UNIVERSITY OF CALIFORNIA,
MERCED

DISSERTATION

Mathematical Models of Natural Language Evolution

by

Alex John Quijano

A technical report submitted
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Applied Mathematics

2021

Committee Members:
Professor Suzanne S. Sindi, Chair
Professor Rick Dale
Professor Arnold D. Kim
Professor Roummel F. Marcia

The Dissertation of Alex John Quijano is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Committee Member:

Professor Roummel F. Marcia

Committee Member:

Professor Arnold D. Kim

Committee Member:

Professor Rick Dale

Committee Chair / Research Advisor:

Professor Suzanne S. Sindi

Date

Dedication

This dissertation is dedicated to my sister Philline Quijano, mom Maria Farley, and my grandparents Cleopatra Veloso and Antonio Veloso.

Acknowledgments

First, I am grateful to my advisor Prof. Suzanne Sindi for her unending encouragement and guidance. Her mentorship provided me the tools and skills for research and guided me to be an effective and empathetic educator. Our shared curiosity and interest in language evolution modeling - and mathematical modeling in general - has been intellectually stimulating and rewarding.

Second, I thank my dissertation committee members. Prof. Rick Dale has been very helpful on expanding my breadth of knowledge in linguistics. Prof. Arnold Kim and Prof. Roummel Marcia have been very helpful on providing me valuable feedback for my work. Additionally, I am thankful for my committee's guidance in career prospects.

Third, I am grateful to my sister Philline Quijano and mom Maria Farley for supporting me throughout my educational and personal journey. My pursuit of obtaining a PhD degree has been a tremendously difficult challenge and I could not have done it without their support. Our travel adventures together and frequent chats have helped me recharge and relax from my occasionally stressful lifestyle. Moreover, I like to thank my stepdad Jim Farley for supporting me and my Mom, and providing us a wonderful home in Tennessee.

Fourth, I like to thank specifically Matea Santiago, Fabian Santiago, Jessica Taylor, and Omar DeGuchy for being such wonderful support network and I am grateful for our friendship and academic collegiality. I also like to thank Jamin Shih, Michelle Yeung, and May Kao Xiong for being such wonderful supportive friends and providing me the knowledge and awareness of social issues. I thank my dad Alexander Quijano for his support and patience. Furthermore, I thank my family and friends for providing me encouragement throughout my years in Graduate school.

Fifth, I like to say thanks to my mentors and research collaborators at Lawrence Livermore National Laboratory, Sam Nguyen, Juanita Ordonez, Brenda Ng, Robert Clayton Blake, and Ben C. Yee. I am grateful for the opportunities provided to me and the experience to work on your team.

Finally, I gratefully acknowledge the financial support from the Applied Mathematics Department at the University of California, Merced. Furthermore, I acknowledge the support from NSF INSPIRE Track 1 grant BCS-1344279 (PI: Prof. Suzanne Sindi), and the NSF RTG grant 1840265 (PI: Prof. Arnold Kim).

Contents

Mathematical Models of Natural Language Evolution

by

Alex John Quijano

Doctorate in Applied Mathematics

Suzanne S. Sindi, Chair

University of California, Merced

2021

Abstract

This dissertation investigates the ways that natural languages evolve and what it means in the overall cultural evolution of society. Computational and modeling advances have made possible to explore large-scale text data and test hypothesis of language evolution. Similar to biological systems, natural languages are evolving systems with words as its measurable units. Words have certain functions within a body of text to convey ideas and thought. The frequency distribution of these words can change based on how it is used at a particular point in time and context. This work incorporates two different sources of text data: 109 year's worth of digitized books and text taken from social media. Given large-scale diachronic corpora, this work focuses on the following topics:

1. Modeling word rank evolution utilizing statistical and data-driven modeling approaches.

2. Exploring the evolution of contextual semantics through the use of distributional semantics and word embedding models.

3. Evaluating the accuracy of reading comprehension tasks by using contemporary machine learning models.

The evolution of language presents a profound problem in natural language processing and cognitive science. The social and cultural aspect of language adds to the problem of how word meanings develop and change. Examining how language evolves, while drawing from both molecular biology and socio-cultural aspects, allows us to explore the ways word meanings form that are influenced by political and social changes.

List of Abbreviations and Symbols

SVD	Singular Value Decomposition
PCA	Principal Component Analysis
DMD	Dynamic Mode Decomposition
NLP	Natural Language Processing
SGNS	Skip-Gram with Negative Sampling
LSA	Latent Semantic Analyis
TF-IDF	Term Frequency Inverse Document Frequency
BERT	Bidirectional Encoder Representations form Transformers
RBO	Rank-Biased Overlap
α	Corpus size rate
β	Initial corpus size
c	Vocabulary size
$N(t)$	Corpus size at time t
a	Shape parameter of the Zipf probability mass function
A	Full square matrix operator
$\widetilde{A}$	Low-rank square matrix operator
$\mathbf{\Phi}$	Full size dynamic modes
$\widehat{\mathbf{\Phi}}$	Low-rank dynamic modes
$\vec{\mathbf{w}}$	Word vector of word w
$\vec{\mathbf{v}}$	Word vector of word v
$P(x)$	Probability of an even x
$P(y\|x)$	Conditional probability of an event y given event x has occurred

List of Figures

List of Tables

Curriculum Vitae

Education

2015-2021 Ph.D., Applied Mathematics.
Advised by Prof. Suzanne Sindi.
University of California, Merced. Merced, CA.

2011-2015 B.S., Mathematics.
Advised by Prof. Michele Joyner.
East Tennessee State University. Johnson City, TN.

2009-2011 A.A.S., Computer Science.
Northeast State Community College. Blountville, TN.

Teaching Experience

2015-2020 Teaching Assistant.
University of California, Merced. Merced, CA.
Subjects: Probability and Statistics, Calculus, Differential Equations, and Linear Algebra.

Chapter 1

Introduction

1.1 Organization of the Dissertation

Chapter 1 introduces the concept of language evolution and its connection to mathematical biology. This chapter also explains the motivation behind language modeling and its significance to the broader field of language evolution. Moreover, the basics of natural language processing is introduced in this chapter, explaining the mathematical basis behind language modeling and its applications.

Chapter 2 is divided into two sections. The first section details our work in statistical modeling of word rank evolution to characterize the stability and volatility conditions of rank change. The second section details our work in applying matrix algebra techniques to understand the unigram time-series behaviors of word frequencies.

Chapter 3 presents two word embedding models and its application to two online social movement hashtags. By using the word embedding models, we explain on how word paths are computed and how it shows the evolution of contextual semantics in time.

Chapter 4 is about the evaluation of reading comprehension tasks using three machine learning language models. These models are trained using long documents data, which contains text with narrative structures paired with questions and answers. This chapter also explains the challenges of machine learning models on reading comprehension tasks.

Chapter 5 summarizes each Chapter and the future work of this dissertation.

1.2 Language Evolution in the Context of Mathematical Biology

Unlike programming languages which are made for the purpose of giving specific instructions to computers, natural languages are both somehow "made" and have emerged naturally from humans. Languages are "made" in a sense that rules are specified to create a combination of words for easy comprehension. Language rules are known as grammar or linguistic style/rhythm like in poetry. Language speakers can even invent new

words to describe something new or to create/derive a new language for the purpose of fiction [1], national identity, or international cooperation [2]. Natural languages are forms of communication that are somehow emerged from an arbitrary intelligible concepts. Furthermore, languages evolve as a result of cultural, environmental, and socio-political changes that influence its structure. People are typically receptive to the ever changing landscape of language. The need to communicate, learn, and establish connections are fundamentally human. The journey of language is not alone, but a consequence of human evolution.

The concept of evolution is relatively simple but often misunderstood. Evolution, the change in a species in time, is often incorrectly thought to mean natural selection. As an example suppose that there are two organisms $\{0, 1\}$ in a population and assume that the frequency distribution at $t = 0$ is 50% 0 and 50% 1. If there is no selection (i.e. 0 and 1 are equally fit), the expected frequency distribution at successive generations $t > 0$ is 50% 0 and 50% 1. Evolutionary processes occur when the frequency distribution changes. The forces that influences these changes are known as drift, flow, mutation, and natural selection. Drift or neutral drift is a process of evolution dominated by random sampling. The organisms at time t are sampled from $t - 1$ and - for example - where it happens by chance that organism 0 is more dominant than organism 1. Flow is what happens when foreign organisms are introduced into the population. For example, the organism set $\{0, 1\}$ becomes $\{0, 1, A\}$, which can lead to changes in the frequency distribution. Mutations or variants are when there is a property change of an organism which can have an effect or no effect in the frequency distribution. Natural selection is a process of evolution by which some organisms are better adapted - or detrimental - with their environment due to some advantageous trait, and thus an increase or decrease in frequency.

Evolution is known in science most associated with molecular biology. For example, evolution is studied in a sequence structure of molecules such as in Deoxyribonucleic Acid (DNA) or in proteins [30]. These molecules are labeled so that researchers can track the order of sequences. Similar to sequences of molecules, language can be thought of as a sequence of words that serve a function within a larger body of complex organism. In a natural language, a word or phrase represents a meaning or concept which can shift as the context changes in time. The shifts in contextual meaning could change the word frequencies causing it to rise and fall.

This dissertation studies written language across time. Although written language is only a part of natural language, it is a great source of empirical data for language analysis. Spoken language poses another level of difficulty in terms of methods on how to take such data.

---

[1] An example of a fictional language is Klingon (1966), as seen from the science fiction television show *Star Trek*, by linguist Marc Okrand. [6]

[2] An example of a constructed language is Esperanto which was originally made to be the world's auxiliary language. Esperanto is a mixture of European languages and it survived political issues at its infancy. It is now widely used. [10]

Background and Motivation

The research on language evolution has been going on for centuries. Major areas of focus include: How did language originate? How did language evolve? How does language continue to evolve? How is human language different from animal communication? What factors lead to cultural changes? How do we learn language and communicate? What does your particular language and style say about how you think and behave? These questions are fundamental to studying language and researchers have explored biological and cognitive approaches to answer these questions.

The state of language evolution is divided into two groups of conflicting approaches. The biolinguistic approach and the usage-based [28, 22], and the saltationist and gradualist approaches [23]. Researchers argue on which approach is best or which theories are correct. The study of language evolution was once banned by the Paris Linguistic Society in 1866 because of the disagreements among researchers during that time [23]. The exact definition of "language evolution" is heavily contested as the words "language" and "evolution" are in fact polysemous words. Below, we discuss a brief definition of the approaches.

The biolinguistic approach stems from the Darwinian view of evolution which began in the late 19th century. It explores if natural selection contributed to the emergence of human language. In general, biolinguistics considers the biological foundations of language, looking at both the genetic and psychological basis of human communication. In contrast, the usage-based approach stems from the idea that language is a complex adaptive system. This means that the evolution of language is deeply influenced by independent socio-cognitive interactions. The phrase "usage-based" reflects that people's word usage and communication styles are influenced by observing and learning. Part of the disconnect between the two is that the usage-based approach seems to be heavily focused on the socio-cognitive factor and the biolinguistics is too focused on the genetics. [28, 22]

Noam Chomsky and Robert Berwick in 2016 [3] argue that natural selection has no contribution to the evolution of language. They share the saltationist view of language evolution: syntax suddenly appeared as a result of genetic minor mutation. In connection to Chomsky's theory of "Universal Grammar", the simplest syntax of language came from a minor mutation. In contrast, the gradualist approach views language evolution as accumulated changes over long periods of time influenced by the biological and cultural aspect of language. [23]

Regardless of the approaches mentioned above, the difficulty of this research lies in the lack of hypothesis testing. Today, we are facing a new paradigm of computational advantages and data rich sources to test hypothesis of language evolution [21]. Newberry et. al. in 2017 [20] for example, provides a method for testing selective theories of language by using a null stochastic model against a large scale temporal text data of historical American English [1]. Karjus et. al. in 2018 [13] discusses the challenges of detecting evolutionary forces of language change by carefully considering different temporal binning of bodies of text. Turney et. al. [27] uses large historical English corpora to explore the evolutionary fitness features of synonyms. All three research

projects showed the value of large-scale diachronic corpora on studying the evolution of language.

1.3 Natural Language Processing for the Application of Language Evolution

The interest in language evolution has grown recently because of the advances in computation and the availability of text data. Researchers can now take wealth of text data from many sources such as the internet and digitized books. Methodological approaches are currently advancing, making it possible to perform a comprehensive analysis of text data. Language contains a wealth of mathematical structures and can be viewed as a statistical and algebraic source of information. The methods in Natural Language Processing (NLP) have made it possible to model and analyze the mathematical structures of language.

NLP first emerged as a field of computer science that sought to design algorithms so that computers can "understand" human language. NLP is divided into two parts: linguistics and computer science. Linguistics is concerned with the syntax and meanings of words and phrases. Computer science is concerned with transforming text data into numerical representations through statistics and machine learning. Both fields are concerned with how do computers "understand" language. While there are many approaches of NLP, we briefly discuss two modeling approaches below.

Statistical Language Modeling

Modeling language can be thought of as computing the probability of sequences of words. Consider a sequence of k words $(w_1, w_2, w_3, \cdots, w_k)$ where w is a word from finite set of vocabulary words V. The probability of this random sequence of words to appear in a text is given by

$$P(w_1, w_2, \cdots, w_k) = P(w_1)P(w_2|w_1) \cdots P(w_k|w_1, w_2, \cdots, w_{k-1}). \tag{1.1}$$

This equation is formed using the chain rule of joint probabilities because of the assumption that current word in the sequence depends on all of its history. Estimating the conditional probabilities becomes problematic when you have very long sequences of observations, especially from real languages that may have very large but finite set of vocabulary words.

To mitigate this problem, the usual formulation of modeling word sequences often assumes the current word only depends on the previous word. This is known as the Markov Assumption [3]. The Markov chain model is a classical approach to modeling and predicting event sequences. This fundamental model is most famous for weather prediction and biological sequence analysis [8].

[3]The Markov assumption and the Markov model is named after the mathematician Andrei Markov who is known for his work in conditional probabilities. [8]

With the Markov assumption, the probability of a sequence of words can now be approximated using a bi-gram model [4]. We write this as

$$P(w_1, w_2, \cdots, w_k) \approx \prod_{i=2}^{k} P(w_i|w_{i-1}) \tag{1.2}$$

where the term $P(w_i|w_{i-1})$ is the conditional probability of observing word w_i given the previous word w_{i-1}. Computing the conditional probabilities can be approximated using Maximum Likelihood Estimation (MLE). The MLE procedure usually just counts bigrams from a given body of text. The bigram model can be further reduced to a unigram model. That is

$$P(w_1, w_2, \cdots, w_k) \approx \prod_{i=1}^{k} P(w_i) \tag{1.3}$$

where $P(w_i)$ is the probability of observing a single word. The bigram and unigram models can be generalized into an n-gram model.

Three weaknesses of n-gram modeling are as follows: (1) it's usually computationally expensive, (2) the model is very sensitive to the training data, and (3) the meaning of words and phrases - including grammar structure - are ignored completely because it only considers the statistical structures of language. While the n-gram model lacks the cognitive model of language grammar, it has been successfully used for speech recognition and sequence modeling. [12]

There are more advanced models using machine learning such as the Recurrent Neural Network (RNN) and the Long-Short Term Memory (LSTM) [11]. These methods deals with long term dependency problems that is inherent with sequences such as natural language.

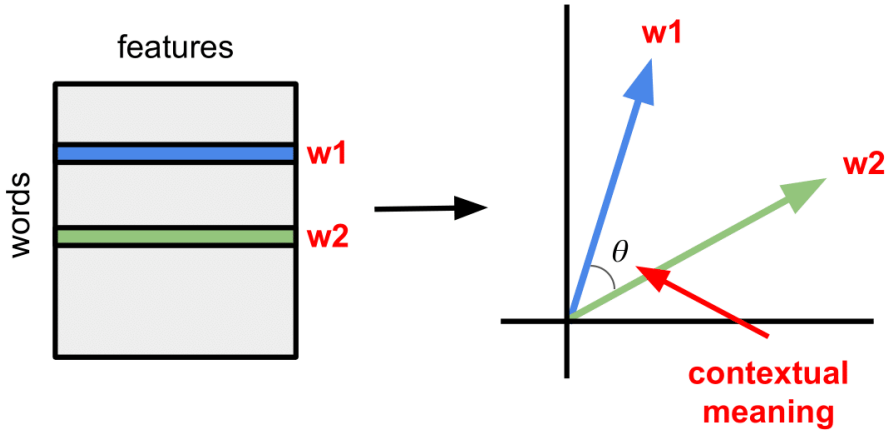
Unigrams and bigrams are particularly useful for modeling historical word frequencies. Historical trends of unigrams are the main focus of this dissertation and we are particularly interested in modeling these trends using statistics. We present our work in unigram time-series modeling and analysis in Chapter 2.

Word Embeddings

Word embeddings are numerical representations of words - also known as word vectors - trained from a given text data. To give a sense of what a word embedding is, we show an abstract example in Fig. 1.1. In this figure, we show that the word embedding can be thought of as a real-valued matrix of rows as words and columns as features. These features include: documents that contain similar words, co-occuring words or bigrams, or a hidden layer of a neural network. There are multiple ways to measure word "meaning". One of which is by measuring the angular distances of words. For example, the words **w1** and **w2** in Fig. 1.1 are contextually similar if their angular distance is close enough compared to other words. Contextual meaning is associated to words due to the content and style of a given text, rather than by their actual definition.

[4] An n-gram is a sub-sequence of words of length n , where a 1-gram (unigram) is a word, 2-gram (bigram) is a subsequence of two words, 3-gram (trigram) is a subsequence of three words, and so on.

Figure 1.1: **Word embedding illustration.** The word embedding matrix is composed of rows as words and columns as features. Each word is represented as a vector and the angular distance between the words can be a metric of contextual meaning.



There are many classes of word embeddings but the most common ones are the word co-occurence matrix and the Term Frequency Inverse Document Frequency (TF-IDF) [16]. These matrices are dimensionally reduced using Singular Value Decomposition (SVD). Word embeddings can be generated by using advanced machine learning models such as the Skip-Gram with Negative Sampling (SGNS) [18]. Word embeddings are a basis of advanced language models such as the Bidirectional Encoder Representations from Transformers (BERT) [7].

The applications of word embeddings have been a success in many NLP tasks such as machine translation [4, 14, 29, 17], question answering [26, 25, 24], and sentence classification [5, 19, 2]. Like many NLP models, word embedding models poses a large and dangerous problem of societal biases. NLP models work through detecting patterns that may contain societal biases already embedded in the text data on which it is trained. For example, biases in word embeddings revealed cultural biases on gender stereotypes [9]. Moreover, text data trained for question and answering tasks uncovered stereotypes in gender, nationality, ethnicity, and religion [15].

The focus of Chapter 3 is using the word embeddings to track the evolving contextual meaning of words. Chapter 4 explores machine learning language models on how they can predict answers to question from long documents with narrative structures.

Bibliography

[1] R. Alatrash et al. “Ccoha: Clean corpus of historical american english”. *Proceedings of The 12th Language Resources and Evaluation Conference*. 2020, pp. 6958–6966.

[2] P. Anand et al. “Cats rule and dogs drool!: Classifying stance in online debate”. *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*. 2011, pp. 1–9.

[3] R. C. Berwick and N. Chomsky. *Why only us: Language and evolution*. MIT press, 2016.

[4] D. Britz et al. “Massive exploration of neural machine translation architectures”. *arXiv preprint arXiv:1703.03906* (2017).

[5] E. Cambria et al. “Sentiment Analysis Is a Big Suitcase”. *IEEE Intelligent Systems* 32.6 (2017), pp. 74–80. DOI: 10.1109/MIS.2017.4531228.

[6] M. Destruel. “Reality in Fantasy: linguistic analysis of fictional languages”. PhD thesis. Boston College, 2016.

[7] J. Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Tech. rep. 2018. arXiv: 1810.04805v2.

[8] P. A. Gagniuc. *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017.

[9] N. Garg et al. “Word embeddings quantify 100 years of gender and ethnic stereotypes”. *Proceedings of the National Academy of Sciences* 115.16 (2018), E3635–E3644.

[10] R. Garvía. *Esperanto and Its Rivals: The Struggle for an International Language*. University of Pennsylvania Press, 2015. ISBN: 9780812247107. URL: http://www.jstor.org/stable/j.ctt14jxw44.

[11] I. Goodfellow et al. *Deep learning*. Vol. 1. 2. MIT press Cambridge, 2016.

[12] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*.

[13] A. Karjus et al. “Challenges in detecting evolutionary forces in language change using diachronic corpora”. *arXiv preprint arXiv:1811.01275* (2018).

[14] G. Lample et al. “Unsupervised machine translation using monolingual corpora only”. *arXiv preprint arXiv:1711.00043* (2017).

[15] T. Li et al. “UNQOVERing Stereotyping Biases via Underspecified Questions”. *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Nov. 2020, pp. 3475–3489. DOI: 10.18653/v1/2020.findings-emnlp.311. URL: https://www.aclweb.org/anthology/2020.findings-emnlp.311.

[16] D. I. Martin and M. W. Berry. “Mathematical foundations behind latent semantic analysis”. *In*. 2007.

[17] T. Mikolov, Q. V. Le, and I. Sutskever. “Exploiting similarities among languages for machine translation”. *arXiv preprint arXiv:1309.4168* (2013).

[18] T. Mikolov et al. “Distributed Representations of Words and Phrases and Their Compositionality”. *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS’13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 3111–3119.

[19] S. M. Mohammad and P. D. Turney. “Crowdsourcing a word–emotion association lexicon”. *Computational intelligence* 29.3 (2013), pp. 436–465.

[20] M. G. Newberry et al. “Detecting evolutionary forces in language change”. *Nature* 551.7679 (2017), pp. 223–226.

[21] J. Nölle, S. Hartmann, and P. Tinits. “Language evolution research in the year 2020: A survey of new directions”. *Language Dynamics and Change* 10.1 (2020), pp. 3–26.

[22] M. Pleyer and S. Hartmann. “Constructing a consensus on language evolution? Convergences and differences between biolinguistic and usage-based approaches”. *Frontiers in psychology* 10 (2019), p. 2537.

[23] L. Progovac. *A critical introduction to language evolution: Current controversies and future prospects*. Springer, 2019.

[24] P. Rajpurkar et al. “SQuAD: 100,000+ Questions for Machine Comprehension of Text” (June 2016). arXiv: 1606.05250. URL: http://arxiv.org/abs/1606.05250.

[25] M. Tapaswi et al. *MovieQA: Understanding Stories in Movies through Question-Answering*. Tech. rep. 2016, pp. 4631–4640. URL: http://movieqa.cs.toronto.edu.

[26] A. Trischler et al. “NewsQA: A Machine Comprehension Dataset” (Nov. 2016), pp. 191–200. arXiv: 1611.09830. URL: http://arxiv.org/abs/1611.09830.

[27] P. D. Turney and S. M. Mohammad. “The natural selection of words: Finding the features of fitness”. *PloS one* 14.1 (2019), e0211512.

[28] S. Wacewicz et al. “Language in language evolution research: In defense of a pluralistic view”. *BIOLINGUISTICS* 14 (2020).

[29] Y. Wu et al. “Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation” (Sept. 2016). arXiv: 1609.08144. URL: http://arxiv.org/abs/1609.08144.

[30] Z. Yang. *Molecular evolution: a statistical approach*. Oxford University Press, 2014.

Chapter 2

Modeling Word Rank Evolution

Natural languages are deeply connected to human culture. There is no doubt that the subtleties of word usage and evolution in a language are very difficult to understand and predict. This is due to multiple reasons. For example, words can naturally gain or lose meanings in time, or words can change their meaning entirely. The word "gay" - for example - may seem out of place to many English speakers in the 21st century. Today, the word "gay" refers widely to homosexuality in all sorts of contexts, as a noun and adjective. In fact, the word came from the old French word "gai" meaning "cheerful" and it was borrowed centuries ago into English as "gay" meaning "happy" or "joyful", among other connotations [28]. Although the word still retains an echo of its original meanings, the sexualization of the word "gay" is a result of cultural change across the past half-century. Word changes can sometimes be pinned to specific cultural events. [1] Important historical events may serve as symbolic beginnings of cultural change that we can now see "echo" from them in word frequency changes. Roughly speaking, the frequency and their ranks showed us some insights into the evolution of word meanings. The rank changes among words are indicators of semantic change. Fully developed languages are structured and have specific rules. For a written language, sentence structure is important and usually, words are specifically chosen by the writer to convey a thought. Humans have written billions of texts in thousands of languages. Despite the complicated nature of written language, the frequency of word usage follows a pattern. Popularized by George Kingsley Zipf, word frequencies on a large body of text follow a power-law $r \propto 1/k$ where r is the relative frequency and k is the rank [85, 84]. We observed from the Google unigram data that the ranks of words change in time. The motivation behind modeling word rank evolution is to explain how and why words change in ranks.

With the inspiration of molecular evolutionary biology, the statistical model presented in Chapter 2.2 offers an explanation of the volatility/stability conditions of word rank evolution.

[1] For example, the Stonewall riots of 1969, in New York City, were a series of events that sparked the modern LGBT rights movement. In New York City, a bar such as the Stonewall Inn is a place of refuge for many LGBT individuals to express their identity openly. After a violent police raid, a rapid series of events there and elsewhere in the United States began to shift public opinion on homosexuality [22, 26]. The Stonewall story is a popular historical event for the LGBT community but it is not the only one [5].

We examine the dynamic behaviors and predictability of word ranks by utilizing a data-driven approach to modeling. With the assumption that words behave similar to a dynamical system, the method of Dynamic Mode Decomposition (DMD) presented in Chapter 2.3 is applied to the unigram time-series data.

2.1 The Google Ngram Corpus

A subset of historical word frequency time-series data was taken from the Google Ngram Corpus, a collection of digitized books summarized by counting n-grams. [2] This data base provides unigram (single-word) occurrences for over one hundred years in eleven languages (Four of these eleven languages are variations of the English language.) In total, there are eight distinct languages in the dataset used here: Chinese, English, French, German, Hebrew, Italian, Russian, and Spanish. Google parsed millions of books from the Google books corpus and counted how many times a unigram occurred in a given year. These datasets were generated in 2012 (version 2 [3]) and in 2009 (version 1). Version 2 includes parts-of-speech annotations [49].

To minimize bias in the data set, we attempted to select common words from each language. For the English language, we selected unigrams that occurred in at least 500 volumes each year. Because volume counts vary across languages, the other languages required different filtering parameters. The raw data from Google went through three layers of processing: (1) filter layer, (2) consolidation layer, and (3) normalization layer. In the filter layer, we select unigram data within the desired year and count range for both unigram count and volume count. In the consolidation layer, we convert each unigram into lowercase and summed the frequencies while listing all part-of-speech (POS) annotations into one vector. For example, the word 'solo', can be a noun, verb, adjective, or adverb. The POS annotation information and the frequency of the unigrams 'solo', 'Solo', or 'SOLO' are then summed into one frequency for the unigram 'solo'. In the normalization layer, we convert unigram frequencies into z-scores as we will describe below.

Following from the work of Sindi and Dale [72], the unigram frequency data is standardized. Given a set of words $V = \{w_1, w_2, \cdots, w_c\}$ and years $Y = \{t_0, t_1, t_2, \cdots, t_{T-1}\}$ where T is the number of years. The frequency of a word w in a corpus at time t, $r_{w,t}$, is the number of occurrences of that word in that corpus. In our analysis of the Google unigram data, we selected only words that occur above a desired frequency in the year interval $(1900, 2008)$. As such, the vocabulary size, c, is fixed as is the number of years T and we thus represent the word frequencies as a matrix: $\mathbf{R} \in \mathbb{R}^{c\times T}$ where

$$\mathbf{R}_{w,t} = r_{w,t}, \quad r_{w,t} \geq 1. \tag{2.1}$$

[2] The Google n-gram data spans from 1-gram (or unigram) to 5-grams but we only consider the unigram data for this dissertation. For easy implementation of downloading and processing the raw files of the Google Ngram data, visit `https://stressosaurus.github.io/raw-data-google-ngram/`.

[3] The Google Ngram Corpus (v2) raw data is available for download here `https://books.google.com/ngrams`

In our normalization process, we first convert the frequency matrix **R** into a proportion (or relative frequency) matrix **P** by normalizing the columns of **R** which normalizes word frequencies by year:

$$\mathbf{P}_{w,t} = p_{w,t}, \quad p_{w,t} = \frac{r_{w,t}}{\sum_{w=1}^{c} r_{w,t}}. \tag{2.2}$$

Finally, we normalize the proportions for each unigram by converting the rows of **P** into z-scores:

$$\mathbf{Z}_{w,t} = z_{w,t}, \quad z_{w,t} = \frac{p_{w,t} - \overline{p_w}}{\sigma_{p_w}} \tag{2.3}$$

where $\overline{p_w}$ is the mean and $(\sigma_{p_w})^2$ is the variance;

$$\overline{p_w} = \frac{1}{T} \sum_{t=0}^{T-1} p_{w,t} \tag{2.4}$$

and

$$(\sigma_{p_w})^2 = \frac{1}{T} \sum_{t=0}^{T-1} \left(p_{w,t} - \overline{p_w}\right)^2. \tag{2.5}$$

In order to do minimal cross-linguistic analysis, a set of Swadesh words and stopwords are used. Swadesh words are words that describe basic concepts - for example, “water” [74]. Stopwords are a set of most commonly used words that has no meaning by itself but are used to form a coherent sentence. For a more complicated cross-word analysis, a set of words are used associated with basic sentiments (e.g. positive and negative).

2.2 A Statistical Model of Word Rank Evolution

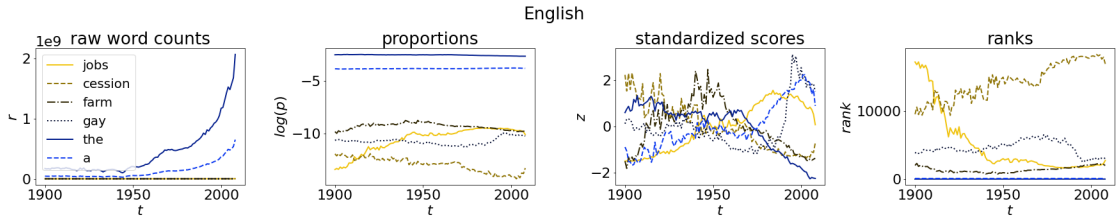
As an example, we show the Google unigram time-series trends of the words “jobs”, “cession”, “farm”, “gay”, “the”, and “a” in Fig 2.1. The first thing to notice that the words “the”, “and” and “a” are examples of the most frequently used word and their ranks in time remained constant. These types of vocabulary words are called **stop words**[4]. This group of words is mostly function words because they have little lexical meaning but serve an important function, to construct a meaningful and grammatical sentence. Other stop word examples are “to”, “and”, and “they”. Second, the word “gay” shows an increase in frequency around the 1960s and its rank went up a few ranks. The most significant rank change for this example is with the word “jobs”. As you can see in the fourth subplot in Fig. 2.1, the rank trend for “jobs” went from one of the lowest ranks in 1900 to one of the highest ranks in the year 2000. Similarly in an opposite way, the word “cession” decreased in rank. In contrast to the rank trend of “jobs”, the rank trend for the word “farm” generally remained in the higher ranks but there are some changes as well. Significant industrial and technological changes in the past have contributed to many

[4]The stopwords list can be obtained from Ranks NL website, `https://www.ranks.nl/stopwords`.

cultures changing and adapt to the modern workforce which explains the word “jobs” has gained ranks significantly.

Words like “water”, “sun”, “moon” and “night” are considered to describe basic concepts. These words are also part of a famous set of words called the **Swadesh words**[5], named after Morris Swadesh, who started creating a list of these words to compare different languages historically and culturally [74]. Some words in this list have multiple meanings which do not have one-to-one relationship between all languages. Intuitively, these words can be stable in ranks because the basic concepts that the words describe tend to be common across languages.

Figure 2.1: **English unigram time-series.** These are time-series of words “jobs”, “cession”, “farm”, “gay”, “the”, and “a”. The 1st subplot is the raw word counts for each year. The 2nd Subplot the proportions of words in log scale for each year. The 3rd subplot is the standardized scores of the words where each time-series is converted into z-scores. The 4th subplot is the rank time-series where each word is assigned a unique rank proportional to its relative frequency. The most frequently used word “the” has a rank of 1.



We consider the Wright-Fisher inspired neutral model as a base model to the Google unigram time-series data. This neutral model is inspired by the theory of molecular evolution by Kimura [42] and the Wright-Fisher model [82, 25]. A version of this model was previously presented by Sindi and Dale [72] who simulated the evolution of a growing corpus where each word in the vocabulary has equal fitness and chosen from the previous generation at random. This type of model is largely dominated by drift. Drift is a neutral process of evolution that is dominated by sampling errors. The outcome of the word frequency at the current generation is determined entirely by chance and not by other evolutionary forces like mutation, migration, natural selection, and random “mating”.

Background Work

Previous work has been done using the Google Ngram dataset to study and model how word frequency evolves in time. Turney et. al. [77] demonstrated that language change is not random but natural selection is the main driver on how language changes. They also observed similar behavior where the rate of change is decreasing overtime of the English language, as also demonstrated by Petersen et. al. [63]. We use unigram time-series similar to the studies involving word frequency change using the Google unigram data

[5]The swadesh words list can be obtained using the Natural Language Tool Kit (NLTK) module [12].

[83, 44, 43, 56, 62, 45, 41, 10, 2, 30, 9, 54], building on our prior work which, like Turney et al., uses Google unigram as a testbed for devising new tests of selection in the cultural domain [72]. There was previous work on modeling the rank diversity of words. Cocho et. al. [18] developed a Gaussian random walk model and compared it to the Google unigram dataset of six European languages. They found that the ranks of words in their experiments approach a log-normal distribution and the diversification of word ranks is the result of the random walks. Morales et. al. [57] continued the rank diversity analysis to include n-grams where $n > 1$. For six European languages, they found that it is important to consider studying languages at higher scales. Bigrams pose challenges, considering standard approaches to automatically quantifying word meanings through *document* statistics[19] and analyzing meaning changes through machine learning [7, 34].

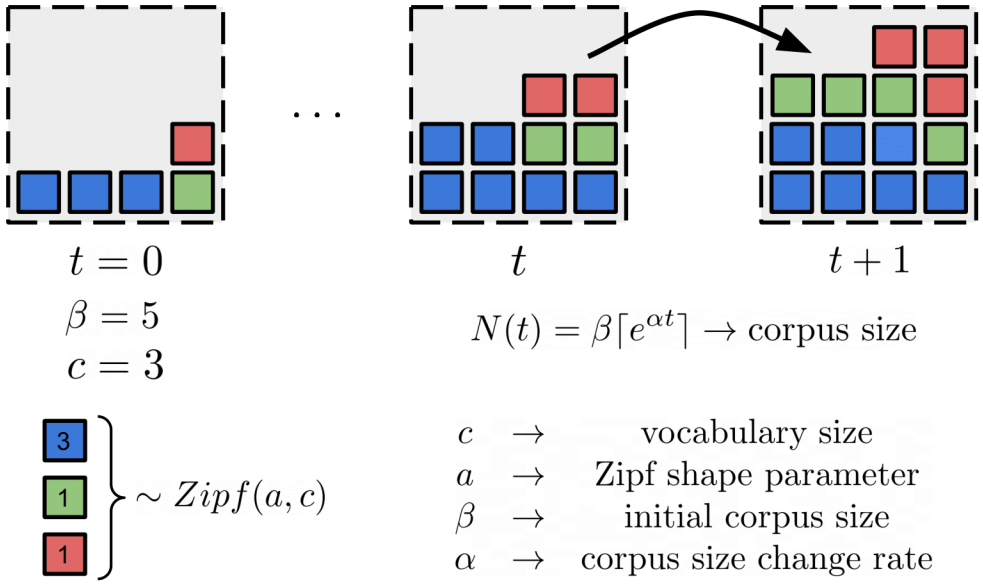
In this section, we analyze a unigram dataset of a century's worth of word rank changes in several languages. These languages are English, Simplified Chinese, French, German, Italian, Hebrew, Russian, and Spanish. We also consider three other variants of English which are American English, British English, and English Fiction. These languages are eight of the most spoken and written in the world, though admittedly they are a small subset of the world's thousands of languages. Nevertheless, they also represent several language families (Indo-European, Sino-Tibetan, Afroasiatic). There are many perspectives of studying language evolution which includes grammatical variations, sentence structure ordering (e.g. subject-verb-object), pronunciation variations, number of speakers, word frequencies, and other linguistic concepts that explains how language works [35]. In our work, we seek to characterize word frequency changes of our chosen languages using word-rank statistics. This section uses the Swadesh words and stopwords to perform a comparative analysis between the eight languages. We first develop a neutral model of frequency change and, consistent with past studies [61, 77, 58, 60, 72, 13], we observe departure from neutrality in all languages we study. We then generalize our neutral model to consider what we believe to be a *minimal* model of linguistic change: a word's frequency changes as a function of its previous frequency and the frequencies of other closely related words. This section also explains the mathematical framework of the model to answer why words change ranks in time. By using word-embedding models of the kind just summarized, we articulate several of the volatility/stability conditions for word-frequency change and show that it is surprisingly consistent across several languages. The result shows that *word community* matters, not just frequency – how a word is situated in the entire space of possible in a language gives it a position in a landscape. Some regions of this landscape may introduce capacity for semantic change, and other regions may restrict it.

2.2.1 Wright-Fisher (WF) Inspired Model

The Wright-Fisher model is an evolutionary model in molecular biology that simulates genetic drift, an evolutionary process of random sampling of alleles. In our work, each generation is sampled from the previous generation while the total number of words is increasing exponentially. As shown in Fig 2.2, at each time, the model

determines an empirical probability for each word in the vocabulary. The model do so at each time by sampling as many words are in our corpus at that time according to the previous generation. The model also ensure that each word is sampled at least once. This method of sampling ensures that no words disappear from our language. The initial generation is sampled from a Zipf distribution. It is known that most of the natural languages follow Zipf's law [85, 84, 52, 64, 6]. As noted above, Zipf's law states that the word rank is inversely proportional to its frequency.

Figure 2.2: **The Wright-Fisher inspired model diagram.** Words (shown as squares) at time $t + 1$ chosen - with replacement - from the previous time t assuming three words in the vocabulary with increasing total number of words. Different colored squared represents different words. Words are sampled at least once at each generation. This will prevent the model to have disappearing words.



More specifically, for a set of vocabulary words $V = \{w_1, w_2, \cdots, w_c\}$ with length c and with corresponding ranks $K = \{k_1, k_2, \cdots, k_c\}$ of the same length, the probability mass function for the Zipf distribution is given by

$$P(Y^{theory} = k_w; a, c) = \frac{1/k_w^a}{\sum_{w=1}^{c}(1/k_w^a)} \tag{2.6}$$

where $a \geq 0$ is the power law parameter characterizing the distribution and Y is the random variable for the word ranks. The Zipf distribution is used to compute the initial probability for each word sample at time $t = 0$. Word in V is sampled randomly based on its probability from the Zipf distribution. Words are sampled until the corpus size. The

corpus size N at time $t+1$ increases exponentially at rate $\alpha \geq 0$ and initial corpus size $\beta \geq c$ is given by

$$N(t;\alpha,\beta) = \lceil \beta e^{\alpha t} \rceil . \tag{2.7}$$

Zipf's Law in Relation to the Corpus Size

Zipf's law (also known as the power law) is defined in our problem as

$$r_{w,0} \propto \frac{1}{k_{w,0}^{a}}, \quad r_{w,0} \geq 1 \tag{2.8}$$

where $k_{w,0}$ is the rank of word w and raw frequency $r_{w,0}$ at $t = 0$. That means the most frequent word $k_1 = 1$ has $r_1 \propto 1$. The initial probability distribution of words of the Wright-Fisher inspired model is based on the Zipf probability mass function (pmf) shown in Eq 2.6. Each word at $t = 0$ are sampled with probability from the Zipf distribution. The expected value (or the expected rank of a randomly sampled word) is given by

$$E[Y^{theory}] = \sum_{w=1}^{c} k_w P(k_w;a,c) = \frac{\sum_{w=1}^{c}(1/k_w^{a-1})}{\sum_{w=1}^{c}(1/k_w^{a})}. \tag{2.9}$$

When sampling from the Zipf distribution, it is possible that some words will have the same proportions. It means that some words can have the same rank but to simplify the problem, the ranks of the words are unique.

Since we know that β is the initial corpus size, then by definition

$$P(Y^{data} = k_{w,0}) = \frac{r_{w,0}}{N(0;\alpha,\beta)} = \frac{r_{w,0}}{\beta} \tag{2.10}$$

at $t = 0$. We can write the expected rank in terms of raw frequency $r_{w,0}$ and β,

$$E[Y_0^{data}] = \sum_{i=1}^{c} r_{w,0} \frac{r_{w,0}}{\beta}. \tag{2.11}$$

There are bound to be some differences between the expected values because of the error in the data. Therefore, the difference can be characterized by

$$E_{diff}[Y] = E[Y^{theory}] - E[Y_0^{data}] = \frac{\sum_{w=1}^{c}(1/k_{w,0}^{a-1})}{\sum_{w=1}^{c}(1/k_{w,0}^{a})} - \sum_{w=1}^{c} r_{w,0}\frac{r_{w,0}}{\beta} \tag{2.12}$$

where as β increases the $E_{diff}[Y] \to 0$ for fixed c. The term $\frac{r_{w,0}}{\beta}$ converges to the theoretical Zipf probabilities. If c increases while β is fixed ($\beta \geq c$), then $r_{w,0} \to 1$ for all w. If $\beta = c$, then $r_{w,0} \propto 1$ for all w. So, $E[Y_0^{data}] \to 1$ and $E_{diff}[Y] = E[Y^{theory}] - 1$ if c increases for fixed β. The corpus size and the vocabulary size is theoretically finite.

The Wright-Fisher (WF) Inspired Model Written as Multinomial Probability Transitions

Consider the state space $\mathbf{x}_t \in \left\{x_{w,t} \in \{1, 2, \cdots, N(t) - 1\} \text{ for } w = \{1, 2, \cdots, c\}\right\}$ for a random variable $X_{w,t}$ where $\mathbf{x}_t$ is a vector of word counts at time t, c is the number of words, and $N(t)$ is the corpus size at time t. We know that $\sum_{w=1}^{c} x_{w,t} = N(t)$. The corpus size N at time t increases exponentially at rate $\alpha \geq 0$ and initial corpus size $\beta \geq c$ is given by Eq 2.7.

The probability on transitioning from $\mathbf{x}_{t-1} = (a_1, a_2, \cdots, a_c)$ to $\mathbf{x}_t = (b_1, b_2, \cdots, b_c)$ is given by the multinomial probability mass function,

$$Mult\left(\mathbf{x}_t = (b_1, b_2, \cdots, b_c) | \mathbf{x}_{t-1} = (a_1, a_2, \cdots, a_c)\right) = \frac{(N(t) - c)!}{b_1! b_2! \cdots b_c!} \left(\frac{a_1}{N(t-1)}\right)^{b_1} \left(\frac{a_2}{N(t-1)}\right)^{b_2} \cdots \left(\frac{a_c}{N(t-1)}\right)^{b_c}. \quad (2.13)$$

with initial state $\mathbf{x}_0 \sim Zipf(a)$ where $a \geq 0$ is the Zipf shape parameter. Recall that we defined the Zipf probibility mass function in Eq 2.6.

The multinomial random vector has components with a binomal distribution for frequency $x_{w,t}$ in $\mathbf{x}_t$,

$$\begin{aligned} X_{1,t} &\sim Bin\left(N(t), \frac{x_{1,t-1}}{N(t-1)}\right) \\ X_{2,t} &\sim Bin\left(N(t), \frac{x_{2,t-1}}{N(t-1)}\right) \\ &\vdots \\ X_{c,t} &\sim Bin\left(N(t), \frac{x_{c,t-1}}{N(t-1)}\right) \end{aligned} \quad (2.14)$$

where $X_{w,t}$ is the random variable of the counts of word w at time t. The binomial probability mass function of $X_{w,t}$ with range $0 \leq x_{w,t} \leq \beta - c$ is given by

$$Bin(X_{w,t} = x_{w,t}; N(t), N(t-1), c) = \binom{N(t) - c}{x_{w,t}} \left(\frac{x_{w,t-1}}{N(t-1)}\right)^{x_{w,t}} \left(1 - \left(\frac{x_{w,t-1}}{N(t-1)}\right)\right)^{N(t) - x_{w,t}}. \quad (2.15)$$

The initial frequencies at time $t = 0$ are sampled with probability mass function given as

$$Bin(X_{w,0} = x_{w,0}; \beta, c) = \binom{\beta - c}{x_{w,0}} p_w^{x_{w,0}} \left(1 - p_w\right)^{\beta - x_{w,0}}. \quad (2.16)$$

where the value of the probability p_w is from the Zipf probability mass function in Eq. 2.6. It is important to note that even though the word frequencies can be independently divided into their binomial components, the overall behavior of the Wright-Fisher model is heavily dependent on the corpus size and the vocabulary size.

The expected value and the variance for the binomial probability mass function are given as follows:

$$E[X_{w,t}] = (N(t) - c)p_{w,t-1} \tag{2.17}$$
$$Var[X_{w,t}] = (N(t) - c)p_{w,t-1}(1 - p_{w,t-1}) \tag{2.18}$$

where the terms $E[X_{w,t}]$ and $Var[X_{w,t}]$ are respectively the expected value and variance. The expected value and variance are heavily dependent on the probability and the corpus size function. Since it is defined that the corpus size function is an exponentially increasing function in time shown in Eq. 2.7, then the expected value and variance also increases in time.

Rank Change Potential

A discrete binomial distribution can be approximated by a normal distribution with mean $\mu_{w,t} = (N(t) - c)p_{w,t}$ and variance $\sigma^2_{w,t} = (N(t) - c)p_{w,t}(1 - p_{w,t})$. Approximately 100% of the distribution lies within the interval $\mu_{w,t} \pm 4\sigma_{w,t}$. Note that $\sum_{w=1}^{c} p_w = 1$ where the values of p_w came from the Zipf probability mass function in Eq. 2.10 with shape parameter a and number of vocabulary words c. The interval is the segment where a word is most likely to have a frequency when randomly sampled. These segments can overlap, and so it becomes likely that words can change ranks. We count the number of overlaps by computing the length of this segment overlap. The following is how we compute the length of the overlap of two segments for word w and word v:

$$l_{wv,t} = min\big(\mu_{w,t} + 4\sigma_{w,t}, \mu_{v,t} + 4\sigma_{v,t}\big) - max\big(\mu_{w,t} - 4\sigma_{w,t}, \mu_{v,t} - 4\sigma_{v,t}\big). \tag{2.19}$$

Let r_w and r_v be the ranks of word w and v respectively. By counting the number of words that overlap word w, the net potential rank change is given by

$$\widehat{ol}_w = \sum_{v=1, v\neq w}^{c} \begin{cases} 1 & l_{wv} > 0 \text{ and } r_w < r_v \\ -1 & l_{wv} > 0 \text{ and } r_w > r_v \\ 0 & otherwise. \end{cases} \tag{2.20}$$

In other words, the net number of words that overlap with word w is by summing the number of overlaps above and below its word rank. If $\widehat{ol}_w > 0$, then word w has the potential to go down in rank. If $\widehat{ol}_w < 0$, then word w has the potential to go up in rank.

Word Rank Change Quantification

The most important aspect of our analysis is the rank of words. Rank is a discrete integer measure that determines how frequently a word is used relative to all other words. Compared to the raw frequency, ranks have the advantage of localizing the relative position of the word frequencies in time. For example, the Table 2.1 below shows the assigned unique rank for example words w_1, w_2, w_3, and w_4. The word w_1 remained

in the 1st rank for all five timestamps while the word w_3 changed rank from 4th in t_1 to 2nd in t_2. The rank for each word is assigned using their proportion and each word has a unique rank. If two words have the same proportion, the words are alphabetized (for actual words) or sorted (for numerals) and then the ranks are assigned in sequence accordingly. This procedure is rare and is unlikely to introduce bias.

Table 2.1: **Word rank matrix example with four words and five time stamps.**

	t_0	t_1	t_2	t_3	t_4
w_1	**1**	**1**	**1**	**1**	**1**
w_2	2	2	3	2	2
w_3	3	**4**	**2**	3	3
w_4	4	3	4	4	4

We also considered a different style of quantifying the ranks. Table 2.2 below shows a ranked list for each timestamp. Instead of assigning an integer for the word ranks in time, the words are positioned in a form of a ranked list. It gives the same information from Table 2.1 but with the words instead of integers. The rank matrix shows the time-series for ranks in each word while the ranked list shows the overall structure of word ranks in time.

