

UVM ScholarWorks

Measuring Linguistic and Cultural Evolution Using Books and Tweets

Item Type	dissertation;article
Authors	Gray, Tyler
Download date	2026-05-08 14:36:50
Link to Item	https://hdl.handle.net/20.500.14849/3040

MEASURING LINGUISTIC AND CULTURAL EVOLUTION USING
BOOKS AND TWEETS

A Dissertation Presented

by

Tyler John Gray

to

The Faculty of the Graduate College

of

The University of Vermont

In Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
Specializing in Mathematical Sciences

August, 2019

Defense Date: June 7, 2019

Dissertation Examination Committee:

Christopher M. Danforth, Ph.D., Advisor

Peter Sheridan Dodds, Ph.D., Advisor

Jacques A. Bailly, Ph.D., Chairperson

Laurent Hébert-Dufresne, Ph.D.

Cynthia J. Forehand, Ph.D., Dean of the Graduate College

© Copyright by
Tyler John Gray
August, 2019
All rights reserved.

ABSTRACT

Written language provides a snapshot of linguistic, cultural, and current events information for a given time period. Aggregating these snapshots by studying many texts over time reveals trends in the evolution of language, culture, and society. The ever-increasing amount of electronic text, both from the digitization of books and other paper documents to the increasing frequency with which electronic text is used as a means of communication, has given us an unprecedented opportunity to study these trends. In this dissertation, we use hundreds of thousands of books spanning two centuries scanned by Google, and over 100 billion messages, or ‘tweets’, posted to the social media platform, Twitter, over the course of a decade to study the English language, as well as study the evolution of culture and society as inferred from the changes in language.

We begin by studying the current state of verb regularization and how this compares between the more formal writing of books and the more colloquial writing of tweets on Twitter. We find that the extent of verb regularization is greater on Twitter, taken as a whole, than in English Fiction books, and also for tweets geotagged in the United States relative to American English books, but the opposite is true for tweets geotagged in the United Kingdom relative to British English books. We also find interesting regional variations in regularization across counties in the United States. However, once differences in population are accounted for, we do not identify strong correlations with socio-demographic variables.

Next, we study stretchable words, a fundamental aspect of spoken language that, until the advent of social media, was rarely observed within written language. We examine the frequency distributions of stretchable words and introduce two central parameters that capture their main characteristics of balance and stretch. We explore their dynamics by creating visual tools we call ‘balance plots’ and ‘spelling trees’. We also discuss how the tools and methods we develop could be used to study mistypings and misspellings, and may have further applications both within and beyond language.

Finally, we take a closer look at the English Fiction n -gram dataset created by Google. We begin by explaining why using token counts as a proxy of word, or more generally, ‘ n -gram’, importance is fundamentally flawed. We then devise a method to rebuild the Google Books corpus so that meaningful linguistic and cultural trends may be reliably discerned. We use book counts as the primary ranking for an n -gram and use subsampling to normalize across time to mitigate the extraneous results created by the underlying exponential increase in data volume over time. We also combine the subsampled data over a number of years as a method of smoothing. We then use these improved methods to study linguistic and cultural evolution across the last two centuries. We examine the dynamics of Zipf distributions for n -grams by measuring the churn of language reflected in the flux of n -grams across rank boundaries. Finally, we examine linguistic change using wordshift plots and a rank divergence measure with a tunable parameter to compare the language of two different time periods. Our results address several methodological shortcomings associated with the raw Google Books data, strengthening the potential for cultural inference by word changes.

CITATIONS

Material from this dissertation has been published in the following form:

Gray, T. J., Reagan, A. J., Dodds, P. S., and Danforth, C. M.. (2018). English verb regularization in books and tweets. *PLOS ONE*, 13(12):1–17.

AND

Material from this dissertation has been submitted for publication to *PLOS ONE* on July 9, 2019 in the following form:

Gray, T. J., Danforth, C. M., and Dodds, P. S.. (2019). Hahahahaha, Duuuuude, Yeeessss!: A two-parameter characterization of stretchable words and the dynamics of mistypings and misspellings. Manuscript submitted for publication.

In loving memory of Mémère,

Louise Bissonnette (1929 – 2018),

who wanted to see me get my doctorate, but didn't quite make it.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisors, Chris Danforth and Peter Dodds, not only for their mentorship while I obtained my Ph.D., but going back through to when we first met while I was still but a wee high school student. They welcomed me into their group, providing continued guidance and friendship throughout the journey, even as I ventured off studying elsewhere, and were happy to have me return and see me through my graduate years. My growth as a scientist and where I am today is in large part due to them. I would also like to thank Jacques Bailly and Laurent Hébert-Dufresne for agreeing to be a part of my defense committee and for their comments and suggestions.

