starting from under-represented species.

Ieremie et al. [7] explore the efficacy of pLMs trained on reduced amino acid alphabets. They find that pLMs trained on the full alphabet capture fine evolutionary details that are lost with reduced alphabets. Although pLMs systematically cluster amino acids, small alphabets distort evolutionary information, leading to degraded performance. Furthermore, reduced alphabets struggle to distinguish between mutations within clusters, limiting their ability to detect fine-grained sequence variations.

Rao et al. [8] introduce five downstream tasks for protein language modeling. While their work doesn't compare different tokenizers and uses outdated models, it sets a benchmark for future studies by establishing a standard evaluation framework.

Tan et al. [9] assess how protein language models perform across various downstream tasks with different vocabulary sizes and tokenization methods. Comparing per-amino acid, BPE, and Unigram methods (with vocabulary sizes ranging from 50 to 3200), the study shows that vocabulary size significantly affects protein representation. Larger vocabularies often worsen optimization in structure prediction datasets, with vocabulary sizes above 800 generally degrading performance.

Similarly, Dotan et al. [10] evaluate the impact of different tokenization methods and vocabulary sizes on PLM performance across downstream tasks. They compare per-amino acid, BPE, WordPiece, and Unigram methods (with vocabulary sizes ranging from 100 to 3200), demonstrating that advanced tokenization strategies can substantially reduce sequence lengths and improve performance. They suggest that tasks involving proteins with similar domains may benefit more from task-specific tokenizers.

4.2. NLP

Petrov et al. [11] demonstrate that disparities in tokenization lengths across languages can lead to unfairness, with differences of up to 15 times in some cases. They argue for multilingually fair subword tokenizers that produce similar encoded lengths for equivalent content across languages. They also discuss fairness implications such as cost, latency, and long-context processing.

Ali et al. [12] show that the choice of tokenizer significantly impacts model performance and training costs. Common metrics like fertility and parity don't always predict downstream performance, and English-centric tokenizers cause performance degradation in multilingual models. They advocate for balanced tokenizers across languages to achieve consistent fertility and parity scores.

Gutierrez-Vasques et al. [13] hypothesize that subword properties derived from BPE compression vary based on a language's morphological type. By creating language vectors based on subword productivity and idiosyncrasy, they demonstrate that subword properties can distinguish morphological language types using raw text.

Incorporating context into subword vocabularies is another important consideration. Yehezkel and Pinter [14] argue that traditional subword tokenizers like BPE and Unigram primarily focus on word frequency statistics, ignoring contextual information. Their SAGE tokenizer incorporates contextual fitness using a modified SkipGram objective, leading to more context-aware subwords. Compared to BPE, SAGE performs better and represents them using token length, frequency, fertility, and contextual exponent.

Lastly, Hiraoka et al. [15] propose a novel method for jointly optimizing a tokenizer and a downstream model to enhance performance across various NLP tasks. By using loss values from the downstream model to train the tokenizer, they enable task-specific tokenization. This method, applicable as a post-processing step for already trained models, shows improvements in text classification and machine translation tasks.

Overall, this body of work provides a comprehensive overview of the impact of tokenization methods on protein sequence analysis and language models, offering a strong foundation for our study focused on the comparative performance of different subword tokenization methods in protein language understanding.

5. CONCLUSION AND FUTURE WORK

In this work, we evaluated Byte-Pair Encoding (BPE), WordPiece, and SentencePiece across varying vocabulary sizes (400, 800, 1600, 3200, and 6400) in the context of protein sequences using the UniRef50 dataset. We assessed each tokenizer through shared token percentages, token length distribution, fertility, and contextual exponence, in addition to examining their behavior under linguistic laws including Zipf's, Brevity, Heaps', and Menzerath's laws. Results illustrate distinct characteristics and efficiencies of BPE, WordPiece, and SentencePiece across various tokenization metrics, each showing strengths and weaknesses depending on the metric and vocabulary size.