Table 2.2: **Word ranked lists example with four words and five time stamps.**

	t_0	t_1	t_2	t_3	t_4
1	$\mathbf{w_1}$	$\mathbf{w_1}$	$\mathbf{w_1}$	$\mathbf{w_1}$	$\mathbf{w_1}$
2	w_2	w_2	$\mathbf{w_3}$	w_2	w_2
3	$\mathbf{w_3}$	w_4	w_2	$\mathbf{w_3}$	$\mathbf{w_3}$
4	w_4	$\mathbf{w_3}$	w_4	w_4	w_4

Formally, we denote this rank information as $\mathbf{K}$ with dimensions $c \times T$ for the word ranks and $\mathbf{RL}$ with dimensions $c \times T$ for the ranked list. There are two metrics that describe the overall structure of the rank changes for each word in $\mathbf{K}$. First, we compute the $\mathbf{K}$ by taking $\mathbf{K_t} - \mathbf{K_{t-1}}$ for each w. The dimensions of $\mathbf{K}$ is now $c \times (T-1)$. The first metric is the sum of rank change of word w which is computed by

$$\sum_{t=0}^{T-2} \Delta k_{w,t} \tag{2.21}$$

where $\Delta k_{w,t}$ is an entry in the matrix $\mathbf{K}$ for word w at time t. The second metric is the rank change variance of word w which is computed by

$$\frac{1}{T-1} \sum_{t=0}^{T-2} \left(\Delta k_{w,t} - \overline{\Delta k_{w,t}} \right)^2 \tag{2.22}$$

where $\overline{\Delta k_{w,t}}$ is the mean computed as

$$\overline{\Delta k_{w,t}} = \frac{1}{T-1} \sum_{t=0}^{T-2} \Delta k_{w,t}. \tag{2.23}$$

The sum of rank change metric is the net change of the ranks within the time-frame which is 109 years. It can capture words that changed in ranks - either monotonically up or down trends - within the time-frame. This type of measure ignore cases where a word changes ranks often or the ranks are going up and down with some variance. For example, if a word initially has rank 10 and changed its rank to 100 in 50 years, and it went back to rank 10 in additional 50 years, the net change would be zero. The rank change variance is a second metric to verify that most of the stopwords and Swadesh words are consistently stable across languages.

Finally, we use the Rank Biased Overlap - or RBO - to measure the overall temporal structure of the ranked list matrix **RL**. The RBO is a similarity measure to compare two indefinite ranked list. For a given ranked list matrix **RL**, we compare $\mathbf{RL}_t$ and $\mathbf{RL}_{t+1}$ to each other, $\mathbf{RL}_t$ and $\mathbf{RL}_{t+10}$ to each other, and $\mathbf{RL}_0$ and $\mathbf{RL}_t$ to each other. That means that the RBO is measured for the matrix **RL** to see the overall collective changes in the ranked list in time. The RBO measure is in the range $[0, 1]$. $RBO = 1$ means that both lists are the same while $RBO = 0$ means that both lists are completely different. Following from the work of Webber et. al. [80] and their variable symbolisms, below is the general computation of the RBO similarity measure of two ranked list.

Given two sets of infinite rankings S and T, the RBO is computed generally as

$$RBO(S, T, p) = (1 - p) \sum_{d=1}^{\infty} p^{d-1} \dot{A}_d \tag{2.24}$$

where the parameter p is the steepness of the weights. This parameter falls in the range $[0, 1]$. Smaller p means that less items are considered in both of the list while larger p means that more items are considered in both of the lists. If $p = 0$, it means that only the top ranked item for both lists are considered which means that the RBO can only be either 0 or 1. If $p = 1$, all items are considered and that RBO fall between the range $[0, 1]$. The term A_d is called the *agreement* measure at some depth d. The *agreement* is computed by

$$A_{S,T,d} = \frac{|S_{:d} \cap T_{:d}|}{d} \tag{2.25}$$

where the operation $|S_{:d} \cap T_{:d}|$ is called the cardinality of the intersection at depth d of the sets S and T. In other words, that term is the number of times the two sets have common items up to the depth of d. For quick and easy implementation of the RBO, you can follow the work by Changyao Chen [17]. They set $p = 1$ as a default.

2.2.2 Results

The results are divided into three parts. Part 1 is about the results of the binomial components of the multinomial distribution. This part explains the effects of changing the parameters to the binomial distribution of each word. Part 2 is about the simulations of the WF inspired model and how it it fits into the theoretical multinomial distribution. This part also explains the effects of changing the parameters to the simulations. Part 3 is about the Google Unigram data and how it fits into the WF Inspired model. This part also explains the similarities and differences between the WF simulations and the Google Unigram data.

Part 1 of 3: WF as Multinomial Probability Transitions

Results summary. Recall that the initial frequency distribution of the WF model is sampled from the Zipf distribution. The probabilities of sampling a word are inversely proportional to its ranks. There are two key results in this section.

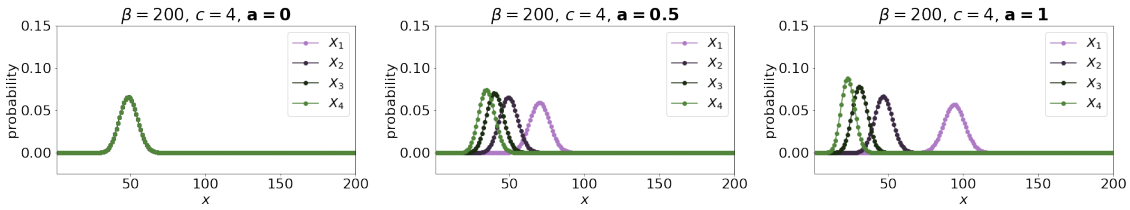
- If a word is assigned a rank of 1, then the probability of sampling that word is $P(Y = 1; a, c)$ which is defined in Eq. 2.6. For multiple independent trials, each word follows the binomial distribution. However, these binomials are overlapping and the chances of having a word having an error from its predefined rank is non-zero.

- The binomial distributions suggest that the overlaps are the main cause of a rank error when sampling at the initial time. The results show that if the corpus size increases, the binomial overlaps decreases while if the vocabulary size gets closer to the corpus size, the binomial overlaps increase.

Below, we explain the details of our findings.

The individual components of the multinomial probability distributions are binomial distributions. As shown in Eq. 2.14, the frequency of a word w_i can be sampled using the binomial distribution with probability mass function shown in Eq. 2.15 (or Eq. 2.16 at $t = 0$). There are three parameters we can vary. The first parameter is the Zipf shape parameter called a. This parameter controls the distance between the success probabilities of the most frequent words versus the rest. These probabilities are inserted into the individual Binomials in Eq. 2.16 for each word in the vocabulary. Fig. 2.3 shows the binomial probabilities of four words. At $a = 0$, the Zipf distribution reduces to the uniform distribution. So, the binomial distributions of these words are the same and since the corpus size, $\beta = 200$ is fixed the modes of each of the words are all $x = 50$. If $a = 0.05$, the binomial distributions of these four words are now separated and they are more separated at $a = 1$. The increased separation of the most frequent word to the rest is expected from the behavior of the Zipf distribution if a is increased. The binomials are also separated based on the shape parameter of the Zipf distribution. The word with the highest probability would have to be the most frequent.

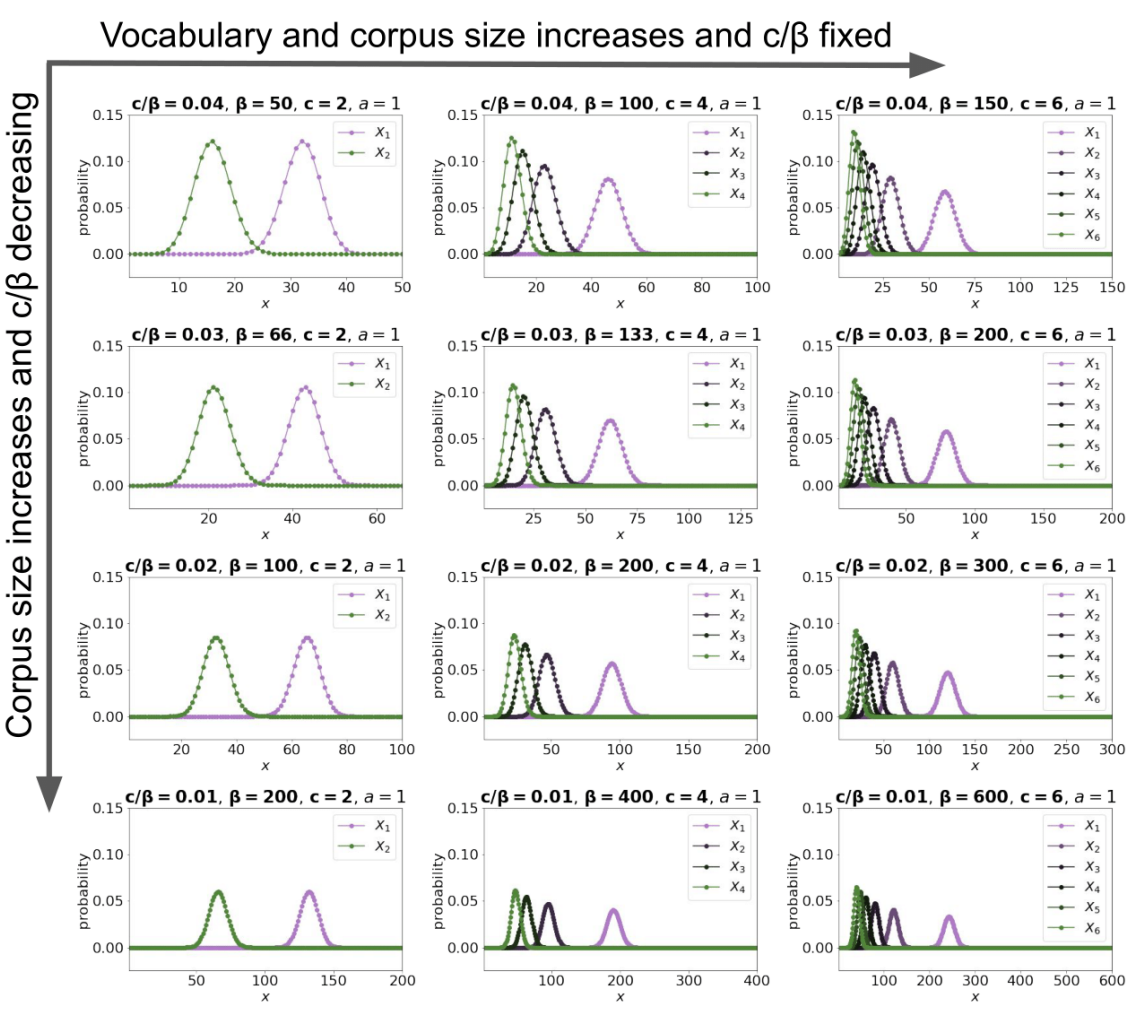
Figure 2.3: **The binomial probabilities at the initial time.** The subfigures below consists three plots where it shows the binomial probabilities of words with varying Zipf shape parameter. The increase in a resulted in separation of the binomials.



Next, we look at the binomials if the vocabulary size is increasing while the rest of the parameters are fixed. Fig. 2.4 shows the binomials of the words as the vocabulary size increases. The binomials at $c = 2$ are as expected since the Zipf distribution with a shape parameter of $a = 1$ or greater would force the two words to be separated. At $c = 4$ where the vocabulary size is four, the binomials are now overlapping since the probabilities of some of these words are close enough for them to have a chance of getting the same for close frequencies. At $c = 6$ where the vocabulary size is six, the binomials of the less frequent words are forced to overlap since the lowest probabilities of the Zipf distributions have shorter separations between them. This will give less frequent words to have higher chances to have the same or close frequency to each other. In short, the binomial curves are shown to overlap if the vocabulary size is increasing while the corpus size is fixed. This captures the intuitive idea that competition increases if vocabulary size increases.

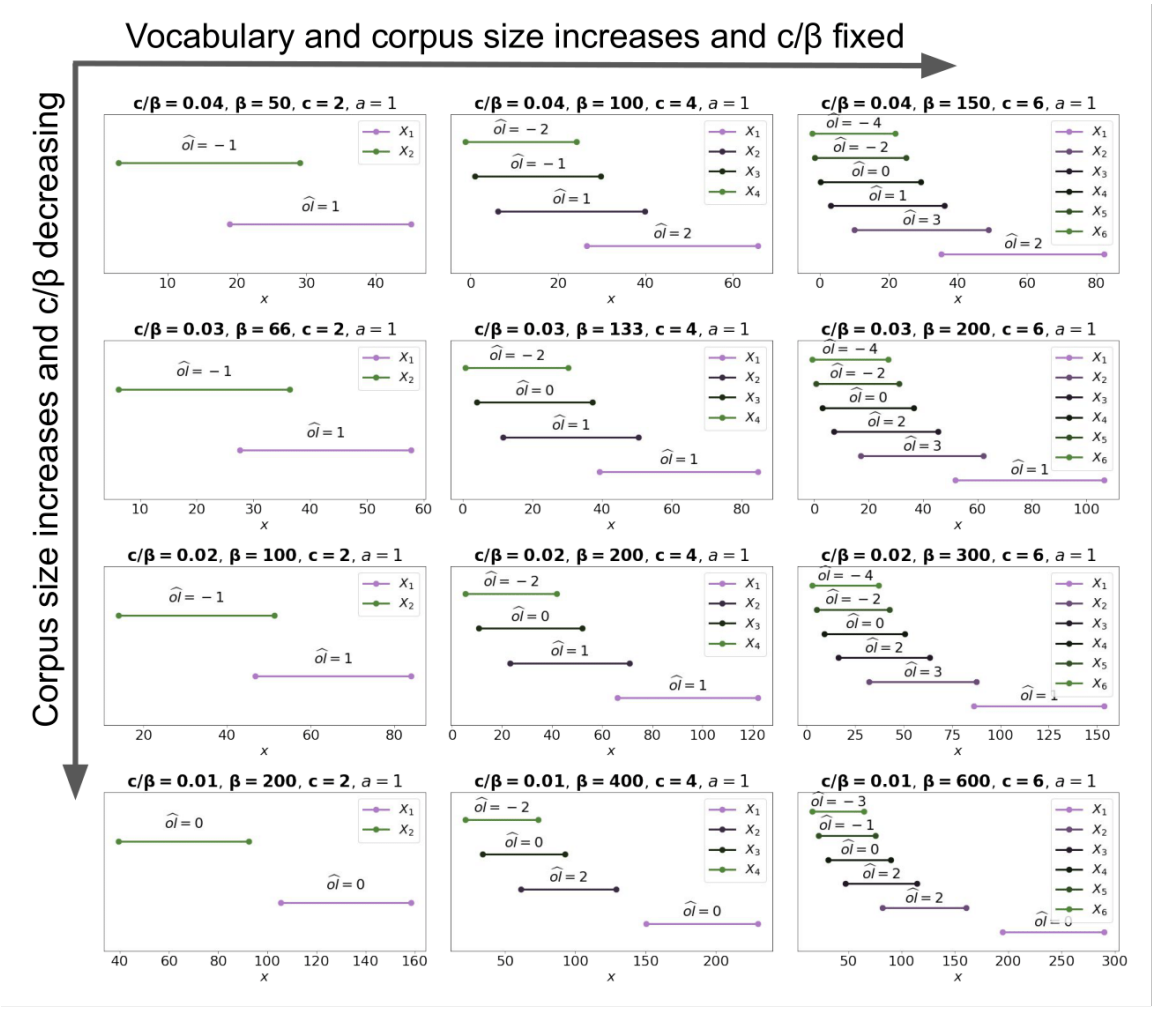
As we have seen from the previous subfigures, the binomials overlap if the Zipf parameter is large enough. Next, we look at the binomials if the corpus size is increasing. Fig. 2.4 shows the binomials of the words as the corpus size increases while the vocabulary size and the Zipf shape parameter is fixed. The corpus size is the total number of words which means that the words have more space to fill in. As seen at $\beta = 600$ the separation between the binomial curves has increased and the overlaps get narrower. This will have less frequent words to have lower chances to have the same or close frequencies. The parameters vocabulary size and corpus size are very important parameters. These parameters can change the separation between the binomials and the probability that two words can overlap in frequency. Fig. 2.4 are showing these behaviors more clearly. It shows that the binomials are more separated if the corpus size is increasing while the chances of any two words overlap in frequency increases if the vocabulary size increases. The ratio of c/β is a metric where it indicates the balance between the vocabulary size and the corpus size. If the ratio is small that means that the corpus size is large enough to indicate that the binomials of words have fewer cases where there are overlapping frequencies.

Figure 2.4: **The binomial probabilities at the initial time.** The subfigures below are cases of different parameter values showing overlapping binomial distributions which contributes to rank errors when sampling. Each row of subfigures have the same ratio. The direction to the right is where the vocabulary size increases while the direction downwards is where the corpus size increases.



Now, we look at the overlaps more closely. In Fig. 2.5, we show the intervals of these binomial cases where each interval is computed with $\mu \pm 4\sigma$ (see Eq. 2.19). Based on the intervals when the corpus size is increasing while the vocabulary size is 2, the separation of these intervals widened and the number of overlaps for each interval reduces to zero. Similarly, with the vocabulary size of 4 and 6, the number of overlaps between the intervals decreased. For the interval for the most frequent word, it looks like the separation from it to other intervals widens faster than the less frequent words as the corpus size increases. There are more overlapping intervals if the vocabulary size is large enough but if the corpus size is much larger than the vocabulary size (or the ratio c/β is low), the number of overlaps of these intervals decreases.

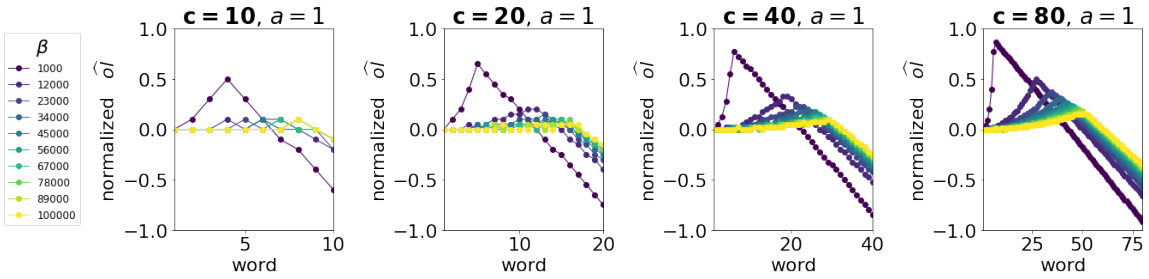
Figure 2.5: **The binomial overlaps at the initial time.** The subfigures below are intervals based on 4 standard deviations from the mean of the binomial distributions from Fig. 2.4. Each interval are computed using $\mu \pm 4\sigma$ (see Eq. 2.19). The value $\widehat{ol}$ as labeled is the net potential rank change (see Eq. 2.20). The sign indicates the direction of the rank change. Positive $\widehat{ol}$ means that a word has the net potential to go down in rank while a negative $\widehat{ol}$ is the opposite.



The net potential rank change - as explained in Eq. 2.20 - is the potential for a word to change ranks based on it's current rank. We look at this net potential values more broadly in terms of the initial corpus size β and the vocabulary size c in Fig. 2.6. For example, the Subfig labeled $c = 20$, the potential at higher ranks are positive which means that these word have the potential to go down in ranks. The words with negative potential means that these words have the potential to go up in ranks. We can see in the subfigs that as the corpus size increases the net potential for each rank decreases. The potential of words to change ranks decreases as the corpus size increases. By comparison, we can see in the subfigs that as the vocabulary size increases the net potential for each ranks increases. Again, more words means more competition and the potential of words to change ranks increases. We observe that there is a global maximum and minimum values for each case

which is the limit of word rank change potential.

Figure 2.6: **The averange potential rank change.** The subfigures below shows the net potential rank change (Eq. 2.20) based on a few examples of the corpus size β and the vocabulary size c. The subfigure indicate that as the corpus size increases the net potential decreases while an increase in vocabulary size increases the net potential. A positive $\widehat{ol}$ means a word has the potential to go down in rank while a negative $\widehat{ol}$ means a word has the potential to go up in rank. The normalization of the $\widehat{ol}$ values is by dividing the values by the vocabulary size.



Part 2 of 3: WF Simulations

Results summary. The Wright-Fisher inspired model is a statistical model that relies on sampling words. As explained in the previous section and the Methods section, the multinomial probability transitions of the Markov chain of transitioning from $t-1$ to t would result in the accumulation of sampling errors. These sampling errors are the fundamental reason for evolutionary drift. As explained in the Methods section, a smaller ratio c/β means that the initial corpus size is significantly larger than the vocabulary size which means that the words are less likely to have rank errors in sampling. If the vocabulary size is closer to the initial corpus size, the ratio c/β goes to 1 which means that the words are forced to have a frequency of just 1. There are two key results in this section.

- We observed that the ratio between c/β is an important indicator of the behavior and structure of a single WF inspired model simulation. A significantly small ratio of around 10^{-5} in magnitude results in the words to change ranks less regardless of their initial ranks.

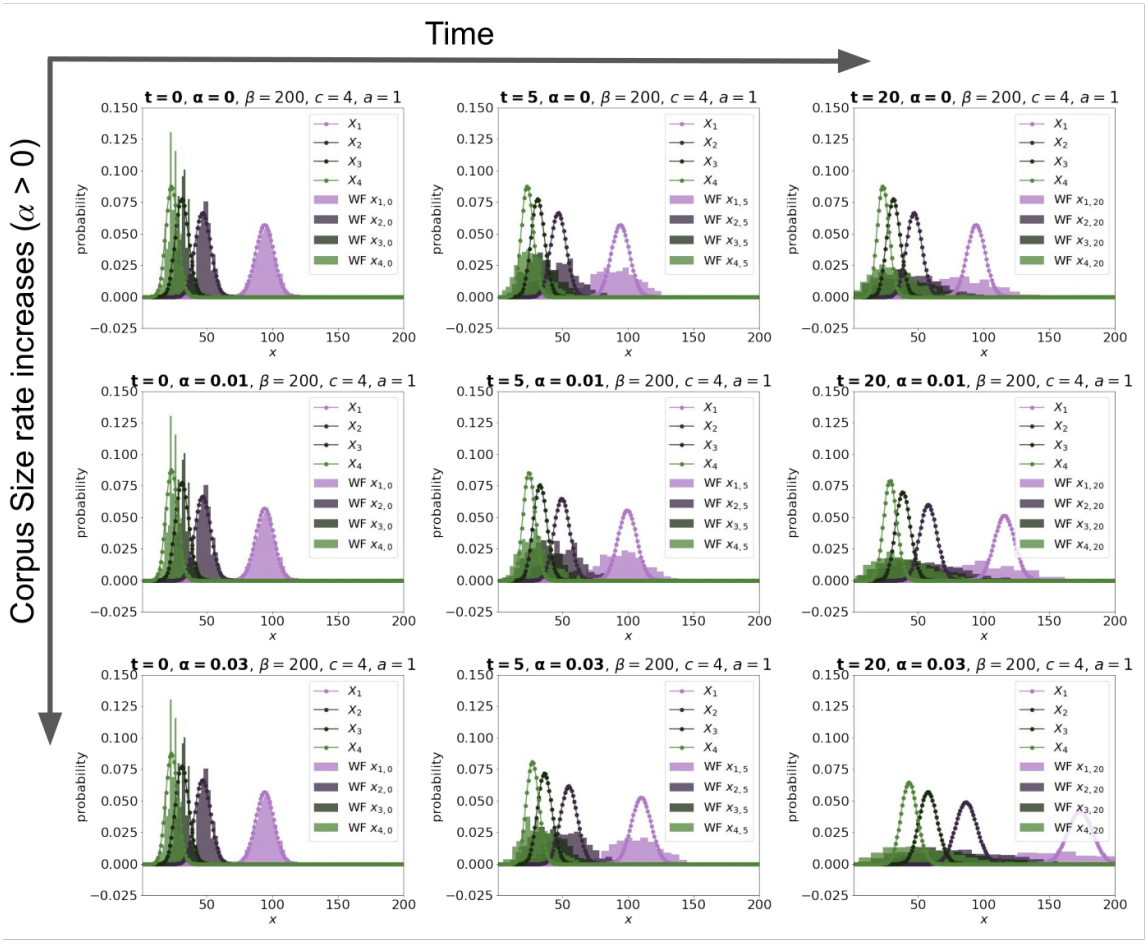
- In contrast, a larger ratio around 10^{-2} in magnitude results in the words to change ranks more often especially for lower-ranked words. This is due to the binomials overlapping for lower-ranked words for vocabulary sizes greater than two.

Below, we show the details of our findings.

First, we look at 1000 WF Simulations with simple parameters $\alpha = 0.01$, $\beta = 200$, $c = 4$, and $a = 1$. For all of the WF simulations shown in this section, the total time is set to be $T = 109$. We compare these simulations against the corresponding binomial curves as discussed in the previous section. The binomial curves are the theoretical distribution of the probabilities only at the initial time. Fig. 2.7 is showing that at $t = 0$ the Wf simulation

are right on the binomial curves as expected with the observed separation between the words and their overlaps. Since the WF is simulated by sampling words from the previous generation, the behavior of the samples at $t = 5$ and $t = 20$ appears to be widening and the chances of any two words to overlap increases.

Figure 2.7: **The binomial probabilities with 1000 WF simulations with parameters $\beta = 200$, $c = 4$, and $a = 1$.** The subfigures below shows the theoretical binomial curves of four words compared against the WF simulations (in shaded bars) at time points $t = 0$, $t = 5$, and $t = 20$ (by row) and at different corpus size rate increase $\alpha = 0$, $\alpha = 0.01$, and $\alpha = 0.03$ (by column). The β parameter is the initial corpus size while the α parameter is the rate of corpus size increase. As time moves forward the WF simulations are widening. We can also see that as α increases the WF simulations distributions move to the right. While the multinomial probability transitions of the Markov chain results in the accumulation of sampling errors, the expected value and variance of the raw counts also increases as the corpus size increases in time. The binomial curves are observed to shift right as time increases and as the corpus size increases.

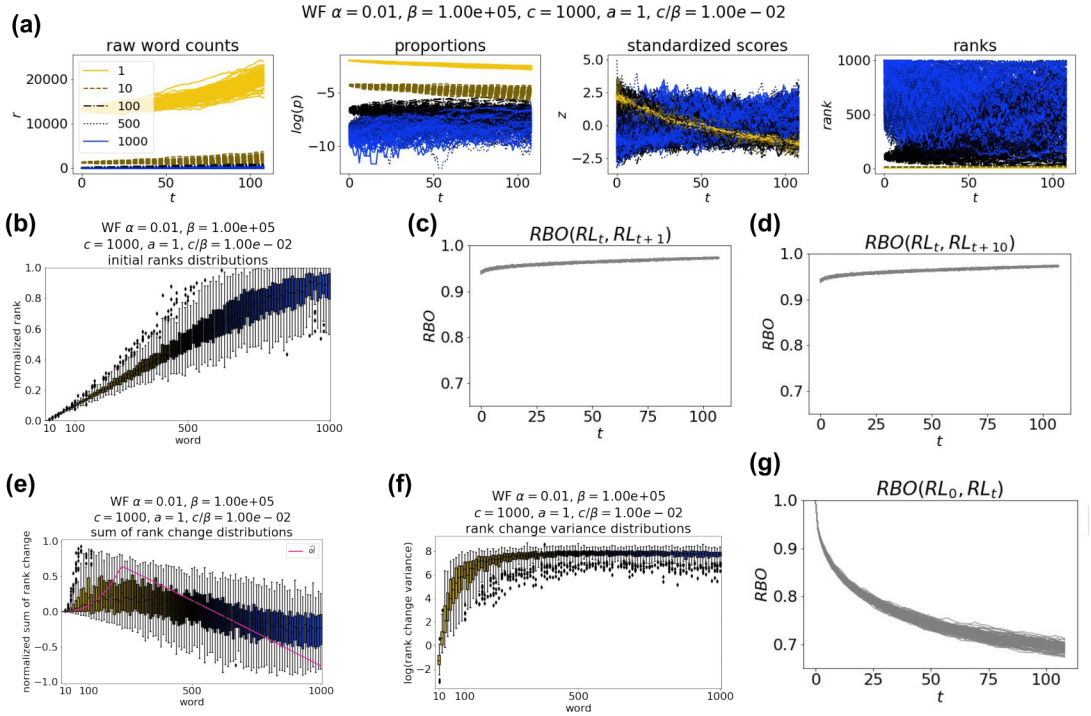


Next, we look at a 100 WF simulation using parameters $\alpha = 0.01$, $\beta = 1.00 \times 10^5$, $c = 1000$, and $a = 1$. The Subfigs in Fig. 2.8 shows the results of these 100 Wf simulations The time-series behaviors in Subfig. **(a)** shows that the high ranked words behave in a

more deterministic way because these words have fewer chances of overlapping with other words for it to change ranks. In contrast, the low ranked words show chaotic behaviors because these words have more chances of changing ranks in time. After all, as we have shown in the binomial analysis - the probabilities overlap increases for lower-ranked words if vocabulary size is large. At the initial, we can see in Subfig **(b)** that the rank distributions of the high ranked words are very narrow while the low ranked words have wider rank distributions. We can also verify that the low ranked words change in ranks more often than the high ranked words by looking at Subfig. **(e)** and **(f)**. For more simulation outcomes with smaller parameter values, see S1 Figure for simulations with varied β and S2 Figure for simulations with varied c.

The RBO trends of the simulations in Fig. 2.8 shows that the ordered rank list show consistent changes in time. It means that - even though the words are changing ranks - the overall structure of the word ranks are predictable. For example, Subfig. **(c)** and **(d)** shows that the RBO trends for RL_t and RL_{t+1} are consistently have the same pattern for all 100 WF simulations. The RBOs are greater than 0.90. It means more than 90% of the words at RBO_t retained their rank or have little change in rank at RBO_{t+1}. In comparison, the RBO for RL_0 and RL_t in Subfig. **(g)** shows decreasing RBO trends which means that the ordered ranks changed significantly since the initial time point. This means that small accumulated rank changes by year result in big changes in ranks for a longer time length. This is due to the accumulated sampling errors of the binomials. We can see that at $t = 100$ there only around 70% of words retained their rank or have little change in rank since the initial. Even though we see over 90% of words retained their rank at the current time from the previous time, the overall structure of the ordered ranks can change significantly through time. This is due to the behavior of the low ranks words to change in ranks more often than high ranked words but the corpus size is large enough for more words to have fewer chances of changing their ranks significantly.

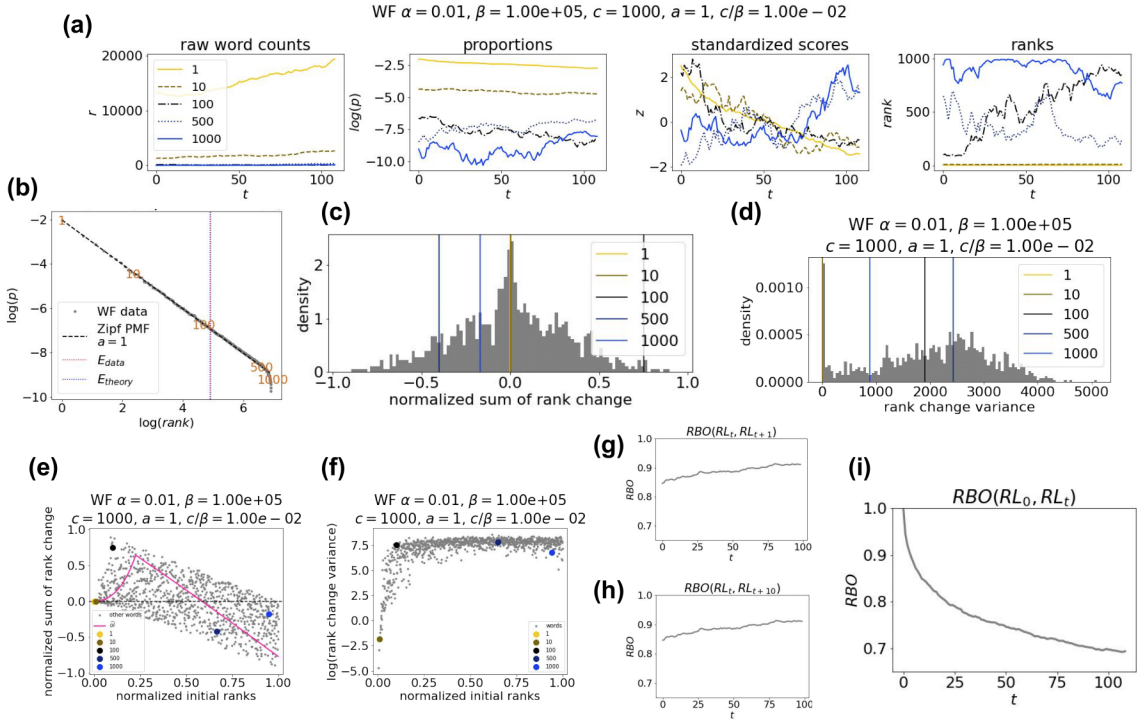
Figure 2.8: **100 WF simulations with parameters $\alpha = 0.01$, $\beta = 1.00 \times 10^5$, $c = 1000$, and $a = 1$.** The subfigures below are the results of 100 WF simulations of the given parameter set. Subfig. **(a)** is the time-series visualization of the raw word counts, proportions, standardized scores, and ranks. These figures show the time-series simulation outcomes of the five example words within the vocabulary. The words shown are the 1st, 10th, 100th, 500th, and 1000th initially ranked words. The results show that the highest initially ranked word has no outcomes where it changed ranks. Subfig. **(b)** shows the box plot rank distributions of the selected words at the initial time. Subfig. **(e)** and **(f)** are the rank change distributions of the selected words. The $\widehat{ol}$ line is the rank change potential from Eq. 2.20. As expected, the rank change distributions of the lower-ranked words have higher variances than the high ranked words. The low ranked words have little or no rank changes and have low variance distributions as expected. Subfig. **(c)**, **(d)**, and **(g)** shows the RBO trends of the WF simulations. It shows that the overall ordered ranks have a consistent pattern for all 100 WF simulations. This particular example have $c/\beta = 1.00 \times 10^{-2}$. Normalize rank means that the ranks are divided by the vocabulary size which is the variable c. For the normalized variance, the variances are divided by the maximum variance.



The results in Fig. 2.8 are the 100 different outcomes of the WF inspired model with parameters $\alpha = 0.01$, $\beta = 1.00 \times 10^5$, $c = 1000$, and $a = 1$. t. Next, we look at just one outcome of this particular WF model with the given parameters. As you can see in Fig. 2.9 Subfig. **(a)**, the rank time-series for the 1st and 10th initially ranked words have no changes in ranks and we know that for high initially ranked words, there is high certainty that these words have no other different outcomes. The 100th, 500th, and 1000th initially ranked words are observed to have rank changes and we know that for low initially ranked words, there are other possible outcomes than the one observed. In this particular case, we look at the structure of the multiple time-series as a whole. On

Subfig. **(b)**, this is the initial rank distribution of the words. Results show that at the lower-ranked words, there is an error in the samples as expected. On Subfig. **(c)**, this is the normalized distribution of the sum of rank change. The left tail represents the words that went up in ranks and the right tail represents words that went down in ranks. The normalization process is by dividing the values by the vocabulary size to make the sums in the range $[-1, 1]$. The distribution indicates that most of the words have little or no rank changes. This includes the 1st initially ranked word or the highest-ranked word which is located at zero. In contrast, the highest-ranked word is also located at zero when looking at the variance distribution shown in Subfig. **(d)**. There are also a lot of words with zero variance in their rank changes. This distribution is also normalized by dividing the values by the maximum variance to make the variances in the range $[0, 1]$. The variance distribution - unlike the sum distribution - is skewed and has two modes in this case. The initial rank distribution is important on how the rank changes in time. We see in Subfig. **(e)** that for words closer to high ranked words tend to go down in ranks while words closer to the low ranked words tend to go up in ranks. There is still uncertainty on how the rank changes based on the rank distribution but for the highest-ranked word, it will certainly remain in rank given a large enough initial corpus size. Similarly, for the lowest-ranked word, the only way it can change is to go up in rank. On Subfig. **(f)**, we see the initial rank versus the rank change variance. This figure tells us the volatility of how to word change in ranks. For example, the 10th initially ranked word is observed to changed down in ranks and have low variance. This means that the word is almost consistent in changing its rank upward. In comparison, the 500th initially ranked word is observed to change rank inconsistently were at around $t = 50$ to $t = 60$, the word radically changed its direction. We see in Subfig **(f)** that the variances in relation to the initial ranks are high and uncertain for lower ranked words.

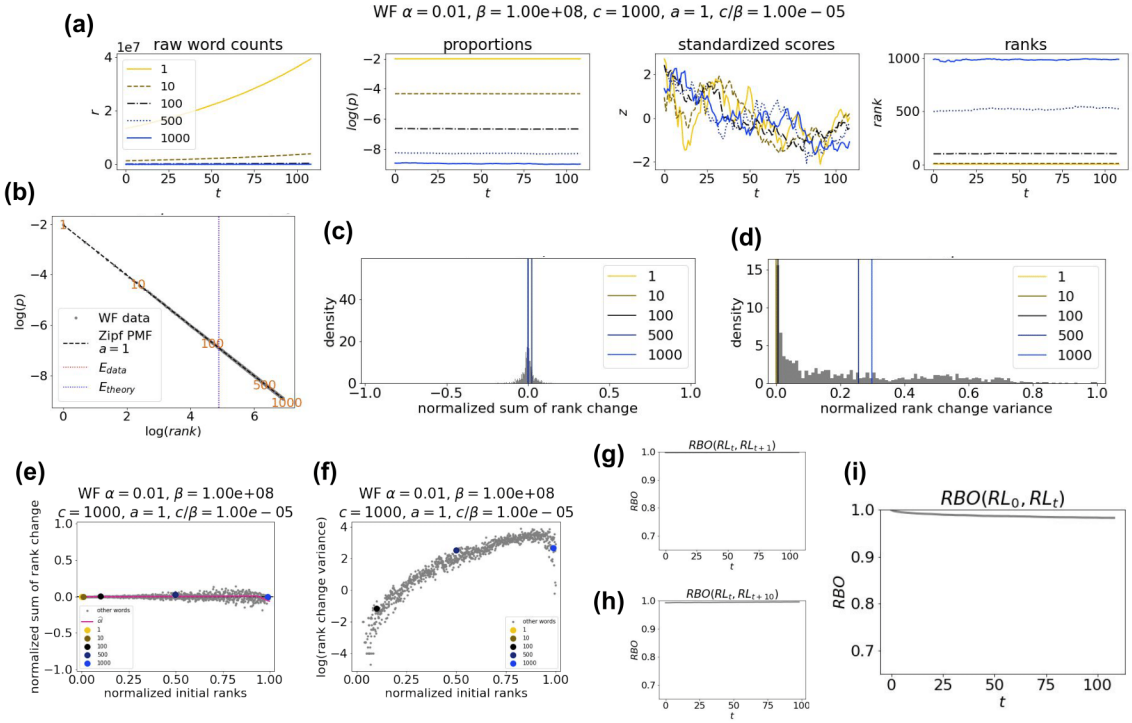
Figure 2.9: **One WF simulations with parameters** $\alpha = 0.01$**,** $\beta = 1.00 \times 10^5$**,** $c = 1000$**, and** $a = 1$**.** The subfigures below are the results of one WF simulation of the given parameter set. Subfig. **(a)** is the time-series visualization of the raw word counts, proportions, standardized scores, and ranks. These figures show one time-series outcome of the five example words within the vocabulary. Subfig. **(b)** is the initial distribution of word ranks with annotated words from Subfig. **(a)**. The label of the word corresponds to its initial rank. Subfig. **(c)** is the normalized distribution of the sum of rank changes of all 1000 words while Subfig. **(d)** is the normalized distribution of the rank change variance. The mode of these distribution is close to zero but the second mode is around 0.50. Subfig. **(e)** is a scatter plot of the initial ranks versus the sum of rank change of the words. The $\widehat{ol}$ line is the rank change potential from Eq. 2.20. It shows that high initially ranked words tend to go down in ranks while low initially ranked words tend to go up in ranks. Similarly in Subfig. **(f)**, high initially ranked words have lower variances while low initially ranked words have higher variances. Here we can see that the words are widely scattered but there is a distinct curve word follow. Subfig. **(c)**, **(d)**, and **(g)** shows the RBO trends of the one WF simulation. This particular example have $c/\beta = 1.00 \times 10^{-2}$. The normalization of the sums is by dividing the values by the vocabulary size while the normalization of the variances is by dividing the values by the maximum variance.



Next, we look at an extreme case of the WF inspired model where we increase the initial corpus size significantly. In Fig. 2.10, we observe a radical change in the behavior of the WF inspired model. As mentioned in the binomial analysis from the previous section, higher corpus size would result in words to a less likely change in ranks because their binomial distributions would overlap less. We can see the effects of an extreme initial corpus size in Subfig. **(a)** where the low ranked words are stabilizing unlike the behaviors in the less extreme case. The high ranked words also remained stable in ranks.

In Subfig **(e)**, the sum of rank changes of words are mostly low and zero sums regardless of their initial ranks. This means that almost all of the words have not changed in ranks significantly. The rank change variance are also low for high ranked words shown in Subfig. **(f)**. The initial ranks versus the rank change variance shown in Subfig **(f)** exhibit a predictable pattern where high ranked words have less variance while low ranked words have more variance. In general, the variances of each word here is low but the normalization process of the values made the curve obvious. The curve also explains that low ranked words still has some amount of variance even with extreme initial corpus size. There is a special case happening for the lowest-ranked words where we observed that the variance starts to go down. It would be reasonable to predict that for an infinite amount of initial corpus size, the sum of rank change variances would go to zero and the variances would also go to zero. For Subfigs **(g)**, **(h)**, and **(i)**, the RBO curves for this extreme case of WF inspired model shifted up because the initial corpus size is large enough for the words to have less rank change.

Figure 2.10: **An "extreme" case of One WF simulations with parameters** $\alpha = 0.01$**,** $\beta = 1.00 \times 10^8$**,** $c = 1000$**, and** $a = 1$**.** The subfigures below are the results of one WF simulation of the given parameter set. This is an extreme case with significantly higher β of the one shown in Fig. 2.9. Subfig. **(a)** is the time-series visualization of the raw word counts, proportions, standardized scores, and ranks. These figures show one time-series outcome of the five example words within the vocabulary. Subfig. **(b)** is the initial distribution of word ranks with annotated words from Subfig. **(a)**. The label of the word corresponds to its initial rank. Subfig. **(c)** is the distribution of the sum of rank changes of all 1000 words while Subfig. **(d)** is the distribution of the rank change variance. The mode of this distribution is still close to zero but the entire distribution contracted towards zero. Subfig. **(e)** is a scatter plot of the initial ranks versus the sum of rank change of the words. The $\widehat{ol}$ line is the rank change potential from Eq. 2.20. It shows that, given that β is extremely high while other parameters are fixed, there a little no rank changes regardless of initial ranks. Similarly in Subfig. **(f)**, high initially ranked words have significantly lower variances ralative to low initially ranked words. We can see clearly where the words follow a curve and the words are not as scattered as from the previous case but in general the words in this case are very low. Subfig. **(c)**, **(d)**, and **(g)** shows the RBO trends of the one WF simulation. This particular example have $c/\beta = 1.00 \times 10^{-5}$.

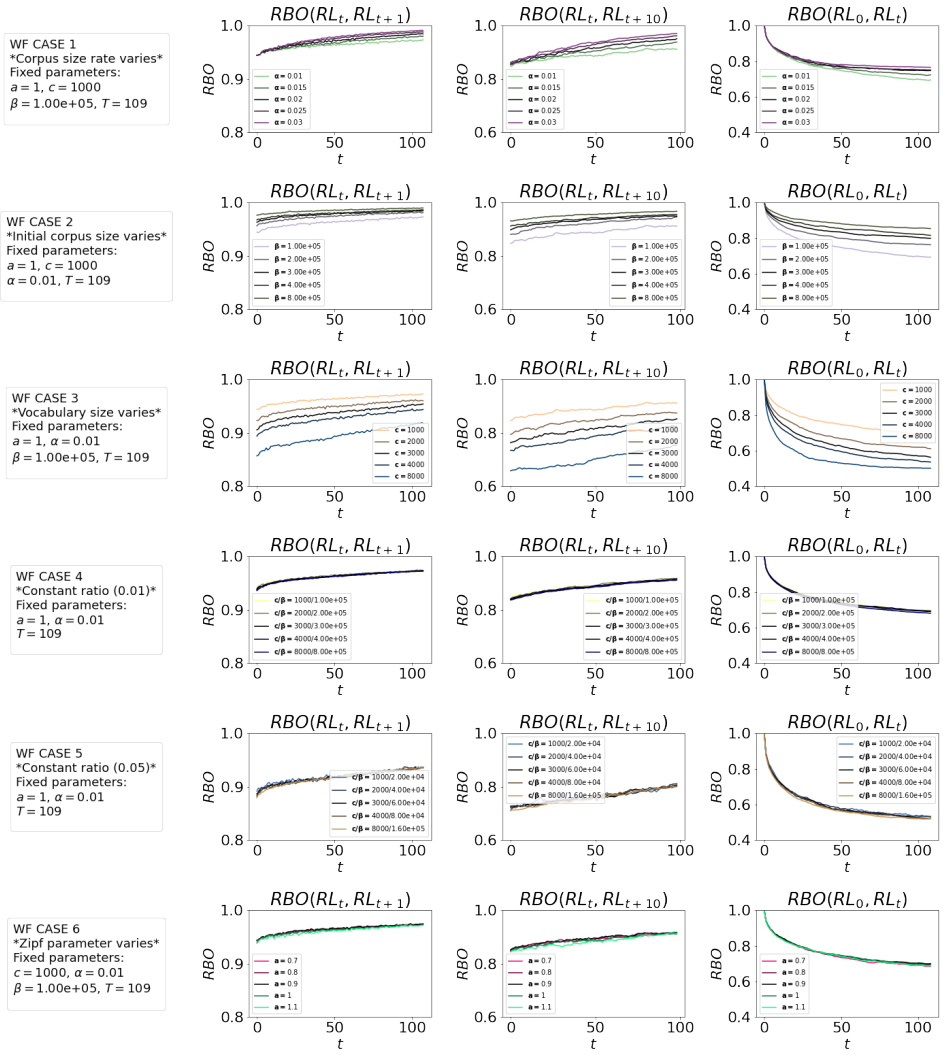


We further investigate the behaviors of the RBO curves by changing the parameters of the WF inspired model. The RBO measure is the similarity between two ranked lists which is defined in Eq. 2.24. In Fig. 2.11, we show six different cases of varied parameter values while other parameters are fixed. Case 1 is when the corpus size rate is varied. This is when the corpus size of the WF inspired model increases exponentially in time. That means that - while the vocabulary size stays fixed - the behavior of the words in time get less likely to change ranks. We see in the RBO trends that the curve slightly

shifts upward as α increases which means that the words are stabilizing their rank in time. Similarly in Case 2, while the initial corpus size is varied and increasing, the entire RBO curves shifted more strongly upward because the corpus size is large enough, to begin with for words to stabilize rank more quickly. In contrast, Case 3 is when the vocabulary size c is varied and increasing. We see in this case that the RBO curves shift downward as c increases. This means that - while the initial corpus size stays fixed - the increase in vocabulary size would let words strongly compete with each other and they would have more than likely to change ranks in time. Cases 4 and 5 in Fig. 2.11 shows that the RBO curves does not shift if the ratio c/β is fixed. Even with the vocabulary size and initial corpus size is increasing while the ratio is fixed, the behavior of the words collectively are consistent throughout. Similar to Cases 4 and 5, Case 6 - where the Zipf shape parameter is varied and increasing - the RBO curves stayed the same which means that no matter what the initial state distribution is defined the RBO curves will be consistent throughout.

For a fixed ratio of c/β while c and β are changing, the overall structure and behavior of the word rank change give consistent RBO curves for the WF inspired model. This means that the ratio of c/β is an indicator of how an entire language space is behaving compared to the WF model. For instance, we expect similar word rank change behaviors for a real language with $c/\beta = 1.00 \times 10^{-5}$ compared to the "extreme" case we showed in Fig. 2.10 with ratio $c/\beta = 1.00 \times 10^{-5}$. The languages we observed do have a small ratio c/β down to 10^{-6} in magnitude but we see different rank change behaviors in the real languages compared to the "extreme" case of the WF inspired model. In the next Section, we explain the details of our results of the languages comparing it to the WF inspired model.

