Modern Byte Pair Tokenizers are Zipfian

Jack Hanke
Northwestern University

Daniel Plotkin
Northwestern University

Nicole Birova
Northwestern University

Abstract

A majority of large language models ingest word fragments called *tokens* produced by a data compression algorithm known as byte pair encoding. This algorithm groups high-frequency letter pairings in natural language into individual units. A natural question is whether the tokens produced by this pairing process deviate significantly from the source language's frequency distribution. Zipf famously showed that many natural language's frequency distribution follows a power law, commonly known as Zipf's Law. We examine two modern tokenizer's adherence to Zipf's law at the token level. We demonstrate that these tokenizers are Zipfian on two corpuses, and speculate as to why this is.

1 Introduction

George Zipf in his hallmark *The Psycho-Biology of Language* (Zipf, 1935) introduced the remarkable trend that the word frequencies in many human languages exhibit the same power law distribution. Specifically, Zipf said that the frequency that a word appears in some large corpus is proportional to the word's frequency rank. Mathematically, Zipf's laws states

$$\text{word frequency} \propto \frac{1}{\text{word rank}}. \tag{1}$$

A more descriptive model of word frequency that is more commonly referenced in linguistics literature is the Zipf-Mandelbrot distribution

$$\text{word frequency} \propto \frac{1}{(\text{word rank} + b)^a}. \tag{2}$$

where a, b are fitted parameters. We say a distribution that follows the trend in Equation 2 with $a \approx 1$ is *Zipf distributed*, or simply *Zipfian*.

The accuracy of the trend in Equation 2 has been examined in 10^8 English words in (Ferrer-i Cancho and Solé, 2000), over 50 languages in (Yu et al., 2018), and written-versus-spoken corpuses in (Lin et al., 2015). Each of these studies demonstrate that Zipf's law is generally exhibitted for common and somewhat-uncommon words, but rare words (words with high rank) appear less frequently than predicted. This deviation is shown to be statistically significant, and appears as two trendlines in the log-log plot. In English corpuses, this transition is found around the 10^5*-th* ranked word.[1] The authors further explore the linguistic relavence and universality of these multiple regimes of words.

Nearly a century after Zipf's discovery, large language models (LLMs) generate text comparable to human communication. However, unlike humans, LLMs digest text using a fixed vocabulary of word fragments called *tokens*. The mapping between natural language and tokens is most commonly computed using the *byte pair encoding algorithm* (Gage, 1994) over some large training corpus. Byte pair encoding identifies the most frequent pair of letters, and creates a new token in the "token vocabulary" to replace that pair. Iterating this procedure until some fixed vocabulary size is reached creates the vocabulary of tokens. Finally, any sequence that does not map to one of the derived tokens is labelled with the <unk> token, standing for unknown.

When considering these ideas in tandem, a natural question arises: given a corpus that appears Zipfian, how does the tokenization process affect this trend?

2 Methods

To explore this question, we choose two modern tokenizers to analyze. We choose the tokenizer for the RoBERTa language model (Liu et al., 2019), as the training data is publicly known. This allows us to conduct frequency analyses on corpuses that are

[1] Some of these studies also find that extremely common words appear slighly more commonly than the Zipfian prediction.

Rank	Words		RoBERTa		GPT-4	
1	.	the	<s>	␣	“	,
2	,	of	\n	\n	i	␣the
3	the	and	<\s>	.	he	.

Table 1: Summary of most common words & tokens for each corpus, where the left subcolumn is bookscorpus and the right subcolumn MiniPile, including control tokens and punctuation

on and off-distribution for the tokenizer. We also choose to compare our results with the tokenizer for the GPT-4 language model (OpenAI et al., 2024) as a representative of an industry-grade tokenizer.

We choose two corpuses based on RoBERTa's training data. We choose the 4.4GB bookscorpus dataset (Zhu et al., 2015), as RoBERTa's tokenizer was trained (in-part) on this corpus. For our off-distribution corpus, we choose the 5.6GB-tying MiniPile dataset (Kaddour, 2023), which was released after the RoBERTa model was released. Note that it is likely that the GPT-4 tokenizer was trained (in-part) on both of these corpuses.

We compute the word and token frequency for both tokenizers on each corpus, where we define a word as anything separated by a space. We then compute the data's Kolmogorov-Smirnov goodness-of-fit statistic for the Zipf distribution, and compare to that for an exponential and log-normal distribution. We do not remove any tokens from consideration unless otherwise stated.