I would like to thank Jake Williams and Cathy Bliss for the examples and template from which this dissertation has been created, Cheryl Ponchin for her dedicated and persistent help in solving the hardest of puzzles provided by the wonderful \LaTeX , Margaret Caron for her general support and her dedicated review and edits of this document, Kelsey Linnell for the forced distractions and enjoyable conversations throughout my graduate years, David Dewhurst for the friendship and the interesting conversations, Karen for the delightful conversations, interesting books, and support at my defense, Jeff Dinitz for the great years at GIV and the support at my defense, Rosi for the guidance as my teaching coordinator and for the continued support and encouragement, and Andi Elledge and Melissa Rubinchuck for keeping Farrell (both the building and the people within) from falling apart.

I would like to thank my teachers through the years, especially Margaret Lima, for being a great lady and for always being there to help in many ways, and Jim Grittner, for the tremendous help providing opportunities for my advancement in mathematics and the engaging and pleasurable discussions over the years.

Finally, I am extremely grateful of my family, both immediate and extended, for all of their support through the years. I would like to especially thank my brother Tim for getting me started in mathematics so many years ago and for teaching me everything he knew, my

sister Melissa for forcing me to go to Governor's Institute, which has led to many great things, including meeting Prof. Danforth, my sister Kristina for the general support and especially for my favorite sweater that she made for me, which always receives compliments every time I wear it and will keep me warm for the rest of my life, and finally my parents, for (besides simply everything) their unconditional support through the years and for the great values they have taught me through the years, including the value of hard work, which has helped me get to where I am now.

TABLE OF CONTENTS

Citations	ii
Dedication	iii
Acknowledgments	iv
List of Figures	viii
List of Tables	xii
1 Introduction	1
1.1 Description of the datasets	1
1.1.1 Google Books	2
1.1.2 Twitter	3
1.2 Verb regularization	4
1.3 Stretchable words	6
1.4 Unlocking Google Books	7
2 English verb regularization in books and tweets	9
2.1 Introduction	10
2.2 Description of datasets	13
2.3 Methods and results	16
2.3.1 Verb regularization using Ngrams and Twitter	16
2.3.2 American and British English	18
2.3.3 Regularization by U.S. county	21
2.4 Concluding remarks	31
2.5 Acknowledgments	32
2.6 References	32
2.7 Appendix	34
2.7.1 Table of verb forms	34
2.7.2 Details on user location matching	40
2.7.3 Average verb regularization for verbs binned by data volume	41
2.7.4 User tweet frequency count	42
3 Hahahahaha, Duuuuude, Yeeessss!: A two-parameter characterization of stretchable words and the dynamics of mistypings and misspellings	43
3.1 Introduction	43
3.2 Description of the dataset and method for extracting stretched words	45
3.3 Analysis and results	50
3.3.1 Distributions	50
3.3.2 Balance	52
3.3.3 Stretch	59
3.3.4 Spelling trees	62
3.3.5 Mistypings and misspellings	65
3.4 Concluding remarks	69

3.5	Acknowledgments	71
3.6	References.	71
3.7	Appendix	73
3.7.1	Alternate balance measure	73
3.7.2	Stretch ratio	78
3.7.3	“Drawing presentable trees” algorithmic bugs	81
4	Culturomics Redux: Rebuilding the Google Books n-gram dataset to reveal the hidden collective voice of English fiction authors	83
4.1	Introduction	84
4.2	Description of the dataset	86
4.3	The problem with using token counts	87
4.4	Unlocking the secrets	94
4.4.1	Using book counts to rank n -grams	94
4.4.2	Subsampling and smoothing	98
4.5	Lexical turbulence	103
4.5.1	Zipf distributions	103
4.5.2	Rank flux	105
4.5.3	Detailed analysis of language changes with rank divergence wordshift plots	107
4.6	Concluding remarks	122
4.7	Acknowledgments	126
4.8	References.	126
5	Concluding remarks	128
	Bibliography	130

LIST OF FIGURES

2.1	Relative word frequencies for the irregular and regular past verb forms for ‘burn’ during the 19th and 20th centuries.	11
2.2	Comparison of verb regularization for Ngrams and Twitter.	17
2.3	American and British English verb regularization fractions for (A) Ngrams and (B) Twitter.	19
2.4	Differences in verb regularization fractions.	20
2.5	(A) The average verb regularization fraction by county for the lower 48 states, along with (B) residuals and (C) Gi^* z -score.	23
2.6	The Gi^* z -score for verb regularization by county for the verb ‘dream’ for the lower 48 states.	25
2.7	Scatter plot of average verb regularization for (A) counties and (B) synthetic counties.	27
2.8	(A) Average verb regularization and (B) average verb regularization residuals for counties as a function of the percentage of civilians employed in agriculture, forestry, fishing, hunting, and mining.	30
2.A1	The scatter plot of average binned verb regularization for counties.	41
2.A2	Frequency counts of tweets by unique users in our Twitter decahose dataset.	42
3.1	Token count distribution for the kernel (to).	48
3.2	Total token counts for stretched versions of all kernels.	49
3.3	Token count distribution for the kernel [g][o][a][l].	51
3.4	Token count distribution for the kernel (ha).	51
3.5	Balance plot for the kernel [g][o][a][l].	53
3.6	Balance plot for the kernel (ha).	55
3.7	Jellyfish plots for kernel balance for (A) all kernels, and (B) excluding kernels with entropy exactly zero.	56