Figure 2.11: **WF inspired model single simulations of different cases of parameter sets showing the RBO trends.** The subplots below show six cases of Wf simulation where each case has fixed parameter values and a varied parameter. Looking at the subplots by row, the RBO curve shows predictable patterns. Case 1 - with the corpus size rate α is varied - showed the RBO curve to "level-off" or shifted slightly upward as α increases which means that as the corpus size increases, the ranks are changing less. Similarly, Case 2 - with the initial corpus size β is varied - showed the RBO curve to shift up more strongly than Case 1. Case 3 - with the vocabulary size c is varied - showed the RBO curve to shift down as c increases. This means that as the vocabulary size increases the words are more likely to change ranks because there is more competition among words. Cases 4 and 5 - with the ratio c/β is constant while c and *beta* are increasing - showed consistent RBO curves. This means that the overall structure of the words remained largely consistent as long the ratio of c/β remained the same regardless c and β increases or decreases. Similarly Case 6 - with the Zipf shape a parameter is varied - showed RBO curves not changing as a increases.



Part 3 of 3: Google Unigram Data

Results summary. The Google Unigram data contains the time-series of unigrams from 1900 to 2008 in eight unique languages. The WF inspired model simulates the drift evolutionary process in which the frequency of the unigrams changes based entirely on frequency-dependent resampling. We compared the languages from the Google Unigram data and the "extreme" case of the WF inspired model and found these three key results.

- Most of the words in each of the eight languages have relatively little or no rank changes compared to other words which suggest that the languages we considered are mostly stable.
- Some words change in ranks significantly from 1900 to 2008 in two ways: (a) by accumulation of small increasing/decreasing rank changes in time and (b) by shocks of increase/decrease in ranks. Most of the stop words and Swadesh words appear to be stable in ranks for each of the eight languages.
- The word ranks in the languages as a whole change more significantly than the "extreme" case of the WF inspired model despite the ratio c/β to be as low as the languages. This suggest that words in each of the languages collectively behave contrary to a neutral evolutionary process.

The above three summaries are the most important results in this section. Below we explain the details of the results and show that the languages deviate from neutral evolution.

First, we look at an overview of the Google Ngram data in Table. 2.4. After processing the Google unigram data, the resulting vocabulary for each language is in the c_{data} column of the Table. The resulting initial corpus size is in the β_{data} column of the Table. The vocabulary sizes of the languages varied a lot where the highest is 18737 for the English language and 180 for the Simplified Chinese language. The overwhelmingly large initial corpus size results in the ratio c/β to be roughly 10^{-6}. The vocabulary sizes are that way since the processing includes where we remove the words that have zero counts in any year. The data we present here is a representative sample of the words that are used every year from 1900-2008 for each of the languages. It is important to mention that there are three other variants in the English language here namely American English, British English, and English Fiction. The English language is the combined information of all variants. They have overlapping vocabulary words. There are also Spanish and French words in the English set because English speakers often borrow words from these languages historically and culturally.

Next, we look at the fits of the corpus function in Eq. 2.7 and the Zipf probability mass function in Eq. 2.6. Recall that the corpus size function governs the corpus size (or total unigram frequency of words) at time t with parameters β which is the initial corpus size and α which is the corpus size rate increase. These parameters are estimated such that these numbers fit into the corpus size time-series data of each language. Using the log transformed corpus size time-series and the log transformed corpus size function, we

fit these parameters using the non-linear least-squares method (See S13 Appendix for more details). Results show that the α values are roughly close to each other between the languages. The language with the highest rate of corpus size change is Simplified Chinese which is much higher than other languages. See S3 Figure for the visualization of the corpus size function fitted against each language data corpus size time-series. The initial Zipf distribution shape parameter a for each language varies but roughly close to 1. The shape parameter was fitted using the *scipy.optimize.curve_fit* module in Python [79] (See S13 Appendix for more details) and the data is truncated such that the highest-ranked word up to the word closest to the expected rank of the data (see Eq. 2.11) is fitted while the rest is ignored. Since we know that lower-ranked words are prone to sampling errors, this method of semi-data truncation will fully minimize the error of the fit to achieve optimal fit for the most frequently used words. Fig. 2.12 Subfig **(b)** shows the fitted log-transformed Zipf function fitted against the English initial log-transformed rank distribution. We can see in this figure that the lower ranks presented a significant error from the fitted line but these data points are ignored in the fit. The higher ranked words are almost perfectly on the fitted line. The resulting fitted Zipf shape parameter for the English initial rank distribution is $0.9923 \approx 1$. See S4 Figure for the visualization of the log-transformed Zipf function fitted against each language data. The Zipf probability mass function Eq. 2.6 is non-linear but applying the log transform of the function yields a linear function where the shape parameter is the slope of that linear function. There are other research works where they discussed fitting the curves of the log-transformed Zipf with a more generalized Zipf distribution called the Zipf-Mandelbrot distribution but in our case, we only focus on the linear aspect [52, 64]. The general assumption of the shape parameter of the Zipf distribution is assumed to be 1 given large text data [6]. However, it is worth mentioning that the value of this parameter is not fixed and it is shown that it varies with time and with linguistic complexity [6, 44]. For our analysis, we have large enough data to assume that this parameter is close to 1 as we have shown in fitting the Zipf distribution on the language data.

We have previously shown six example time-series of English words in Fig. 2.12, we show more details on other words and their rank change behaviors. Subfig **(a)** show that for the most frequent words "the" and "a", their ranks remained constant while lower-ranked words have some changes in their ranks. The word "jobs" initially in the lower rank (Subfig **(b)**) but became high ranked in time. The sum of these rank changes is a metric for each word on how their rank changes in time. For example, the word "jobs" has a negative sum of rank change which is higher than most words. Negative sums mean that a word changes up in ranks while a positive-sum means a word change down in rank. The value of the sum is the magnitude of the change. We can see the normalized distribution of these sums of rank change for the English language in Subfig **(c)**. The sums are normalized by dividing the values by the vocabulary size. This will transform the sums in the range $[-1, 1]$. Shown in Subfig **(c)**, the set **A** of words are the words that changed up in ranks in 109 years. Words such as "jobs", "job", "user", "users", "marketing", and "housing" are words with socio-economic meanings. Because of the advancements of technology in the past century, these words changed up in ranks and

Table 2.3: **Table of the Google unigram data vocabulary sizes (c_{data}) and initial corpus sizes (β_{data}).** This Table includes the number of available stop words (c_{stop}) and Swadesh words (c_{swad}) in each language. The last column is the vocabulary to corpus size ratio c_{data}/β_{data}.

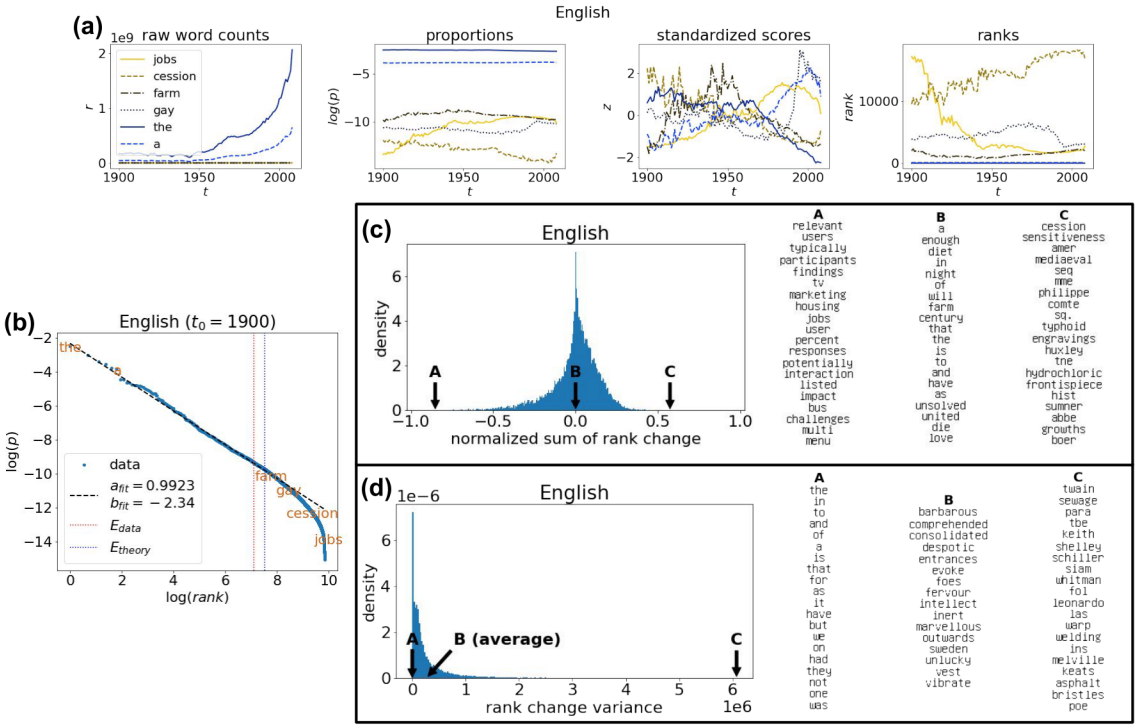
Language	c_{data}	c_{stop}	c_{swad}	$\ln(\beta_{data})$	$\frac{c_{data}}{\beta_{data}}$
English	18737	571	202	21.4598	$9e^{-6}$
American English	16410	568	202	21.3127	$9e^{-6}$
British English	4759	592	171	20.2960	$7e^{-6}$
English Fiction	5651	478	193	19.0564	$3e^{-5}$
Simplified Chinese	180	49	30	13.7613	$1.9e^{-4}$
French	12168	116	193	20.5723	$1.4e^{-5}$
German	5871	113	142	19.7012	$1.6e^{-5}$
Italian	4446	123	121	18.9443	$2.6e^{-5}$
Hebrew	3000	313	144	16.8523	$1.44e^{-4}$
Russian	828	238	53	18.0485	$1.2e^{-5}$
Spanish	10661	140	174	19.1625	$5.1e^{-5}$

became widely used. On the opposite end of the distribution, the set **C** are words that changed down in ranks in 109 years. These words became less frequent in usage because of changes in culture. For example, the word “mediaeval” with the emphasis in the extra letter “a” between “i” and “e” became less frequent in usage compared to the word “medieval”. Probably because of a spelling preference between American versus British where more people prefer writing it without the “a” for simplicity. Other words such as “cession” and “typhoid” are also words that went down in ranks significantly. In addition, words such as “phillipe”,“huxley”, “sumner”, “abbe”, and “boer” are names that became less popular today than 100 years ago. The set **B** of words is mostly the stop words such as “a”, “in”, “off”, “that”, “the”, “is”, and “it”. These are the most frequently used words and more stable in ranks than other words. On Subfig **(d)**, this is the normalized distribution of the rank change variance. Normalized means that the values are divided by the maximum variance. We can see this Subfig that set **A** of words are mostly stop words because they are stable in ranks. The set **B** are words with an average variance while the set **C** are words with the highest variance. For words with the highest variance, they are mostly names and nouns such as “twain”, “keith”, and “shelly”. This is probably because names are more susceptible to changes based on seasonality. Furthermore, the words listed in these lists for both Subfig **(c)** and **(b)** are words in the English set of the Google Ngram data. The American English, British English, English Fiction are special sets of English Language data to separately distinguish different vocabulary usage of American culture, British culture, and the words used in fictional writings. The English set is a combination of all three. The distributions of other English dataset and other languages are shown in S5 Figure and S6 Figure.

Table 2.4: **Table of the fitted parameter values of the corpus size function and the Zipf probability mass function.** This table shows the fitted estimate of the parameter values for the corpus size function from Eq. 2.7 and the Zipf shape parameter for the initial distribution in Eq. 2.6. The 99% confidence intervals are computed using the student-t distribution.

Language	$\ln(\beta_{fit})$ (99% CI)	$\frac{c_{data}}{\beta_{fit}}$	α_{fit} (99% CI)	a_{fit} (99% CI)
English	20.7866 (20.5867, 20.953)	$1.80e^{-5}$	0.0239 (0.021, 0.0267)	0.9923 (0.9879, 0.9967)
American English	20.6247 (20.4173, 20.7964)	$1.80e^{-5}$	0.0215 (0.0185, 0.0245)	1.0056 (1.0016, 1.0095)
British English	19.5205 (19.2605, 19.7267)	$1.6e^{-5}$	0.0179 (0.0142, 0.0215)	1.0388 (1.0316, 1.0461)
English Fiction	18.0306 (17.735, 18.2585)	$8.30e^{-5}$	0.0306 (0.0265, 0.0347)	1.0776 (1.0662, 1.0889)
Simplified Chinese	9.4742 (8.595, 9.9347)	$1.38e^{-2}$	0.1068 (0.0975, 0.1162)	0.8244 (0.6986, 0.9501)
French	19.8126 (19.5795, 20.0015)	$3.0e^{-5}$	0.0109 (0.0075, 0.0142)	0.9969 (0.9881, 1.0058)
German	19.3325 (19.0549, 19.5496)	$2.4e^{-5}$	0.0120 (0.0082, 0.0159)	1.0317 (1.0201, 1.0432)
Italian	18.3050 (18.1225, 18.4593)	$5.0e^{-5}$	0.0208 (0.0181, 0.0235)	0.9948 (0.9856, 1.0039)
Hebrew	15.8291 (15.5314, 16.0581)	$4.01e^{-4}$	0.0342 (0.03, 0.0383)	0.9771 (0.9664, 0.9878)
Russian	17.3915 (17.0154, 17.6642)	$2.30e^{-5}$	0.0319 (0.0269, 0.0369)	0.9151 (0.8937, 0.9365)
Spanish	18.9342 (18.8706, 18.994)	$6.4e^{-5}$	0.0265 (0.0255, 0.0274)	0.9378 (0.9267, 0.9489)

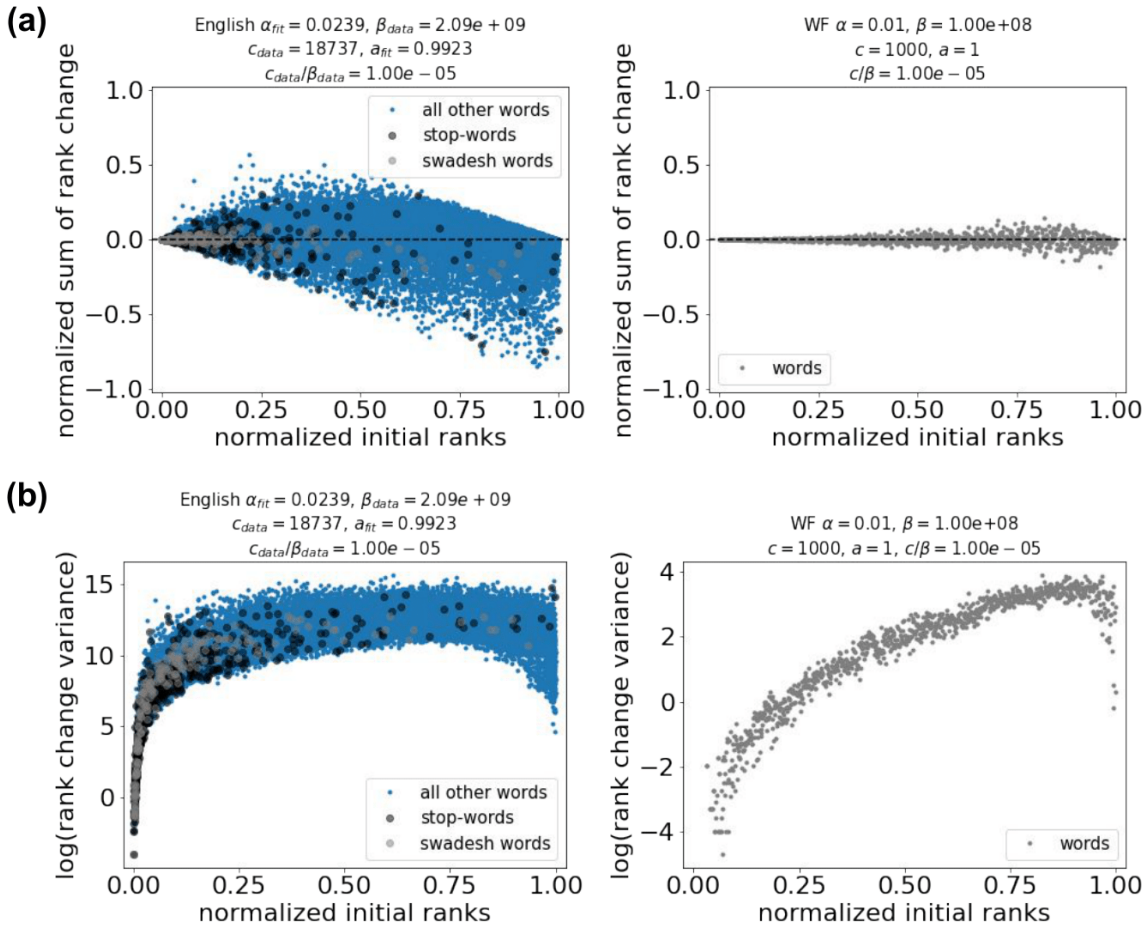
Figure 2.12: **Time-series visualization of six example words of the English data and the distributions of sum of rank change and rank change variance of all English words in the data.** Subfig **(a)** is the time-series of six example words in the English vocabulary. This figure shows the words "a", "the", "gay", "farm", "cession", and "jobs". We see that the most frequent words "a" and "the" did not change ranks in time. Subfig **(b)** is the initial rank distribution with the fitted Zipf shape parameter *a*. Subfig **(c)** is the normalized distribution of the sum of rank changes showing words that went up in ranks (list **A**), little or no rank change (list **B**), and words that went down in ranks (list **C**). The sums are normalized by dividing the values by the vocabulary size. We see that most words in the list (list **A**) are stop words and Swadesh words. Subfig **(d)** is the normalized distribution of the rank change variances showing words with little or no variance (list **A**), words with average variances (list **A**), and words with extreme variances (list **C**). We also see that all of the words in (list **A**) are stop words and Swadesh words. The variances are normalized by dividing the values by the maximum variance.



Now that we considered the distributions of the sum of the rank change, next we consider the initial rank distribution compared to the sums. From our analysis in the WF simulations, the initial distribution is important since it determines the overall behavior on how the rank changes in time. We show the results of the English dataset compared with the "extreme" case of the WF inspired model single simulation in Fig. 2.13. Subfig **(a)** shows the scatter plot of the initial rank versus the sum of rank change. It shows that, for the English language, the words in the high ranks initially tend to go down in ranks while the words in the low ranks initially tend to go up in ranks. Most of the words have sums closer to zero which means that the words in the English language want to be stable. However, some words did change in ranks significantly. We can also see that the stop and Swadesh words are in the high ranks and most of them have little or zero sums

of rank change which means that these words are stable. The English dataset has ratio $c/\beta = 1.00{\times}10^{-5}$. That means that there is a significantly smaller lexicon than corpus size. Compared to the "extreme" case of the WF model where we set the ratio to be the same $c/\beta = 1.00 \times 10^{-5}$, the results are different from the English data. *Why?* As we mentioned in the Methods section, the binomial components of the multinomial distributions get separated as the corpus size increases. For smaller corpus sizes, the binomials overlap, and there are higher chances for words to change ranks. The "extreme" case of the WF model has significantly more corpus size than the vocabulary size. That is why the sum of rank change for each word is close to zero. In comparison to the English data where it also has more corpus size than vocabulary size, the English sum of rank changes have words with high sums and most of its words have little sums. Even though the ratio is the same the difference between the English data and the WF inspired model is striking. The WF inspired model mainly models purely the drift evolutionary process. The smaller the corpus size, the stronger the drift effect. The higher the corpus size the drift effect is weaker. The English data shows that some words are behaving contrary to drift.

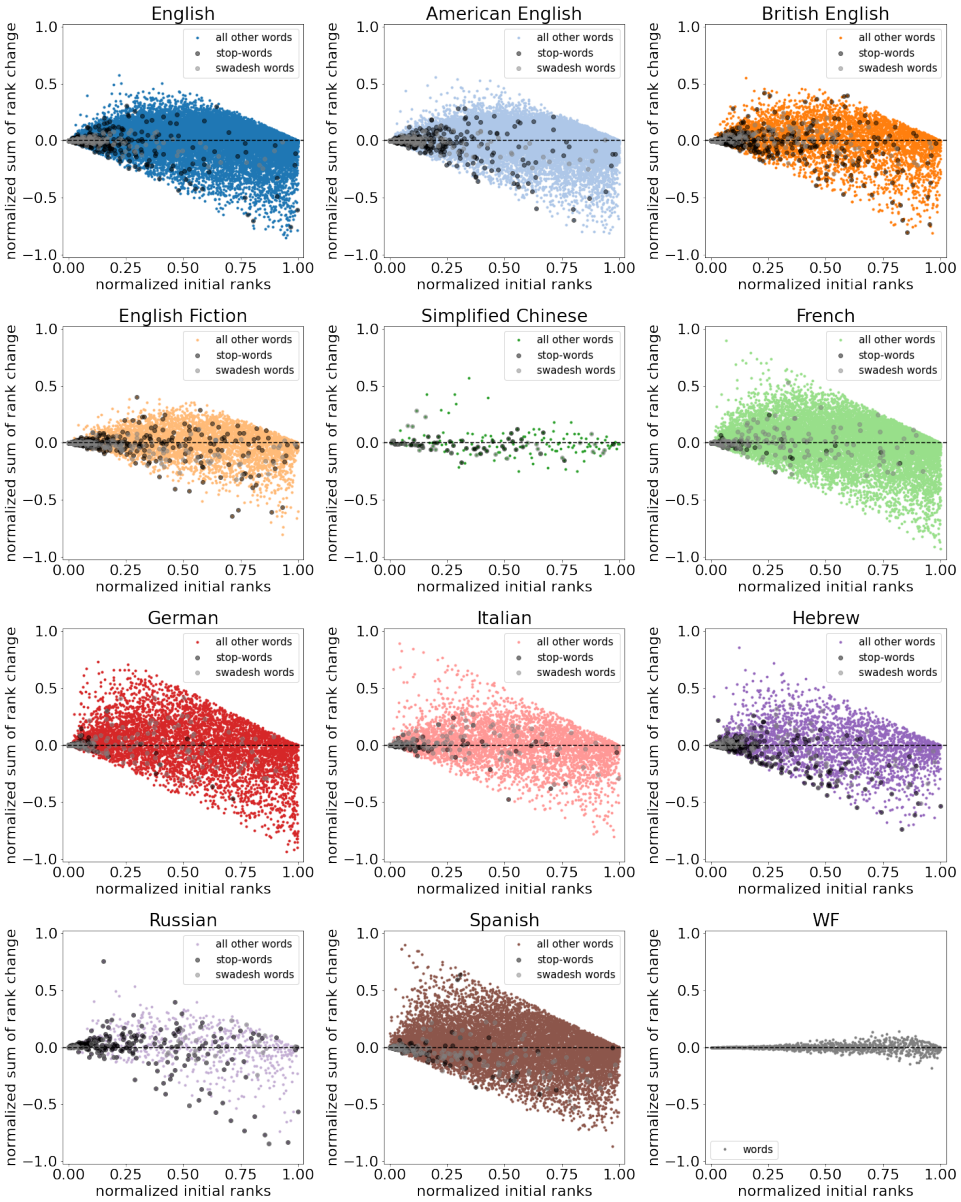
Figure 2.13: **The initial ranks versus the sum of rank change and rank change variance of the English data compared with the WF inspired model.** Looking at the subplot by row, Subfig **(a)** shows the initial ranks versus the sum of rank changes of the English data compared with the "extreme" case of the WF inspired model. This figure also shows the stop words and Swadesh words on where they are located in the English scatterplot. Subfig **(b)** shows the initial ranks versus the rank change variance of the English data compared with the extreme case of the WF inspired model. It shows that these groups of words are the most frequently used words and most of them have little or no change in ranks with minimal variance. We also see a big differences in the shape of the scatterplots between the English data and the WF inspired model even though they have both almost the same ratio down to $c/\beta = 1.00 \times 10^{-5}$. The normalization of the sums is by dividing the values by the vocabulary size while the normalization of the variances is by dividing the values by the maximum variance. Most of the words for both the English and the "extreme" case of the WF model have very low rank change variances. We note that due to the max normalization of the variances, the English data appears to be flat compared to the WF model mainly because English and other languages have outliers and most of the variances are low.



In Fig. 2.14, it shows the initial ranks vs the sum of rank change for each language in our work. First, most of the stop and Swadesh words are consistently in the high ranks with low sums for all of the languages. Second, the rectangular pattern is present in each

language where it means that the high initially ranked words tend to go down in ranks while low initially ranks words tend to go up in ranks. Finally, the Simplified Chinese have the lowest vocabulary size and the results show that some words have high sums while most of the words have small sums. The rectangular pattern is not a coincidence. It is the effect of the multinomial Markov chain where the initial frequency is sampled from a Zipf distribution. The Zipf distribution has a shape parameter a that governs how the probability of the most frequent and the smallest are separated. For $a \gg 0$, the most frequent word can only go down in ranks while the smallest can only go up in ranks.

Figure 2.14: **The initial ranks versus the sum of rank change of the Language data compared with the extreme case of the WF inspired model.** Each of the subplots below shows the initial rank versus the sum of rank change for each language in the Google unigram data with stop words and Swadesh words annotated on the scatterplot. First, we see that these group of words are consistent for all languages where they are the highest ranks and they have little or no rank change in time. Second, all of them exhibits a rectangular shape showing that high ranked words tend to go down in ranks while low-rank words tend to go up in ranks. Finally, the extreme case of the WF inspired model shows a radically different outcome. The ratios c/β are low for all the languages and the "extreme" case of WF inspired model.

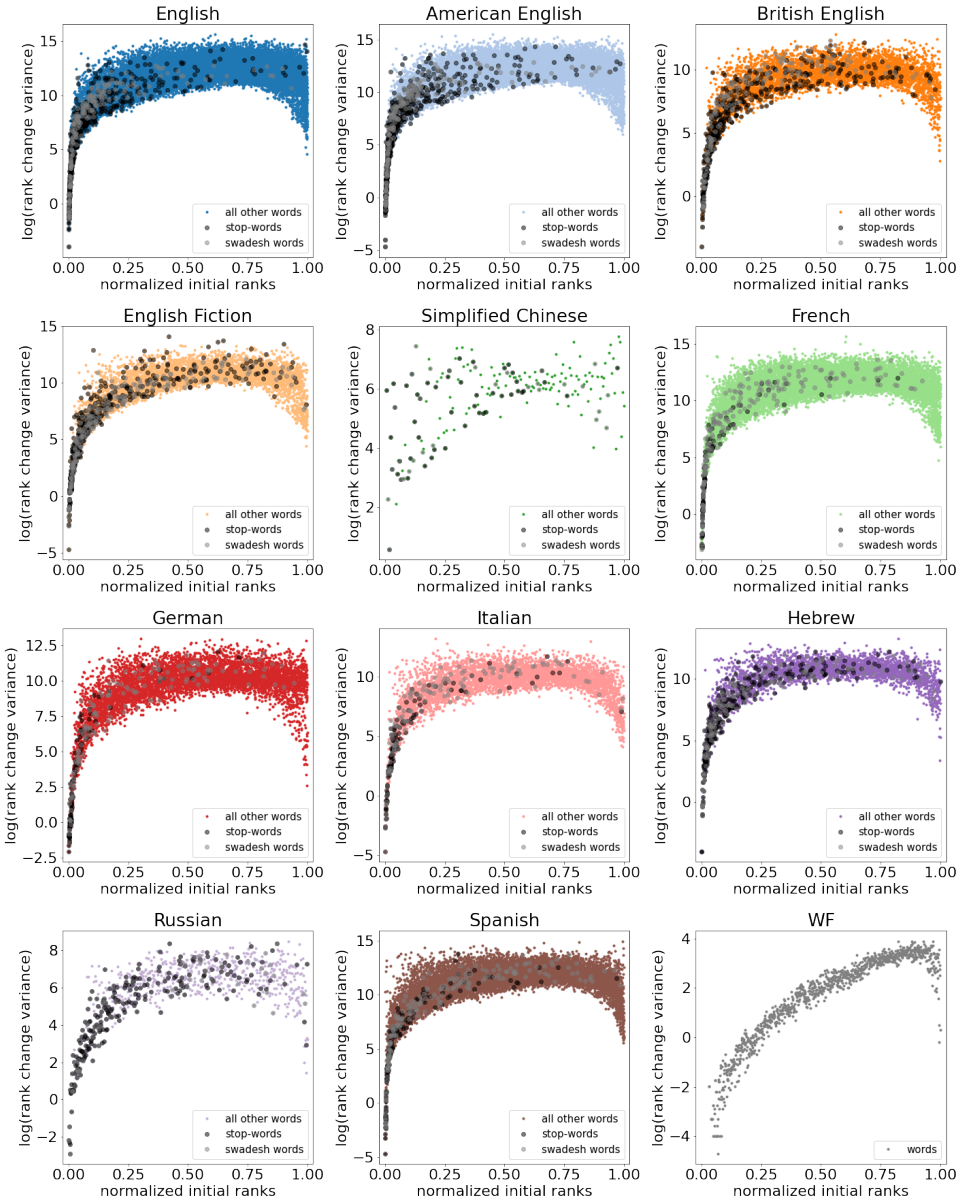


Going back in Fig. 2.13, next we look at the initial ranks versus the rank change variance in Subplot **(b)**. The WF inspired model results on the right Subplot show a

curved pattern where the initially high ranked words have low variance relative to the initially low ranked words. If the corpus size goes to infinity, the curve would go to zero. The English data with the same ratio $c/\beta = 1.00 \times 10^{-5}$ shows some similarities and differences. First, there is still a small curve exhibited by the English data. The high ranked words have mostly low variance while some of the low ranked words have high variance. This is the effect of normalization where the words with very high variances skewed the curve to appear flat compared to the "extreme" case of the WF model. Second, most of the words in English have low variance regardless of where the word is ranked initially. This is similar to the "extreme" case of the WF model where all of of the words have low variances. Finally, the stop words and Swadesh words are in the high ranks at the initial with a low variance which is expected. Since the WF inspired model is the null model for the drift evolutionary process, the comparison suggests that some words in the English data behave contrary to drift. Similar to the results in Subfig **(a)**, most of the words have low-rank change variance which means that most of the words in the English data are stable in ranks.

In Fig. 2.15, it shows the initial ranks versus the rank change variance of the languages we consider in our work. First, each language has this flat curve compared to the WF inspired model. Again, this is due to the normalization process. This curve is consistent for all the languages which mean that most words in the languages are stable regardless of their initial ranks. There are words in the languages with high variance which skewed the curve to appear flat compared to the "extreme" case of the WF model. Second, the stop words and Swadesh words in each language are consistently in the high ranks at the initial step and so have low variance. These groups of words are expected to be stable in meaning and in frequency and it is shown that most of these words have low sums and low variances in rank change. The results of languages clearly showed that some of the words have high variance regardless of initial ranks compared to the WF inspired model. Generally, both the languages and the "extreme" case of the WF model have low variances because the initial corpus size is very large relative to the vocabulary size. The WF inspired model has the ratio $c/\beta = 1.00 \times 10^{-5}$ which is as low as the real languages. However, the differences in the structure of the scatterplots suggest that the languages have words - with very high variance relative to other words - behave contrary to a drift evolutionary process.

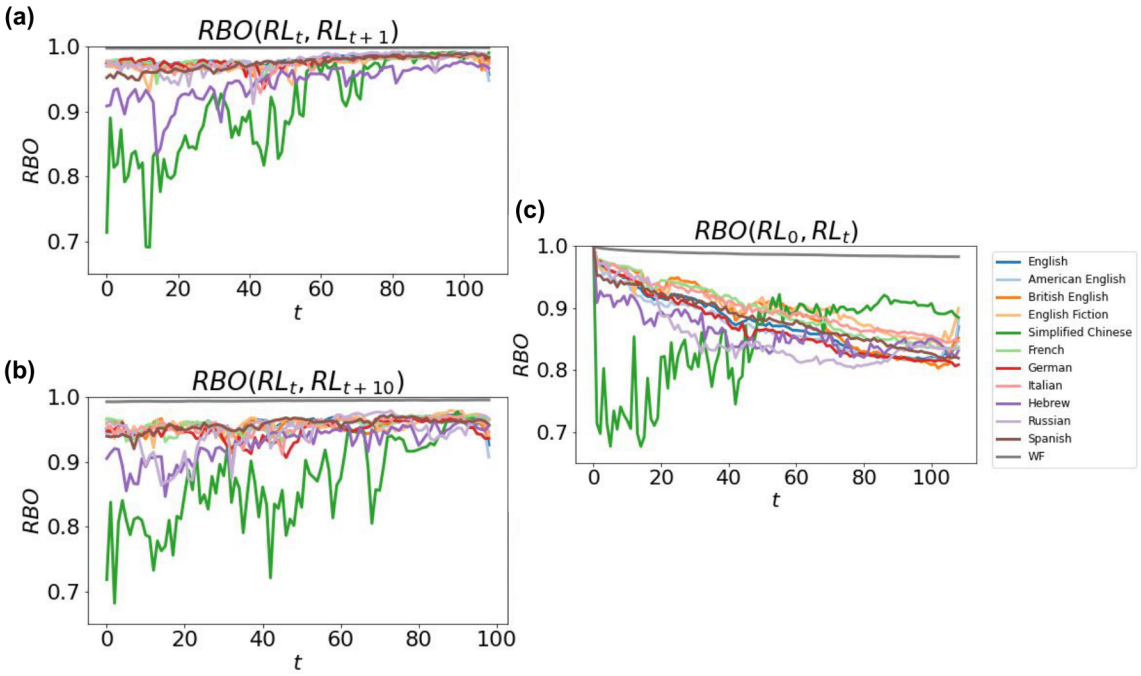
Figure 2.15: **The initial ranks versus the rank change variance of the Language data compared with the extreme case of the WF inspired model.** Each of the subplots below shows the initial rank versus the rank change variance for each language in the Google unigram data with their respective stop words and Swadesh words. First, we observe that there is a subtle curve exhibited by the languages. If look closely at the bottom of each scatterplot, you will see the curve. Second, the curve of the WF inspired model is much more obvious compared to the Language data even though the ratios c/β are extremely low for all the languages and the extreme case of WF inspired model. Finally, the stop words and Swadesh words for each of the languages show consistent behavior where they are the highest-ranked words with low variances.



To further verify that the languages are different from the WF inspired model, we again look to the results of the rank biased overlap (RBO) curves. RBO computes the

similarity between two ranked lists which is defined in Eq. 2.24. Fig. 2.16 show the RBO curves of the languages compared to the "extreme" case of the WF inspired model. Subfig **(a)** are RBO curves where the RL_t and RL_{t+1} is compared and in Subfig **(b)** is where RL_t and RL_{t+10} is compared. We see in the results that the WF inspired model has an RBO curve that is visually constant close to 1 which means that the ranks of words did not change that much. In comparison, the RBO curves of the languages are much lower which means that words are changing in ranks significantly although the "extreme" case of the WF model and the languages have a small ratio c/β down to 10^{-5} in magnitude. Back in Fig. 2.11, we showed that as long as the ratio is the same while the vocabulary size and corpus size is different, the RBO curves should be consistent. This is contrary to what we see in the RBO curves of the languages compared to the "extreme" case of the WF inspired model as shown in Fig. 2.16. We also observed that the RBO curves appear to be leveling off to the value of $RBO = 1$. This is due to the corpus size to be increasing in time at the rate of α. Recall that as the corpus size increases the chances of words to change ranks decreases. In Subfig **(c)** in Fig. 2.16, it shows RBO curves where the RL_0 and RL_{t+1}. In other words, this computation is comparing the initial ranks and the ranks at time t. The results show that as t increases the RBO is decreasing. It means that the ranks are gradually changing in time. Our observations of the RBO curves tell us that the ranks of the overall structure of the languages are variable. A small accumulation of rank changes leads to the overall ranks to change significantly from the initial ranks and we observed in the results that the RBO curves appear to be leveling off to an unknown RBO value.

Figure 2.16: **The RBO curves of the Language data with the extreme case of the WF inspired model.** Subfig **(a)** shows the RBO curves computed by taking the *RBO* metric of the ranked list *RL* at time t versus the ranked list at $t + 1$. Subfig **(b)** shows the RBO curves computed by taking the *RBO* metric of the ranked list *RL* at time t versus the ranked list at $t + 10$. For both Subfigs **(a)** and **(b)**, the RBO curves for the Languages are lower than the extreme case of the WF inspired model which means that the Languages have words changed in ranks more often than the WF model. Subfig **(c)** shows the RBO curve computed by taking the *RBO* metric of the ranked list at the initial time versus the ranked list at time t. This also shows that the Languages have words change in ranks much more extremely than the WF inspired model. It is expected for the extreme case of the WF inspired model to behave like the Languages since we set the ratio c/β to be close enough like the ratios of the languages. However, we observed a different result which means that some words in the Languages behave differently from a drift evolutionary process.



2.2.3 Conclusion

We conclude that a word's rank change shows one of two types of possible characteristics in these data: (1) the increase or decrease in rank is monotonic, or (2) the net rank stays the same. High-ranked words tend to be more stable while low-ranked words tend to be more volatile. Among those words that change rank, some change in two ways: (a) by accumulation of small increasing/decreasing rank changes in time and (b) by sudden shocks of increase/decrease in ranks. Most of the stopwords and Swadesh words are observed to be stable in ranks across eight languages (this is not meant to imply that these groups have the same meaning). In general, the WF model captures some but not all of these trends, as the sudden change that some words are given to depart from the neutral

WF model.

In our analysis, we investigated a large set of unigram frequency data from the Google unigram corpus in eight unique languages in 109 years (1900-2008). We presented a minimal model that simulates unigram frequency evolution while looking at the rank changes in time. We presented the mathematical framework of the Wright-Fisher (WF) inspired model that explains why word rank change. The results using the Wright-Fisher inspired model are consistent with those shown by Sindi and Dale [72] where they concluded that many words exhibit deviations from neutrality. We extended this study by considering multiple languages in the dataset and incorporating different parameter values. The model we presented was inspired by the neutral theory of molecular evolution by Kimura [42] and the Wright-Fisher model [25]. In our model, there are four parameters that we can adjust to observe different outcomes of the model. The Zipf shape parameter a governs the initial frequency distribution with c vocabulary size. The distribution at time $t + 1$ is then sampled using the proportions from t with increased corpus size. The corpus size is governed by an exponential function with β as the initial corpus size and α as the corpus size rate of increase. The model simulates unigram frequency evolution dominated by drift evolutionary process which is a process dominated by frequency-dependent sampling. We demonstrated in our model that smaller corpus size gives stronger drift effects which are consistent with the neutral theory of molecular evolution.

As explained in the results, the binomial curves of words show overlaps between words. These overlaps are stronger when there is a low initial corpus size and weaker when there is a high initial corpus size. The overlaps are the reason why there is a chance that any two words change ranks. Overlaps happen more frequently for low ranking words. Because of the sampling errors in the WFWright-Fisher model, these overlaps in the binomials get stronger and the sampling errors accumulate in time. Therefore, the unigram ranks for low initially ranked words have higher chances to change ranks than the high initially ranked words. If the ratio c/β gets closer to 0 - or β is significantly larger than c - the probability of any two words to change ranks decreases. If c/β gets closer to 1 - or c gets as large as β - the probability of any two words to change ranks increases. The real languages have extreme β and with very low c/β. The WF model would predict that the rank changes in the real languages would be low and the rank change variance would be low if a word is initially high ranked. However, the results show that the languages behave contrary to what the WF model predicts using extreme parameter values of β. The deviations from neutrality that we conclude are consistent with results of previous studies [61, 77, 58, 60, 72, 13]. Similar behavior of stable and fluctuating word frequencies of languages was also observed by past studies [44, 63]. Our work provides a mathematical framework from biological evolution to explain natural language word rank behaviors. In sum, the lexicons of all languages behave as if they are projected in a much smaller corpus – they are given to considerably more fluctuation than their massive corpus would predict. The corpus size is such as that the WF model predicts extreme stability. Instead, languages may be adapting to fluctuations in cultural and other environmental features that drive rank-order changes.

The Ranked Biased Overlap (RBO) metric - which measures the similarity of two ranked lists - showed predictable curves. By adjusting the value of the initial corpus size β and the vocabulary size c, the RBO curves of the WF inspired model can be predicted and have minimal variance. Higher β shifts the curve upwards (small or no change in ranks) while higher c shifts the curve downwards (more change in the ranks). The RBO curves of the languages showed similar curves but with more variance. This is because the WF model with extreme values of β would let the words have less likely to change ranks in time. The languages have some words changed in ranks significantly after the fact that they have extreme values of β.

The unigram frequency and ranks showed deviations from drift evolutionary process. Due to their functionality, stop words and Swadesh words might be the words that are "fixed" to maintain stability in a language system. Many words have low-rank changes similar to stop words and Swadesh words. We also have seen words that have changed up or down in ranks significantly for all the languages we considered. Since language is tied to culture, these words are probably "selected" to serve an important function in cultural change. Did these words change in ranks entirely by chance or is it the result of natural selection? It is difficult to say with certainty that the words in the data that behave in this particular way are naturally selected. Testing for selection requires more than just comparing the data to a null model that simulated drift behavior. However, we are certain that the unigram frequency of words and their ranks does not just behave like the drift evolutionary process but also show peculiar behaviors unexplained by drift. The deviations from neutrality that we conclude are consistent with results of previous studies [77, 60, 72]. Similar behavior of stable and fluctuating word frequencies of languages was also observed by past studies [44, 63]. Our work provides a simple mathematical framework of the Wright-Fisher inspired model to explain natural language word rank behaviors.

2.2.4 Future Work

First, the Wright-Fisher inspired model can be modified further to incorporate selective variation and binning time into different time-scales. This modification could test the model whether behaviors in the real languages are caused by external forces such as the environment. The diversification of language may be the product of environmental factors [11]. A recent study suggest that care should be exercised when binning text data into different time-scales because it may introduce errors on interpreting the results when testing for selection [39].

Second, the Google Books Corpus is not the only large scale historical linguistic corpora. It would be a good idea to see other datasets such as the Corpus of Historical American English (COHA) [4] and the Standardized Project Gutenburg Corpus (SPGC) [29].

Third, linguistic datasets with temporal features taken from online social media platforms such as Twitter, Facebook, or Reddit are possible datasets to consider when studying language evolution in shorter time-scales. These sources of datasets can reveal

interesting patterns of word ranks since these potential datasets are a representative sample of 21st-century human cultural phenomena around the world. The mathematical framework of the model we present could be used to analyze the language change of social movements, popular culture, and political discourse.

Fourth, the words in the Google Ngram Data were annotated with Part-of-Speech (POS) tags but the frequencies of these tags are combined. It would interesting to see the differences in rank changes of these POS word groups.

Finally, the next step to studying word rank evolution is to consider the n-grams for $n > 1$ to see if the n-grams rank changes behave similarly to the Wright-Fisher inspired model.

2.3 A Data-Driven Approach to Word Rank Evolution

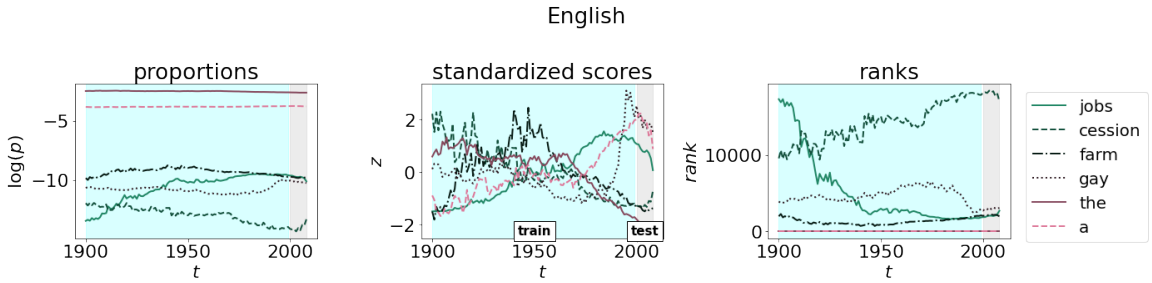
The evolution of language provides insight into cultural shifts through the increase and decrease of word usage across time. Recent availability of large diachronic linguistic data sets allows for rich analysis of language evolution to include broad cultural population size [31] and more computational approaches to features-of-interest within the data [27]. As linguistic data sets increase in size and complexity, advances in language evolution will require the development of increasingly sophisticated and scalable computational approaches to aid in extracting latent features from such data.

Within language, emotion is a well-known cultural experience that has been linked to public opinion [14], political polling [59], predicting economic behaviors [15], and consumer behavior [66]. NLP often looks at emotion through a broad approach, defined as sentiment analysis (also known as opinion mining), which describes emotion along a single valence scale such as “good”-“bad”. Sentiment analysis is commonly based on human curated word-lists that have been assigned to emotional categories where the frequency of a set of emotionally tagged words indicates the emotional valence of a data set. While the sentiment of some words can change drastically over time (e.g. terms such as “gay” or “sick”), the sentiment of most words evolves slowly. Sentiment has been analyzed via both categorical and continuous valence scales but comparison between data sets of different sentiment types [21] and prediction of sentiment change are still hard tasks. Access to large data sets of textual social media content has enabled sentiment analysis to rapidly become one of the largest areas of research in NLP today [81, 50, 51].

In this section, we apply techniques from numerical linear algebra to the study of evolution of word frequencies in the English language. We perform comparative analysis between the words by using sets of vocabulary words. These group of words are stopwords and Swadesh words. We also use different emotional categories of words. More specifically, we use the annotated word emotion lexicon from Mohammad and Turney 2013 [55] to evaluate the ability of the techniques from numerical linear algebra to characterize, reconstruct, and predict the time-series associated with positive, negative and “neutral” words. We present the mathematical algorithms associated with Principal Component Analysis (PCA) and the Dynamic Mode Decomposition (DMD). Both methods are linear models that approximates the best linear fit of given data.

The data for this analysis is the standardized scores - or z-scores - of the unigram time-series data (see Eq. 2.3) for the English language. The times-series data is separated into train and test sets. The time-regime 1900-1999 is used as a training set for PCA and DMD. The time-regime 2000-2008 is used as a test set for future predictions using DMD. An example of the data set separation into train and test set is shown in Fig. 2.17.

Figure 2.17: **English unigram time-series with annotated time regimes for train and test sets for PCA and DMD analysis.** These are time-series of words “jobs”, “cession”, “farm”, “gay”, “the”, and “a” which is the same as Fig. 2.1. The standardized scores of the unigram time-series data is used for PCA and DMD analysis. The time-regime 1900-1999 (highlighted in cyan) is used as a training set for PCA and DMD. The time-regime 2000-2008 (highlighted in grey) is used as a test set for future predictions using DMD.



Background Work

Principal Component Analysis (PCA) is a well known statistical method for dimensionally reducing high-dimensional data for the purpose of visualization and modeling [37, 1]. The method of SVD for computing PCA has been used in the field of computational linguistics since Deerwester et. al. [19]. While PCA has a history of use in the study of linguistic data [24], it is not able to generate predictions outside the period of study. In contrast, DMD is capable of both reconstruction and prediction.

The method of DMD has been applied to high-dimensional fluid dynamical systems. It was first presented and demonstrated by Peter J. Schmid [67, 69, 68] to extract dynamical information of fluid flows from data. The data used into DMD can be from a numerical solution or actual data points taken from a physical experiment. The method has been improved by promoting sparsity to compensate for under-sampled data or lack of data [38]. A randomized DMD was also developed to compensate for problems from over abundant data or “big” data. [23] The method is popular in fluids research [46, 76, 69] but it is also been applied to time-series data related to infectious diseases [65], pattern recognition in videos [48, 32], modeling blood flows in cardiac cycles [33], analysis of brain electrical activities [73, 16], and modeling power systems [3, 8]. DMD was also applied into detecting background and foreground regions of static images [71].

2.3.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique by projecting data points onto vector subspaces called principal components [37]. The significant variations within the data are preserved while noise can be rejected. We explain the process of computing the principal components via Singular Value Decomposition (SVD) below.

Consider a data matrix X of size $M \times N$. For this project, the M is the number of words and the N is the number of years. Applying SVD on X gives us

$$\underset{M \times N}{\begin{bmatrix} | & | & \\ x_0 & x_1 & \cdots \\ | & | & \end{bmatrix}} = \underset{M \times M}{\begin{bmatrix} | & | & \\ u_0 & u_1 & \cdots \\ | & | & \end{bmatrix}} \underset{M \times N}{\begin{bmatrix} \sigma_0 & 0 & \cdots \\ 0 & \sigma_1 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}} \underset{N \times N}{\begin{bmatrix} - & v_0 & - \\ - & v_1 & - \\ & \vdots & \end{bmatrix}} \tag{2.26}$$

where $\sigma_0 \geq \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_N$ are the singular values and the matrices U and V are unitary matrices containing the singular vectors. The $*$ denotes the conjugate transpose where it means that the columns of U and V are orthonormal such that $U^*U = I$ and $V^*V = I$ respectively. See Trefethen et. al. [75] for the mathematical details of SVD computation. SVD computation can be easily implemented using the NumPy module in Python programming language [36].