3 Results

We find that token vocabularies are remarkably Zipfian, excluding extremely rare tokens. This holds true for both tokenizers and both corpuses studied, including out-of-distribution text. In the log-log plots in Figure 1, we show sharp declines from the Zipfian trend only for the final few tokens in the RoBERTa tokenizer for both corpuses and the GPT-4 tokenizer for the MiniPile corpus. However, the GPT-4 tokenizer trend on the bookscorpus seems to deviate more dramatically, similar to the multiple regime studies.

The lowest rank words and tokens for each tokenizer and each corpus, including control tokens and punctuation, are summarized in Table 1.

The Kolmogorov-Smirnov test statistics for each considered distribution are summarized in Table 3.

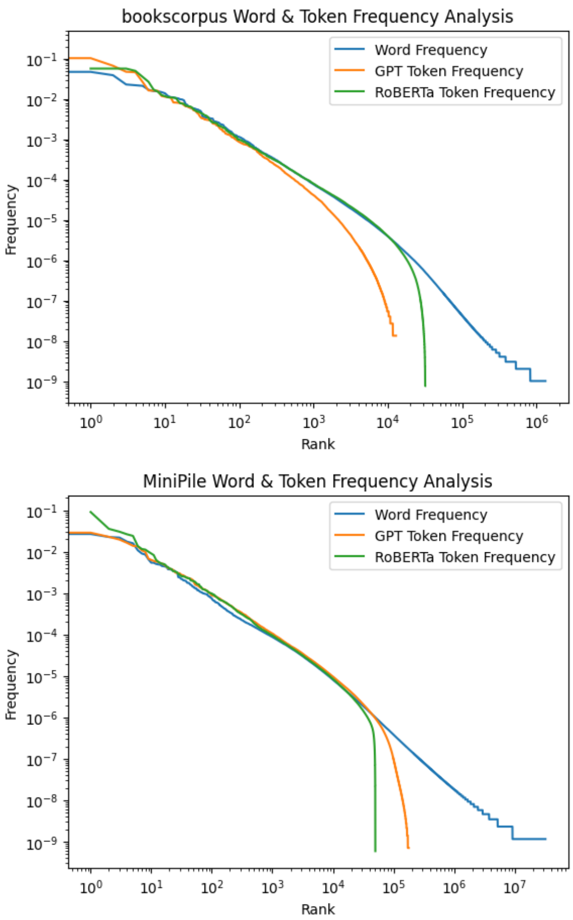

Figure 1: Log-log plots of word & token rank vs word & token frequency

4 Conclusions

Despite

5 Future Work

TODO

How do token vocabularies differ from english vocabularies? Fixed vocabulary size, catchall token, functional tokens

Rank	Words		RoBERTa		GPT-4	
1	the	the	the	the	i	the
2	to	of	to	of	he	of
3	i	and	and	and	she	and

Table 2: Summary of most common words & tokens for each corpus, where the left subcolumn is bookscorpus and the right subcolumn MiniPile, *not* including control tokens and punctuation

Distribution	Words		RoBERTa		GPT-4	
Zipf						
Exponential	0.874	0.808	0.580	0.482	0.718	0.563
log-normal	0.379	0.451	0.032	0.050	0.397	0.283

Table 3: Kolmogorov-Smirnov test statistics for distribution fit, where the left subcolumn is bookscorpus and the right subcolumn MiniPile

References

Ramon Ferrer-i Cancho and Ricard Solé. 2000. Two regimes in the frequency of words and the origins of complex lexicons: Zipf's law revisited. *Santa Fe Institute, Working Papers*.

Philip Gage. 1994. A new algorithm for data compression. *C Users J.*, 12(2):23–38.

Jean Kaddour. 2023. The minipile challenge for data-efficient language models. *Preprint*, arXiv:2304.08442.

Ruokuang Lin, Qianli D. Y. Ma, and Chunhua Bian. 2015. Scaling laws in human speech, decreasing emergence of new words and a generalized model. *Preprint*, arXiv:1412.4846.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Shuiyuan Yu, Chunshan Xu, and Haitao Liu. 2018. Zipf's law in 50 languages: its structural pattern, linguistic interpretation, and cognitive motivation. *Preprint*, arXiv:1807.01855.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *Preprint*, arXiv:1506.06724.

G. K. Zipf. 1935. *The Psycho-Biology of Language*. Houghton Mifflin, Boston.