3.8	Jellyfish plots for kernel stretch as measured by the Gini coefficient, G , of its token count distribution.	59
3.9	Kernels plotted in Balance-Stretch parameter space.	60
3.10	Spelling tree for the kernel (ha).	63
3.11	A collection of example spelling trees.	66
3.12	(A) Token count distribution, (B) balance plot, and (C) spelling tree for the kernel n[o](io).	67
3.13	Spelling tree for the kernel hear(ta)ck.	68
3.A1	Balance plot for the kernel (pa).	74
3.A2	Jellyfish plots for kernel balance based on an alternate entropy measure for (A) all kernels, and (B) excluding kernels with entropy exactly zero.	75
3.A3	Kernels plotted in Balance-Stretch parameter space using an alternate measure of normalized entropy for balance.	77
3.A4	Jellyfish plots for kernel stretch ratio, ρ , as given by the ratio of the sum of the kernel's stretched tokens to the sum of its unstretched tokens.	78
3.A5	Scatter plot comparing two measures of stretch for each kernel.	79
4.1	Yearly book counts for the Google Books English Fiction 2012 corpus.	88
4.2	Zipf distributions for (A) 2-grams and (B) 3-grams based on token counts in the Google Books English Fiction 2012 corpus.	89
4.3	The top contributors to the Jensen-Shannon divergence (JSD) between subsampled frequency distributions of 2-grams for the periods 1934–1943 and 1944–1953.	90
4.4	Relative token frequency time series for the bigram ‘Raintree County’.	91
4.5	The top contributors to the JSD between the relative token frequency distributions for 3-grams in 1979 and 1980.	92
4.6	Relative token frequency time series for a select collection of 3-grams.	93
4.7	A comparison of token count rank and composite rank for bigrams in 1948.	95
4.8	Relative token frequency time series for the 1-grams ‘Lanny’ and ‘Hitler’.	97

4.9	Relative book frequency time series for the 1-grams ‘Lanny’ and ‘Hitler’ . . .	97
4.10	Flux time series for 1-grams.	98
4.11	Flux for 1-grams between consecutive years using the sampled version of the data.	101
4.12	Flux for 1-grams using the sampled, combined data.	102
4.13	Zipf distributions for (A) 1-grams, (B) 2-grams, and (C) 3-grams based on book counts in our subsampled version of the data.	104
4.14	Rank flux for (A) 1-grams, (B) 2-grams, and (C) 3-grams using the sampled, combined data comparing distributions combined over 10 years and separated by 10 years.	106
4.15	The top 60 1-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	109
4.16	The top 60 1-grams contributing to rank divergence with $\alpha = 0.2$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	110
4.17	The top 60 1-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	111
4.18	The top 60 2-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	113
4.19	The top 60 2-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	114
4.20	The top 60 3-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	115
4.21	The top 60 3-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935.	116

4.22	The top 60 1-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990.	118
4.23	The top 60 1-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990.	119
4.24	The top 60 2-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990.	120
4.25	The top 60 2-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990.	121
4.26	The top 60 3-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990.	123
4.27	The top 60 3-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990.	124
4.28	The top 60 3-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1947 to that starting in 1957.	125

LIST OF TABLES

2.1	A summary of the verb datasets.	15
2.2	A summary of the average regularization fractions for AE and BE on Twitter and Ngrams.	21
2.3	Top demographic variables sorted by the magnitude of their partial correlation with verb regularization in U.S. counties.	29
2.A1	A tabulation of all verb forms in this study.	34
3.1	Examples of distilling tokens down to their kernels.	46
3.2	Top 10 kernels by normalized entropy, H	58
3.3	Bottom 10 kernels by normalized entropy, H	58
3.4	Top 10 kernels by Gini coefficient, G	61
3.5	Bottom 10 kernels by Gini coefficient, G	61
3.A1	Top 10 kernels by an alternate normalized entropy, H_{alt}	76
3.A2	Bottom 10 (nonzero) kernels by an alternate normalized entropy, H_{alt}	77
3.A3	Top 10 kernels by stretch ratio, ρ	80
3.A4	Bottom 10 kernels by stretch ratio, ρ	80

CHAPTER 1

INTRODUCTION

Human language is in a constant state of evolution, partly driven both by random drift (Newberry et al., 2017) and by cultural and societal factors. Words, or more generally, ‘*n*-grams’, are the building blocks of language, and changes in their meanings and usage patterns reflect, and provide insight into, cultural and societal evolution. The increase in digitized text, both current and historical, allows for the large-scale analysis of language, both of its current state and of its evolution, which, in turn, provides a lens through which to observe current and changing sociocultural trends.