Assume that X is centered, then the $N \times N$ covariance matrix is given by

$$C = \frac{X^T X}{M - 1}. \tag{2.27}$$

This follows that

$$C = \frac{V\Sigma U^* U\Sigma V^*}{M - 1} = V\frac{\Sigma^2}{M - 1}V^* = VDV^*. \tag{2.28}$$

where D is a diagonal matrix of eigenvalues and V is a matrix containing the eigenvectors. This shows that the covariance matrix C is diagonalizable and the dominant singular value σ_0 with corresponding eigenvector v_0 explains the most variance in the data. Therefore, the rows of V^* are the principal directions and the columns of $XV = U\Sigma$ are the principal components. The individual explained variance can be computed as a ratio, written as

$$\frac{\sigma_i^2}{\Sigma_{i=1}^N \sigma_i^2}, \tag{2.29}$$

where σ_i is the i-th singular value.

To reduce dimensionality, we choose the first r singular vectors of U and the upper left $r \times r$ matrix of Σ. Therefore, the r principal components are given by

$$\underset{M \times r}{\begin{bmatrix} | & | & \\ pc_1 & pc_2 & \cdots \\ | & | & \end{bmatrix}} = \underset{M \times r}{\begin{bmatrix} | & | & \\ u_0 & u_1 & \cdots \\ | & | & \end{bmatrix}} \underset{r \times r}{\begin{bmatrix} \sigma_1 & 0 & \cdots \\ 0 & \sigma_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}} \tag{2.30}$$

where the column vectors $pc_1, pc_2, \cdots, pc_r$ are the principal components.

The number of PC can be chosen arbitrarily but typically 2 PC are used if the explained variance is high enough. We can also choose r according to the desired percentage variance ratio;

$$\rho_r = \frac{\Sigma_{i=1}^{r} \sigma_i^2}{\Sigma_{i=1}^{N} \sigma_i^2}, \tag{2.31}$$

we choose r when ρ_r reaches a particular percentage of choice.

PCA Reconstructions

We can reconstruct the data matrix X using the PC_r vectors and the reduced unitary vector V^*. The reconstructed X is given by

$$\begin{array}{ccccc} \widehat{X} & & = & PC_r & V^* \\ \begin{bmatrix} | & | & \\ \widehat{x_0} & \widehat{x_1} & \cdots \\ | & | & \end{bmatrix} & & = & \begin{bmatrix} | & | & \\ pc_1 & pc_2 & \cdots \\ | & | & \end{bmatrix} & \begin{bmatrix} - & v_0 & - \\ - & v_1 & - \\ & \vdots & \end{bmatrix}. \\ M \times N & & & M \times r & r \times N \end{array} \tag{2.32}$$

2.3.2 Dynamic Mode Decomposition (DMD)

Dynamic Mode Decomposition (DMD) is a data-driven, equation free method to model dynamical systems from high-dimensional data [47]. Dynamical systems are often modeled using a system of ordinary differential equation

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}, t; \theta) \tag{2.33}$$

where $\mathbf{x}(t) \in \mathbb{R}^m$ is the state of the system at time t in a form of a vector of size m, θ is the parameters of the model, and $f(\cdot)$ represents the dynamics. Real dynamical systems are often non-linear and so $f(\cdot)$ is a nonlinear function and $\mathbf{x}$ has dimension $m \gg 1$.

The DMD method approximates the nonlinear system as a locally linear dynamical system

$$\frac{d\mathbf{x}}{dt} = \mathbb{A}\mathbf{x} \tag{2.34}$$

with initial condition $\mathbf{x}(0)$, and the matrix $\mathbb{A}$ is the coefficients. This is a homogeneous, first-order system of differential equations. The well known solution to this is

$$\mathbf{x}(t) = \Phi e^{(\Omega t)} \mathbf{b} \tag{2.35}$$

where the columns of Φ are the eigenvectors of $\mathbb{A}$ and Ω is the eigenvalues of $\mathbb{A}$. The term $\mathbf{b}$ are from $\mathbf{x}(\mathbf{0})$ defined as

$$\mathbf{b} = \Phi^{\dagger} \mathbf{x}(0). \tag{2.36}$$

where the superscript † is the moore-penrose pseudoinverse.

Given a real complex system, we typically get discrete samples for every Δt so that we can represent the discrete time-series data as

$$\mathbf{x}_{n+1} \approx \mathbf{A}\mathbf{x}_n \tag{2.37}$$

where $\mathbf{x}_n$ are samples collected at time t_n for $n = \{0, 1, 2, \cdots, N-1\}$, N is the total time, and $\mathbf{A}$ is the linear operator matrix. We can define the initial conditions as $\mathbf{x}(t_0) = \mathbf{x}_0$.

Let X be an $M \times N$ data matrix where we denote $\mathbf{x}_n$ as the nth column. Recall that M is the number of words and the N is the number of years. We define $M \gg N$. DMD requires two sets of data,

$$X = \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{N-1} \\ | & | & & | \end{bmatrix} \quad \text{and} \quad X' = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \\ | & | & & | \end{bmatrix} \tag{2.38}$$

where $k = N - 1$. The goal of the DMD is to find the best fit linear operator matrix $\mathbf{A}$ so that

$$X' \approx \mathbf{A}X. \tag{2.39}$$

The DMD solution we present here is

$$\mathbf{A} = X'X^{\dagger} \tag{2.40}$$

where we can compute a low-rank approximation of $\mathbf{A}$ such that we can minimize the error

$$\|X' - \mathbf{A}X\|_F. \tag{2.41}$$

In a case where $M \gg N$, the resulting matrix A is a large $M \times M$ matrix. We find the DMD solution via SVD which is detailed by Kutz et. al. [47] and Tu et. al. [76]. Below is the steps (Alg. 1) for computing the standard DMD solution where the matrix operator A is reduced to a low rank approximation called $\widetilde{\mathbf{A}}$ of size $r \times r$ where r is the rank of X.

Algorithm 1 Standard DMD Algorithm [47] (For the SVD and eigendecomposition methods, see Harris et. al. [36] for an easy implementation and See Trefethen et.al. [75] for mathematical details)

- **Step 1:** Organize the data matrix into X and X' (see Eq. 2.38). The size of the data matrix is $M \times N$ where $M \gg N$.
- **Step 2:** Compute the SVD of X:

  $$X = U\Sigma V^*$$

  where $*$ denotes the conjugate transpose, U is $N \times r$, Σ is a diagonal matrix of size $r \times r$, V is $N \times r$, and r is the rank of X.
- **Step 3:** Compute the reduced $r \times r$ matrix operator $\widetilde{\mathbf{A}} \in \mathbb{R}^{r\times r}$, writing

  $$\widetilde{\mathbf{A}} = U^* X' V \Sigma^{-1}. \tag{2.42}$$

- **Step 4:** Compute the eigendecomposition of $\widetilde{\mathbf{A}}$, writing

  $$\widetilde{\mathbf{A}} W = W\Lambda \tag{2.43}$$

  where the diagonal matrix Λ contains the complex eigenvalues corresponding to the eigenvector columns of W.
- **Step 5:** Compute the dynamic modes $\widehat{\mathbf{\Phi}} \in \mathbb{C}^{M\times r}$, writing

  $$\widehat{\mathbf{\Phi}} = X' V \Sigma^{-1} W. \tag{2.44}$$

The eigendecomposition of the full $M \times M$ matrix operator $\mathbf{A}$ is given by

$$\mathbf{A}\Phi = \Phi\Omega \tag{2.45}$$

where Ω is a diagonal matrix with complex eigenvalues corresponding to the eigenvector columns in $\Phi \in \mathbb{C}^{M\times r}$. However, since $\mathbf{A} \in \mathbb{R}^{M\times M}$ is a large matrix computing the eigendecomposition is expensive. The eigenvalues and eigenvectors of $\widetilde{\mathbf{A}}$ in Eq. 2.42 is equal to the nonzero eigenvalues and eigenvectors of $\mathbf{A}$. The proof is in page 6 of Tu et. al. [76]). Since we are not interested in the zero eigenvalues, these are already rejected when computing $\widetilde{\mathbf{A}}$. The matrix Φ is called the dynamic modes while $\widehat{\Phi}$ in Eq. 2.44 is the low-rank dynamic modes.

For a case when N is significantly large, the Eq. 2.42 in Algorithm 1 can be reduced further by defining r according to a defined percentage variance (see Eq. 2.31).

DMD Reconstructions and Predictions

Given $\widehat{\Phi}$ in Eq. 2.44, we can reconstruct the original data and approximate a solution at all future times.

First, we compute the continuous time frequencies:

$$\omega_i = \frac{\ln(\lambda_i)}{\Delta t} \tag{2.46}$$

where λ_i is the nonzero eigenvalues of $\widetilde{\mathbf{A}}$ and Δt is the change in time. This approximates the solution in Eq. 2.35 at all future times given by

$$\hat{\mathbf{x}}(t) \approx \widehat{\Phi} e^{\Lambda t} \mathbf{b} \tag{2.47}$$

where Λ is a diagonal matrix in Eq. 2.43.

Next, we compute the initial coefficient values b_i in the vector $\mathbf{b}$:

$$\hat{\mathbf{b}} = \widehat{\Phi}^{\dagger} \mathbf{x}_0 \tag{2.48}$$

where $\hat{\mathbf{b}}$ is the solution of the linear matrix equation $\mathbf{x}_0 = \widehat{\Phi}\mathbf{b}$. This can be solved by a standard linear least-squares method.

Cosine Similarity

The relationships between words can be measured using the cosine of their vectors. These word vectors can be the principal components or the dynamic modes trained on a given data. The cosine similarity metric is a distance metric on how two word vectors are similar in pattern or somehow correlated with each other. Given two words w and v, the cosine similarity between their word vectors is given by the following:

$$\text{cos-sim}_{v,w} = \frac{\langle \vec{\mathbf{w}}, \vec{\mathbf{v}} \rangle}{\|\vec{\mathbf{w}}\| \|\vec{\mathbf{v}}\|} = \frac{\sum_{i=1}^{N} \vec{\mathbf{w}}_i \vec{\mathbf{v}}_i}{\|\vec{\mathbf{w}}\| \|\vec{\mathbf{v}}\|}, \tag{2.49}$$

where $\vec{\mathbf{w}}$ and $\vec{\mathbf{v}}$ are the word vectors of length N for words w and v respectively and $\|\cdot\|$ is the vector 2-norm. That is,

$$\|\vec{\mathbf{w}}\| = \left(\sum_{i=1}^{N} \vec{\mathbf{w}}_i^2 \right)^{\frac{1}{2}}. \tag{2.50}$$

The range of the cosine similarity is $[-1, 1]$ where a value close to 1 means that the word vector for w is similar to the word vector of v. A value close to -1 means that the two vectors are distant in cosine.

2.3.3 Results

The results are divided into four parts. Part 1 is the PCA results of the Google Unigram time-series data showing the structure and interpretation of the principal components. Part 2 is the DMD results detailing the structure and interpretation of the dynamic modes. Part 3 is about a comparative analysis of the time-series patterns extracted from PCA and DMD. Part 4 is about the residual analysis of the time-series reconstructions generated by the PCA and DMD.

Part 1 of 4: PCA Results

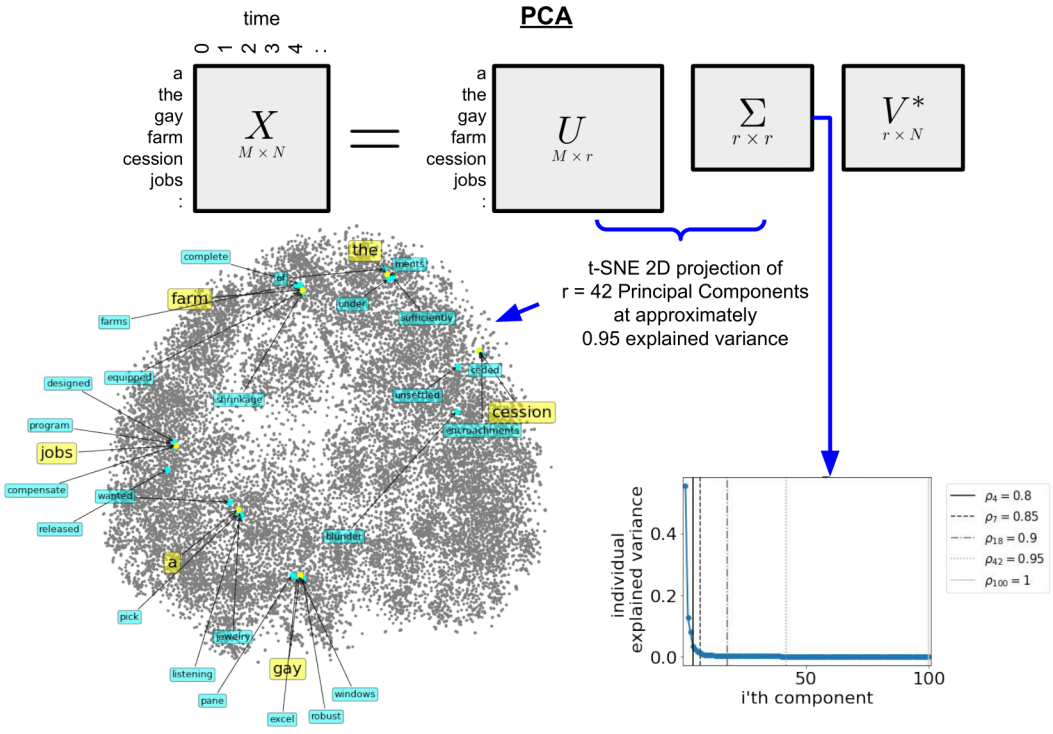
Results summary. Recall that PCA is a dimensional reduction technique utilizing SVD to extract the dominant features of the data. We apply PCA onto the unigram z-score time-series data (See Eq.2.3). There are two main types of information we can extract from the given data using PCA:

- The relationships between the time-series can be extracted and projected onto a smaller dimensional space.
- There are some cases of words that share semantically similar meanings and have similar time-series patterns. However, there are some cases that, although the two words have similar time-series patterns, they don't share semantically similar meanings but rather share a culturally significant meanings.

We explain the details of the results below.

To understand PCA in the context of the unigram time-series data, we need to look at the individual words and how it is projected onto the principal components. We show an illustration of the SVD and a 2D projection of the principal components in Fig. 2.18. In this figure (Fig. 2.18), the proportion time-series data matrix is factorized as written in Eq. 2.26. First, we look at the individual explained variance (Eq. 2.29) at the lower right subplot in Fig. 2.18. For the English dataset, the dominant singular value accounts for approximately 55.4% of the variance while the second singular value accounts for approximately 12.7% of the variance. There is a significant drop in explained variance which means that for only two principal components, that already accounts for a total of 68.1% of the variance. With 4 principal components, that is 80% of the variance. With 42 principal components, that is 95% of the variance.

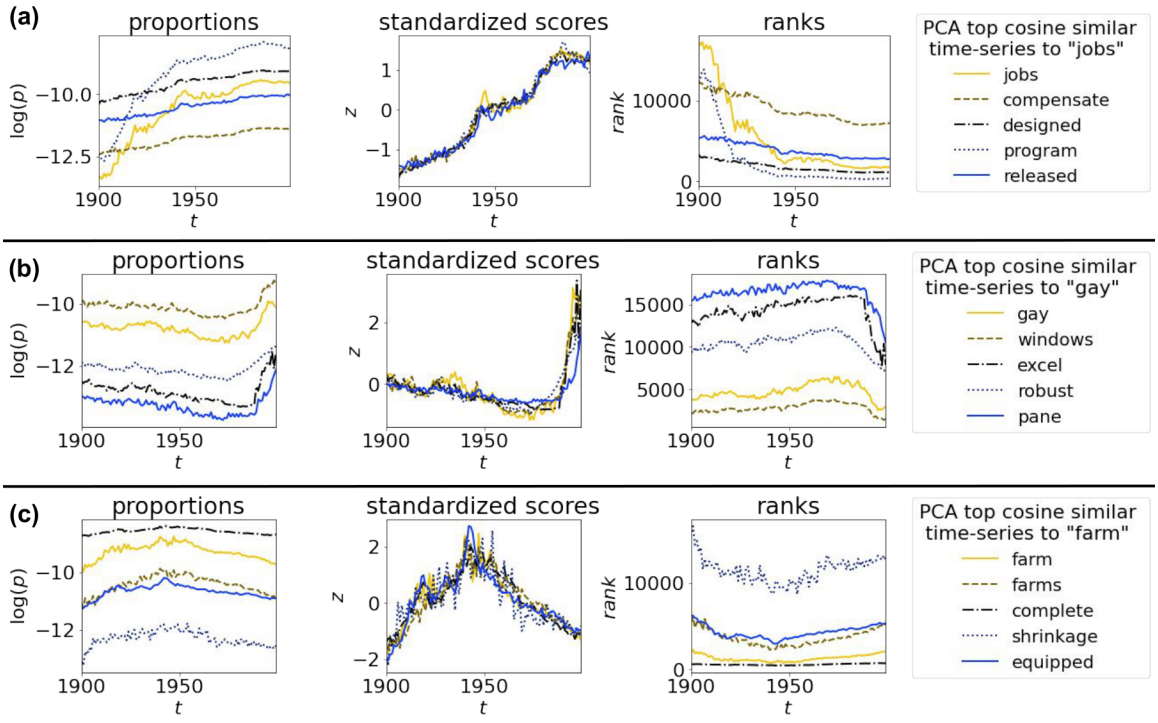
Figure 2.18: **PCA of the English unigram time-series.** This figure includes an illustration of the SVD computation of the z-score time-series of English. The lower right plot is the individual explained variance (Eq. 2.29) plot for each *i*th principapl component. The vertical lines within this plot indicate the number of principal components to get to a percentage variance (Eq. 2.31) as labeled in the legends. The lower left figure is a 2D projection (using t-SNE) of 42 principal components. That means it takes 42 PC to get 95% explained variance. See Maaten et. al. for t-SNE details [78]. The yellow annotated words are the example words shown in Fig. 2.17 and the words in cyan are the top cosine similar words (Eq. 2.49). The dimensionality reduction technique using PCA captures the pattern similarities of the z-score time-series.



We apply Eq. 2.49 to the computed PC shown in Fig. 2.18 of English and show three example words in Fig. 2.19. The top 4 cosine similar words to "jobs" are "compensate", "designed", "program", and "released". The standardized score (or z-scores) time-series of these words have the same increasing pattern as shown in the middle plot in Subfig. **(a)**. The dominant increasing pattern is also present in the proportions time-series shown in the left plot in Subfig. **(a)**. Since these words are increasing, we can see a decreasing rank in the right plot of Subfig. **(a)** which means that the word "jobs" and its cosine similar words changed from higher rank to lower rank. We can also say that the word "compensate" is culturally similar to "jobs" because both words are related in terms of "to give money" in exchange for "work". However, this is not always the case. For subfig. **(b)** in Fig. 2.19, we can see a different z-score time-series pattern for the word "gay" and the cosine similar words to it are "windows", "excel", "robust", and "pane". The word "gay" is not semantically similar to "windows". The words "windows" and "excel" with increasing trends refers to the computer operating system and application due

to the rise of technology. For subfig. **(c)** in Fig. 2.19, we can see that the word “farm” is semantically similar to the word “farms” because it is the plural form of singular “farm”. The cosine similarity measure only captures the similarities of time-series patterns but not the actual cultural and semantic meanings of the words. There are some cases that the cosine similarity of z-scores can capture coherent words together but not all of them.

Figure 2.19: **English unigram time-series of “jobs”, “gay”, and “farm” with their top 4 cosine similar words using there principal component vectors.** Each subfigure below show the unigram time-series data in three forms: the proportions (Eq. 2.2), standardized scores (Eq. 2.3), and ranks (See Table. 2.1). By computing the cosine similarities (Eq. 2.49) of the three example words from all other words, what is shown in each subfigure are the words with the closest cosine distance for each word.



Part 2 of 4: DMD Results

Results summary. Recall that the DMD is a data-driven approach to model a dynamical system. Although the DMD method is a linear model used on a non-linear system, the number of parameters is large enough to model the non-linear behaviors of the system. We applied DMD on the unigram time-series data and there are three main results that we found:

- Similar to PCA, the relationships between the time-series can be projected onto a smaller dimensional space.
- The DMD method describes dynamic behaviors based on the complex eigenvalues

of the matrix operator $\widetilde{\mathbf{A}}$ (Eq. 2.42). The non-zero imaginary part of an eigenvalue describes a group of times-series with sinusoidal behaviors while the eigenvalues with zero imaginary part describes a group of increasing/decreasing time-series.

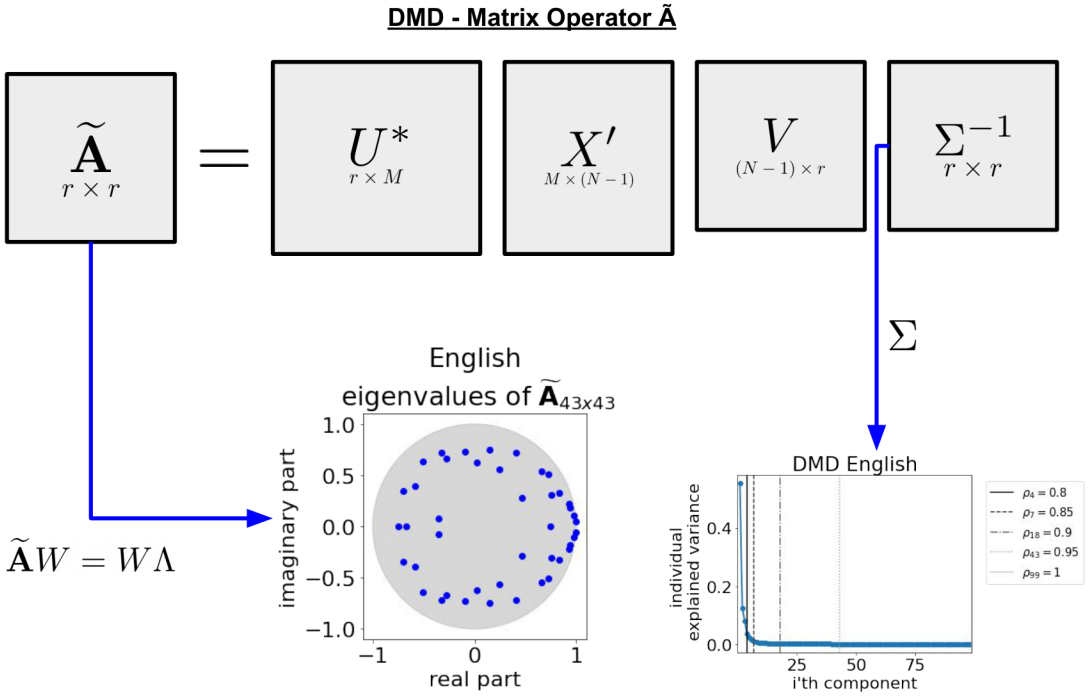
- The dynamic modes Φ (Eq. 2.44) of the DMD describes the collective behaviors of the times-series and it can extract pattern similarities in time-series.

We explained the details of the results below.

DMD takes in data with the assumption that the data is from a dynamical system. Natural Languages are thought of as a complex system and evolves dynamically with human evolution. The unigram time-series is assumed to be a non-linear dynamical system with underlying system of equations that governs its behavior. The size and complexity of the unigram time-series data is too large for explicitly defining a system of ordinary differential equation and solve it numerically. The DMD method is a data-driven approach to extract the dynamic behavior of the unigram time-series data without the need of a system of equations. In contrast, PCA is a more simplistic method than DMD because PCA is a more straightforward approach to describing the dominant linear time-series patterns. However, DMD is more complicated because it can describe and model the collective dynamic behavior of the linear and non-linear time-series of words.

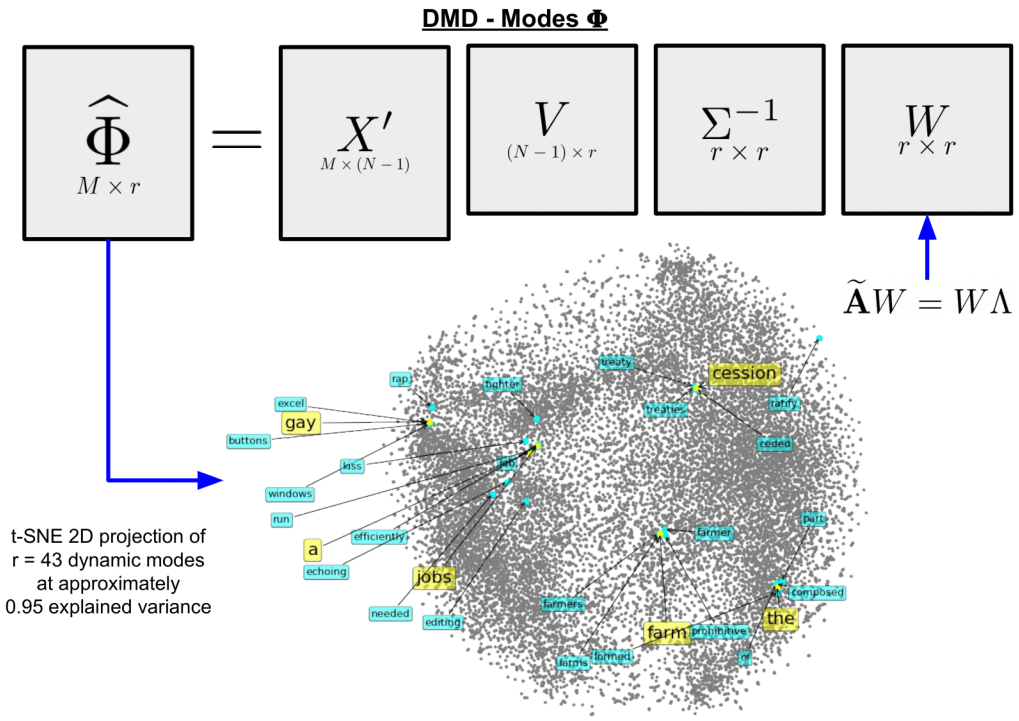
As explained in Alg. 1, the low-rank matrix operator $\widetilde{\mathbf{A}}$ is constructed by computing the SVD of the data matrix X (see Eq.2.38). We first look at the low-rank matrix operator $\widetilde{\mathbf{A}}$ in Fig. 2.20. For a matrix size of 43×43, this means that 43 components are used in the SVD that explains 95% of the variance. For a dynamic system, the eigenvalues tells the story of the collective behavior of the time-series of words. For example, in Fig. 2.20, the eigenvalues of $\widetilde{\mathbf{A}}$ of size 43×43 is complex and the modulus of each eigenvalues is less than 1 which means that the long term behavior of the model is stable. The eigenvalues close to the real number line (with zero complex part) describes a group of words with linear behaviors in their time-series. The eigenvalues far away from the real line (with non zero complex part) describes a group of words with sinusoidal behavior. To be clear, the eigenvalues shown in Fig. 2.20, does not describe individual words but rather a group of words. Each eigenvalue is associated with the dynamic modes that describes the driven behaviors of the unigram time-series of words.

Figure 2.20: **The matrix operator $\widetilde{\mathbf{A}}$ of DMD in the context of the English data.** This diagram shows the computation of the low-rank matrix operator $\widetilde{\mathbf{A}}$ using the SVD of X (see Eq. 2.42). Using the English data the complex eigenvalues of $\widetilde{\mathbf{A}}$ is shown. The size of $\widetilde{\mathbf{A}}$ is 43 × 43 which is using the 43 components of the SVD that explains 95% of the variance.



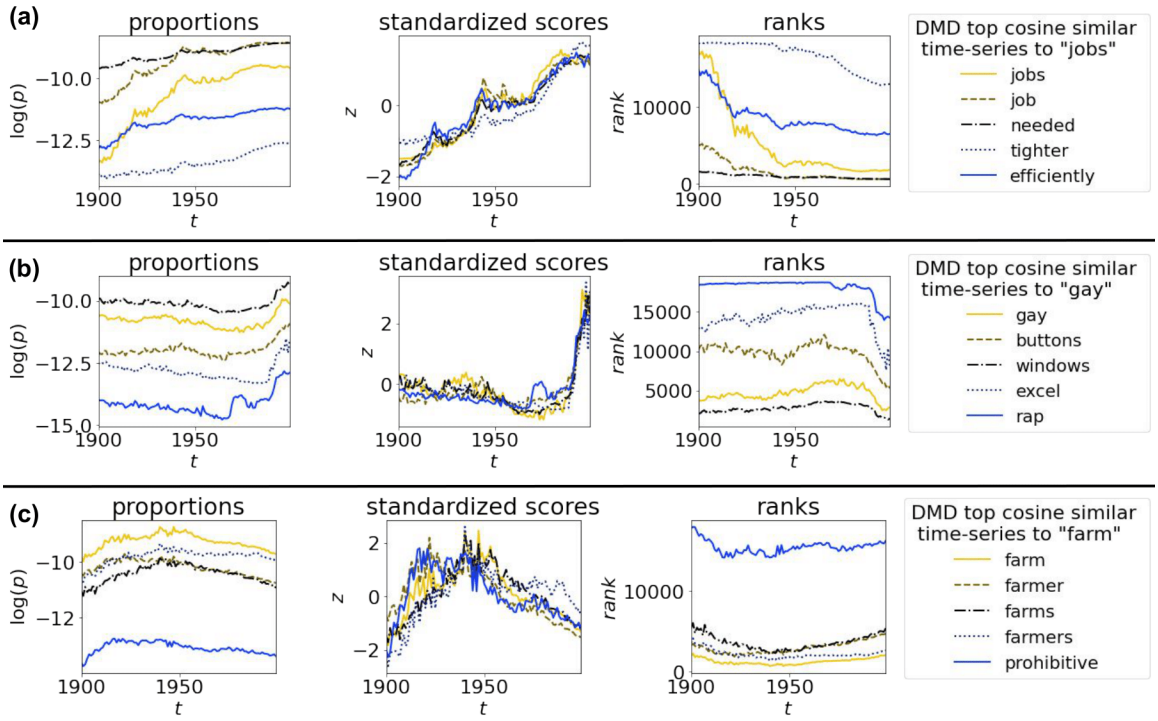
DMD computes the dynamic modes $\mathbf{\Phi}$ as described in Eq. 2.44 that is associated with the growth/decay rates of the linear time-series of words in the matrix operator $\widetilde{\mathbf{A}}$, as well as the oscillation frequencies of non-linear behaviors of words. The dynamic modes also can extract the relationships of words similar to the PCA. In Fig. 2.21, the 2D projection of the dynamic modes of the English data indicate that words with similar time-series patterns are clustered together. For example, the word “cession” is cosine similar to “ceded”, “treaty”, “treaties”, and “ratify”. The word “cession” itself means a formal act of assigning a territory or property to another entity which is related to the word “treaty” which is an agreement to said formal acts. In comparison, the PCA results only includes the word “ceded”. Another example is the word “jobs” which is cosine similar to “job”, “needed”, “tighter”, and “efficiently”. Unlike PCA, the word “jobs” (plural) is similar to “job” (singular) as opposed to “compensate” in PCA. The most exciting result is the word “farm” which is cosine similar to “farmer”, “farms”, “farmers”, and “prohibitive” because compared to PCA the word “farm” was similar only “farms” and some other unrelated words.

Figure 2.21: **The real-part dynamic modes $\widehat{\Phi}$ of the English data.** This diagram shows the computation of the dynamic modes (Eq. 2.44) using the SVD of X and the eigenvalues of the low-rank matrix operator $\widetilde{\mathbf{A}}$. As an example, the number of modes is 43 because that is the number of components of the SVD that explains 95% of the variance. The bottom figure is a 2D projection (using t-SNE) of 43 dynamic modes. See Maaten et. al. for t-SNE details [78]. The yellow annotated words are the example words shown in Fig. 2.17 and the words in cyan are the top cosine similar words (Eq. 2.49). The DMD method captures the pattern similarities of the z-score time-series similar to PCA.



Next, we look at the time-series patterns of the example words “jobs”, “gay”, and “farm” in Fig. 2.22. Similar to the results using PCA, the DMD captures the similarities in the standardized score (or z-score) time-series of words. For example, the word “jobs” in subfig. **(a)** in . 2.22 shows a dominant increasing pattern and the cosine similar words “job”, “needed”, “tighter”, and “efficiently” also shows similar increasing pattern. The ranks of these words changed from high rank to low rank except for the word “needed” and “tighter” which only has minimal rank changes. Compared to PCA in Fig. 2.19, the cosine similar words using DMD in Fig. 2.22 showed some differences. For example, the word “gay” is cosine similar to “buttons”, “windows”, “excel”, and “rap” in DMD as opposed to “gay” is cosine similar to “windows”, “excel”, “robust”, and “pane” in PCA. Both PCA and DMD captured the similarities of time-series patterns of these words.

Figure 2.22: **English unigram time-series of "jobs", "gay", and "farm" with their top 4 cosine similar words using there dynamic mode vectors.** Each subfigure below show the unigram time-series data in three forms: the proportions (Eq. 2.2), standardized scores (Eq. 2.3), and ranks (See Table. 2.1). By computing the cosine similarities (Eq. 2.49) of the three example words from all other words, what is shown in each subfigure are the words with the closest cosine distance for each word.



Part 3 of 4: Pattern Analysis

Results summary. The PCA and DMD methods are both utilizing dimensionality reduction technique using SVD to extract the most dominant features of the data. The SVD technique allows these methods to project the data onto smaller dimensional spaces called principal components in PCA and the dynamic modes in DMD. We probe both of these matrix structures by comparing them side by side and we have found three results:

- The DMD dynamic modes (See Eq. 2.44)) provided a more temporally meaningful interpretation of the time-series trends of the English data than PCA. For example, stopwords and swadesh words are spatially close together within the DMD 2D projection than in PCA.
- There are two main trends of the z-score time-series which are increasing and decreasing patterns. There are more decreasing trends than increasing trends.
- Although more words are shown to have decreasing trends, there are relatively more positive words among increasing trends and there are more negative words among decreasing trends.

The details of our findings are shown below.

The z-score time-series exhibits three kinds of basic patterns: decreasing, increasing, or neither. We have seen these kinds of patterns by inspection from Fig. 2.19 and Fig. 2.22. Though not obvious, we also have seen two large regions in the 2D projections of PCA and DMD in Fig. 2.18 and Fig. 2.21 respectively. These two regions corresponds to increasing and decreasing time-series patterns. We show this by performing a separate statistical test for monotonic time-series trends.

We are interested in detecting a time-series data whether it is a monotonically increasing or decreasing trend. We begin by defining a null and alternative hypothesis.

$$H_0 : \text{The time-series is non-monotonic.}$$

$$H_a : \text{The time-series is monotonic.}$$

We initially assume the null hypothesis to be true and we wanted to produce a p-value using a statistical method to decide if we want to reject the null hypothesis and conclude monotonicity. We use the Mann-Kendall (MK) test for monotonicity for both exploration and labeling.

The MK test is a non-parametric test to asses if there is an upward or downward monotonic trend in a given time-series. This method assumes that the observed time-series is not correlated over time (i.e. non-monotonic). The method is implemented by Schramm [70] using Python programming language which is verified [20]. The Author's code is based on the original publications about the method by Mann [53] and Kendall [40]. Below is the summary of the method.

First, we need to compute the sum of the positive and negative differences which is given by

$$S = \sum_{k-1}^{N-1} \sum_{j-k+1}^{N} sgn(x_j - x_k) \tag{2.51}$$

where x is the given time-series data and N is the number of observations. Second, we compute the variance of S which is given by

$$\sigma(S) = \frac{1}{18}\left[N(N-1)(2N+5) - \sum_{p-1}^{g} t_p(t_p-1)(2t_p+5)\right]. \tag{2.52}$$

The second term of the above equation is when there are "tied" groups. A "tied" group is when there are two more identical observations in the data. For example, there are $g = 2$ tied groups in the sequence $\{2, 2, 3, 3, 3, 1, 4\}$ because there are $t_1 = 2$ for the value 2 and $t_2 = 3$ for the value 3. Third, we compute the test statistic which was labeled as z for "z-score" but we relabeled this as MK to distinguish it from other symbols. This is computed by

$$MK = \begin{cases} \frac{S-1}{\sqrt{\sigma(S)}} & S > 0 \\ 0 & S = 0 \\ \frac{S+1}{\sqrt{\sigma(S)}} & S < 0 \end{cases}. \tag{2.53}$$

We use a two tail test to compute the p-value;

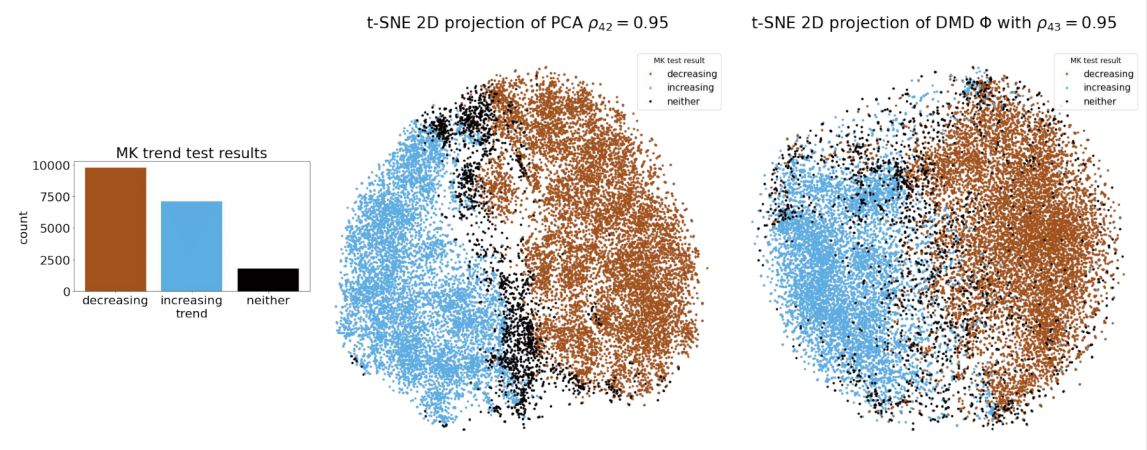
$$\text{p-value} = 2(1 - F_{MK}(0, 1)), \tag{2.54}$$

where $F_{MK}(0, 1)$ is the cumulative density function of the standard normal distribution. Given that the p-value is less than some significance value, the MK test predicts that the time-series trend is increasing if $MK > 0$ or decreasing if $MK < 0$. The MK test only detects monotonic trends, not trends with seasonality behaviors.

We set a typical significance value of 0.05. It means that any p-value less than 0.05 is a deciding factor that a time-series is considered monotonic. In other words, we can reject the null hypothesis that the time-series is non-monotonic and there is enough evidence to suggest that the time-series has a monotonic trend. If a time-series is considered monotonic, the sign of the test statistic determines whether the trend is increasing ($+MK$) or decreasing ($-MK$). We apply the MK method to label each z-score time-series individually in the English data. We include stopwords lexicon, the swadesh words lexicon, and the sentiment lexicon in our analysis to perform a comparative analysis between word groups.

Fig. 2.23 shows that there are more decreasing z-score time-series patterns in the English data. The middle and right subplots of Fig. 2.23, we can see and confirm that the two regions we observed in the 2D projections of PCA and DMD corresponds to the increasing and decreasing patterns. For the PCA projection, the “neither” pattern appears to be clearly in between the two regions of increasing and decreasing trends. In comparison, the DMD projection for the “neither” pattern appears to be scattered. Recall that PCA is a statistical technique that extracts the most dominant features of the data which is projected onto orthogonal vectors known as principal components. In this case, the most dominant features within the z-score time-series data is the increasing and decreasing trends. In comparison, the DMD method is a modeling technique that utilizes dimensionality reduction to characterize temporal behaviors as opposed to PCA which does not. Essentially, the DMD method gives us a more temporally meaningful interpretation of the dominant time-series patterns of the English data. The MK trend test results indicate that there are more decreasing words than increasing words.

Figure 2.23: **MK test for monotonic trends results for the English data.** The subfigures below shows the distribution of the time-series trends of the English language. It also shows the two regions in the 2D projections of the PCA and DMD shown in Fig. 2.18 and Fig. 2.21 respectively which confirms that these regions captures the dominant trends of the z-score time-series.

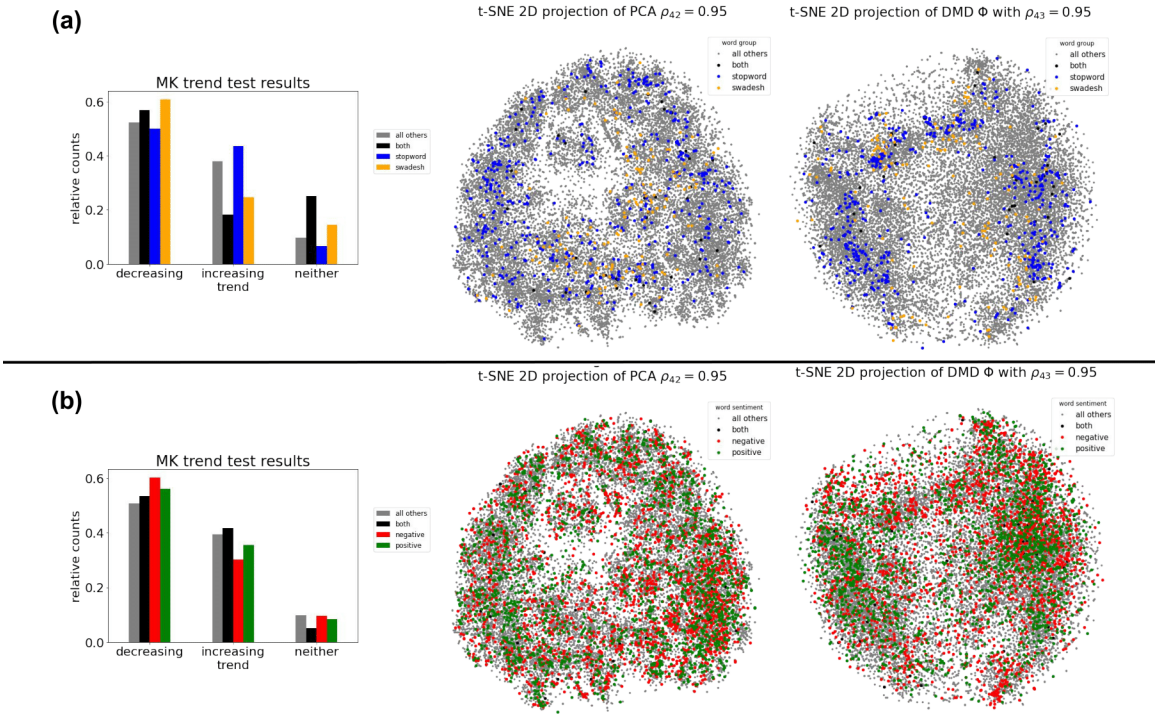


Since the DMD is temporally meaningful than PCA, then the time-series patterns for semantically similar words should be better clustered than in PCA. We can see that the stopwords and swadesh words are more clustered within the large regions of the 2D projection of the DMD shown in the right subplot of Fig. 2.24 Subfig. **(a)**. This means that these group of words have temporally similar structures in there z-score time-series. The MK trend test results shows us that there are 60.75% decreasing trends among swadesh words while there are only 49.90% decreasing trends among stopwords. In comparson, there are 24.60% increasing trends among swadesh words and there are 43.45% increasing trends among stopwords. This means that among stopwords, there is almost an equal balance of increasing and decreasing words compared to swadesh words which is dominated by decreasing patterns.

Stopwords and swadesh words are words with the simplest meanings but some swadesh words can have complicated and multiple meanings. For example, the word "black" and "white" are both swadesh words associated with color but in American culture, these words are used associated with racialized groups. The results show that the swadesh words in the DMD projections are more spread out than stopwords. Positive and negative sentiment words can be difficult to characterize because of its volatile nature. For example, the word "sick" is naturally a negative sentiment associated with the feelings of illness but - until recently - the word changed into a positive sentiment closely associated with the word "cool" or "very good." Without context, the word "sick" can be ambiguous to decipher when it is used in a sentence such as "that's so sick." Because of that ambiguity in usage of positive and negative sentiment words - also we are limited into just using unigrams, both PCA and DMD are having difficulty clustering these two word groups together as shown in Fig. 2.24 Subfig. **(b)**. The results of MK trend test indicate that there are 60.2% decreasing trends among negative words while there are 56.06% decreasing trends among positive words. In comparison, there are 30.19% increasing

trends among negative words while there are 35.53% increasing trends among positive words. This means that - although more words are shown to have decreasing trends - there are more positive words among increasing trends and there are more negative words among decreasing trends. Technically, words categorized as both positive and negative sentiment are shown to have the highest percentage among increasing trend. This could mean that these words are often used because of it's ambiguity and flexibility when used.

Figure 2.24: **MK test for monotonic trends results for the English data with labeled stopwords and swadesh words and also the words associated with positive and negative sentiments.** Subfig. **(a)** is the MK trend test results with labeled stopwords and swadesh words. The left figure is the relative counts of each word group. Relative counts means that, the sum of the yellow/blue/black/gray bars is one. For example, there are 60.75% decreasing trends among swadesh words compared to 49.90% decreasing trends among stopwords. Similar interpretation can be said in Subfig. **(b)**. The middle and right figures of each subplot is the 2D projection for PCA and DMD with labeled word groups and sentiments.



Part 4 of 4: Residual Analysis

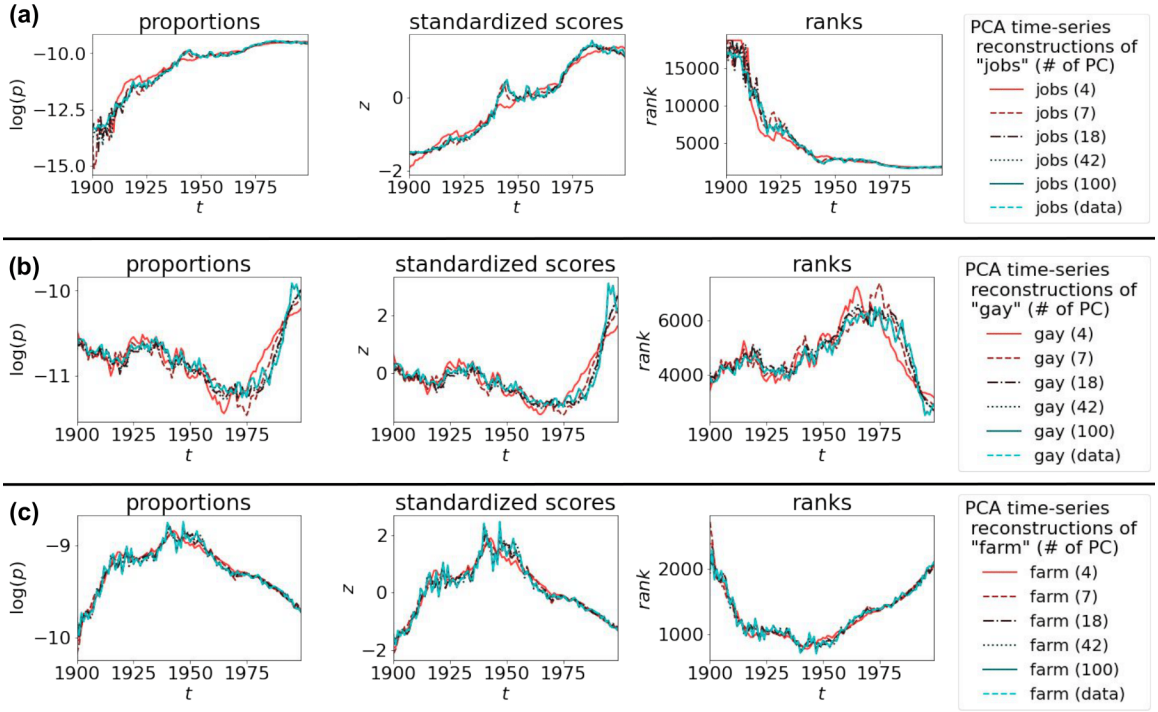
Results summary. PCA and DMD have different methods on reconstructing the time-series. PCA reconstructs the English time-series data by a simple matrix operation using the principal components. DMD reconstructs the English time-series data by using the low-rank matrix $\widetilde{\mathbf{A}}$, the dynamic modes $\widehat{\mathbf{\Phi}}$, and an initial condition. we apply the PCA and DMD and reconstructed the English time-series data and found four results.

- PCA reconstructed the time-series more accurately than DMD.

- The DMD produces a more smoothed time-series reconstruction curves than PCA.
- Stopwords are more accurately reconstructed than swadesh words and the positive and negative sentiment words.
- DMD is having a hard time predicting future time-series values.