Our analysis highlights notable differences in the makeup of vocabularies, distributions of token lengths, and the efficiency of sequence representation among the tokenizers studied. Although variations are evident, each tokenizer conforms to Zipf's and Heaps' laws. BPE and WordPiece, however, align more closely with the Brevity law than SentencePiece, which does not adhere as well to this principle. None of the tokenizers effectively follow Menzerath's Law. Additionally, BPE is distinguished by its better context usage, WordPiece strikes a balance between efficient encoding and diversity of tokens, and SentencePiece stands out due to its encoding efficiency and distinct characteristics in token frequency distribution.

Overall, none of the tokenizers fully meet all the statistical metrics or adhere to the linguistic laws assessed. This indicates the need for a more effective tokenizer specifically tailored for protein sequences, especially if we continue to model proteins as analogous to linguistic systems.

Here are the things that we plan to do in the following months:

- Repeat experiments for larger vocabulary sizes.
- Include different metrics such as parity, productivity, and idiosyncracy.

- Compare these results to our proposed Evolutionary Subword Tokenization approaches.
- Treat species as different languages and work on specie-focused tokenizers. Train specie-specific tokenizers and
  - apply tests that are available to multilingual approaches.
  - combine them for a multi-specie tokenizer and compare it with the generic approach.

REFERENCES

1. Sennrich, R., B. Haddow and A. Birch, “Neural machine translation of rare words with subword units”, *arXiv preprint arXiv:1508.07909*, 2015.

2. Wu, Y., M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation”, *arXiv preprint arXiv:1609.08144*, 2016.

3. Kudo, T. and J. Richardson, “Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing”, *arXiv preprint arXiv:1808.06226*, 2018.

4. Kudo, T., “Subword regularization: Improving neural network translation models with multiple subword candidates”, *arXiv preprint arXiv:1804.10959*, 2018.

5. Consortium, T. U., “UniProt: the universal protein knowledgebase in 2021”, *Nucleic Acids Research*, Vol. 49, No. D1, pp. D480–D489, 11 2020, `https://doi.org/10.1093/nar/gkaa1100`.

6. Ding, F. and J. N. Steinhardt, “Protein language models are biased by unequal sequence sampling across the tree of life”, *bioRxiv*, pp. 2024–03, 2024.

7. Ieremie, I., R. M. Ewing and M. Niranjan, “Protein language models meet reduced amino acid alphabets”, *Bioinformatics*, Vol. 40, No. 2, p. btae061, 2024.

8. Rao, R., N. Bhattacharya, N. Thomas, Y. Duan, P. Chen, J. Canny, P. Abbeel and Y. Song, “Evaluating protein transfer learning with TAPE”, *Advances in neural information processing systems*, Vol. 32, 2019.

9. Tan, Y., M. Li, P. Tan, Z. Zhou, H. Yu, G. Fan and L. Hong, “PETA: Eval-

uating the Impact of Protein Transfer Learning with Sub-word Tokenization on Downstream Applications", *arXiv preprint arXiv:2310.17415*, 2023.

10. Dotan, E., G. Jaschek, T. Pupko and Y. Belinkov, "Effect of tokenization on transformers for biological sequences", *Bioinformatics*, Vol. 40, No. 4, p. btae196, 2024.

11. Petrov, A., E. La Malfa, P. Torr and A. Bibi, "Language model tokenizers introduce unfairness between languages", *Advances in Neural Information Processing Systems*, Vol. 36, 2024.

12. Ali, M., M. Fromm, K. Thellmann, R. Rutmann, M. Lübbering, J. Leveling, K. Klug, J. Ebert, N. Doll, J. S. Buschhoff *et al.*, "Tokenizer Choice For LLM Training: Negligible or Crucial?", *arXiv preprint arXiv:2310.08754*, 2023.

13. Gutierrez-Vasques, X., C. Bentz and T. Samardžić, "Languages through the looking glass of bpe compression", *Computational Linguistics*, Vol. 49, No. 4, pp. 943–1001, 2023.

14. Yehezkel, S. and Y. Pinter, "Incorporating context into subword vocabularies", *arXiv preprint arXiv:2210.07095*, 2022.

15. Hiraoka, T., S. Takase, K. Uchiumi, A. Keyaki and N. Okazaki, "Joint optimization of tokenization and downstream model", *arXiv preprint arXiv:2105.12410*, 2021.