The PCA time-series reconstructions are done through a simple matrix computation written in Eq. 2.32. This allows a quick computation for each time-series. Depending on the chosen rank r the PCA reconstructions depends on the amount of variance captured. For example, the PCA reconstruction in Fig. 2.25 Subfig. **(a)** is showing that as the number of principal components are used the reconstructions get closer to the actual data. This means that the larger the r, more noise are captured within the data. Choosing $r = 42$ with 95% variance captured rejects the 5% noise within the data. As shown, the reconstruction of the time-series of "jobs" with $r = 42$ closely follows the actual data. Similarly with the word "farm" in Subfig. **(c)**, the PCA time-series reconstruction almost follows the actual data.

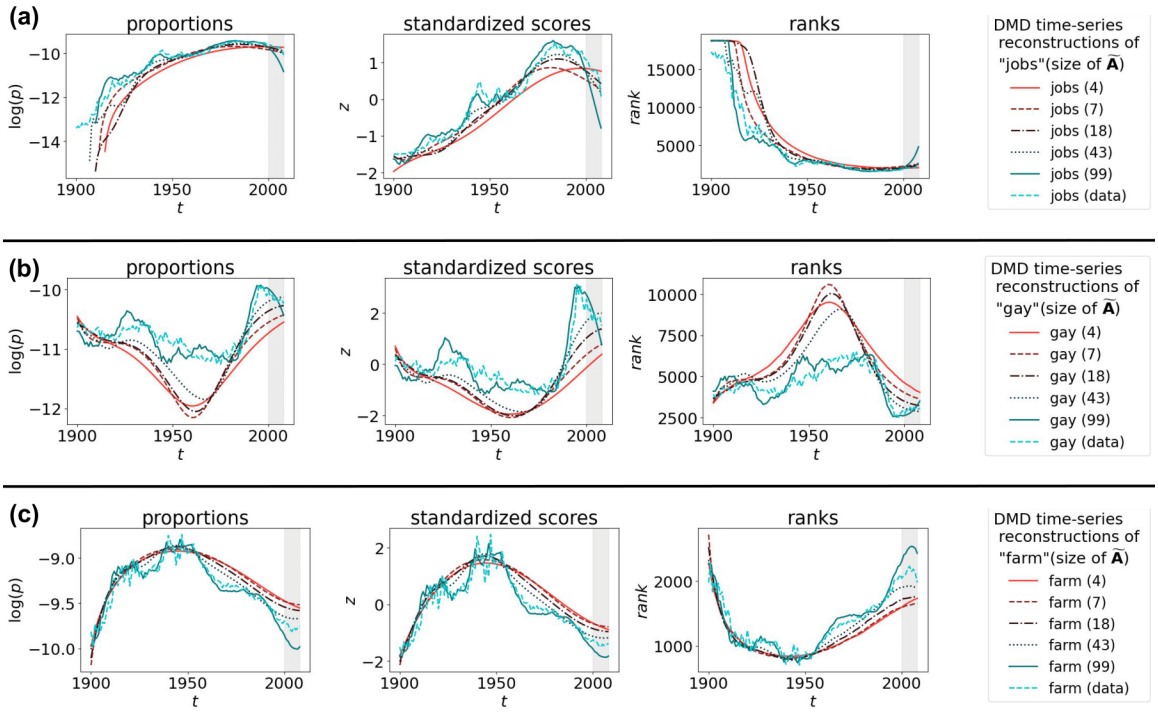
Figure 2.25: **PCA reconstructions of three example words "jobs", "gay", and "farm" within 1900-1999.** The legends of each subfigure shows a different number of principal components used to reconstruct the z-score time-series. For example, "jobs (42)" means that the word "jobs" is reconstructed using 42 principal components. Recall that the PCA is trained using the standardarized scores (or z-scores) within the time frame 1900-1999. The time-series is then reconstructed using Eq. 2.32. The proportions are reconstructed by using the data mean and variance, and the ranks are reconstructed using the proportions. The results show that as more principal components are used, the reconstructions gets closer to the real data. Less principal components are used means that less noise is captured.



Recall that PCA is a dimensional reduction technique that captured the most important variance in the data and the reconstruction steps of the data is simple and direct. This allows the reconstructions to be quite good as shown in Fig. 2.25. In comparison, the DMD method has more extra steps on reconstructing the time-series data. Reconstructing the time-series using DMD is done using Eq. 2.47. This equation uses the reduced rank matrix $\widetilde{\mathbf{A}}$ and the dynamic modes $\widehat{\mathbf{\Phi}}$ which are computed using the data via SVD. The main difference between PCA and DMD is that DMD accounts for the temporal features of the data as opposed to PCA which only accounts for the variance and the correlations between time-series. DMD is temporally meaningful than PCA because DMD is a data-driven approach to modeling dynamical systems inspired by ordinary differential equations and linear algebra. As discussed in the previous results, the eigenvalues of $\widetilde{\mathbf{A}}$ is associated with the growth and decay rates of the collective behavior of the times-series as well as the oscillation behaviors if it exists. The dynamic modes also extracts the similarities between time-series.

We show the DMD time-series reconstruction and predictions in Fig. 2.26. The results show that the DMD reconstructions produce smoother curves than using PCA. In Subfig. **(a)**, we can see that the DMD reconstruction for the time-series of "jobs" follows a smooth curve along with the actual data if the size of $\widetilde{\mathbf{A}}$ is 4. It also means that for $r = 4$, there are 4 columns in $\widehat{\mathbf{\Phi}}$ are used. It also show that the curve is slightly underestimated. For $r = 43$, the curve is still smooth and it follows the data more closely. For $r = 99$, the DMD reconstruction starts to very closely follow that noise of the data. Similar behavior can be seen in Subfig. **(c)**. In Subfig. **(b)**, the DMD reconstructions for the time-series of "gay" shows a different behavior where there the DMD curve does not closely follow the data. It does follow the general curve of the data but not exactly.

Figure 2.26: **DMD time-series reconstructions of three example words "jobs", "gay", and "farm" within 1900-1999.** The legends of each subfigure shows a different sizes of the low-rank matrix operator $\widetilde{\mathbf{A}}$ used to reconstruct the z-score time-series. For example, "jobs (43)" means that the word "jobs" is reconstructed using 43 $\times$ 43 matrix $\widetilde{\mathbf{A}}$ and the dynamic modes with 43 columns. Recall that the DMD is trained using the standardized scores (or z-scores) within the time frame 1900-1999. The time-series is then reconstructed using Eq. 2.47 including the future times 2000-2008 (shaded in gray). The proportions are reconstructed by using the data mean and variance, and the ranks are reconstructed using the proportions. The results show that as more dynamic modes are used, the reconstructions gets closer to the real data. Less dynamic modes means that the curves of the predictions are averaged and looks smoother than PCA.



These are three example DMD time-series reconstructions shown in Fig. 2.26. We test the accuracy of the PCA and DMD reconstructions by computing th Root Mean Squared Error (RMSE) of each time-series in the data. Recall that the English data has 18737

words and that means there are 18737 time-series reconstructions. As explained in the previous sections, we trained and reconstruct all 18737 time-series using PCA and DMD within the time frame 1900-1999. The RMSE value allows us to evaluate and compare the general reconstructing power of PCA and DMD. The RMSE distributions are shown in Fig. 2.27. The RMSE distributions are grouped according to the labeled trends which are determined using the MK test for monotonicity as explained in the previous sections. The distributions are also grouped by stopword or swadesh words and by positive or negative sentiments. The word groupings allows us to do comparison within the word groups on how well the PCA and DMD reconstructs these types of words.

Shown in Fig. 2.27 Subfig. **(a)** and Subfig. **(b)**, the immediate difference we can see between PCA and DMD is that DMD has a higher RMSE values than PCA for the same captured explained variance of 95%. This corresponds to the difference in reconstruction steps. PCA requires only a matrix operation for computing every value for reconstructions while the DMD reconstruction requires generating temporal computations using an initial condition. In any case, both PCA and DMD reconstructs the English time-series data well enough to produce decent curves. DMD produces a more smoother curves and can produce curves at all future times.

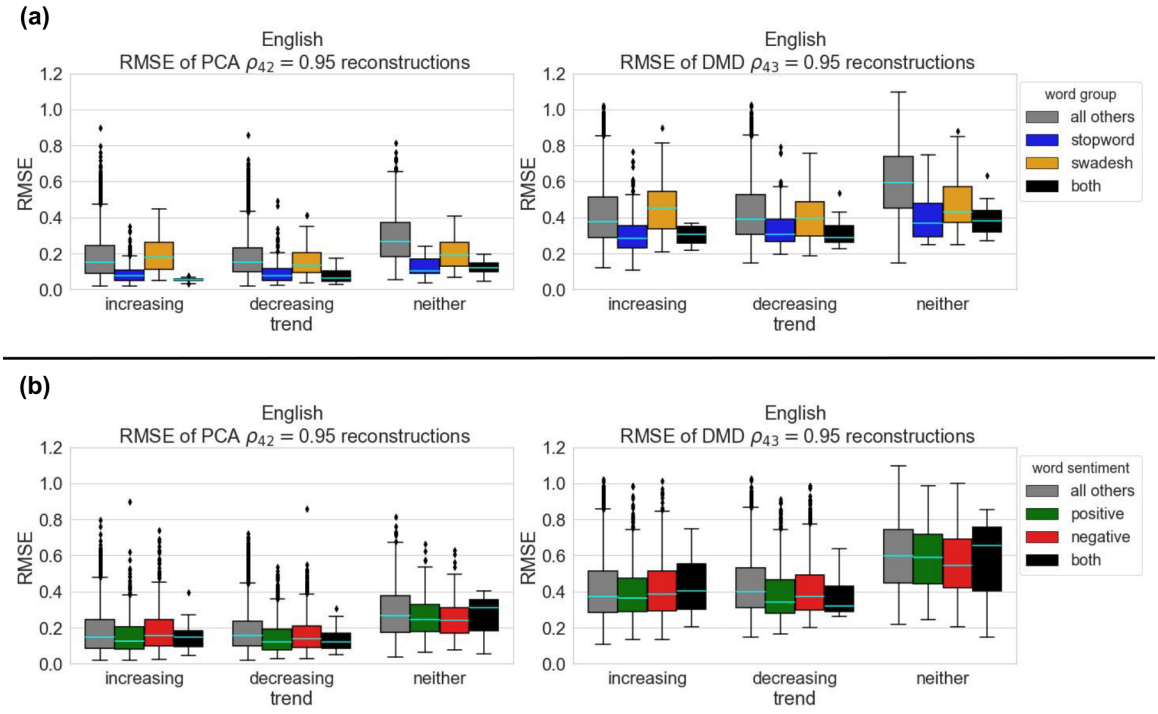
Recall that the z-score time-series was labeled using the MK test for monotonicity. These trend labels are then used to separate the PCA and DMD reconstructions into three groups of increasing, decreasing, and "neither". As we can see in Fig. 2.27 Subfig. **(a)** left and right figure, the "neither" RMSE distributions are higher than its increasing and decreasing counterparts. For example in PCA RMSE results in Subfig. **(a)**, the gray boxplot in the increasing and decreasing trend labels has a RMSE median of less than 0.20 while the gray box plot in the "neither" trend label has a RMSE median above 0.20. Similar can be said in the DMD results in the right figure in Subfig. **(a)** and the figures in Subfig. **(b)** in Fig. 2.27. This is an expected results since PCA and DMD are both linear methods and the "neither" trends are time-series with random patterns or oscillating pattern with no clear increasing or decreasing trends.

The RMSE distribution of the stopword and swadesh words show distinct difference. Shown in Fig. 2.27 Subfig. **(a)**, the blue boxplot - which corresponds to the stopwords - have lower RMSE median than the median of the yellow boxplot - which correspond to the swadesh words. This means that the time-series for the stopwords are well reconstructed using PCA or DMD compared to the time-series of the swadesh words. For the words that are both stopword and swadesh word, the RMSE median is as low as the stopword. Recall that stopwords are words that are the most frequently used words with no particular meaning by itself. Because of the simplicity of stopwords and how it is the most frequently used words, the PCA and DMD is better at reconstructing their time-series. In comparison, the swadesh words have simple meanings but also have some words with ambiguous meanings which contributed to the complexity of their usage in time. For example, the word "black" and "white" are both simple color words but can also be used based on context other than the colors.

The RMSE distribution of the positive and negative sentiment words show no significant difference. Shown in Fig. 2.27 Subfig. **(a)**, the green boxplot - which

corresponds to the positive words - have almost the same RMSE distribution as the red boxplot - which corresponds to the negative words. We see this similarities in increasing, decreasingm and neither trends labels. Compared to the stopwords, the sentiment words have higher RMSE median. This means that the sentiment words are harder to reconstruct due to their complexity in meaning and in word usage.

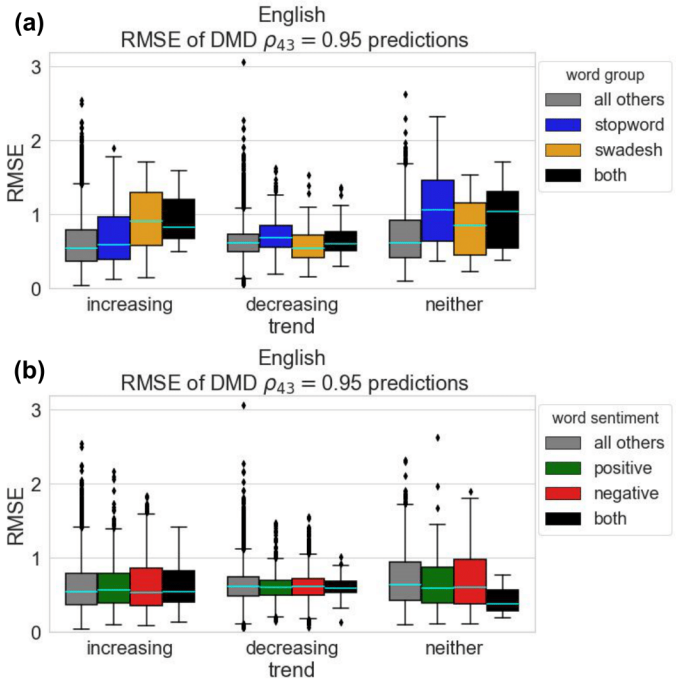
Figure 2.27: **RMSE distributions of the PCA and DMD time-series reconstructions within 1900-1999.** The Root Mean Squared Error (RMSE) is computed for each word time-series reconstructions by PCA and DMD. The left side of Subfig. **(a)** is the RMSE distributions of the PCA reconstructions grouped by trends and by stopword and swadesh word. The right side of Subfig. **(a)** is the RMSE distributions of the DMD reconstructions with the same word groupings. Similarly with Subfig. **(b)** but with the sentiment word groupings. The results show that the DMD RMSE distributions are higher than PCA due to the fact that the reconstruction method is different for DMD and PCA. DMD has a more smoother reconstructions than PCA for the same variance captured. In addition, the stopwords are more accurately reconstructed than the swadesh words while the "neither" trends are less accurately reconstructed than the rest.



Recall that the DMD method can predict the unigram time-series at all future time. We test the accuracy of the future predictions by computing the RMSE values for each time-series reconstruction. Recall that the DMD is trained using the time-series within the time frame 1900-1999 while the time frame 2000-2008 is set aside for evaluating the accuracy of the predictions. That is 8 years for the DMD to predict. We show the RMSE distributions of the DMD predictions in Fig. 2.28. Compared to the RMSE distributions of DMD reconstructions in Fig. 2.27, the RMSE distributions of th DMD predictions is much higher. This is observed for all increasing, decreasing, and "neither" trend

labels and for all word groupings. The results show that the DMD is having a hard time predicting future values of the time-series precisely but it does capture the most important temporal structure of the time-series of words.

Figure 2.28: **RMSE distributions of the DMD predictions at future times 2000-2008.** The Root Mean Squared Error (RMSE) is computed for each word time-series predictions DMD. Recall that the DMD is trained using the standardized scores (or z-scores) within the time frame 1900-1999. The time-series is then reconstructed using Eq. 2.47 for all future times 2000-2008. The RMSE distributions is much higher than the RMSE reconstructions which indicates that the prediction value of DMD using the unigram time-series data is unclear.



2.3.4 Conclusion

In our analysis, we have applied PCA and DMD to the English unigram time-series data. The standardized scores (or z-scores) are used with the time frame 1900-1999 as training data for PCA and DMD. The time frame 2000-2008 is used as test data for the DMD predictions. Both PCA and DMD uses the SVD technique to dimensionaly reduce the data to extract the most important features of the time-series data. From our results, the most dominant time-series patterns are increasing and decreasing trends. These trends are captured by the PCA and DMD. The MK trend test is used separately to label the time-series individually to verify that the two regions we observed in the PCA and DMD projections corresponds to the increasing and decreasing trends. By grouping the words into sentiments, we observed in our results that there are relatively more positive words among increasing trends and there are more negative words among decreasing trends.

With the assumption that the unigram time-series is a dynamic system, DMD provides a more temporally meaningful interpretation of the results than PCA. There is more accuracy in the PCA reconstructions than using DMD. The dynamic modes $\widehat{\mathbf{\Phi}}$ trained on the English data shows that it can capture the time-series similarities among words and - in some cases - it can capture semantically meaningful words. For example, the stopwords are closely clustered together in DMD than compared to PCA. DMD also can describe the collective behavior of the time-series by looking at the eigenvalues of the low-rank matrix operator $\widetilde{\mathbf{A}}$.

2.3.5 Future Work

Although our analysis demonstrates the power of the DMD on linguistic data there is more work to be done. In particular, we note that we may need a more refined time-series to capture temporal variations with the DMD. For example, with Twitter or other more regularly sampled data we may be able to pick up seasonal behaviors particularly in terms of mood that themselves might reflect processes such as the weather, elections or other external drivers of discourse. Another limitation of our work is the use of unigram data. We also only used the real-part of the dynamic modes. It would be interesting to study the imaginary part and how to relates to the language data. Future work using larger sizes of n-gram data will allow DMD to be applied for underlying contextual linguistic artifacts such as semantic meaning and discourse features.

Bibliography

[1] H. Abdi and L. J. Williams. “Principal component analysis”. *Wiley interdisciplinary reviews: computational statistics* 2.4 (2010), pp. 433–459.

[2] A. Acerbi et al. “The expression of emotions in 20th century books”. *PloS one* 8.3 (2013), e59030.

[3] A. Alassaf and L. Fan. “Dynamic mode decomposition in various power system applications”. *2019 North American Power Symposium (NAPS)*. IEEE. 2019, pp. 1–6.

[4] R. Alatrash et al. “Ccoha: Clean corpus of historical american english”. *Proceedings of The 12th Language Resources and Evaluation Conference*. 2020, pp. 6958–6966.

[5] E. A. Armstrong and S. M. Crage. “Movements and memory: The making of the Stonewall myth”. *American Sociological Review* 71.5 (2006), pp. 724–751. DOI: 10.1177/000312240607100502.

[6] J. Baixeries, B. Elvevåg, and R. Ferrer-i-Cancho. “The evolution of the exponent of Zipf’s law in language ontogeny”. *PloS one* 8.3 (2013), e53227.

[7] R. Bamler and S. Mandt. “Dynamic word embeddings”. *International conference on Machine learning*. PMLR. 2017, pp. 380–389.

[8] E. Barocio et al. “A dynamic mode decomposition framework for global power system oscillation analysis”. *IEEE Transactions on Power Systems* 30.6 (2014), pp. 2902–2912.

[9] R. A. Bentley et al. “Word diffusion and climate science”. *PloS one* 7.11 (2012), e47966.

[10] R. A. Bentley et al. “Books Average Previous Decade of Economic Misery”. *PLOS ONE* 9.1 (Jan. 2014), pp. 1–7. DOI: 10.1371/journal.pone.0083147. URL: https://doi.org/10.1371/journal.pone.0083147.

[11] C. Bentz et al. “The evolution of language families is shaped by the environment beyond neutral drift”. *Nature Human Behaviour* 2.11 (2018), pp. 816–821.

[12] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. ” O’Reilly Media, Inc.”, 2009.

[13] R. A. Blythe. “Neutral evolution: a null model for language dynamics”. *Advances in complex systems* 15.03n04 (2012), p. 1150015.

[14] J. Bollen, H. Mao, and A. Pepe. “Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena”. *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 5. 1. 2011.

[15] J. Bollen, H. Mao, and X. Zeng. “Twitter mood predicts the stock market”. *Journal of computational science* 2.1 (2011), pp. 1–8.

[16] B. W. Brunton et al. “Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition”. *Journal of neuroscience methods* 258 (2016), pp. 1–15.

[17] Changyaochen. *Implementation of Rank-biased Overlap*. 2018. URL: `https://github.com/changyaochen/rbo`.

[18] G. Cocho et al. “Rank diversity of languages: generic behavior in computational linguistics”. *PLoS One* 10.4 (2015), e0121898.

[19] S. Deerwester et al. “Indexing by latent semantic analysis”. *Journal of the American society for information science* 41.6 (1990), pp. 391–407.

[20] *Design Trend Mann-Kendall*. `https://vsp.pnnl.gov/help/Vsample/Design_Trend_Mann_Kendall.htm`.

[21] P. S. Dodds and C. M. Danforth. “Measuring the happiness of large-scale written expression: Songs, blogs, and presidents”. *Journal of happiness studies* 11.4 (2010), pp. 441–456.

[22] N. C. Edsall. *Toward Stonewall : homosexuality and society in the modern western world*. University of Virginia Press, 2003, pp. 249–275, 314–333. ISBN: 9780813922119.

[23] N. B. Erichson et al. “Randomized dynamic mode decomposition”. *SIAM Journal on Applied Dynamical Systems* 18.4 (2019), pp. 1867–1891.

[24] N. E. Evangelopoulos. “Latent semantic analysis”. *Wiley Interdisciplinary Reviews: Cognitive Science* 4.6 (2013), pp. 683–692.

[25] W. J. Ewens. *Mathematical population genetics 1: theoretical introduction*. Vol. 27. Springer Science & Business Media, 2012.

[26] F. Fejes. *Gay rights and moral panic : the origins of America’s debate on homosexuality*. Palgrave Macmillan, 2008, pp. 1–9. ISBN: 9781403980694.

[27] A. Gandomi and M. Haider. “Beyond the hype: Big data concepts, methods, and analytics”. *International journal of information management* 35.2 (2015), pp. 137–144.

[28] *gay, adj., adv., and n.* 2008. URL: `http://www.oed.com/viewdictionaryentry/Entry/77207?print` (visited on 09/25/2018).

[29] M. Gerlach and F. Font-Clos. “A standardized Project Gutenberg corpus for statistical analysis of natural language and quantitative linguistics”. *Entropy* 22.1 (2020), p. 126.

[30] Y. Goldberg and J. Orwant. “A Dataset of Syntactic-Ngrams over Time from a Very Large Corpus of English Books”. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*. Atlanta, Georgia, USA: Association for Computational Linguistics, June 2013, pp. 241–247. URL: https://www.aclweb.org/anthology/S13-1035.

[31] P. M. Greenfield. “The changing psychology of culture from 1800 through 2000”. *Psychological science* 24.9 (2013), pp. 1722–1731.

[32] J. Grosek and J. N. Kutz. “Dynamic mode decomposition for real-time background/foreground separation in video”. *arXiv preprint arXiv:1404.7592* (2014).

[33] M. Habibi, S. Dawson, and A. Arzani. “Data-driven pulsatile blood flow physics with dynamic mode decomposition”. *Fluids* 5.3 (2020), p. 111.

[34] W. L. Hamilton, J. Leskovec, and D. Jurafsky. “Diachronic word embeddings reveal statistical laws of semantic change”. *arXiv preprint arXiv:1605.09096* (2016).

[35] H. Hammarström. “Linguistic diversity and language evolution”. *Journal of Language Evolution* 1.1 (2016), pp. 19–29.

[36] C. R. Harris et al. “Array programming with NumPy”. *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[37] I. T. Jolliffe and J. Cadima. “Principal component analysis: a review and recent developments”. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (2016), p. 20150202.

[38] M. R. Jovanović, P. J. Schmid, and J. W. Nichols. “Sparsity-promoting dynamic mode decomposition”. *Physics of Fluids* 26.2 (2014), p. 024103.

[39] A. Karjus et al. “Challenges in detecting evolutionary forces in language change using diachronic corpora”. *arXiv preprint arXiv:1811.01275* (2018).

[40] M. Kendall. “Rank correlation methods. 2nd impression”. *Charles Griffin and Company Ltd. London and High Wycombe* (1975).

[41] Y. Kim et al. “Temporal analysis of language through neural language models”. *arXiv preprint arXiv:1405.3515* (2014).

[42] M. Kimura. *The neutral theory of molecular evolution*. Cambridge University Press, 1983.

[43] A. Koplenig. “The impact of lacking metadata for the measurement of cultural and linguistic change using the Google Ngram data sets - Reconstructing the composition of the German corpus in times of WWII”. *Digit. Scholarsh. Humanit.* 32 (2017), pp. 169–188.

[44] A. Koplenig. “Using the parameters of the Zipf–Mandelbrot law to measure diachronic lexical, syntactical and stylistic changes – a large-scale corpus analysis”. *Corpus Linguistics and Linguistic Theory* 14 (2018), pp. 1–34.

[45] V. Kulkarni et al. “Statistically significant detection of linguistic change”. *Proceedings of the 24th International Conference on World Wide Web*. 2015, pp. 625–635.

[46] J. N. Kutz, X. Fu, and S. L. Brunton. “Multiresolution dynamic mode decomposition”. *SIAM Journal on Applied Dynamical Systems* 15.2 (2016), pp. 713–735.

[47] J. N. Kutz et al. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[48] J. N. Kutz, J. Grosek, and S. L. Brunton. “Dynamic mode decomposition for robust pca with applications to foreground/background subtraction in video streams and multi-resolution analysis”. *CRC Handbook on Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing* (2016).

[49] Y. Lin et al. “Syntactic Annotations for the Google Books NGram Corpus”. *Proceedings of the ACL 2012 System Demonstrations*. Jeju Island, Korea: Association for Computational Linguistics, July 2012, pp. 169–174. URL: https://www.aclweb.org/anthology/P12-3029.

[50] B. Liu. “Sentiment analysis and opinion mining”. *Synthesis lectures on human language technologies* 5.1 (2012), pp. 1–167.

[51] A. Maas et al. “Learning word vectors for sentiment analysis”. *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. 2011, pp. 142–150.

[52] D. Y. Manin. “Mandelbrot’s Model for Zipf’s Law: Can Mandelbrot’s Model Explain Zipf’s Law for Language?” *Journal of Quantitative Linguistics* 16.3 (2009), pp. 274–285.

[53] H. B. Mann. “Nonparametric tests against trend”. *Econometrica: Journal of the econometric society* (1945), pp. 245–259.

[54] J.-B. Michel et al. “Quantitative analysis of culture using millions of digitized books”. *science* 331.6014 (2011), pp. 176–182.

[55] S. M. Mohammad and P. D. Turney. “Crowdsourcing a word–emotion association lexicon”. *Computational intelligence* 29.3 (2013), pp. 436–465.

[56] M. A. Montemurro and D. H. Zanette. “Coherent oscillations in word-use data from 1700 to 2008”. *Palgrave Communications* 2.1 (2016), pp. 1–9.

[57] J. A. Morales et al. “Rank dynamics of word usage at multiple scales”. *Frontiers in Physics* 6 (2018), p. 45.

[58] M. G. Newberry et al. “Detecting evolutionary forces in language change”. *Nature* 551.7679 (2017), pp. 223–226.

[59] B. O'Connor et al. "From tweets to polls: Linking text sentiment to public opinion time series". *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 4. 1. 2010.

[60] J. P. O'Dwyer and A. Kandler. "Inferring processes of cultural transmission: the critical role of rare variants in distinguishing neutrality from novelty biases". *Philosophical Transactions of the Royal Society B: Biological Sciences* 372.1735 (2017), p. 20160426.

[61] M. Pagel et al. "Dominant words rise to the top by positive frequency-dependent selection". *Proceedings of the National Academy of Sciences* 116.15 (2019), pp. 7397–7402.

[62] E. A. Pechenick, C. M. Danforth, and P. S. Dodds. "Characterizing the Google Books corpus: Strong limits to inferences of socio-cultural and linguistic evolution". *PloS one* 10.10 (2015), e0137041.

[63] A. M. Petersen et al. "Languages cool as they expand: Allometric scaling and the decreasing need for new words". *Scientific reports* 2.1 (2012), pp. 1–10.

[64] S. T. Piantadosi. "Zipf's word frequency law in natural language: A critical review and future directions". *Psychonomic bulletin & review* 21.5 (2014), pp. 1112–1130.

[65] J. L. Proctor and P. A. Eckhoff. "Discovering dynamic patterns from infectious disease data using dynamic mode decomposition". *International health* 7.2 (2015), pp. 139–145.

[66] M. Salehan and D. J. Kim. "Predicting the performance of online consumer reviews: A sentiment mining approach to big data analytics". *Decision Support Systems* 81 (2016), pp. 30–40.

[67] P. J. Schmid. "Application of the dynamic mode decomposition to experimental data". *Experiments in fluids* 50.4 (2011), pp. 1123–1130.

[68] P. J. Schmid. "Dynamic mode decomposition of numerical and experimental data". *Journal of fluid mechanics* 656 (2010), pp. 5–28.

[69] P. J. Schmid et al. "Applications of the dynamic mode decomposition". *Theoretical and Computational Fluid Dynamics* 25.1 (2011), pp. 249–259.

[70] M. Schramm. *Mann-Kendall-Trend*. `https://github.com/mps9506/Mann-Kendall-Trend`. 2015 (accessed April 16, 2018).

[71] O. Sikha, S. S. Kumar, and K. Soman. "Salient region detection and object segmentation in color images using dynamic mode decomposition". *Journal of Computational Science* 25 (2018), pp. 351–366.

[72] S. S. Sindi and R. Dale. "Culturomics as a data playground for tests of selection: Mathematical approaches to detecting selection in word use". *Journal of Theoretical Biology* 405 (2016), pp. 140–149. ISSN: 0022-5193. DOI: `https://doi.org/10.1016/j.jtbi.2015.12.012`.

[73] M. S. J. Solaija et al. “Dynamic mode decomposition based epileptic seizure detection from scalp EEG”. *IEEE Access* 6 (2018), pp. 38683–38692.

[74] M. Swadesh. *The Origin and Diversification of Language*. Transaction Publishers, 1971.

[75] L. N. Trefethen and D. Bau III. *Numerical linear algebra*. Vol. 50. Siam, 1997.

[76] J. H. Tu et al. “On dynamic mode decomposition: Theory and applications”. *Journal of Computational Dynamics* 1.2 (2014), pp. 391–421. ISSN: 2158-2491. DOI: 10.3934/jcd.2014.1.391.

[77] P. D. Turney and S. M. Mohammad. “The natural selection of words: Finding the features of fitness”. *PloS one* 14.1 (2019), e0211512.

[78] L. Van der Maaten and G. Hinton. “Visualizing data using t-SNE.” *Journal of machine learning research* 9.11 (2008).

[79] P. Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[80] W. Webber, A. Moffat, and J. Zobel. “A similarity measure for indefinite rankings”. *ACM Transactions on Information Systems (TOIS)* 28.4 (2010), pp. 1–38.

[81] R. West et al. “Exploiting social network structure for person-to-person sentiment analysis”. *Transactions of the Association for Computational Linguistics* 2 (2014), pp. 297–310.

[82] Z. Yang. *Molecular evolution: a statistical approach*. Oxford University Press, 2014.

[83] N. Younes and U.-D. Reips. “Guideline for improving the reliability of Google Ngram studies: Evidence from religious terms”. *PloS one* 14.3 (2019), e0213554.

[84] G. K. Zipf. *Human behavior and the principle of least effort*. John Wiley & Sons Inc, 1950.

[85] G. K. Zipf. *The Psycho-Biology of Language*. Houghton, Mifflin, 1935.

Chapter 3

Evolving Contextual Semantics

The rise and fall of words exhibits interesting structures similar to evolving complex biological systems [33, 32, 26, 15]. The use of text data for the study of language evolution can be done by considering the change in frequency of words in time. Information and social influence is propagated through a social network by frequently using novel and existing words. This process can contribute to language change [11]. Words as language features are units passed on between people is the process called language transmission. Bryden et. al. [4] showed that the processes of horizontal transmission in biological systems are similar to language transmission. By using Twitter[1] text data, they found that individuals who engage in online conversation more often, they will use similar words in other conversations outside of their group. Language transmission is interesting because it is the basis of learning word meanings by encounter. We focus on the evolution of contextual semantics of words especially in social media platforms that can capture social change in real time.

Semantics is the study of meaning. As language speakers, we draw meanings from words by taking into account their relationships with other words. For example, the word "left" has meanings depending on its association with other words; "I want to turn *left* at that road" versus "His views are *left* leaning". These example sentences convey the literal meaning and the associative meaning of the word "left". Both example sentences echoes the literal meaning of "left". However, the associative meaning of "left" depends on the cultural context. From our example, "*left* leaning" combined with the word "views" is typically aligned with political beliefs associated with social equality and progressive ideals. Languages - like English - are complicated because of words with polysemous meanings. However, we can infer words' contextual meaning through the use of document statistics and machine learning.

Document statistics is a method of discovering the structural similarities of words within multiple documents. The method used for document analysis is the Latent Semantic Analysis (LSA) [17, 6]. LSA uses matrix algebra to extract vector representation of words. The underlying assumptions of LSA is that each document

[1]Twitter is an online social media platform - similar to Facebook and Reddit - that users can send short text messages to the public called tweets.

contains a mixture of words and similar word meanings have similar distributions. LSA is a classical method in the field of distributional semantics - which is also part of the field of natural language processing (NLP). Machine learning algorithms made its way in NLP. The method called Skip-Gram with Negative Sampling (SGNS) [20] is a classical neural network architecture which is fundamental to many sophisticated language models in NLP. SGNS uses bayesian statistics to extract distributed representations of words from bodies of text. Similar to LSA assumptions, the SGNS utilizes context word sampling under the assumption that similar words have similar distributions. Both LSA and SGNS are classical models in NLP and language modeling in general. Although both models are simple, they give reasonable and meaningful results. LSA and SGNS produce word vectors (also known as word embeddings) as numerical representations. The angular distance between word vectors is a measure of contextual meaning. The angular distances between words are observed to change in time as demonstrated by Hamilton et. al. [13] and Bamler et. al. [1]. The changes in angular distances of word embeddings in time are called semantic change.

In this chapter, we apply LSA and SGNS to study the evolution of contextual semantics within bodies of English texts taken from Twitter. This project's motivation is to study the emergence and structure of online social movements particularly the use of hashtags. Hashtags are labels with "#" character at the beginning and all of them are one word or multiple words with no spaces. For example, "#blacklivesmatter" is one word with words "black", "lives" and "matter" combined. Also, "#metoo" has "me" and "too" combined. They are used in social media to efficiently access specific content and to propagate information. Like many words, hashtags can be treated as actual words with contextual meanings. Hashtags have a unique functions in language and are evolving [5]. We explore the social movement hashtags "#blacklivesmatter" and "#metoo", and we compare the quality of LSA and SGNS results on these particular hashtags.

Background Work

The idea of temporal word embeddings to study language change in decade/year time-scale has been done recently and has grown more popular [30, 35, 2, 1, 13]. Studying the evolution of word meanings has been applied using Twitter in month/week time-scales. Temporal word embeddings has been shown to be useful on tracking Tweets related to important events in Europe for over a two-week period [27]. Garg et. al. uses word embeddings to measure the biases of historical gender and ethnic stereotypes [9]. Kulkarni et. al. [16] uses word embeddings for detection of linguistic change in Twitter and Quillot et. al. [28] uses the same method to track tweet content on specific cultural events.

The study of hashtag social movements has grown popular due to the increase of online social media usage. The use of hashtags on Twitter made users conveniently track events and information online. In particular, social movements have recently utilizing hashtags to spread information and gain support on a cause related to social justice. For example, Ince et. al. [14] uses Twitter data to study the language used by people

interacting to tweets with #blacklivesmatter. Gallagher et. al. [8] examines the differences in discourse of #blacklivesmatter versus #alllivesmatter movements by using Twitter text data. Lingren [18] examines the challenges of the #metoo campaign at its early stages, and they use Twitter data to analyze the discourse related to hate speech and disagreement. Moreover, Modrek et. al. [22] characterized and quantified words of early tweets with #metoo.

In Section 3.1, we explain how we scraped millions of public tweets, specifically those tweets with hashtags. We also explain how we organize the tweets into different data structures.

We examine the contextual semantic changes in time by utilizing both LSA and SGNS. We explain how we train the word embeddings in Section 3.2. In Section 3.3, we track the word angular positions and then we apply a simple cosine similarity method to identify the word path of contextual meaning in time.

In addition to Section 3.3, we apply the Linguistic Inquiry and Word Count (LIWC) to further analyze and discuss the psychological meanings of the Twitter data. LIWC is a linguistic method of content analysis by computing the degree of word usage based on word categories [24]. These word categories within their massive vocabulary are made through decades of empirical research. LIWC can compute a summarized metric to describe how function words are used, the level of sentimentality within a text, thinking styles, and even the given text's social value. Compared to LSA and SGNS, LIWC is not a "black-box" model. LIWC provides a reasonable level of word sense disambiguation to perform content analysis. We discuss the results in Section 3.4, conclusions and future word in Sections 3.5 and 3.6 respectively.

3.1 The Twitter Hashtag Corpus

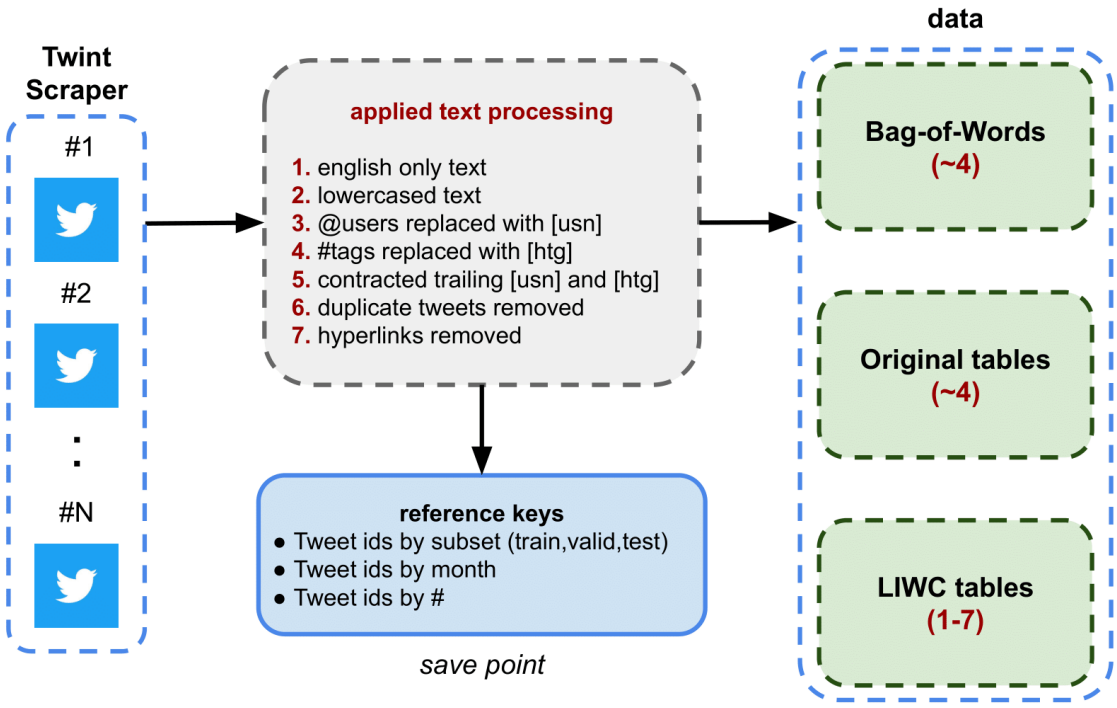
The data for this analysis are texts from Twitter. The Tweets from Twitter are scraped using the Twint software [2]. When scraping, a specific hashtag is entered as a query to request relevant tweets that contain that hashtag at a particular range of time. In this case, we have 94 hashtags scraped from January 2013 to December 2020. Fig. 3.1 is a diagram of the text processing of tweets. Twint's purpose is to scrape tweets without the use of a Twitter API to avoid most of the limits. It uses the search function on twitter and scrapes the results accordingly.

The diagram below in Fig. 3.1 illustrates the processing pipeline where the green boxes indicate separate data savepoints. The Bag-of-Words save point is where only the processed text is saved. The original tables savepoint is where the text and other tweet information is saved (e.g. favorite counts, reply counts, etc). The LIWC tables savepoint is where the text and LIWC metrics are saved. For all savepoints, texts was lowercased and the hyperlinks are removed. The scraper can take tweets in several languages but in our case, we only take the English texts. The username tag (e.g. @username) is

[2]Twint is an advanced Twitter scraping tool that doesn't use Twitter's API. This allows a user to scrape a Twitter user's followers, following, Tweets and more while avoiding most API limitations. Twint Python source codes are available in Github: `https://github.com/twintproject/twint`.

anonymized by replacing it by the character "[usn]". For all trailing usernames, it was contracted into the "[usn]" character. The hashtags (e.g. #hashtag) in the tweet are not fully recognized within the vocabulary of LIWC. Therefore, the hashtags are replaced with the character "[htg]" and trailing hashtags are contracted similar to the usernames for LIWC tables save point only. The processed text is entered into LIWC software and the results are organized into a table where the rows are labeled with unique tweets ids and the columns contain text and the LIWC metrics.

Figure 3.1: **Twitter text data collection and processing.** The labels (#1,#2,⋯,#*N*) above the Twitter icons on the left indicates the $N = 94$ number of hashtag queries used to scrape Tweets. The collected Tweet text undergoes into a text cleaning process (in red) and goes into three separate data structures (shaded green boxes) (the " 4" means "not 4"). The Bag-of-Words data contains only bodies of text while the Original tables data has more information for each Tweet (e.x. favorite and reply counts, and time stamp). The LIWC tables contains results from LIWC software where the columns corresponds to the LIWC dimensions (e.x. Analytic, Clout, Tone, etc.).



The 94 hashtags are chosen because they are related to popular culture, sports, public health, social movements, and/or general elections. For purpose of visualization and convenience, the hashtags are organized into groups based on their similarities about what they mean. For example, the hashtags #election2016, #election2018, #election2020, and #uselection are grouped together because they are all about the United States general elections. Table 3.1, we show all of the chosen hashtags organized into 20 groups. The hashtags in group 20 may not be related because each of these hashtags can be its own group.

Hashtags can have common tweets. For example, the tweet "with love, anything is possible! #loveislove #lovewins" belongs in the #loveislove and #lovewins. Tweet intersections of hashtags like the one mentioned above can produce duplicates when scraping. Each hashtag have different number of tweets; See S14 Figure to view the tweet frequency distribution of hashtags.

Table 3.1: **Table of chosen hashtags organized into groups.** The hashtags are organized into groups according to their similarities in content.

Group	Hashtags
1	#covid, #covid19, #coronavirus, #ebola, #h1n1, #pandemic
2	#lockdown, #lockdownnow, #openitup, #opensafely, #openup, #openitup
3	#covidiots, #socialdistancing, #stayhome, #stayhomestaysafe, #flattenthecurve, #alonetogether
4	#election2016, #election2018, #election2020, #uselection
5	#asiangames, #nfl, #olympics, #usopen, #worldcup
6	#asiangames2018, #rio2016, #sochi2014
7	#zootopia, #starwars, #babyyoda
8	#blackhistorymonth, #pridemonth, #womenshistorymonth
9	#prayforamazonia, #prayforboston, #prayfororlando, #prayforparis, #prayfortheworld, #prayforvegas
10	#ifidieinaschoolshooting, #orlandoshooting, #marchforourlives, #neveragain, #notonemore
11	#guncontrol, #guncontrolnow, #2a, #2ndamendment
12	#loveislove, #gaypride, #lovewins, #straightpride, #cheerstosochi
13	#standfortheanthem, #standfortheflag, #takeaknee, #taketheknee, #boycottnfl
14	#blacklivesmatter, #blm, #alllivesmatter, #alm, #bluelivesmatter, #backtheblue, #crimingwhilewhite
15	#womensmarch, #metoo, #whyistayed, #shoutyourabortion, #himtoo, #survivorprivilege
16	#notallmen, #allmencan, #yesallwomen, #sayhername, #sayhisname, #saytheirname
17	#maga, #makeamericagreatagain, #trumptrain
18	#impeachtrump, #notmypresident, #resist
19	#climatechange, #globalwarming, #climatestrike
20	#cancelculture, #covfefe, #fakenews, #nodapl, #flintwatercrisis, #obamacare

Tokenizer

The tokenizer is essential to creating word vectors in any NLP modeling. Tokenizers convert text to a list of words or phrases. Text tokenization can be done in multiple ways such as using n-grams or by utilizing word stems. The tokenizer we use is the *TweetTokenizer* function of the Natural Language Tool Kit (NLTK) module [3]. This

tokenizer is specifically made for tweets but it can not ignore stopwords and punctuations. The NLTK tweet tokenizer is special because it can tokenize emojis (pictograms) and hashtags properly. We use a separate function to ignore the most frequently used words known as stopwords[3] and punctuations. For example, given the lower-cased text in the following.

```
[usn] my dog is very timely,
and i want to to live in a nice hotel. :)
it is the random tv shows, man.
#fishbus #randomtweet
```

The tokenized version is written as:

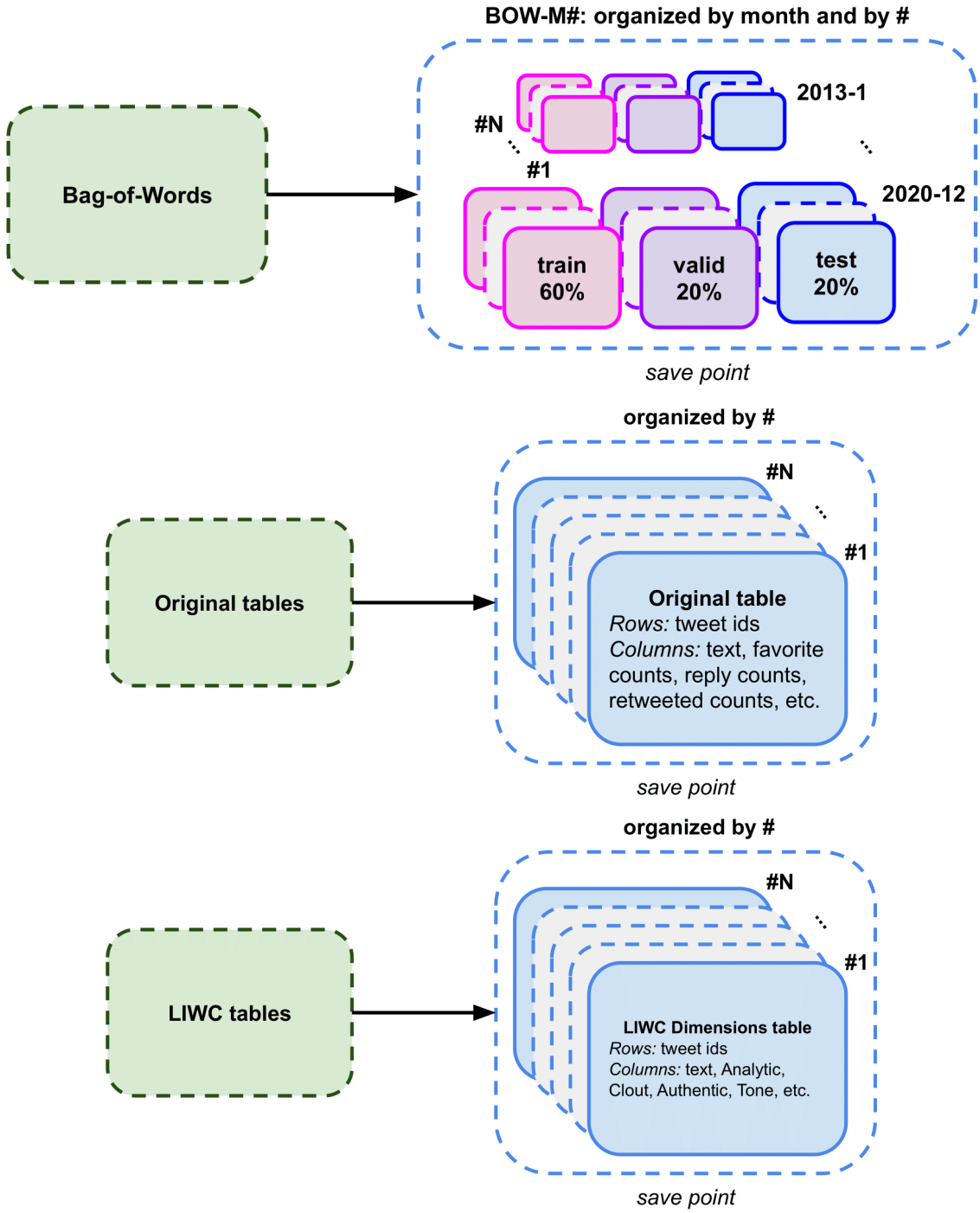
```
dog timely
live nice hotel :)
random tv man
#fishbus #randomtweet
```

Twitter Data Structures

In Fig. 3.1, we illustrated that the twitter data is scraped and processed and saved into a Bag-of-Word data structure. These bodies of text are divided into months and by hashtag. We illustrate the Bag-of-Words data organization in Fig. 3.2. The data is also divided into training, validation, and test sets by uniform random sampling tweets in each hashtag. Any words that occurred less than 100 times within 1 month of data is removed prior to training. The training set is used to train the LSA and SGNS models. These models are unsupervised, the evaluation method is typically done after the model is trained and by performing some specific task. The validation and test sets are set aside for later evaluation of the models.

[3]The stopwords list is obtained from Ranks NL website, `https://www.ranks.nl/stopwords`.

Figure 3.2: **Data stuctures of Bag-of-Words (labeled as BOW-M#), Original tables, and LIWC tables.** There 94 hashtags and 96 months. The data is divided into training, validation, and test set by uniform random sampling Tweets in each group and then they are divided by month. Any words with frequency less than 100 - within a month - is removed prior to training. The Original and LIWC tables are straightforwardly organized by hashtag with each its own rows and columns.



3.2 Word Embedding Models

We use LSA and SGNS models to create temporal word embedding matrices using the Twitter training data. The next two sections explains the mathematics and details on how to train the models.

3.2.1 Latent Semantic Analysis (LSA)

LSA is a method that extracts the relationships of documents and the words they contain [17, 6]. The end goal of LSA is to compute word representations as vectors. The mathematical foundations of LSA is explained by Martin and Berry 2007 [19] but we use the existing Python code implementation using the Scikit-Learn software [23]. The typical pipeline is usually by first computing the Term Frequency Inverse Document Frequency (TF-IDF) matrix and then apply the Singular Value Decomposition (SVD) on the TF-IDF matrix.

For computing the TF-IDF matrix, first we need to create the TF matrix. This is done by counting the individual words in each document and arranging them into a matrix. Let w be a word and d be a document. The elements of the TF matrix is computed by

$$tf(w,d) = \frac{x_{w,d}}{\sum_{w=1}^{M} x_{w,d}} \tag{3.1}$$

where the expression $x_{w,d}$ is the raw count of word w in document d and M is the number of words. We set a lower bound of $x_{w,d} >= 100$. Second, we compute the IDF matrix. let D be the set of documents. The elements of the IDF matrix is given by

$$idf(w,d) = \log\left(\frac{N}{|d \in D : w \in d| + 1}\right) \tag{3.2}$$

where N is the number of documents and the expression $|d \in D : w \in d|$ is the number of documents where word w occurred. The smoothing constant $+1$ in the denominator avoids the problem when $tf(w,d) = 0$.

The completed TF-IDF matrix is the product of the TF matrix and the IDF matrix which is an $M \times N$ matrix with elements

$$tfidf(w,d) = tf(w,d) \cdot idf(w,d) \tag{3.3}$$

where the operation $\cdot$ is the element-wise product.

In our work, the hashtags are considered as documents. For example, the tweets with #blacklivesmatter is considered one document. Since there are 94 hashtags, there are 94 documents or $N = 94$. The number of words is typically large. This results in a large and very sparse TF-IDF matrix. The last step is to apply SVD on the TF-IDF matrix. In our situation, the SVD is computed using randomized SVD [12]. This is done using the Scikit-Learn module [23]. The dimensionality reduction using SVD applied on the TF-IDF matrix allows for the reduction of the number of columns while retaining the most important features in the data. This is done by selecting top r most dominant

singular values and choosing the corresponding columns. The columns of the resulting SVD computation are known as “latent topics”. A “latent topic” is defined as a set of words with the top highest values of the elements of an SVD column.

LSA typically reduces the TF-IDF matrix by the rows and extracts similarities between documents. The main focus of using LSA in this work is the rows of the resulting SVD of the TF-IDF matrix. These rows are the vector representation of words and by computing the angular distance between words - known as the cosine similarity - is the measure of contextual meaning. We apply the LSA method for each month of the Bag-of-Words data. With M vocabulary words and 96 months of Twitter data, this yields 96 LSA word embedding matrices of size $M \times r$ where r is the number of topics chosen. We arbitrarily chose $r = 20$ to correspond the 20 hashtag groupings.

3.2.2 Skip-Gram with Negative Sampling (SGNS)

The SGNS model is a type of word embedding model that represents words into a vector space to extract meaning. The model was introduced by Mikolov et. al. [20]. Given a body of unstructured text, SGNS efficiently learns high quality vector representation of words. The SGNS model is derived from the original Skip-Gram (SG) model introduced by the same author [21]. We explain the mathematics of SG and SGNS in the following.

Given a sequence of words $w_1, w_2, \cdots, w_T$, the training objective of the SG model is given by

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-q\leq j\leq q, j\neq 0}\log\left(p(w_{t+j}|w_t)\right) \tag{3.4}$$

where q is known as the window size with center word w_t. The term $p(w_{t+j}|w_t)$ is the probability of encountering word w_{t+j} given the center word w_t. Let $\vec{\mathbf{v}}$ be the word vector of the context word v. The context word exist within the window of the target word w with word vector $\vec{\mathbf{w}}$. The probability $p(v|w)$ is computed using the softmax function:

$$p(v|w) = \frac{e^{\langle\vec{\mathbf{w}},\vec{\mathbf{v}}\rangle}}{\sum_{w\in\mathbf{V}} e^{\langle\vec{\mathbf{w}},\vec{\mathbf{v}}\rangle}} \tag{3.5}$$

where $\mathbf{V}$ is the vocabulary words set of length M. This is the case where Maximimum Likelihood Estimation (MLE) is used to estimate the objective. The total number of vocabulary words M is typically large and so computing the normalizing constant in Eq. 3.5 in the denominator is proportional to M.

Mikolov et. al. [20] presented the “negative” sampling approach which converts the objective into a logistical function. Let $P(y = 1|w, v)$ be the probability that the context word v is within the training window of the target word w. Let $P(y = 0|w, v)$ be the probability that the context word is outside the training window. With this approach, these probabilities are defined as sigmoid functions written as

$$\begin{aligned} P(y = 1|v, w) &= \frac{e^{\langle\vec{\mathbf{w}},\vec{\mathbf{v}}\rangle}}{e^{\langle\vec{\mathbf{w}},\vec{\mathbf{v}}\rangle} + 1} \text{ and} \\ P(y = 0|v, w) &= \frac{1}{e^{\langle\vec{\mathbf{w}},\vec{\mathbf{v}}\rangle} + 1}. \end{aligned} \tag{3.6}$$

This leads to the negative sampling objective function as

$$-\sum_{i=1}^{M}\left[\log\left(P(y=1|v,w)\right)+\sum_{j=1}^{k}\log\left(P(y=0|v,w)\right)\right]. \tag{3.7}$$

where k is the number of "negative" samples. The "negative" sampling approach avoids computing the normalizing constant in Eq. 3.4 by randomly sampling context words outside the training samples. The negative sampling approach allows one to update a small percentage of the weights (or the word vectors). The first term of Eq. 3.7 is maximized while the second term is minimized through iteration of random words taken from a unigram distribution. Words in real languages are typically not uniformly distributed. The most frequently occurring words are stopwords (e.g. "the", "a") provides little or no meaning in the context of other words. The SGNS model uses a simple subsampling approach to level the imbalance between frequently occurring words and rare ones. Thus, "negative" samples are purposely sampled words that would output $y = 0$ (words outside the training window), and using those samples to maximize the probability for $y = 1$ (words inside the training window).

The value of k is small typically in the range 5-20 for small training data set and 2-5 for large data sets. The SGNS model approximates the word vectors $\vec{\mathbf{w}}$ of some size l using stochastic gradient descent so that the negative sampling objective function is minimized. The length of word vectors is typically around 100-300 for large datasets. The contextual meaning of words can be measured by computing the cosine similarity between word vectors.

In our work, we apply the SGNS model for each time point of the Bag-of-Words data illustrated in Fig. 3.2. We chose the word vector length as $l = 100$, training window size of $q = 10$, and negative samples of $k = 5$. The SGNS models are trained using an existing module in Python called Gensim [29]. Gensim has the function called *Word2Vec* with hyperparameters vector length, training window size, and negative samples. With M vocabulary words and 96 months of Twitter data, there are 96 SGNS word embedding matrices of size $M \times 100$. Unlike the LSA word embeddings, the columns of SGNS are considered as features rather than "latent topics".

3.3 Word Angular Positions in Time

In the previous two subsections, we explained the formulation of word embeddings. These word embeddings are approximated using two separate methods called LSA and SGNS. Since there are 96 months of Twitter data, there are 96 matrices of LSA and 96 matrices of SGNS. The goal is to measure the word angular positions in time using the LSA and SGNS embeddings. The word embeddings are not aligned across time due to the stochastic nature of the SGNS and the SVD's non-uniqueness of its orthonormal basis. However, the length and angles of the embeddings are preserved. The steps we detail in this section are (1) the union of vocabulary words, (2) the alignment of the word embeddings, and (3) tracking the angular position of words.

Let $\mathbf{W}^{(t)} \in \mathbb{R}^{M \times N}$ be a word embedding matrix at time t trained using either LSA or SGNS. The total number of vocabulary words is M and N is the number of columns .

Union of vocabulary words

Some vocabulary words at time t may not exist at time $t + 1$. In order to have the word embedding matrices with the same row length, we take the union of vocabulary words for all t. The set of all vocabulary words for all t is

$$\mathbf{V} = \bigcup_{t=0}^{N} v^{(t)} \tag{3.8}$$

where $v^{(t)}$ is the set of vocabulary words at time t and the length of $\mathbf{V}$ is M which is the total number of vocabulary words.

For each word that did not exist at time t, we assign a vector to that word by computing the average vector of the trained embeddings. That is taking the mean of each column. We take the average vector and place that vector into the final word embedding $\mathbf{W}^{(t)}$. For example, if there are 400 total vocabulary words for all t and there are 300 existing words at time $t = 0$, then we take the average vector using the 300 trained word vectors at $t = 0$ and assign the resulting average vector to every remaining 100 words. This is a crucial step in order for us to align the embeddings.

Alignment of the word embeddings

We follow the method used by Hamilton et. al. [13] where they aligned their historical word embeddings by means of the orthogonal procrustes. The learned word embeddings $\mathbf{W}^{(t)}$ is aligned by finding the best rotation matrix at each time point so that the word embeddings for all time points have a common frame of reference. The orthogonal procrustes method preserves the length and angular information of the word vectors.

The rotation matrix $\mathbf{R}^{(t)}$ of size $N \times N$ is approximated by minimizing the error

$$E^{(t)} = \mathbf{W}^{(t)}\mathbf{R}^{(t)} - \mathbf{W}^{(t+1)} \tag{3.9}$$

were $\mathbf{W}^{(t)}$ is the learned word embeddings at time t. Approximating $\mathbf{R}^{(t)}$ is equivalent to finding the nearest orthogonal matrix to $\left(\mathbf{W}^{(t)}\right)^T \mathbf{W}^{(t+1)}$ which is an $N \times N$ matrix. We can minimize the error by optimizing

$$\mathbf{R}^{(t)} = \underset{\mathbf{Q}^T\mathbf{Q}=I}{\arg\min} \|\mathbf{W}^{(t)}\mathbf{Q} - \mathbf{W}^{(t+1)}\|_F \tag{3.10}$$

where the term $\|\cdot\|_F$ is the Frobenius matrix norm. That is,

$$\|\mathbf{W}\|_F = \left(\sum_{i=1}^{M}\sum_{j=1}^{N} w_{i,j}^2\right)^{\frac{1}{2}}. \tag{3.11}$$

The solution to the orthogonal procrustes problem can be efficiently solved through SVD [31].

Tracking the word angular positions

Given the trained and aligned word embeddings $\mathbf{W}^{(t)}$, we compute the angular position from the all-ones vector to each of the word vectors. The angular position of word w at time t is

$$\theta(w,t) = \frac{2\cos^{-1}\left(\frac{\langle \vec{\mathbf{w}}^{(t)}, J_N \rangle}{\|\vec{\mathbf{w}}^{(t)}\| \|J_N\|}\right)}{\pi} \tag{3.12}$$

where J_N is the all-ones vector of size N which is equal to the size of vector $\vec{\mathbf{w}}^{(t)}$ of word w. The range of $\theta(t,w)$ is $[0, 2\pi]$. The term $\|\cdot\|$ is the vector 2-norm.

Tracing Word Paths

A word path is the trajectory of a word's angular position through local neighborhoods of words. All words are moving together but each word takes a different path. Each word is located within a neighborhood at t. When a word moves at $t+1$, that word takes a path through different or similar neighborhood. We determine the temporal sets of word neighborhood below.

Recall in Eq. 3.8 that v_t is the set of vocabulary words at time t. We compute the cosine similarities of word $w \in v^{(t)}$ and $v \in v^{(t)}$. Given two words w and v, the cosine similarity between their word vectors at time t is given by the following:

$$\text{cos-sim}_{v,w}^{(t)} = \frac{\langle \vec{\mathbf{w}}^{(t)}, \vec{\mathbf{v}}^{(t)} \rangle}{\|\vec{\mathbf{w}}^{(t)}\| \|\vec{\mathbf{v}}^{(t)}\|} = \frac{\sum_{i=1}^{N} \vec{\mathbf{w}}_i^{(t)} \vec{\mathbf{v}}_i^{(t)}}{\|\vec{\mathbf{w}}^{(t)}\| \|\vec{\mathbf{v}}^{(t)}\|}, \tag{3.13}$$

where $\vec{\mathbf{w}}^{(t)}$ and $\vec{\mathbf{v}}^{(t)}$ are the word vectors of length N at time t for words w and v respectively, and $\|\cdot\|$ is the vector 2-norm. The range of the cosine similarity is $[-1, 1]$ where a value close to 1 means that the word vector for w is similar to the word vector of v. A value close to -1 means that the two vectors are distant in cosine.

To detect words with the most similar to a given word w at time t, we take words within a specified tolerance τ. At time t, the set of vocabulary words closest to the word w is given by

$$S_w^{(t)} = \left\{ s_v \in v_t \mid \text{cos-sim}_{v,w}^{(t)} >= 1 - \tau \text{ and } `\#' \notin v \right\} \tag{3.14}$$

where the term $`\#' \notin v$ means that words with the "#" character is excluded. Next, we separate the words in set $S_w^{(t)}$ into novel words (words that w sees for the first time) and repeated words (words that w have seen before). We write that as

$$\begin{aligned} \text{seen-}S_w^{(t)} &= \bigcup_{i=0}^{t} S_w^{(i)} \\ \text{novel-}S_w^{(t)} &= \left\{ s_v \notin \text{seen-}S_w^{(t)} \right\} \\ \text{repeated-}S_w^{(t)} &= \left\{ s_v \in \text{seen-}S_w^{(t)} \right\}. \end{aligned} \tag{3.15}$$

The word of interest w can be a hashtag. In other words, other hashtags are ignored. Our work sets $\tau_{LSA} = 0.05$ and $\tau_{SGNS} = 0.50$ to track the word paths of selected hashtags. Any word that is within the tolerance value is included in the local neighborhood of word w at time t. The resulting temporal sets of words is the path taken of word w.

Linguistic Inquiry and Word Count (LIWC)

We look into LIWC measurements for an independent temporal assessment of the tweet hashtag data. The data we collected from Twitter yields rich information about the collective psychological meanings of words. LIWC is a "transparent" text analysis software that measures the psychological implications from a given collection of words. Through their massive vocabulary, LIWC counts words and summarizes them according to four categories. The following list contains four LIWC categories of summarized metrics and their descriptions as explained by Pennebaker et. al. [25]. We chose LIWC 2015 version to perform textual analysis of tweets [24].

1. **Analytic - analytical thinking (range of 0-100).** A value of 0 means people are speaking from personal experience. A value of 100 means that people are communicating in a way that is logical and/or factual.

2. **Clout - relative social status, confidence, or leadership (range 0-100).** A value of 0 means that people are talking in a way with no indication of entitlement or attempt to influence. A value of 100 indicates authority or sense of entitlement.

3. **Authentic - authenticity and vulnerability (range 0-100).** The value of 100 means that people are speaking in a way that is more personal and sincere. The value close to zero means that the language use is more emotionally detached or deceptive.

4. **Tone - emotional tone of negative or positive sentiment (range 0-100).** A value of 0 means negative tones (e.g. anger, anxiety, sadness) while 100 means positive tones (e.g. happiness, joy, optimistic).

In our work, we apply the LIWC software to compute the summarized metrics for each of the tweets collected. That includes all tweets in train, valid, and test sets.

3.4 Results

The results are divided into three parts. The first part show an overview of the Twitter data by showing the tweet frequency of two hashtag groups. In parts 2 and 3, we examine the word paths and LIWC results of hashtags "#blacklivesmatter" and "#metoo".

Part 1 of 3: Tweet Frequency Time-Series Overview

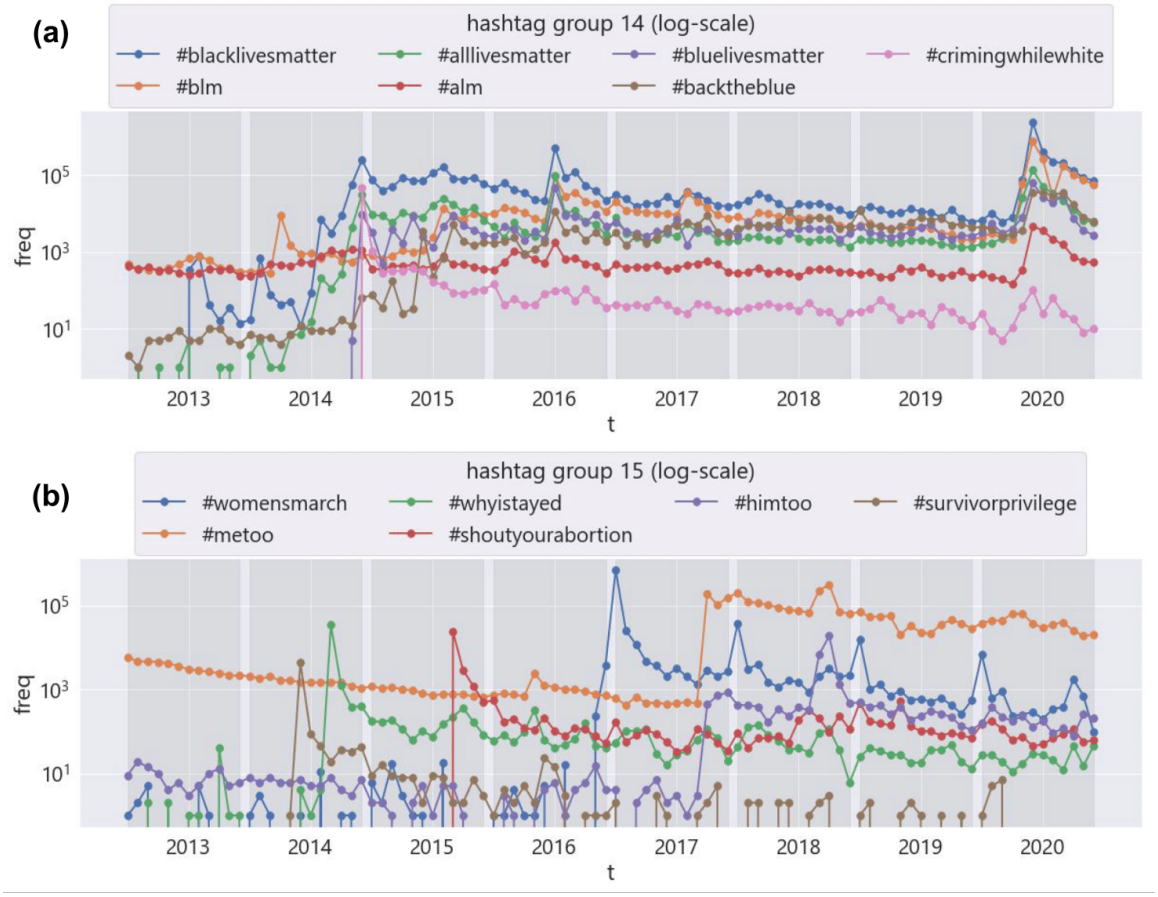
The 94 hashtags listed in Table. 3.1 are organized into 20 groups based on the hashtag's common themes. Our focus is the hashtags #blacklivesmatter and #metoo tweets and these hashtags belongs to Groups 14 and 15 respectively.

We present an overview of the Twitter data in Fig. 3.3. Specifically, we show the tweet frequency time-series of groups 14 and 15. By counting the number of tweets for each hashtag in these groups, we are able to track the trend of these hashtags from January

2013 to December 2020. These trends exhibit different kinds of behaviors. The hashtags in group 15 (Subfig. **b** in Fig. 3.3) shows different kinds of peaks for each hashtag. For example, #shoutyourabortion started to appear in September 2015 and eventually became steady in time. Another example is the time-series trend of #womensmarch where it started in January 2017 and starts to decrease in time but with some peaks annually every January. Moreover, the time-series in group 14 (Subfig. **a** in Fig. 3.3) shows patterns closely following the time-series pattern of #blacklivesmatter. There are other tweet frequency time-series pattern we observed such as cycle and constant trend (See supplementary figures S15 Figure and S16 Figure).

Our main focus here is to see the contextual semantic changes of the hashtags #blacklivesmatter and #metoo.

Figure 3.3: **Tweet frequency time-series of hashtag groups 14 and 15.** Subfig. **(a)** shows the tweet frequency time-series of the #blacklivesmatter and #alllivesmatter social movements related hashtags. We can see that #blacklivesmatter started to appear in July 2013 and reach three peaks in December 2014, July 2016, and June 2020. Subfig. **(b)** shows the tweet frequency time-series of the #metoo and women's rights movements related hashtags. We can see that the #metoo increased significantly in October 2017, and #womensmarch reach a peak in January 2017 which is the United States presidential inauguration month.



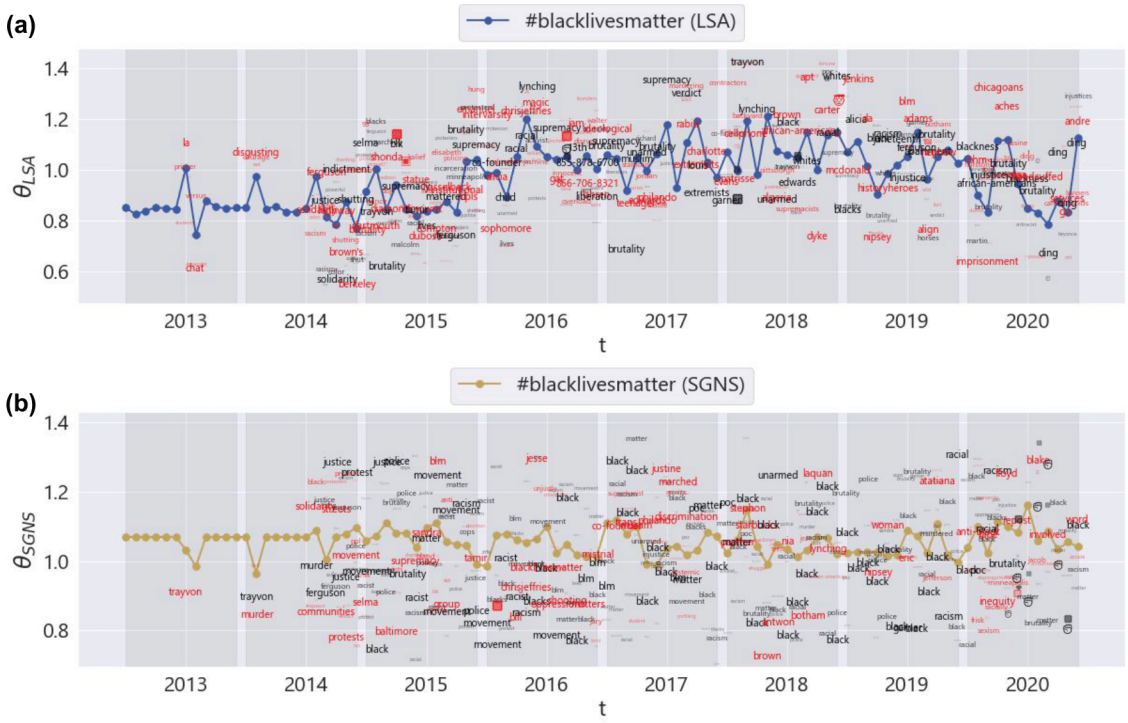
Part 2 of 3: Contextual Word Path of #blacklivesmatter

The Black Lives Matter movement - through the use of #blacklivesmatter - is part of a broader social movement focused on advocating racial justice and civil rights. The hashtag also spreads awareness of police brutality or the excessive use of force by law enforcement. This form of brutality is most often used against Black and African-American citizens motivated by conscious and unconscious form of racism.

In this section, we show and discuss the contextual semantics path of #blacklivesmatter from January 2013 to December 2020. In Fig. 3.4, we show the time-series angular position (Eq. 3.12) of #blacklivesmatter computed from using LSA and SGNS models. The differences between the LSA and SGNA results includes difference novel words. For example, at time point May 2020, the novel words in the word path of #blacklivesmatter using LSA model is "chicagoans" while using SGNS, it encountered the word "floyd". Similarly, in February 2014, LSA showed words "disgusting" while SGNS showed "murder". The differences in results is due to the difference in methods of LSA and SGNS. SGNS is a probabilistic model while LSA is a matrix model. The SGNS model captures the context of the target word based on the surroundings of the target word while the LSA uses multiple hashtags (or documents) to extract context words of the target word. In other words, LSA relies on other hashtags to infer the contextual meaning of #blacklivesmatter as opposed to SGNS which uses a more probabilistic approach. The LSA and SGNS has shown some biases based on the data on which it is trained as seen in the changes of novel words.

We do see common words captured by LSA and SGNS. The words "racism", "brutality", and "black" which indicates that both LSA and SGNS can independently capture the main context of #blacklivesmatter. SGNS shows a more refined results where similar repeated words are consistently captured in time. This indicates that SGNS can capture perpetuating context words deemed important to the central discourse of #blacklivesmatter. The models also captured context words related to specific events. For example, the word "trayvon" refers to Trayvon Martin murdered by George Zimmerman who was acquitted in August 2013 [14]. The acquittal triggered months of protests in response to Trayvon's murder. In subsequent months, we see other names such as "tamir" which refers to Tamir Rice who have died at the hands of law enforcement [14]. In addition, we see the word "ferguson" which refers to the city who had a social unrest in response to the shooting of Micheal Brown by a police officer [14]. These specific event markers indicate that the LSA and SGNS models can reasonably capture the related context words of #blacklivesmatter. In response to the #blacklivematter, other opposing hashtags have appeared such as #alllivesmatter (See S17 Figure).

Figure 3.4: **Word path of the temporal contextual semantics of #blacklivesmatter.** Using the LSA (Subfig. **(a)**) and SGNS (Subfig. **(b)**) models, we show the time-series path of the word angular position of the #blacklivesmatter. The word path taken of #blacklivesmatter is a series of words where it encounters a novel word (colored in red) and a repeated word (colored in black). We see in the word path of #blacklivesmatter with words "justice", "brutality", "supremacy", "black", "white", and "racism". Both the LSA and SGNS models picks up similar words related to the hashtag but they show some differences in novel words. The models captures the most important context words related to the #blacklivesmatter. We see that the repeated words in LSA and SGNS models both captured to words "brutality", "racism", and "black" which suggest that both models captured similar contextual semantics.

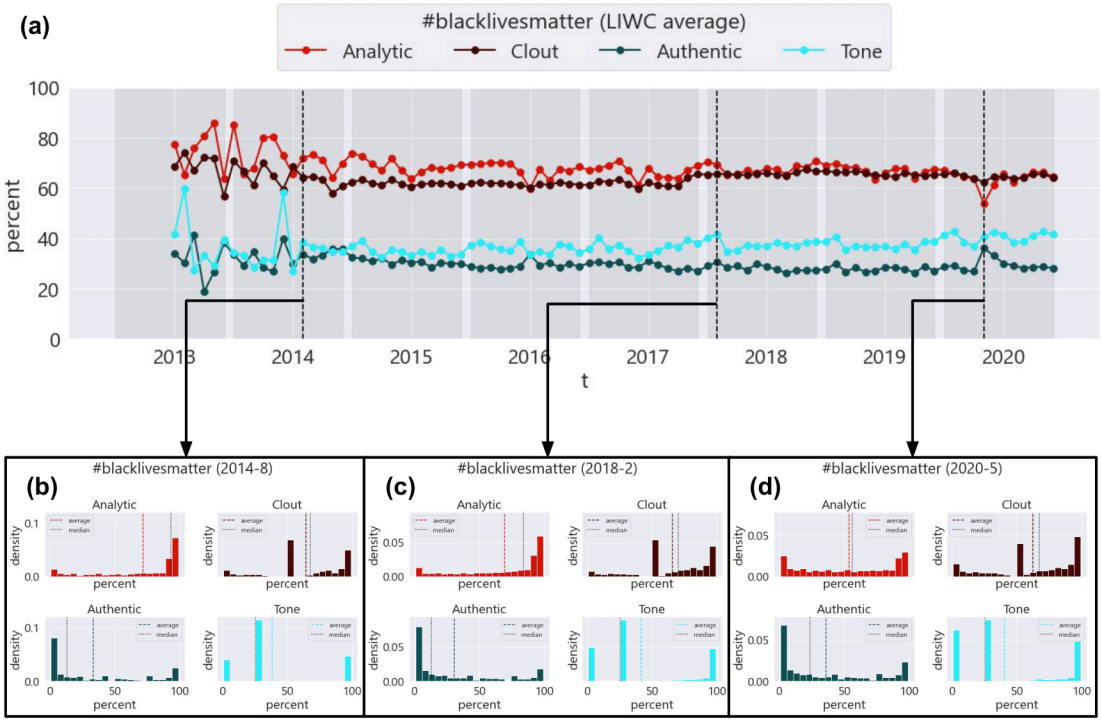


Next, we look at the LIWC time-series averages in Fig. 3.5. In Subfig. **(a)**, we see that the time-series for Analytic, Clout, Authentic, and Tone remained generally constant. However, we see a slight change in some time points like March 2018 and May 2020. Without knowing the specific historical events related to #blacklivesmatter, these small changes largely did not affect the time-series at future times. For example, the Analytic average remained above 50% which indicates that tweets with #blacklivesmatter uses language in a way that is factual rather than from personal experience. Although we see no significant changes of the Analytic average, we see that the actual distribution structure changed. In Subfig. **(b)** the Analytic distribution at August 2014 shows that most of the tweets have values near 100% but in May 2020 in Subfig. **(c)** the distribution appears to spread out into the values near 0%. This indicates that the language use has changed from factual to more personal accounts. In general, all of the LIWC distributions appears to be multimodal at any time-point which indicates that the tweets with #blacklivesmatter is a

mixture of different language usage that needs further investigation.

The LIWC results we see here is the direct consequence of their vocabulary set and formulas to compute their metrics. For example, the Tone distribution at time point May 2020 in Subfig. **(c)** appears to have a three modes and looks likes most of these tweets have the same Tone value. Although the highest mode is in the middle, we see an increase in density of Tone values close to 0% from February 2018 to May 2020. This indicates that the general emotions of tweets became negative.

Figure 3.5: **LIWC averages of the #blacklivesmatter.** Each tweet for each month with #blacklivesmatter is entered into the LIWC software to generate four summarized LIWC metrics which are Analytic, Clout, Authentic, and Tone. Subfig. **(a)** shows the time-series LIWC averages of the #blacklivesmatter. The Subfigs **(b)**, **(c)**, and **(d)** shows the actual distributions of the LIWC results at selected time-points in August 2014, February 2018, and May 2020 respectively. Although the times-series averages appears to be constant, the actual distributions of the Tone and Analytic metrics have changed in structure.



Part 3 of 3: Contextual Word Path of #metoo

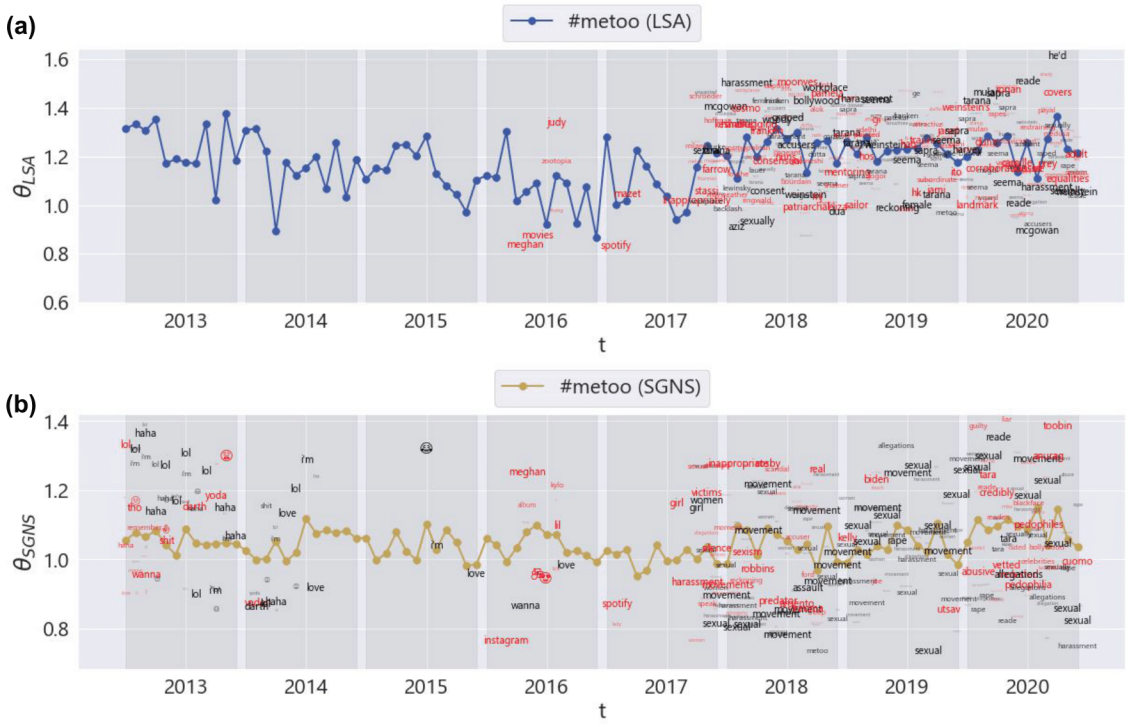
The Me Too movement is a social movement against sexual harrassment. Through the use of #metoo on Twitter, people tweet and share their experiences of sexual assault, mostly from women, to empower other women through solidarity. The original use of the term “Me Too” was by civil rights activist Tarana Burke to raise awareness and support for sexual violence survivors [22]. In October 2017, American actress and author Alyssa Milano used the phrase “Me too” have gained global attention which led to people sharing

their experiences of sexual assault on Twitter, and the subsequently created #metoo [22].

In this section, we show and discuss the contextual semantics path of #metoo from January 2013 to December 2020. In Fig. 3.6, we show the time-series angular position (Eq. 3.12) of #metoo computed from using LSA and SGNS models. The general similarities between LSA ans SGNS results of the #metoo word path are the context words “sexual”, “harassment”, and “rape”. These words remained consistently seen by the word path of #metoo which indicates that the LSA and SGNS can independently capture related context words to #metoo.

Similar to the results of #blacklivesmatter word path, we see that the #metoo word path captures novel words. Using LSA, the captured novel words in 2016 is similar to the novel words using SGNS. Based on the SGNS word path of #metoo, the contextual meaning of the hashtag have changed from “lol” and “haha” to “sexual”, and “harassment”. The original use of #metoo in 2013 only means that the person who says #metoo is saying that they are agreeing to someones views or they are expressing similar experience as the other person. Starting in October 2017, both LSA and SGNS starts to capture context words “harrassment”, “sexual”, and “inappropriately” which corresponds to Alyssa Milano’s original tweet publication. In a sense, the original meaning of #metoo is retained but in a different context because the Me Too movement’s central purpose is for people to share their experiences that they too have similar experience. In response to the #metoo - similar to #blacklivesmatter - other opposing hashtags have appeared such as #himtoo (See S18 Figure).

Figure 3.6: **Word path of the temporal contextual semantics of #metoo.** Using the LSA (Subfig. **(a)**) and SGNS (Subfig. **(b)**) models, we show the time-series path of the word angular position of the hashtag #metoo. The word path taken of #metoo is a series of words where it encounters a novel word (colored in red) and a repeated word (colored in black). Beginning at October 2017, we can see in the path of #metoo with words "harassment", "sexually", "sexual", and "rape". The SGNS model picks up more words than the LSA. Before October 2017, We can see different contextual words which are "haha" and "lol".



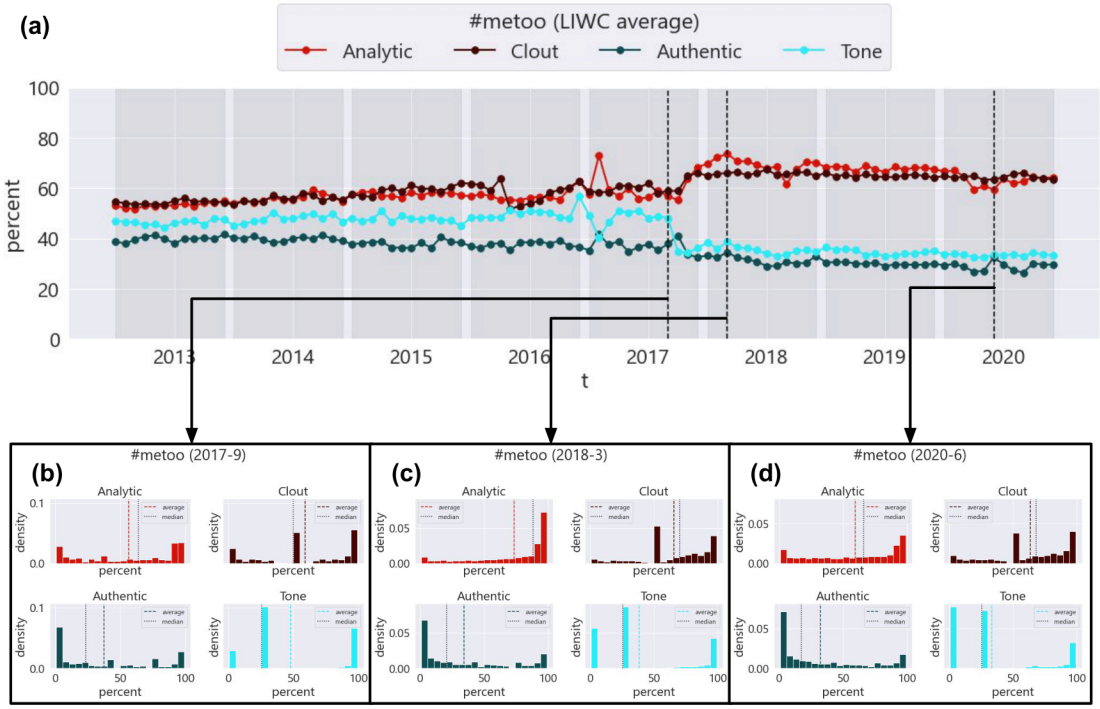
Next, we look at the LIWC time-series averages in Fig. 3.7. We immediately see in Subfig. **(a)** that there is a change at time point October 2017 which corresponds to Alyssa Milano's original tweet. Prior to October 2017, the LIWC time-series are generally constant. After the change in October 2017, the time-series is generally constant as well. If we look at the actual distributions of the LIWC metrics shown in Subfig. **(b)**, **(c)**, and **(d)**, we see a change in structure of the distribution. For example, the Tone distribution at September 2017 have more tweets with Tone values closer to 100%. This indicates that the tweets had generally positive sentiments. In March 2018 and June 2020, the Tone distributions have changes where the tweets are shown to have more Tone values of closer to 0%. This indicates that the tweets after October 2017 had generally negative sentiments. The trimodal structure in Tone is similar to what we have seen in #blacklivesmatter in Fig. 3.5.

The Authentic and Clout metrics are observed to be largely unchanged except for the Analytic metric. The Analytic metric distribution at March 2018 shows that most tweets have Analytic value close to 100%. This means that the language used in this time point is mostly factual rather than personal. The #metoo tweets with Analytic value close to

100% probably mentioning facts about sexual assault and many people are refrained speaking from personal experience. Similar to LIWC results of #blacklivesmatter, we observed that the LIWC distributions are largely multimodal. This indicates that the #metoo tweets contains an underlying mixture of different language usage that needs further investigation.

Furthermore, looking at the Authentic averages for both #blacklivesmatter in Fig. 3.5 and #metoo in Fig. 3.7 seems to be counter-intuitive to the actual usage of #blacklivesmatter and #metoo. For example, #metoo is used for expressing there personal experiences which should give an Authentic value of close to 100%. However, the Authentic distribution at June 2020 in Subfig. **(d)** in Fig. 3.7 shows that the majority of tweets have Authentic values close to 0%. Does this indicate that the majority of #metoo or #blacklivesmatter tweets contains emotionally detached or deceptive language styles? Not necessarily. Due to how LIWC categorize their words into these metrics can lead to biases in the results that may not reflect the actual usage of the given hashtag. We see in the Twitter data that there are mixtures of polysemous word meanings which can lead to misinterpretation of the LIWC results.

Figure 3.7: **LIWC averages of the #metoo.** Each tweet for each month with #metoo is entered into the LIWC software to generate four summarized LIWC metrics which are Analytic, Clout, Authentic, and Tone. Subfig. **(a)** shows the time-series LIWC averages of the #metoo. The Subfigs **(b)**, **(c)**, and **(d)** shows the actual distributions of the LIWC results at selected time-points in September 2017, March 2018, and June 2020 respectively. We can see in the time-series of LIWC averages, there is change right after October 2017. This change is a response to a usage increase of #metoo to spread the social movement against sexual abuse and harrassment.



3.5 Conclusion

We applied LSA and SGNS to compute the contextual semantics word paths of #blacklivesmatter and #metoo. We have observed in the results that these two terms have word paths with novel and repeated words. The repeated words shown in the word path of #blacklivesmatter indicates that the Black Lives Matter movement context of these term remained largely consistent since its original use. The word path of #metoo have change from its generic meaning to a contextual meaning related to the Me Too movement. Both LSA and SGNS produced reasonable word paths for #blacklivesmatter and #metoo that largely corresponds to the real social movement's original meaning and purpose. Although both LSA and SGNS are two of the most simple word embedding models, our recommendations is to use SGNS for a more refined results.

Based on our results, the repeated words that remained consistent throughout time roughly corresponds to the idea that language transmit information similar to horizontal transmission in biological systems [4]. The #blacklivesmatter word path have shown repeated words like "racist", "brutality", and "black" could indicate a strong language transmission has happened. The methods we used and the results we show seems to support the theory of language transmission whereby individuals use similar words to conversations when encountered from where they got it from.

Both hashtags in our example have garnered popularity. Most likely that the repeated words encountered by the word paths of #metoo are due to a strong social influence that made words like "sexual" and "harassment" consistently appear after October 2017. Thus, leading to a change in context of the use of #metoo [11].

3.6 Future Work

The methods presented in this chapter can greatly help social movement organizers on how to improve their online communication and engagement. The critical analysis of social movements can benefit from interdisciplinary collaboration.

There are more areas to explore of the Twitter data. The use of LSA and SGNS to extract evolving contextual semantics of #blacklivesmatter and #metoo can reveal insights into the evolution of word meanings and social movements. These methods can help the study of language evolution - especially social movements that trigger significant changes in culture - by focusing on the novel and repeated words in the contextual word paths. We can ask the question of how often does a word encounter novel context words? What is the probability when novel context words becomes repeated at future times? At what conditions does context words becomes repeated after first encounter? The angular position of words can be used to compute the rate at which a word changes in time. Modeling semantic change of words is a challenging task. With the combination of the angular position and the contextual word paths, this work can lead into the direction of characterizing, quantifying, and modeling semantic change by using a data-driven approach such as the Dynamic Mode decomposition (DMD).

There is an ongoing problem of evaluating unsupervised word embedding models

because of the complicated nature of language [34, 10]. Evaluation by the use of word analogy tasks has been done for the SGNS model [20]. Other evaluation models like concept categorization can also be used to group words with similar concepts [34]. Generally, the evaluation is done by doing a task like classification, translation, or question answering. In our work, we can evaluate the LSA and SGNS models using the validation and test data sets by performing simple multi-class classification task using a naive Bayes classifier.

Finally, there are more advanced word embedding and language models such as the Bidirectional Encoder Representations from Transformers (BERT) [7]. BERT is a neural network model with multiple layers originally designed for machine translation. The architecture of BERT - which is based from the SGNS model architecture - is now widely used as a basis of more advanced language models today. This work can benefit of using BERT to further study social movements hashtags, and can contribute to the study of language evolution in general.

Bibliography

[1] R. Bamler and S. Mandt. “Dynamic word embeddings”. *International conference on Machine learning*. PMLR. 2017, pp. 380–389.

[2] R. C. Barranco et al. “Tracking the evolution of words with time-reflective text representations”. *2018 IEEE international conference on big data (big data)*. IEEE. 2018, pp. 2088–2097.

[3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. ” O’Reilly Media, Inc.”, 2009.

[4] J. Bryden, S. P. Wright, and V. A. Jansen. “How humans transmit language: horizontal transmission matches word frequencies among peers on Twitter”. *Journal of The Royal Society Interface* 15.139 (2018), p. 20170738.

[5] E. Cunha et al. “Analyzing the dynamic evolution of hashtags on twitter: a language-based approach”. *Proceedings of the workshop on language in social media (LSM 2011)*. 2011, pp. 58–65.

[6] S. Deerwester et al. “Indexing by latent semantic analysis”. *Journal of the American society for information science* 41.6 (1990), pp. 391–407.

[7] J. Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Tech. rep. 2018. arXiv: 1810.04805v2.

[8] R. J. Gallagher et al. “Divergent discourse between protests and counter-protests:# BlackLivesMatter and# AllLivesMatter”. *PloS one* 13.4 (2018), e0195644.

[9] N. Garg et al. “Word embeddings quantify 100 years of gender and ethnic stereotypes”. *Proceedings of the National Academy of Sciences* 115.16 (2018), E3635–E3644.

[10] S. Ghannay et al. “Word embedding evaluation and combination”. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*. 2016, pp. 300–305.

[11] R. Goel et al. “The social dynamics of language change in online networks”. *International conference on social informatics*. Springer. 2016, pp. 41–57.

[12] N. Halko, P.-G. Martinsson, and J. A. Tropp. “Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions” (2009).

[13] W. L. Hamilton, J. Leskovec, and D. Jurafsky. “Diachronic word embeddings reveal statistical laws of semantic change”. *arXiv preprint arXiv:1605.09096* (2016).

[14] J. Ince, F. Rojas, and C. A. Davis. “The social media response to Black Lives Matter: How Twitter users interact with Black Lives Matter through hashtag use”. *Ethnic and racial studies* 40.11 (2017), pp. 1814–1830.

[15] S. Kirby, M. Dowman, and T. L. Griffiths. “Innateness and culture in the evolution of language”. *Proceedings of the National Academy of Sciences* 104.12 (2007), pp. 5241–5245.

[16] V. Kulkarni et al. “Statistically significant detection of linguistic change”. *Proceedings of the 24th International Conference on World Wide Web*. 2015, pp. 625–635.

[17] T. K. Landauer, P. W. Foltz, and D. Laham. “An introduction to latent semantic analysis”. *Discourse processes* 25.2-3 (1998), pp. 259–284.

[18] S. Lindgren. “Movement mobilization in the age of hashtag activism: examining the challenge of noise, hate, and disengagement in the# MeToo campaign”. *Policy & Internet* 11.4 (2019), pp. 418–438.

[19] D. I. Martin and M. W. Berry. “Mathematical foundations behind latent semantic analysis”. *In*. 2007.

[20] T. Mikolov et al. “Distributed Representations of Words and Phrases and Their Compositionality”. *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS’13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 3111–3119.

[21] T. Mikolov et al. “Efficient estimation of word representations in vector space”. *arXiv preprint arXiv:1301.3781* (2013).

[22] S. Modrek and B. Chakalov. “The# MeToo movement in the United States: text analysis of early twitter conversations”. *Journal of medical Internet research* 21.9 (2019), e13837.

[23] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[24] J. W. Pennebaker et al. *Linguistic Inquiry and Word Count: LIWC2015*. `https://www.LIWC.net`. 2015.

[25] J. W. Pennebaker et al. *The development and psychometric properties of LIWC2015*. Tech. rep. 2015.

[26] A. M. Petersen et al. “Languages cool as they expand: Allometric scaling and the decreasing need for new words”. *Scientific reports* 2.1 (2012), pp. 1–10.

[27] L. Phillips et al. “Intrinsic and extrinsic evaluation of spatiotemporal text representations in Twitter streams”. *Proceedings of the 2nd Workshop on Representation Learning for NLP*. 2017, pp. 201–210.

[28] M. Quillot et al. “Exploring temporal analysis of tweet content from cultural events”. *International Conference on Statistical Language and Speech Processing*. Springer. 2017, pp. 82–93.

[29] R. Řehůřek and P. Sojka. “Software Framework for Topic Modelling with Large Corpora”. English. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[30] M. Rudolph and D. Blei. “Dynamic embeddings for language evolution”. *Proceedings of the 2018 World Wide Web Conference*. 2018, pp. 1003–1011.

[31] P. H. Schönemann. “A generalized solution of the orthogonal procrustes problem”. *Psychometrika* 31.1 (1966), pp. 1–10.

[32] S. S. Sindi and R. Dale. “Culturomics as a data playground for tests of selection: Mathematical approaches to detecting selection in word use”. *Journal of Theoretical Biology* 405 (2016), pp. 140–149. ISSN: 0022-5193. DOI: https://doi.org/10.1016/j.jtbi.2015.12.012.

[33] P. D. Turney and S. M. Mohammad. “The natural selection of words: Finding the features of fitness”. *PloS one* 14.1 (2019), e0211512.

[34] B. Wang et al. “Evaluating word embedding models: methods and experimental results”. *APSIPA transactions on signal and information processing* 8 (2019).

[35] Z. Yao et al. “Dynamic word embeddings for evolving semantic discovery”. *Proceedings of the eleventh acm international conference on web search and data mining*. 2018, pp. 673–681.

Chapter 4

Question Answering on Long Documents

Natural Language Processing (NLP) has come a long way from minimal content understanding to machine translation on multiple languages. The rise of sophisticated machine learning models and algorithms has made significant advances in language modeling. Although the accuracy of language models are improving, there is still problems on capturing the subtleties and complexities of language. For example, question answering tasks are one of the most difficult problems in NLP along with machine translation. Part of the difficulty within these tasks is that languages are very complex and usually, written language has natural narrative structures. Written language on a long document may contain evolving word meanings. Language models may not be able to capture the temporal structure of very long documents with narrative structures because usually these models can only infer from short snap shots of written texts. In addition, machine translation also faces similar challenges because translating from a language to another may require more than a one-to-one direct translation of word meaning and sentence structure.

This chapter tests the question answering task accuracy of machine learning language models on long documents with narrative structures. The title of this project is **"Grid Search Hyperparameter Benchmarking of BERT, ALBERT, and LongFormer on DuoRC"** which is a collaborative work by Alex John Quijano, Sam Nguyen, and Juanita Ordonez [13].

Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory (LLNL) under contract DE AC52 07NA27344. Alex John Quijano was supported as a summer intern at LLNL. LLNL-TR-817729.

4.1 Abstract

The purpose of this project is to evaluate three language models named BERT, ALBERT, and LongFormer on the Question Answering dataset called DuoRC. The language model task has two inputs, a question, and a context. The context is a paragraph or an entire document while the output is the answer based on the context. The goal is to perform grid search hyperparameter fine-tuning using DuoRC. Pretrained weights of the models are taken from the Huggingface library. Different sets of hyperparameters are used to fine-tune the models using two versions of DuoRC which are the SelfRC and the ParaphraseRC. The results show that the ALBERT (pretrained using the SQuAD1 dataset) has an F1 score of 76.4 and an accuracy score of 68.52 after fine-tuning on the SelfRC dataset. The Longformer model (pretrained using the SQuAD and SelfRC datasets) has an F1 score of 52.58 and an accuracy score of 46.60 after fine-tuning on the ParaphraseRC dataset. The current results outperformed the results from the previous model by DuoRC.

4.2 Introduction

Question Answering (QA) is a fundamental task in reading comprehension in humans. Comprehending sentences and paragraphs requires a level of abstract understanding that only humans are capable of doing. Asking a question is a way to evaluate a reader on how they understood a given passage and it is also a way to see if a given passage contains information that the question asks. Reading Comprehension (RC) tasks in Natural Language Processing (NLP) is one of the most challenging problems in computer science. RC typically involves making machines understand and comprehend sentences and paragraphs with key features like narration and complex reasoning. The RC tasks also have major problems with synthesizing answers which require background and common-sense knowledge that goes beyond the given paragraphs. There has been significant progress in the past that involves RC with the goal of improving QA tasks. A particular QA dataset called SQuAD [14] (Stanford Question and Answering Dataset) is a popular dataset and - which over the past four years - it has been used to build and improve NLP models. NLP models such as the BERT [4] (Bidirectional Encoder Representation from Transformers) model are widely used in developing and training for the purpose of QA. Our particular focus is to fine-tune and perform hyperparameter benchmarking using pretrained BERT-based models on an RC dataset called DuoRC [15].

With the goal of evaluating BERT-based models on DuoRC, this requires fine-tuning these models with a set of hyperparameters. The BERT-based models in question are the original BERT [4], ALBERT [9], and LongFormer [2]. The pre-trained weights of these models are available through the Huggingface library [20]. These pretrained weights are then fine-tuned using the DuoRC dataset. Fine-tuning means that we further train the model weights so that it fits into the DuoRC dataset. In Section 4.3, we explain the background work in terms of RC and QA and the novel datasets used in the past. We also explain the previous work done of DuoRC and the performance of the models used in this dataset. In Section 4.4, the DuoRC dataset is explained in detail and the

preprocessing done for the fine-tuning tasks. We then proceed to explain the mechanisms of the language models BERT, ALBERT, and Longformer in Section 4.5. The results are then presented in Section 4.6 and we discuss the results in detail on what it entails about the DuoRC dataset. Finally, we conclude in Sections 4.7 and 4.8 explaining the outcomes and future work of this research work respectively.

4.3 Related Work

There are many QA datasets that have been developed over the past few years and the NLP model development also have made significant progress on improving performance on these datasets. For example, the DuoRC [15] dataset attempts to push the challenges of RC. Most QA datasets have short passages that contain descriptive passages rather than passages with temporal reasoning and narration. The SQuAD is a particular example of a QA dataset that has short descriptive passages. Similar datasets are the TriviaQA [6], HotpotQA [22], and MS MARCO [1]. Usually, the passages have lexical overlap with the questions which made the current models achieve high reasonable performance. The answers to the questions are typically longer - full sentences - than the DuoRC's answers. Other similar datasets to DuoRC are the MovieQA [18], NarrativeQA [8], and NewsQA [19]. There are other datasets with a much more complex structure. The dataset called Discrete Reasoning Over Paragraphs (DROP [5]) contains paragraphs, questions, and answers that involve the reader to make discrete operations such as addition, subtraction, comparison, and coreference resolution. There are also recent datasets that focus on temporal and coreferential reasoning which are the TORQUE [11] and Quoref [3].

Our particular focus is to evaluate BERT, ALBERT, and Longformer on the DuoRC dataset. We want to know how the hyperparameters influence the performance of the models when fine-tuning the pre-trained model weights. Similar studies have been done, e.g., A Robustly Optimized BERT Pretraining Approach which is also known as RoBERTa[10]. Their studies involved different approaches to pretraining the BERT model architecture to yield the best performance. They indicated that the choice of hyperparameters - similarly here - significantly impacts the performance of the model. The performance was improved by training the model longer and with a larger training batch size. In comparison to our study, we evaluate three BERT-based models on the same dataset rather than evaluating one model on multiple datasets

4.4 Dataset

DuoRC Description. DuoRC [15] is a Reading Comprehension (RC) dataset that contains question-answer pairs that are created from pairs of documents containing movie plots. These pairs of documents contain two different versions of the same movie narrative created by different authors. The documents are paired as short and long plots. The short plots are called the original (labeled as "selfRC" in DuoRC source code) and

the long plot is called "paraphraseRC". Both of the documents have the same narrative but with different lengths and word usage. Each pair of documents have the same set of questions and the answers are based on the plot. Fig. 4.1 is an illustration of the DuoRC dataset where one plot example is shown with its question and answers. What's shown in this Figure are two versions of the same plot where one is with a short plot taken from Wikipedia and the other is a long plot taken from IMBD. With this Figure, there are four example questions (both plots have the same questions) and answers underneath each plot. For the Short plot, the question Q1 *"Who is the owner of the funeral home?"* has the answer *"Eliot Deacon"* which refers to the context *"...owner of the funeral home, Eliot Deacon..."*. In contrast, the longer plot actually refers to *"Eliot Deacon"* as the *"funeral director"* instead of the *"owner"*. The authors of DuoRC [15] collected this dataset where they obtained 7680 movie pairs of long and short plots with 186,089 unique question-answer pairs. The short plots had an average of 580 words while the long plots had an average of 926 words. The QA pairs are created by crowd workers from the Amazon Mechanical Turk (AMT). They first showed the short plot to the first set of workers (2559) to generate QA pairs and they showed the longer plot to a different set of workers (8021) where they answer the questions generated from the first set. The second set of workers also indicated in their answer whether the question is answerable or not based on the context plot. There only 703 workers who are in the first and second sets. The workers are asked to give answers as short as possible.

SelfRC vs ParaphraseRC. One of the main differences between the short and long plot is the sentences referred to in the plot for a given question. For example, the question Q2 *"What killed Paul?"* - with answer *"A car accident"* in the short plot and *"Car accident"* in the long plot - refers to two different context sentences from the two plots. The short plot with sentence annotated as **Q2[...]** (Fig. 4.1 left panel) answers the question more directly. In comparison, the long plot with sentence annotated as **Q2[...]** (Fig. 4.1 right panel) has a longer context and did not explicitly say *"Car accident"* but both answers are essentially the same. Our observations and the observations of the authors of DuoRC indicate that there are some inconsistencies and weaknesses in the dataset. Two of them are (1) the plots sometimes have unstructured and inexplicable sentences and (2) some of the answers to the same questions are different between the short and long plot. The different answers are not necessarily a weakness but some have answers which are not correct. On Fig. 4.1 (right panel) with **Q2[...]** annotation, the sentence reads - on some parts - it does not make any sense. Second, the answers to questions Q3-Q4 are different. The short plot answers to Q3-Q4 is consistent with the short plot. However, the answers to the longer plot tell a different outcome. Even though the questions are the same for Q3 *"Whose funeral does Anna Taylor attend"* the answers *"her piano teacher"* versus *"Her own"* refers to two different contexts from the two plots. There is also a wide disagreement on Anna Taylor's profession. The question Q4 *"What is Anna Taylor's profession"* has answer *"Teacher"* in the short plot and *"funeral director"* in the long plot. This is probably a result of human error from the AMT. There is no indication of human performance measures on this particular dataset. In a more broad overview of the data, 40.7% of the questions have the same answer between the short and long plots,

37.8% have overlapped, and 21% have partial overlap. The rest are no-answer questions.

Figure 4.1: **An example of DuoRC QA pairs of plots with short and long plots.** For illustration purposes, the highlighted colored texts (short plot in blue and long plot in green) are the relevant spans of the question and answers shown at the bottom of each plot. Even though the same questions for both plots are the same, some answers (in red) appears to be inconsistent between the plots. See the DuoRC [15] paper for more examples.

Span and Full Subsets. As mentioned in the DuoRC paper, the SelfRC and DuoRC are divided into subsets called "Span" and "Full". The "Span" subset is a set of plots where only the relevant sentences to the questions are extracted. The sentence was

considered relevant if at least 50% of the question words (stop words excluded) overlap with words in the sentence. If there were no 50% overlap words, the threshold is reduced to 30% overlap words. The words are considered overlapping if (1) they are verbatim, (2) they are word morphemes, and (3) they are semantically similar, and (4) two words are the same in WordNet. Word morphemes are words that have a root word with pre-fixes, post-fix, or compound words. For example, the words "unread", "readable", and "unreadable" have the same root word "read". The semantically similar words are determined using the Glove [12] and the Skip-thought [7] word embeddings. A word is similar to a word if within the top 50 nearest neighbors of the embedding space. WordNet is a word database - almost similar to a thesaurus - that match a meaningfully similar word. After we additional data processing, the dataset we used for fine-tuning is that we only include questions whose answers are completely verbatim in the context and the un-answerable questions are kept.

Previous Models and Reported Performance Results. The DuoRC paper has its own model that they trained on their dataset. The model in question is the BiDAF [17] (BiDirectional Attention Flow) where the model architecture is a multi-stage hierarchical process with bidirectional attention flow. The first BiDAF model is called *SpanModel*. The second BiDAF model is called *GenModel* where the model goes through two stages of processing, span prediction using BiDAF, and answer generation using the span and the question. The performance results of these models on DuoRC are below than the performance on the SQuAD. The BiDAF model was originally trained on the SQuAD dataset. The performance is low due to the fact that the DuoRC dataset has a longer context with a complex structure compared to the SQuAD. We can see in Table 4.1 the reported results of the performance of the models trained on DuoRC.

Table 4.1: **The reported results of the DuoRC [15] paper where the performance measures are for the test sets.** The values in [···] are reported results in their GitHub repository, `https://duorc.github.io/`. The bolded text with a '*' is the measure we can compare with our results in Tables 4.2 and 4.3.

DuoRC	subset	model	acc.	F1
SelfRC	**Span**	**SpanModel**	**46.14***	**57.49***
		GenModel	**16.45**	**26.97**
	Full	SpanModel	[37.53]	[50.56]
		GenModel	[15.31]	[24.05]
ParaphraseRC	**Span**	**SpanModel**	**27.49***	**35.10***
		GenModel	**12.66**	**19.48**
	Full	SpanModel	[14.92]	[21.53]
		GenModel	[5.42]	[9.64]

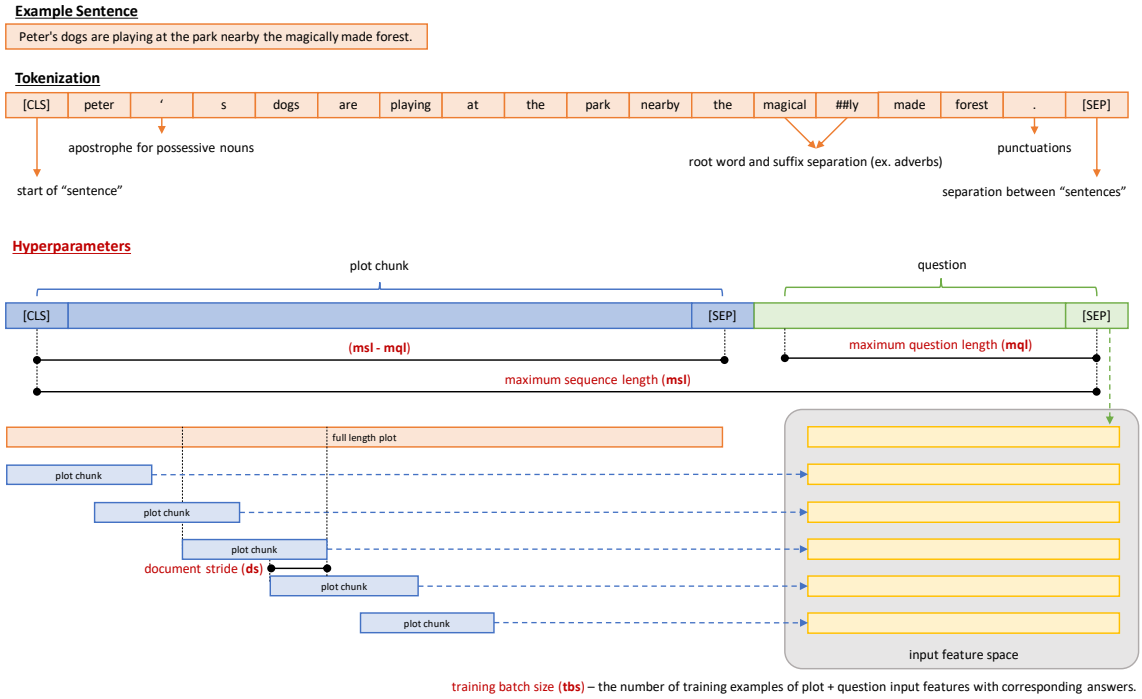
This study only uses the Span subset of the SelfRC and ParaphraseRC for fine-tuning the BERT, ALBERT, and LongFormer Models.

4.5 Models

Tokenization and Hyperparameters. The tokenization is an important pre-step process for any NLP models. There are many different ways tokenization is done. Specifically for the BERT model, the tokenization is based on word and word structures like the apostrophe for possessive nouns and suffixes of adverbs (see Fig. 4.2). The entire tokenization process is done using WordPiece tokenization [16]. This is a subword segmentation algorithm that was originally used for Neural Machine Translation (NMT) tasks and the algorithm is then used in the BERT model [21]. An additional token "[CLS']" is added at the beginning of the sequence and a token "[SEP]" which indicated a separation of two "sentences" or at the end. A "sentence" is applied loosely as an actual sentence or a few paragraphs. The hyperparameters are parameters that we can adjust prior to training. The maximum sequence length (**msl**) is the maximum length - or number of tokens - of the input on a given model which includes the question and the context. The maximum question length (**mql**) is the maximum length - or number of tokens - of the question input. If the input plot is longer than the maximum sequence length then the approach is to take chunks of the sequence to the max length with a given document stride. The document stride (**ds**) is the length of each stride on how the pre-processing of the long context turned into features (see Fig. 4.2). The training batch size (**tbs**) is the number of training examples of plot and question input features with corresponding answers. It is also the number of batches per GPU. We used 4 GPUs when fine-tuning. There are effectively 4×**tbs** actual batch size. These hyperparameters are standard for training QA models using the Huggingface library [20].

BERT. The Bidirectional Encoder Representations from Transformers (BERT) [4] is a language representation model that uses encoder-decoder architecture. The general structure of these neural networks processes language in sequence and the input passes through a layer called an "encoder" where the words are communicating information with each other for the model to generate a semantic structure. For the model to perform a task such as QA, the flow of information then passes through a "decoder" where it performs the reverse operation while it recovers the information from the "encoder". This type of neural network structure is known as Transformers in NLP. This is a novel architecture that is designed to process sequential data. Similar architectures such as the recurrent neural networks (or RNNs), typically require it to process the sequence data from beginning to end but Transformers avoids this requirement, and this led to a novel ground-work for faster models to train on large datasets. We refer the reader to the original paper for details. In short, the BERT model has two versions. The first version is $\mathbf{BERT_{BASE}}$ with 12 layers with hidden layer sizes of 768 and 12 attention layers. The second version is $\mathbf{BERT_{LARGE}}$ with 24 layers with hidden layer sizes of 1024 and 16 attention layers. The parameters are approximately 110M and 340M respectively. Self-attention layers use dot-product similarity scores to augment each

Figure 4.2: **BERT tokenization process and hyperparameters.** This diagram shows the tokenization process and it shows how the hyperparameter during training works. The hyperparameter labels are in bolded red text.



token's representation with information from other tokens that are similar or contextually informative. Crucially, self-attention layers are fully parallelizable because there is no recursiveness, unlike RNNs.

ALBERT. The A Lite BERT (ALBERT) [9] is a smaller modified version of BERT. It uses transformer neural network architecture just like BERT. In ALBERT, the factorization of the embedding parameters is implemented. The model decomposes the embedding space into two lower dimensional matrices before projecting it into the hidden space. In this case, the number of parameters of ALBERT is reduced compared to the original BERT. Similar to BERT, ALBERT has two versions which are ALBERT-large and ALBERT-base. It also has two other versions which are ALBERT-xlarge and ALBERT-xxlarge. The ALBERT-base has approximately 12M parameters with 12 layers where each layer has 768 hidden layer sizes. The ALBERT-large has approximately 18M parameters with 24 layers where each layer has 1024 hidden layer sizes. One key additional feature for ALBERT is the cross-layer parameter sharing. The purpose of the parameter sharing is to improve parameter efficiency. The parameters are shared across layers using a feed-forward network (FFN).

LongFormer. The Long document TransFormer (LongFormer) [2] is designed to deal with long documents. For BERT and ALBERT, the number of token inputs for the model is limited to an upper bound of 512 tokens, the LongFormer model has input tokens upper bound of size 4096. Similar to BERT, the LongFormer has Transformers neural network architecture style with self-attention mechanisms. The key additional feature for LongFormer is the "Attention Pattern". Due to the importance of local context, the LongFormer model builds a sliding window approach for attention. The layers are stacked using the windowed attention that scales linearly based on sequence length. A "global attention" is inserted into selected input locations so that the model is flexible for training a specific task like QA. Due to this approach, the number of parameters for LongFormer is approximately 149M.

Pretrained Weights and Performance Measures. The pretrained weights for the BERT, ALBERT, and LongFormer models are freely accessible from the Huggingface library [20]. The performance measure used in this study is the F1 and accuracy scores. In classification modeling, the F1 score is a measure of accuracy where precision and recall are used to compute it. The precision score is the fraction of true positives over the total of true positive and false positives. The recall score is the fraction of true positive over the total of relevant items. The F1 is the harmonic mean of precision and recall. The accuracy score is correctly classified items over the total.

4.6 Results

SelfRC. We see a significant improvement of the performances across all models for the SelfRC datasets compared to the original performances from Table 4.1. Table 4.2 shows that the best model is the ALBERT model pretrained with the SQuAD2 dataset. The best F1 score is 76.4 and the accuracy score is 68.52 using the validation set. The results also indicate that increasing the hyperparameters improves the performance of

the models. For example, the highest **msl** hyperparameter we tested yielded the highest performance value across all models. The models with **tbs** of 4 took longer to fine-tune than the models with **tbs** of 10. The larger the **tbs** the better the performance will be. Recall that the **tbs** is the batch size per GPU. Since we used 4 GPUs during training, it is actually 4×**tbs** batch size. However, due to memory constraints, we chose a **tbs** of 4 for the LongFormer models with **msl** of 768.

Table 4.2: **Short Plot (SelfRC) of DuoRC Fine-Tuning Results.** Each row corresponds to a combination of hyperparameters used to fine-tune the models using the Self DuoRC dataset. The bolded rows indicate the best F1 values for the validation set for each model while the best model overall is marked with a '*'.

model	msl	ds	mql	tbs	F1 val	F1 test	acc. val	acc. test
albert-base-v1 (SQuAD1 pretrained)	384	128	64	4	70.8	70.87	62.89	62.78
				10	71.13	71.03	63.28	63.2
	512	128	64	10	74.45	74.21	66.37	66.16
		384	**64**	**10**	**74.47**	**74.33**	**66.17**	**66.23**
albert-base-v2* (SQuAD2 pretrained)	384	128	64	4	68.35	69.46	60.38	61.78
				10	70.91	71.56	62.67	63.81
	512*	128	64	10	75.33	75.07	67.48	67.22
		384*	**64***	**10***	**76.4***	**76.29***	**68.52***	**68.32***
bert-base-uncased (BooksCorpus and English Wiki pretrained)	384	128	64	4	70.69	70.2	62.5	62.01
				10	70.56	70.62	62.26	62.35
	512	**128**	**64**	**10**	**74.14**	**73.72**	**65.82**	**65.25**
		384	64	10	73.63	73.07	65.06	64.72
longformer-squad1 (SQuAD1 pretrained)	384	128	64	4	67.99	67.83	59.82	59.36
		128	64	10	67.47	68.16	59.38	59.77
	512	128	64	10	71.07	70.19	62.72	61.76
		384	64	10	70.92	70.59	62.59	62.21
	768	128	64	4	73.47	73.39	65.08	64.97
		384	**64**	**4**	**75.9**	**75.06**	**67.24**	**66.41**
longformer-squad2 (SQuAD2 pretrained)	384	128	64	4	67.92	67.56	59.33	59.02
				10	69.06	69.08	60.71	60.83
	512	128	64	10	71.82	71.33	63.21	62.66
		384	64	10	70.86	69.86	62.4	61.53
	768	**128**	**64**	**4**	**74.35**	**73.64**	**65.7**	**65.22**
		384	64	4	73.92	73.82	65.34	65.15

ParaphraseRC. Table 4.3 shows that the best model is the LongFormer model pretrained with the SelfRC dataset. The best F1 score is 52.78 and the accuracy score is 46.60 using the validation set. The hyperparameters used in fine-tuning the ParaphraseRC dataset is chosen from the results of the SelfRC dataset fine-tuning. The F1 scores for the

Paraphrase RC are lower than the results for the SelfRC as expected. Compared to the selfRC, the ParaphraseRC dataset contains longer plots with questions that may or may not have an overlapping vocabulary.

Table 4.3: **Long Plot (ParaphraseRC) DuoRC Fine-Tuning Results.** Each row corresponds to a combination of hyperparameters used to fine-tune the models using the Paraphrase DuoRC dataset. The bolded rows and marked with '*' indicates the best F1 value for the validation set.

model	msl	ds	mql	tbs	F1 val	F1 test	acc. val	acc. test
albert-base-v1 (SQuAD1 pretrained)	512	384	64	10	50.31	51.08	45.15	45.86
albert-base-v2 (SQuAD3 pretrained)	512	384	64	10	50.14	50.91	45.15	45.6
albert-selfduorc-v1 (Self DuoRC pretrained)	512	384	64	10	51.25	51.74	45.94	46.31
albert-selfduorc-v2 (Self DuoRC pretrained)	512	384	64	10	51.41	51.92	46.34	46.48
bert-base-uncased (BooksCorpus and English Wiki pretrained)	512	128	64	10	48.8	48.22	43.1	42.63
bert-selfduorc-uncased (Self DuoRC pretrained)	512	128	64	10	51.24	51.09	45.37	45.2
longformer-squad1 (SQuAD1 pretrained)	768	384	64	4	50.68	50.07	44.06	43.55
longformer-squad2 (SQuAD2 pretrained)	768	384	64	4	51.68	50.69	45.26	44.17
longformer-selfduorc1* (Self DuoRC pretrained)	**768***	**384***	**64***	**4***	**52.78***	**51.94***	**46.60***	**45.22***
longformer-selfduorc2 (Self DuoRC pretrained)	768	384	64	4	52.28	52	45.82	45.52

4.7 Conclusion

We have performed a grid search hyperparameter benchmarking on three models on the DuoRC dataset. The models we evaluated are the BERT, ALBERT, and LongFormer models which are transformer-based neural network models. The DuoRC dataset contained two main components for each unique plot. The SelfRC has shorter plot lines while the Paraphrase has longer plot lines. The dataset was reduced into subsets called the "span" and "Full" where the "span" subset is the set of plots where only the relevant sentences to the questions are extracted. The best performing model is the ALBERT

model which was pretrained using the SQuAD1 and fine-tuned on the SelfRC. The best performing model fine-tuned on the ParaphraseRC is the LongFormer model which was pretrained using the SelfRC.

4.8 Future Work

Our work is only a step toward improving NLP models for QA and RC in general. Increasing the **msl** hyperparameter for the LongFormer model would definitely improve the performance. The LongFormer model is specifically designed for datasets with longer context documents and should be considered on model development for RC. Datasets like DuoRC contain mostly words with narrative structures but other datasets that contain scientific vocabulary, mathematical equations, logic may be problematic when fine-tuning models for RC. Pretrained models may not always work on a different dataset which is why fine-tuning is the key to improve models to do a specific task.

Bibliography

[1] P. Bajaj et al. "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset". *CEUR Workshop Proceedings* 1773 (Nov. 2016). arXiv: 1611.09268. URL: http://arxiv.org/abs/1611.09268.

[2] I. Beltagy, M. E. Peters, and A. Cohan. "Longformer: The Long-Document Transformer" (Apr. 2020). arXiv: 2004.05150. URL: http://arxiv.org/abs/2004.05150.

[3] P. Dasigi et al. "Quoref: A reading comprehension dataset with questions requiring coreferential reasoning". *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*. Association for Computational Linguistics, 2020, pp. 5925–5932. ISBN: 9781950737901. DOI: 10.18653/v1/d19-1606. arXiv: 1908.05803.

[4] J. Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Tech. rep. 2018. arXiv: 1810.04805v2.

[5] D. Dua et al. "DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs" (Mar. 2019). arXiv: 1903.00161. URL: http://arxiv.org/abs/1903.00161.

[6] M. Joshi et al. "TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension". *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)* 1 (May 2017), pp. 1601–1611. arXiv: 1705.03551. URL: http://arxiv.org/abs/1705.03551.

[7] R. Kiros et al. *Skip-Thought Vectors*. Tech. rep. 2015, pp. 3294–3302.

[8] T. Kočiský et al. "The NarrativeQA Reading Comprehension Challenge". *Transactions of the Association for Computational Linguistics* 6 (Dec. 2018), pp. 317–328. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00023. arXiv: 1712.07040. URL: http://deepmind.com/publications.

[9] Z. Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations" (Sept. 2019). arXiv: 1909.11942. URL: http://arxiv.org/abs/1909.11942.

[10] Y. Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach" (July 2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692.

[11] Q. Ning et al. *TORQUE: A Reading Comprehension Dataset of Temporal Ordering Questions*. Tech. rep. 2020. arXiv: 2005.00242v1. URL: http://dickens.seas.upenn.edu:4006/.

[12] J. Pennington, R. Socher, and C. D. Manning. *GloVe: Global Vectors for Word Representation*. Tech. rep., pp. 1532–1543. URL: http://nlp..

[13] A. J. Quijano, S. Nguyen, and J. Ordonez. "Grid Search Hyperparameter Benchmarking of BERT, ALBERT, and LongFormer on DuoRC". *arXiv preprint arXiv:2101.06326* (2021).

[14] P. Rajpurkar et al. "SQuAD: 100,000+ Questions for Machine Comprehension of Text" (June 2016). arXiv: 1606.05250. URL: http://arxiv.org/abs/1606.05250.

[15] A. Saha et al. "DuoRC: Towards Complex Language Understanding with Paraphrased Reading Comprehension". *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)* 1 (Apr. 2018), pp. 1683–1693. arXiv: 1804.07927. URL: http://arxiv.org/abs/1804.07927%20https://duorc.github.io/.

[16] M. Schuster and K. Nakajima. "Japanese and Korean voice search". *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. 2012, pp. 5149–5152. ISBN: 9781467300469. DOI: 10.1109/ICASSP.2012.6289079.

[17] M. Seo et al. "Bidirectional Attention Flow for Machine Comprehension" (Nov. 2016). arXiv: 1611.01603. URL: http://arxiv.org/abs/1611.01603.

[18] M. Tapaswi et al. *MovieQA: Understanding Stories in Movies through Question-Answering*. Tech. rep. 2016, pp. 4631–4640. URL: http://movieqa.cs.toronto.edu.

[19] A. Trischler et al. "NewsQA: A Machine Comprehension Dataset" (Nov. 2016), pp. 191–200. arXiv: 1611.09830. URL: http://arxiv.org/abs/1611.09830.

[20] T. Wolf et al. "HuggingFace's Transformers: State-of-the-art Natural Language Processing" (Oct. 2019). arXiv: 1910.03771. URL: http://arxiv.org/abs/1910.03771.

[21] Y. Wu et al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation" (Sept. 2016). arXiv: 1609.08144. URL: http://arxiv.org/abs/1609.08144.

[22] Z. Yang et al. "HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering" (Sept. 2018). arXiv: 1809.09600. URL: http://arxiv.org/abs/1809.09600.

Chapter 5

Conclusion

5.1 Summary

Language is a product of human culture. We communicate in many different ways and one of the ways we communicate is with words. Words are units of language that constantly appear in many forms, in many different parts of the world, and in cyberspace. These words can be treated as data and can be analyzed. Language evolve by changing and creating words, and by changing its meanings. These linguistic changes are analogous to the forces of biological evolution which are mutation, flow, drift, and natural selection. In many cases, studying the nature of language requires us to take many samples of words in a form of text data. Most often that these text data are unstructured, unlabeled, and unknown. When studying language, the initial assumption that researchers make is that it contains a rich statistical structures [7, 2, 8, 9, 3]. Treating text data as bag-of-words is a reasonable approach when it comes to extracting latent topics and contextual meaning. Researchers have also shown that words can be algebraic. By representing words as vectors, we can apply simple vector addition and subtraction to predict coherent answers to word analogies and categories [1, 6, 5].

In this dissertation, we explored statistical and algebraic techniques for large scale text data and for modeling language evolution. We focused on the unigram (words) time-series data taken from the Google ngram corpus, a large scale text data with 100 year's worth of word frequencies in eight unique languages. We also focused on applying modern word embedding models to study the evolution of contextual semantics. Using text data from an online social media platform, we attempted to create word paths of contextual meanings in time. We focused on prominent social movement hashtags because we suspect that social change can lead to change in word meanings. This work attempts to use statistical modeling and word embedding models to study the nature of language and how it evolves.

The following summarizes our contributions of the dissertation towards language evolution modeling:

- **Chapter 2 Section 2.2:** We proposed a statistical model for modeling unigram time-series by looking at the ranks of words rather than solely on frequencies. The statistical model we proposed is a modified Wright-Fisher model of neutral

evolution and we articulated the mechanism of how word ranks change in time. We presented that - with our initial hypothesis of neutrality - the results from our model and the data suggests that real language behave contrary to a neutral evolution.

- **Chapter 2 Section 2.3:** We proposed a data-driven approach to modeling unigram time-series data where we applied the Dynamic Mode Decomposition (DMD) to model and analyze the dynamics of language. We demonstrated - with the assumption that language behave like a dynamical system - the ability of DMD to extract temporally meaningful interpretations of the unigram time-series data.
- **Chapter 3:** We applied word embedding models - which are the Latent Semantic Analysis (LSA [5]) and Skip-Gram with Negative Sampling (SGNS [6]) - to create temporal word paths of contextual semantics. We demonstrated in our work that by looking at the word paths of prominent social movement hashtags, we can see the emergence of an established contextual meanings of those particular hashtags.
- **Chapter 4:** We performed model evaluations of recent language models such as Bidirectional Encoder Representations from Transformers (BERT [1]) to determine its accuracy on predicting answers to questions from long documents. Although we see some improvements on the model accuracy, there is still more work to be done.

5.2 Future Work

The study of language and its evolution has been going on for centuries, but recent advances in technology have made it possible to do a more comprehensive study of language evolution. We believe that the combination of Natural Language Processing (NLP) and concepts from evolutionary biology can lead to further advance the study of language. Therefore, we propose the following future work to further extend the applications of mathematical modeling of language:

- The Wright-Fisher inspired model is a model of neutral evolution that simulates drift. By modifying the model to incorporate selection, it would be interesting to study the behavior of the model and compare it with the language data. With this modification, we can infer from or model to identify which words specifically are neutral or non-neutral.
- The DMD [4] model has shown tremendous potential in applications of modeling linguistic change. We have shown that the technique can extract semantics based on the time-series data. While the DMD is widely used in the fluid dynamics community, understanding language evolution would greatly benefit from this type of model.
- We focused on the simple and straightforward models such as LSA and SGNS to extract word contextual paths of evolution. More advanced language models such as the Bidirectional Encoder Representations from Transformers (BERT [1]) have

shown to improve context word understanding in several languages. The use of more advance language model would greatly improve the quality of generating word paths.

- The challenge of using deep learning models to read text and answer questions, the Question Answering task, remains an open and intriguing scientific problem. Question Answering is a critical procedure on evaluating reading comprehension in language models. One area rich for opportunity is to improve these approaches is by considering changes in contextual word meanings.

Looking into the future of language evolution modeling, we hope to contribute to the advancement of machine translation models by leveraging the evolutionary aspect of language.

Finally, the field has been largely driven by available corpora which have been highly focused on the English language and other European languages. In most cases, the study of language evolution has been from a Eurocentric perspective. We hope to expand our work into languages other than English and create a diverse interdisciplinary collaboration.

Bibliography

[1] J. Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Tech. rep. 2018. arXiv: 1810.04805v2.

[2] A. Karjus et al. "Challenges in detecting evolutionary forces in language change using diachronic corpora". *arXiv preprint arXiv:1811.01275* (2018).

[3] V. Kulkarni et al. "Statistically significant detection of linguistic change". *Proceedings of the 24th International Conference on World Wide Web*. 2015, pp. 625–635.

[4] J. N. Kutz et al. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[5] D. I. Martin and M. W. Berry. "Mathematical foundations behind latent semantic analysis". *In*. 2007.

[6] T. Mikolov et al. "Distributed Representations of Words and Phrases and Their Compositionality". *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 3111–3119.

[7] J. A. Morales et al. "Rank dynamics of word usage at multiple scales". *Frontiers in Physics* 6 (2018), p. 45.

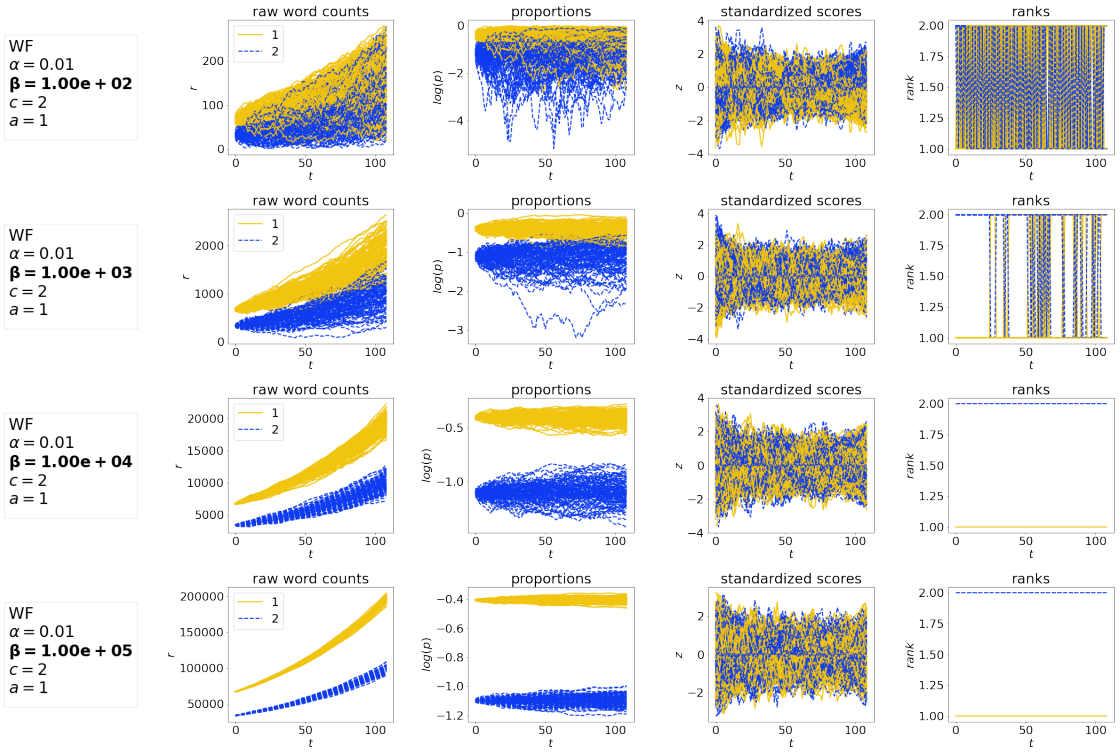
[8] M. G. Newberry et al. "Detecting evolutionary forces in language change". *Nature* 551.7679 (2017), pp. 223–226.

[9] S. S. Sindi and R. Dale. "Culturomics as a data playground for tests of selection: Mathematical approaches to detecting selection in word use". *Journal of Theoretical Biology* 405 (2016), pp. 140–149. ISSN: 0022-5193. DOI: https://doi.org/10.1016/j.jtbi.2015.12.012.

Supplementary Materials

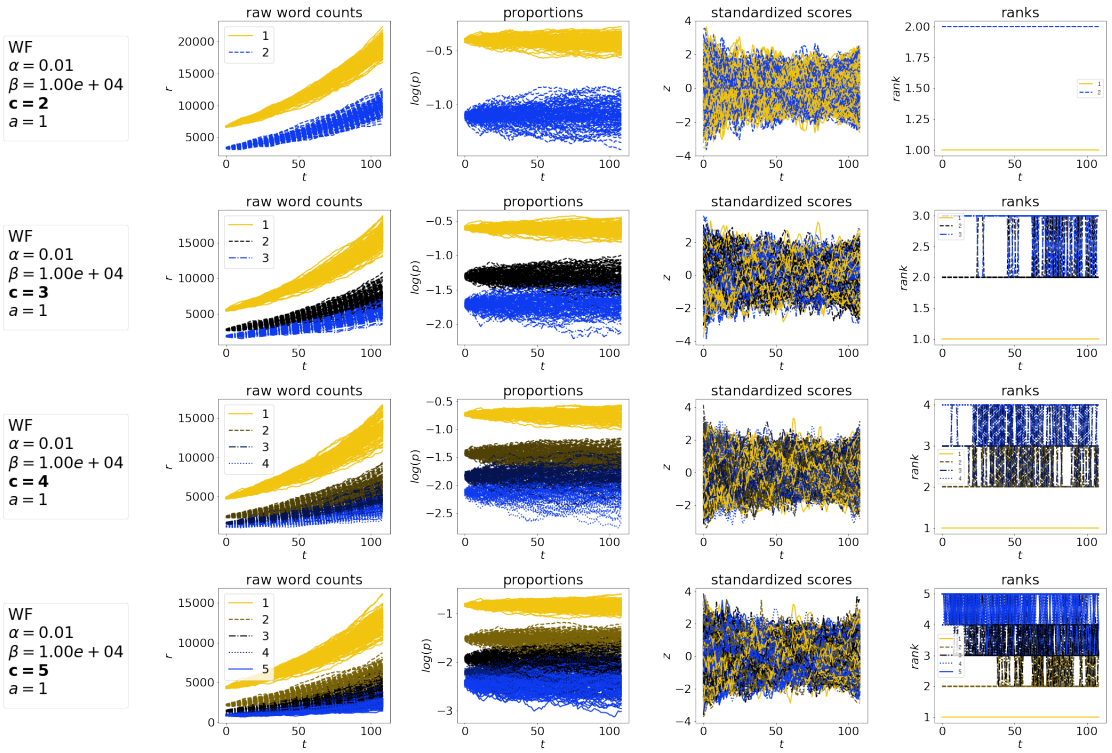
S1 Figure

Figure 5.1: **100 Simulations of the WF inspired model with β varied and fixed $\alpha = 0.01$, $c = 2$, and $a = 1$.** Looking the subplots below from top to bottom, the simulations for the largest initial corpus size showed that the words stayed in their initial ranks in time. In the smallest initial corpus size, there are outcomes where the words changed ranks in time.



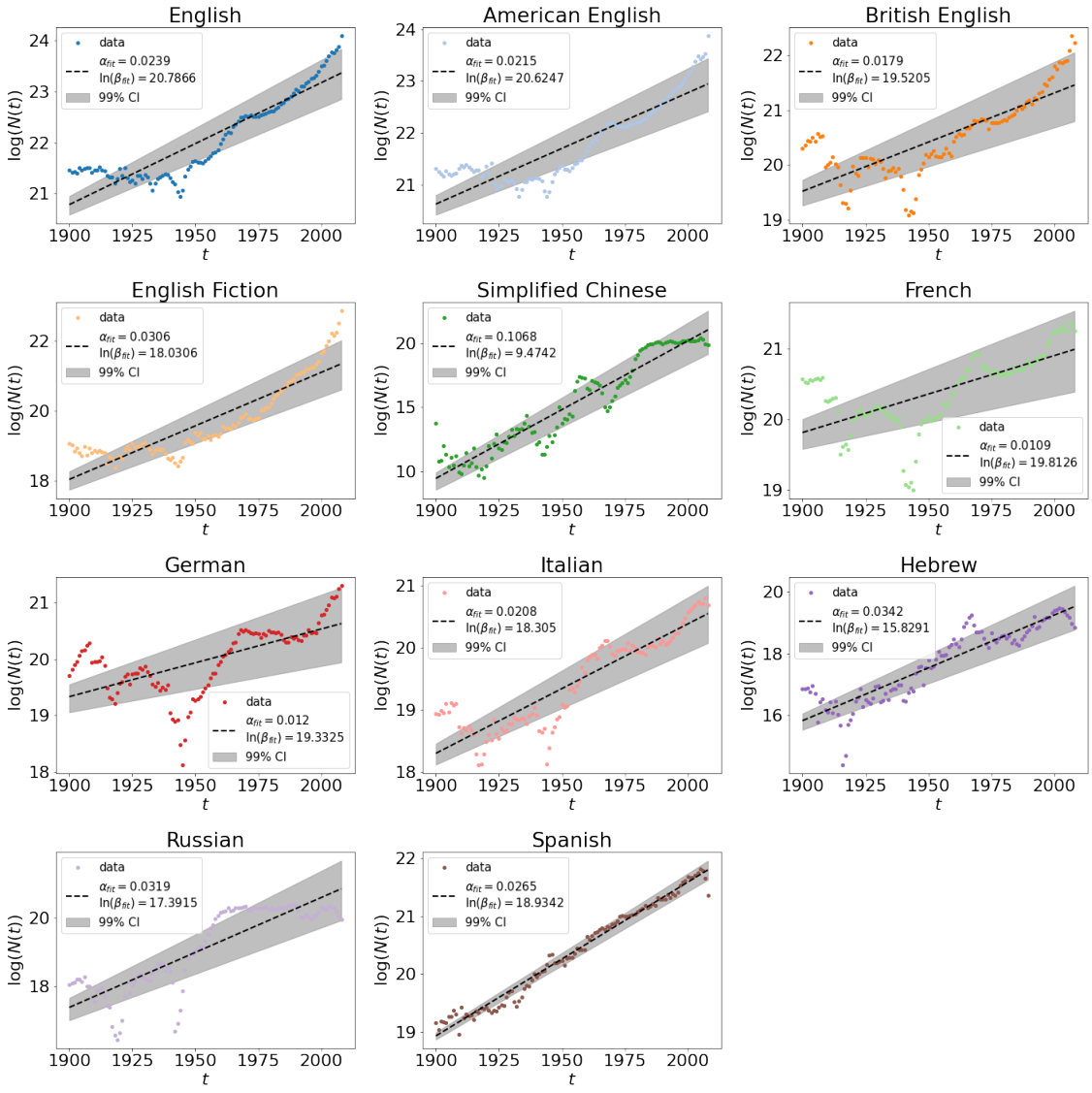
S2 Figure

Figure 5.2: **100 Simulations of the WF inspired model with c varied and fixed $\alpha = 0.01$, $\beta = 1.00 \times 10^4$, and $a = 1$.** Looking the subplots from top to bottom, the simulations for the increasing vocabulary size showed that there are outcomes for the lower ranked words that changed ranks in time.



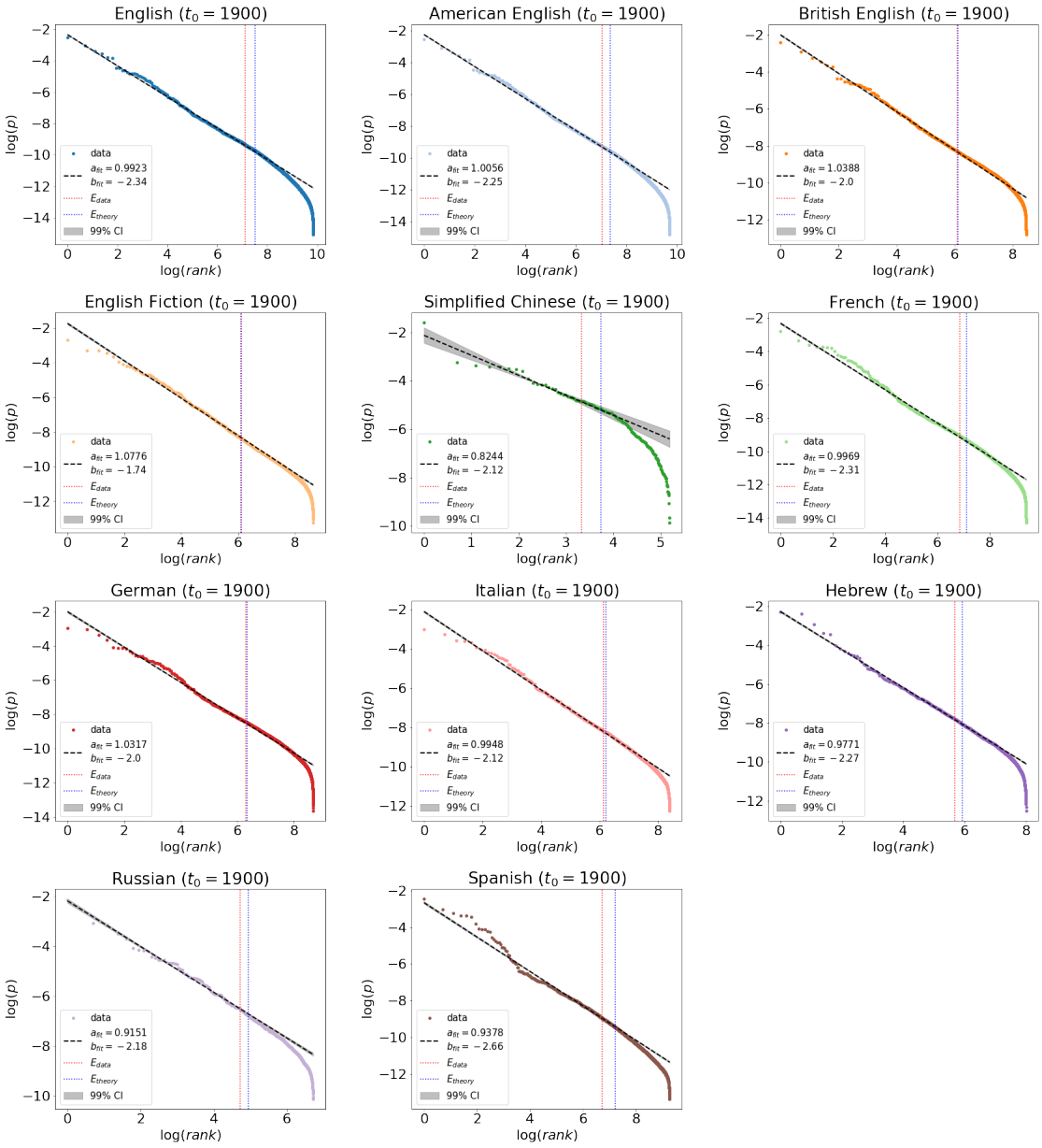
S3 Figure

Figure 5.3: **Log transformed corpus size function fits against the log transformed language data.** Each yearly corpus size for each language was used to estimate the parameters α and β in the corpus size function in Eq. 2.7. The parameter estimation of the log transformed exponential function was done using generalized least squares (see **??** for the details).



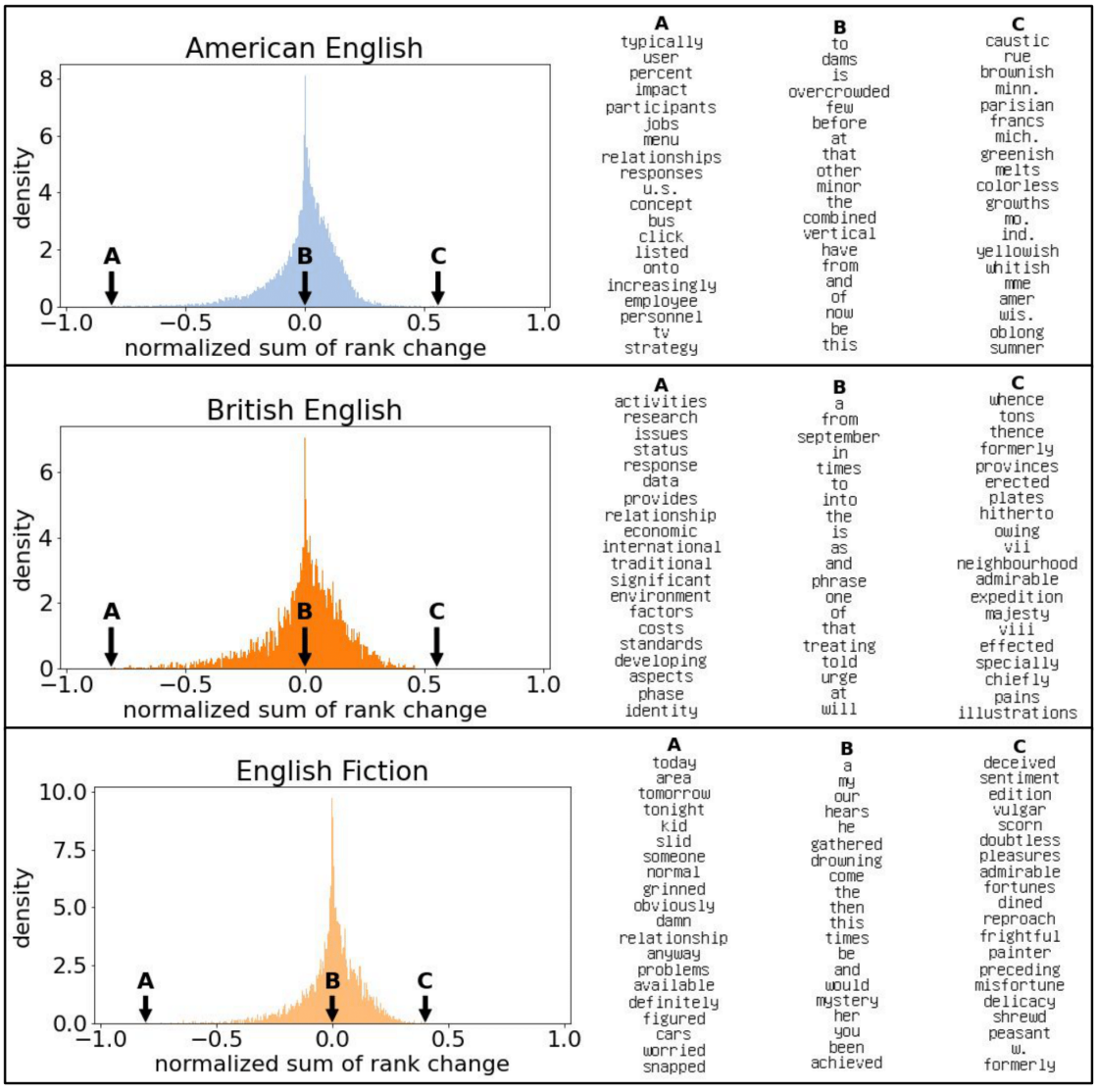
S4 Figure

Figure 5.4: **Log transformed Zipf function fits against the log transformed language data.** The initial rank distribution was used to estimate the shape parameter a of the Zipf probability mass function in Eq. 2.6. The data and the Zipf function are log transformed prior to fitting using generalized least squares method (see **??** for the details).



S5 Figure

Figure 5.5: **The sum of rank change distributions of the languages American English, British English, and English Fiction.** Each distribution below are the sums of the rank change of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the sum. The words in list **A** are words that changed up in ranks. The words in list **B** are words that have little or no change in ranks. The words in list **C** are words that changed down in ranks.



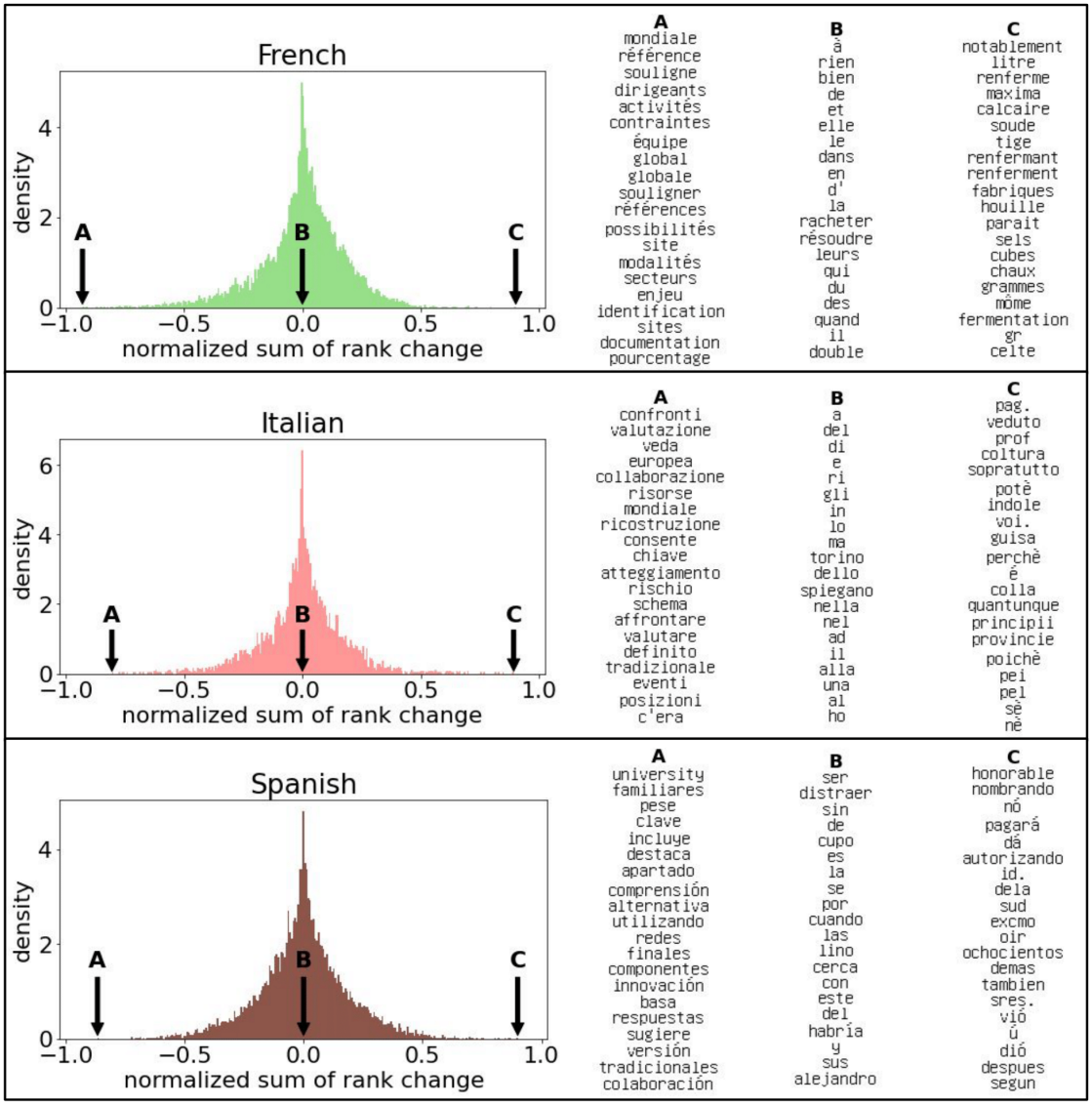
A	B	C
typically	to	caustic
user	dams	rue
percent	is	brownish
impact	overcrowded	minn.
participants	few	parisian
jobs	before	francs
menu	at	mich.
relationships	that	greenish
responses	other	melts
u.s.	minor	colorless
concept	the	growths
bus	combined	mo.
click	vertical	ind.
listed	have	yellowish
onto	from	whitish
increasingly	and	mme
employee	of	amer
personnel	now	wis.
tv	be	oblong
strategy	this	sumner

A	B	C
activities	a	whence
research	from	tons
issues	september	thence
status	in	formerly
response	times	provinces
data	to	erected
provides	into	plates
relationship	the	hitherto
economic	is	owing
international	as	vii
traditional	and	neighbourhood
significant	phrase	admirable
environment	one	expedition
factors	of	majesty
costs	that	viii
standards	treating	effected
developing	told	specially
aspects	urge	chiefly
phase	at	pains
identity	will	illustrations

A	B	C
today	a	deceived
area	my	sentiment
tomorrow	our	edition
tonight	hears	vulgar
kid	he	scorn
slid	gathered	doubtless
someone	drowning	pleasures
normal	come	admirable
grinned	the	fortunes
obviously	then	dined
damn	this	reproach
relationship	times	frightful
anyway	be	painter
problems	and	preceding
available	would	misfortune
definitely	mystery	delicacy
figured	her	shrewd
cars	you	peasant
worried	been	w.
snapped	achieved	formerly

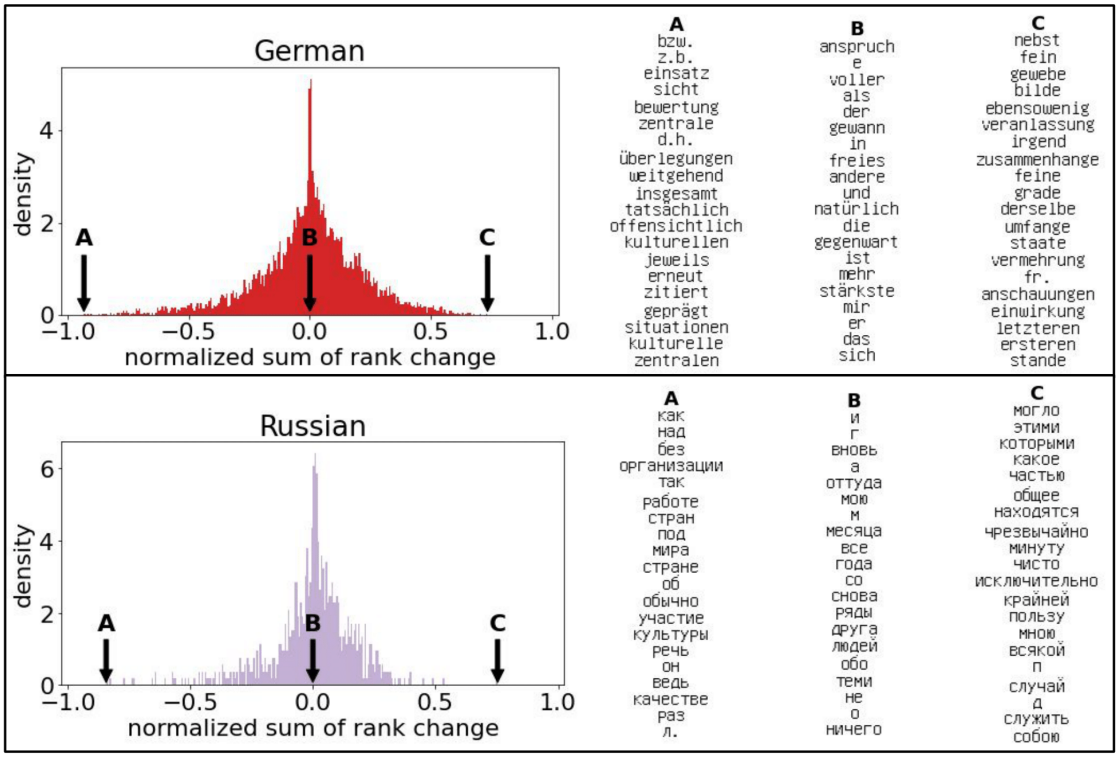
S6 Figure

Figure 5.6: **The sum of rank change distributions of the languages French, Italian, and Spanish.** Each distribution below are the sums of the rank change of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the sum. The words in list **A** are words that changed up in ranks. The words in list **B** are words that have little or no change in ranks. The words in list **C** are words that changed down in ranks.



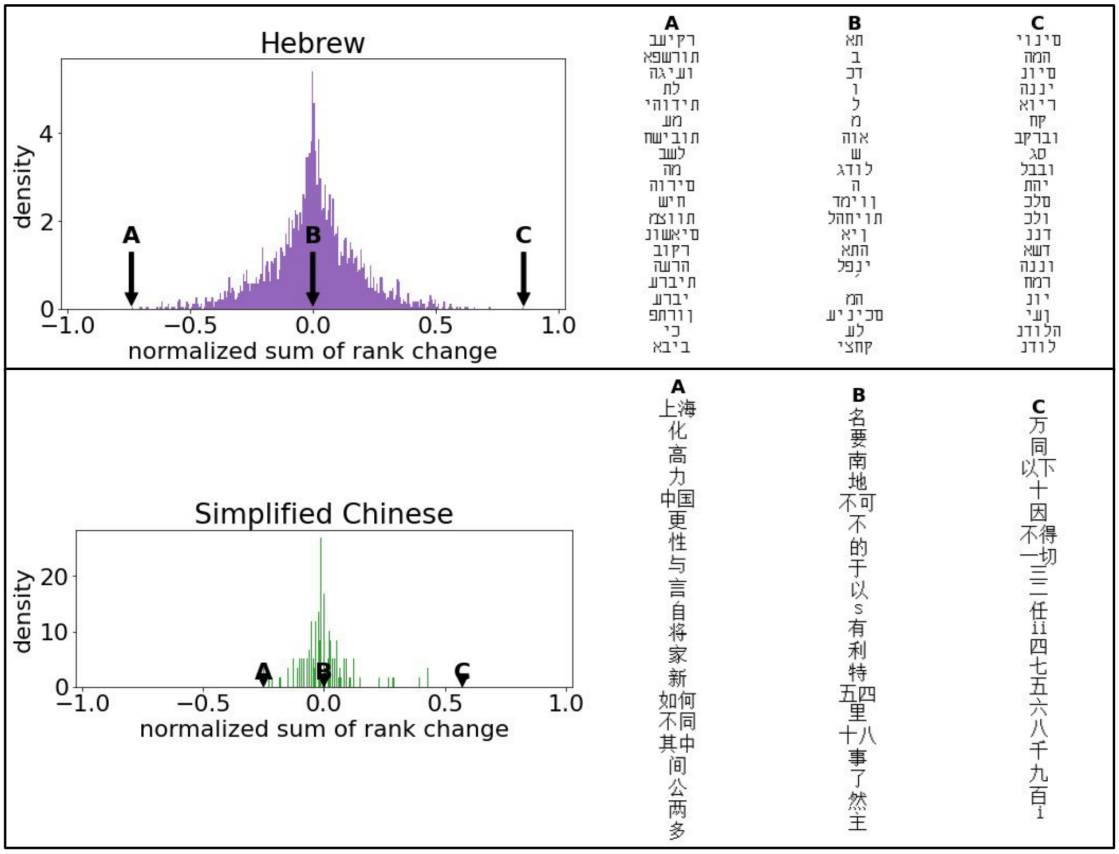
S7 Figure

Figure 5.7: **The sum of rank change distributions of the languages German and Russian.** Each distribution below are the sums of the rank change of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the sum. The words in list **A** are words that changed up in ranks. The words in list **B** are words that have little or no change in ranks. The words in list **C** are words that changed down in ranks.



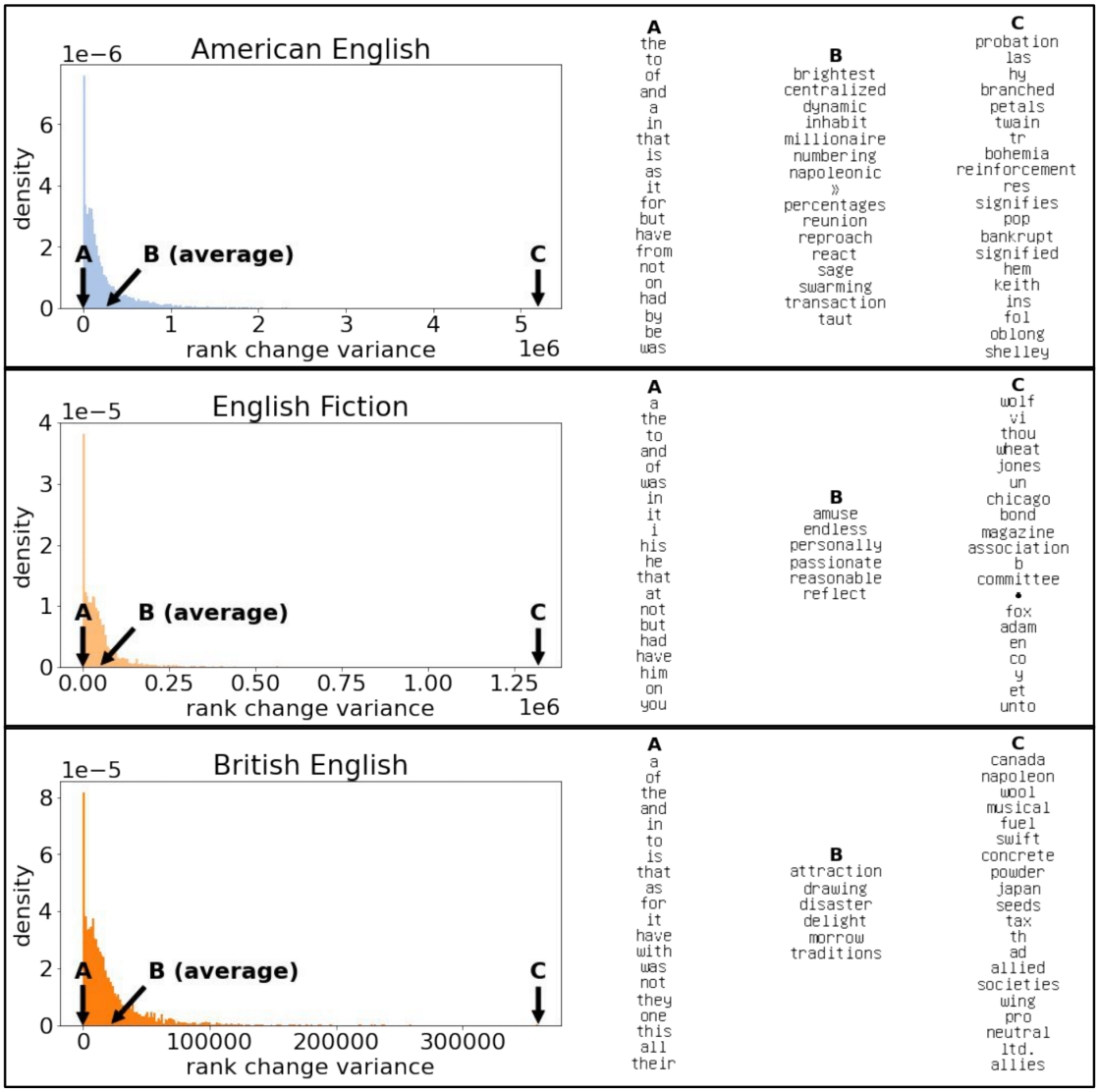
S8 Figure

Figure 5.8: **The sum of rank change distributions of the languages Hebrew and Simplified Chinese.** Each distribution below are the sums of the rank change of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the sum. The words in list **A** are words that changed up in ranks. The words in list **B** are words that have little or no change in ranks. The words in list **C** are words that changed down in ranks.



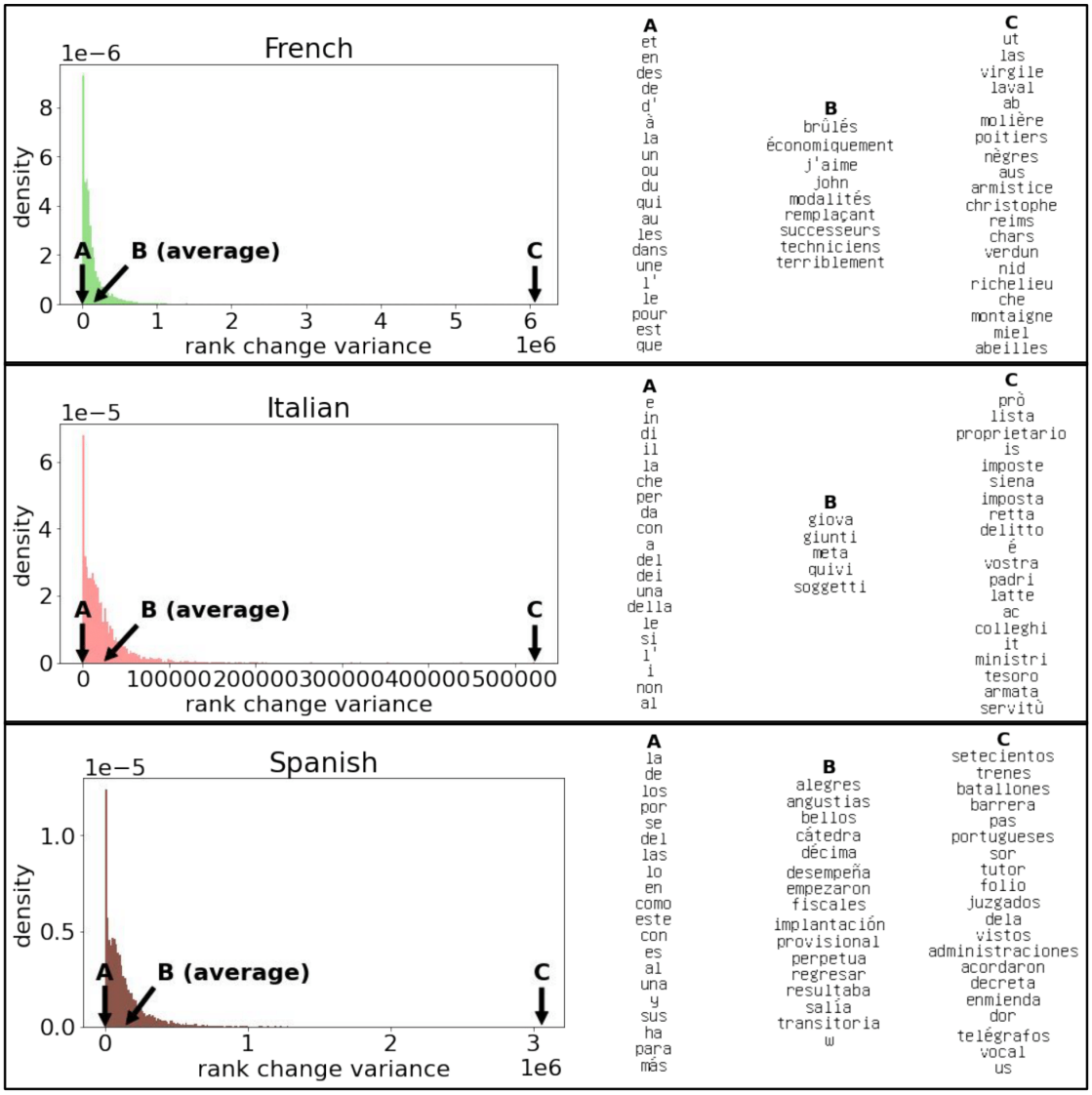
S9 Figure

Figure 5.9: **The rank change variance distributions of the languages American English, English Fiction, and British English.** Each distribution below are the rank change variances of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the variance. The words in list **A** are words that have little or no variance in their rank change. The words in list **B** are words with average variances. The words in list **C** are words that have high variances in their rank change.



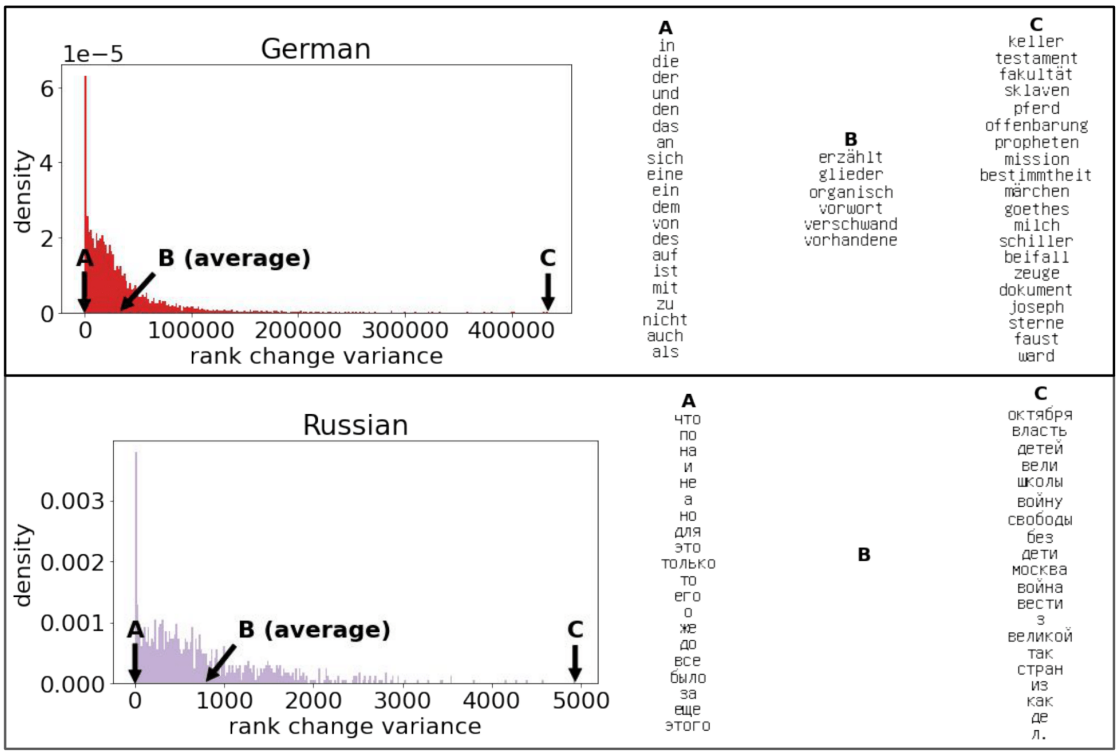
S10 Figure

Figure 5.10: **The rank change variance distributions of the languages French, Italian, Spanish.** Each distribution below are the rank change variances of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the variance. The words in list **A** are words that have little or no variance in their rank change. The words in list **B** are words with average variances. The words in list **C** are words that have high variances in their rank change.



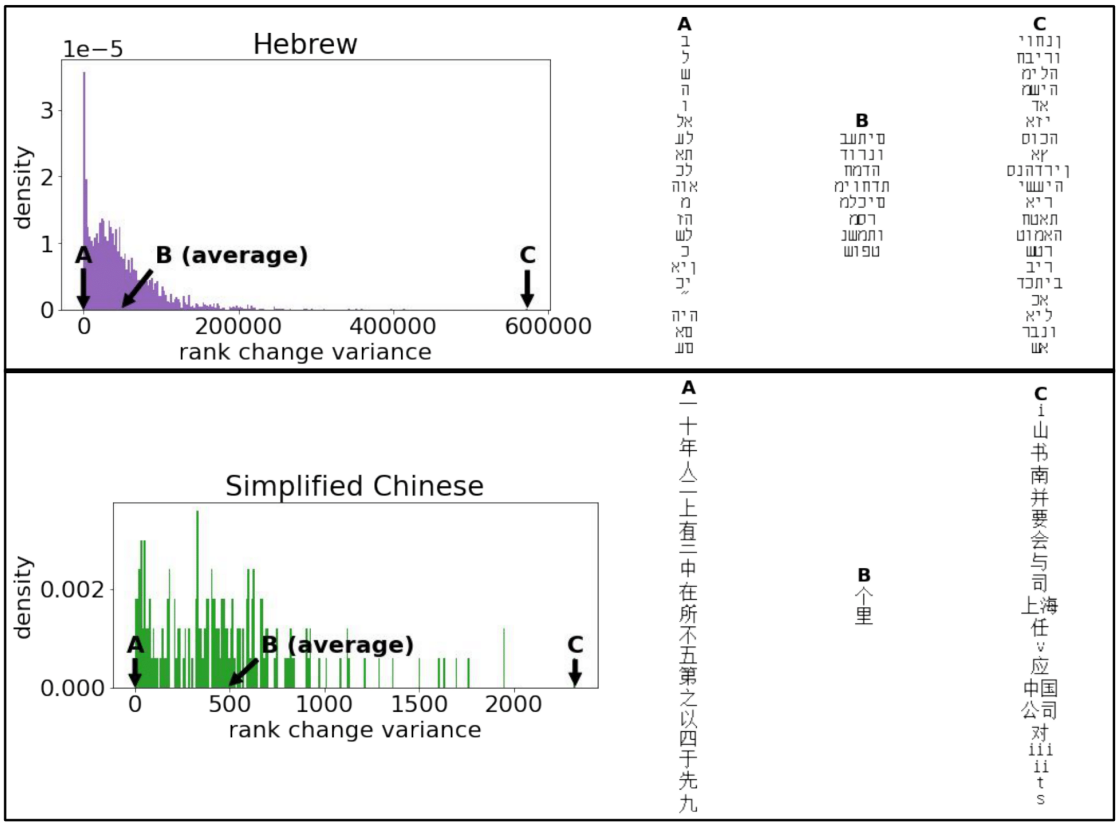
S11 Figure

Figure 5.11: **The rank change variance distributions of the languages German and Russian** Each distribution below are the rank change variances of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the variance. The words in list **A** are words that have little or no variance in their rank change. The words in list **B** are words with average variances. The words in list **C** are words that have high variances in their rank change.



S12 Figure

Figure 5.12: **The rank change variance distributions of the languages Hebrew and Simplified Chinese.** Each distribution below are the rank change variances of the words in the languages in the Google Ngram data. For each language, the distributions are annotated on the left tail (**A**), center (**B**), and right tail (**C**) of the distribution to show the list of words corresponding to the values of the variance. The words in list **A** are words that have little or no variance in their rank change. The words in list **B** are words with average variances. The words in list **C** are words that have high variances in their rank change.



S13 Appendix

To fit the corpus size time-series into the corpus size function, we use a log transform on Eq. 2.7 to make it linear.

$$\ln\left(N(t)\right) = \alpha t + \ln\left(\beta\right) \tag{5.1}$$

where $N(t)$ is the corpus size at time t, α is the rate of increase, and β is the initial corpus size.

To fit the initial frequencies into the Zipf probability mass function, we use a log

transform on Eq. 2.6.

$$\ln\left(P(Y^{theory})\right) = -a\ln\left(r_w\right) - \ln\left(\sum_{w=1}^{c}\left(1/r_w^a\right)\right) \tag{5.2}$$

where Y^{theory} is the random variable for the ranks, c is the vocabulary size, and r_w is the rank of word w. The above equation indicate that the log transform is a linear equation with slope $-a$ based on the first term. The second term is the intercept. We can rewrite this as

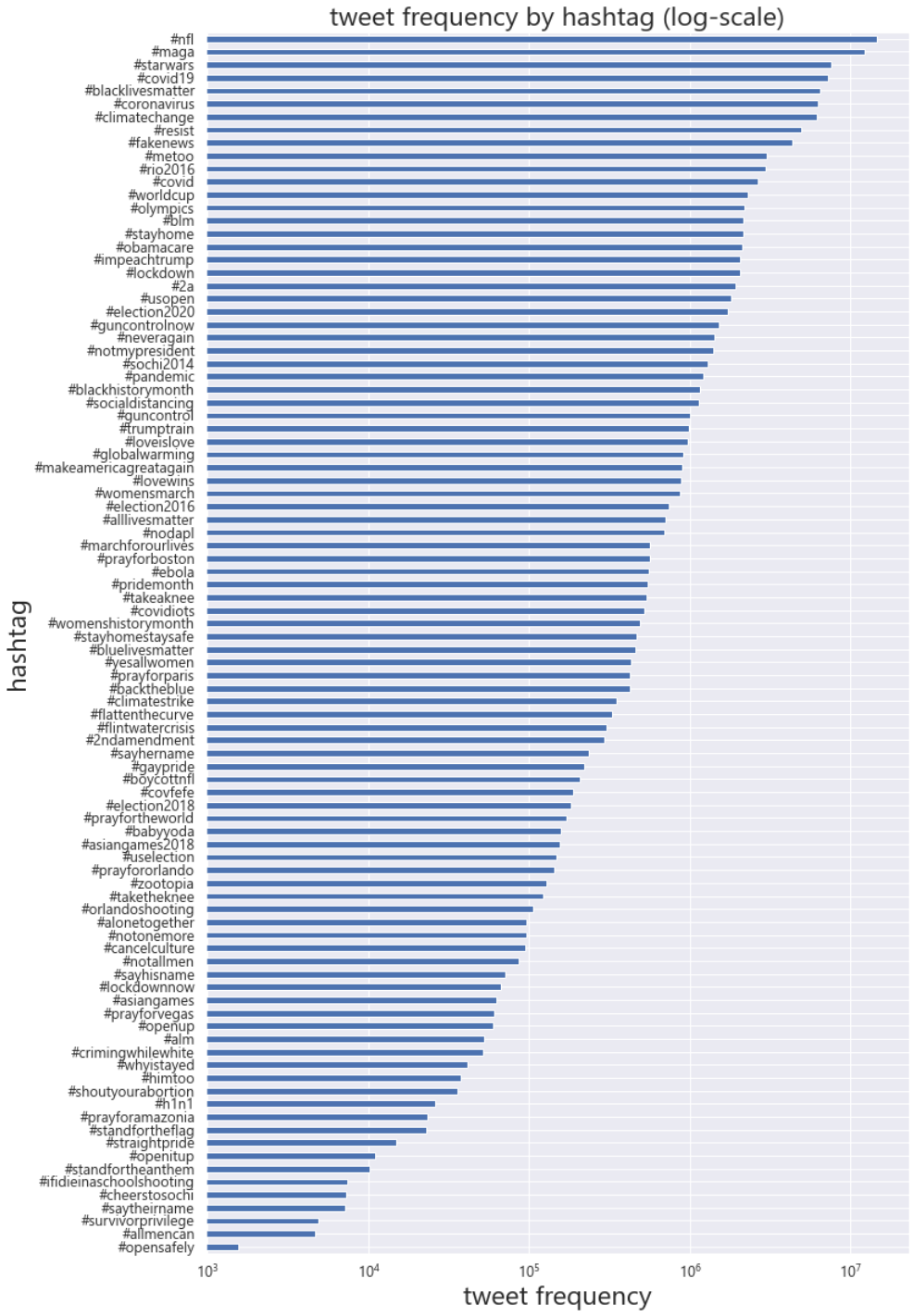
$$y = -a\ln\left(r_i\right) + b \tag{5.3}$$

where a and the b are the parameters we can estimate using the log-transformed data. The shape parameter a is always positive and b can be a negative number.

The linear models (Eq. 5.1 and 5.3) are the equations used to fit the corpus size time-series and the initial frequency distribution respectively using the *scipy.optimize.curve_fit* module in Python [1].

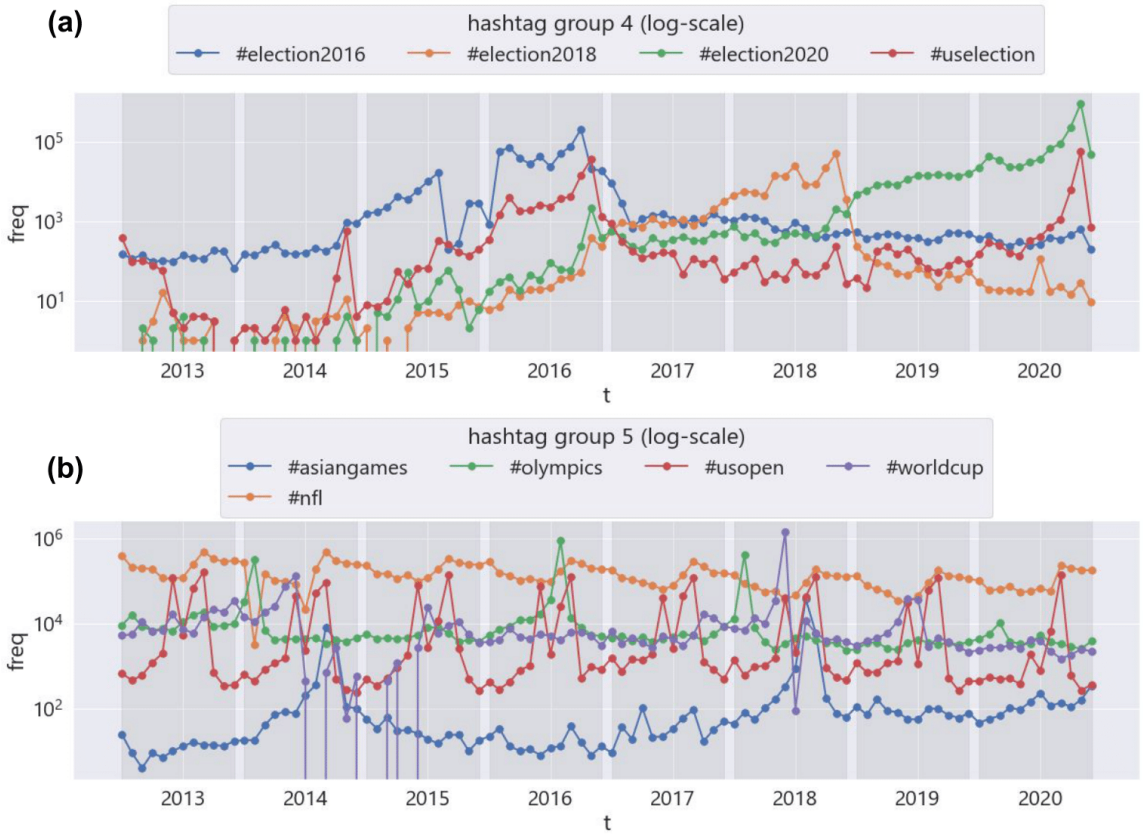
S14 Figure

Figure 5.13: **The log-scaled tweet frequency distribution of hashtags.** Each hashtag is used as a query to scrape tweets that contain the hashtag. The hashtag with the most tweets is #nfl and the hashtag with the lowest tweets is #opensafely.



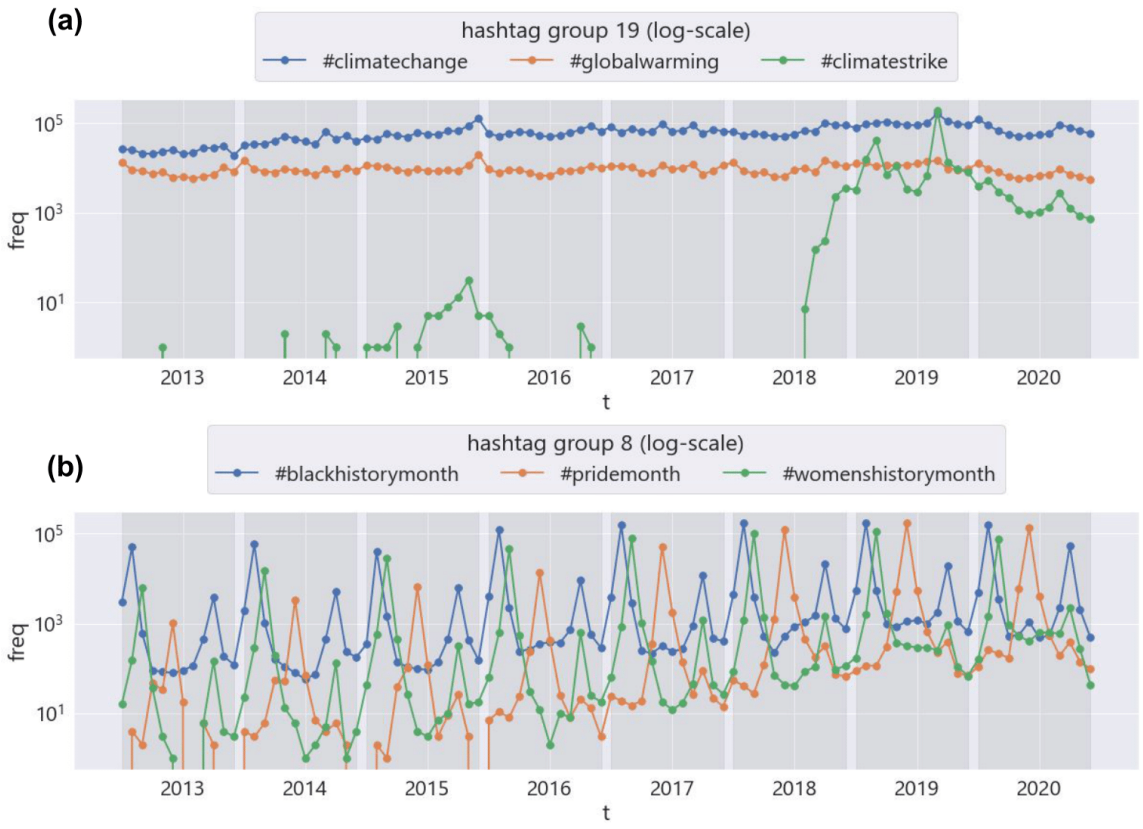
S15 Figure

Figure 5.14: **Tweet frequency time-series of hashtag groups 4 and 5.** Subfig. **(a)** shows the tweet frequency time-series of the us election related hashtags. We can see that - corresponding to the year shown on each hashtag except of #uselection - the tweet frequency reaches at its peak. Subfig. **(b)** shows the tweet frequency time-series of the sports event related hashtags. We can see that the frequency exhibits cycles corresponding to the sports events. For example, #nfl happens annually around September.



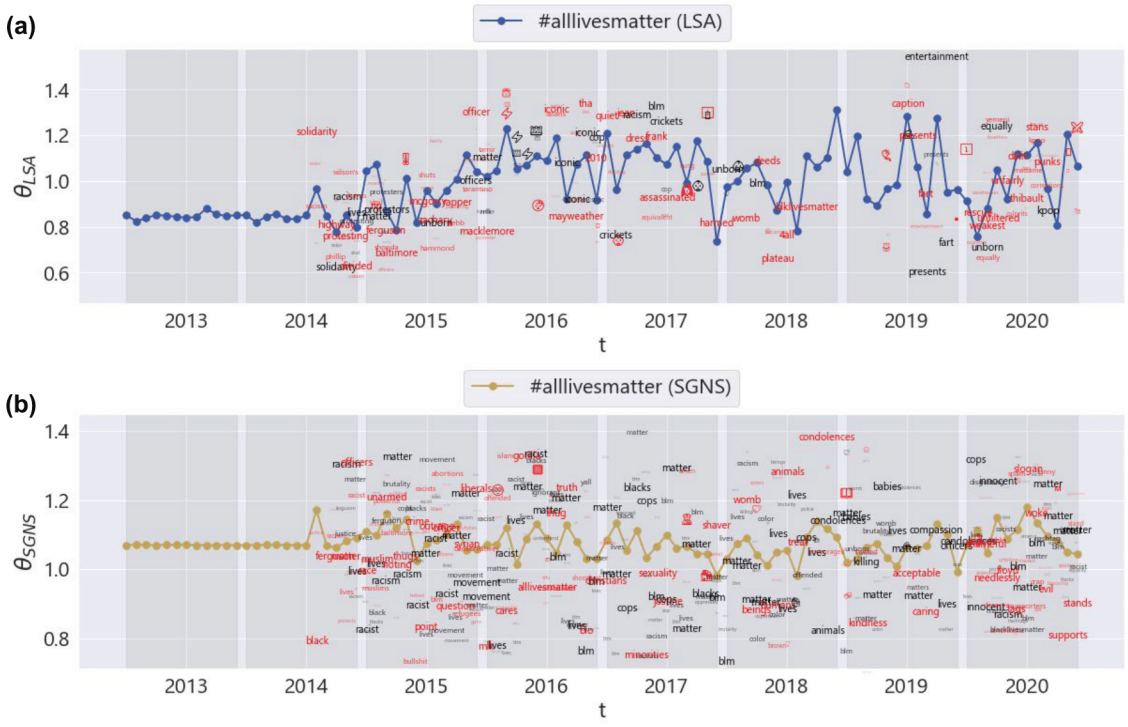
S16 Figure

Figure 5.15: **Tweet frequency time-series of hashtag groups 19 and 8.** Subfig. **(a)** shows the tweet frequency time-series of the climate change related hashtags. We can see that #climatechange and #globalwarming exhibits a constant trend while #climatestrike became popular in 2018. Subfig. **(b)** shows the tweet frequency time-series of the #"month" related hashtags. We can see that the frequency exhibits cycles corresponding to the annual event. For example, #blackhistorymonth happens annually in February and we can see peaks at those time-points very year.



S17 Figure

Figure 5.16: **Word path of the temporal contextual semantics of #alllivesmatter**. Using the LSA (Subfig. **(a)**) and SGNS (Subfig. **(b)**) models, we show the time-series path of the word angular position of the #alllivesmatter. The word path taken of #alllivesmatter is a series of words where it encounters a novel word (colored in red) and a repeated word (colored in black). We see in the word path of #alllivesmatter with words "racism", "racist", "cops", and "officers". The #alllivesmatter is a hashtag used by the All Lives Matter discourse - which is a response to the #blacklivesmatter of the Black Lives Matter movement - arguing that all lives - not just black lives - matter, and the hashtag calls for racial equality. The supporters of Black Lives Matter movement sometimes uses the #alllivesmatter hashtag to critique it. They argue that #alllivesmatter rhetoric ignores and dismisses the embedded racism in society, existing laws, and law enforcement - while not fully understanding the main purpose of the #blacklivesmatter.



S18 Figure

Figure 5.17: **Word path of the temporal contextual semantics of #himtoo**. Using the LSA (Subfig. **(a)**) and SGNS (Subfig. **(b)**) models, we show the time-series path of the word angular position of the hashtag #himtoo. The word path taken of #metoo is a series of words where it encounters a novel word (colored in red) and a repeated word (colored in black). We see in the word path of #himtoo with words like “accusations”, “publicity”, “misconduct”, “innocent”, “sexually”, “rape”, and ”predator”. The #himtoo is a hashtag used by the Him Too movement - which is a response to the #metoo of the Me Too movement - arguing that men are wrongfully accused of sexual harassment and “him too” can also be a victim of sexual assault.