In this dissertation, we use both hundreds of thousands of digitized books and over 100 billion social media messages to study various aspects of the English language, and use this to gain insights into cultural evolution. We begin with a study of verb regularization, followed by an analysis of stretchable words, and finish with an investigation into the lexical turbulence of language. The remainder of this chapter provides an overview of our data and each of these three studies.

1.1 DESCRIPTION OF THE DATASETS

In this paper, we use subsets of two different large corpora: (1) Two centuries of published books scanned by Google (1800–2008), and (2) a decade of social media messages posted to Twitter (2008–2017).

1.1.1 GOOGLE BOOKS

Google underwent a process whereby they scanned millions of books and digitized them using optical character recognition (OCR), creating the Google Books corpus (Michel et al., 2011). They chose a subset of these digitized books based on their metadata and the quality of their OCR and split the texts into ‘ n -grams’, creating the Google Books n -grams datasets (Lin et al., 2012; Michel et al., 2011). A 1-gram is often, but not always, the same as a word. A punctuation mark is a 1-gram. Google also chose to split certain words apart into multiple 1-grams. For example, possessives are split off from the word as a separate 1-gram. ‘Bob’s’ would get split into the two 1-grams ‘Bob’ and ‘s’. Similarly, words like ‘can’t’ get split into the two 1-grams ‘can’ and ‘not’. Hyphenated words get split into three 1-grams, with the hyphen, ‘-’, being one of them. An ‘ n -gram’ is made up of n 1-grams. For example, ‘on the way’ is a 3-gram made up of the three 1-grams ‘on’, ‘the’, and ‘way’, and can be split into the two 2-grams ‘on the’ and ‘the way’.

A ‘token’ is an individual occurrence of a word or n -gram. For example, the phrase ‘the cat and the dog’ is a 5-gram made up of five tokens but four words, or four individual 1-grams, with the 1-gram ‘the’ showing up twice, giving it a token count of two. The other three 1-grams each have a token count of one.

For each n -gram that occurs within the corpus at least 40 times overall, the total number of individual occurrences for that n -gram in each year (token count) and the total number of books it was found in for each year (book count) were recorded by Google and are freely available to the public (The Google Ngram Viewer Team, 2013c). A number of recent studies have used the Google Books data to study different aspects of language and culture (Gerlach and Altmann, 2013; Gray et al., 2018; Michel et al., 2011; Pechenick et al., 2017; Petersen et al., 2012a,b).

However, a closer inspection of the data has identified troubling issues with the Google Books n -grams corpus. There is a large prevalence of scientific works in both the 2009 and

2012 versions of the English corpus and in the 2009 version of the English Fiction corpus (Pechenick et al., 2015). Furthermore, due to its library-esque nature, where each book contributes roughly one time (not counting reprints and different editions), the data fails to reflect information about the popularity of n -grams based on readership views (Pechenick et al., 2015).

Despite these setbacks, as we show in Chapter 4, the Google Books n -grams data still contains a wealth of information from which we are able to extract meaningful results if we keep the preceding limitations in mind. In this dissertation, when studying English in general, we use the English Fiction 2012 corpus as suggested by Pechenick et al. (2015), which uses “books predominantly in the English language that a library or publisher identified as fiction.” In order to allow for the study of regional variations in language, we also use both the American English 2012 corpus, which uses “books predominantly in the English language that were published in the United States,” and the British English 2012 corpus, which uses “books predominantly in the English language that were published in Great Britain” (The Google Ngram Viewer Team, 2013a).

1.1.2 TWITTER

Twitter is a social media site that allows people to post short messages, called ‘tweets’, of originally up to 140 characters in length, and more recently, up to 280 characters in length. Although not all demographic groups are equally represented, in 2015 20% of adult Americans used Twitter (Duggan, 2015), providing a large sample of English language data of a more colloquial nature ripe for linguistic analysis. As such, in recent years there have been a number of papers studying different aspects of language with Twitter data (Donoso and Sanchez, 2017; Eisenstein et al., 2010, 2014; Gonçalves et al., 2018; Gonçalves and Sánchez, 2014; Gray et al., 2018; Huang et al., 2016).

Since September of 2008 we have collected a 10% random sample of tweets from Twitter’s ‘gardenhose’ (now called ‘decahose’) dataset, totaling over 100 billion tweets from over 750 million distinct accounts. Furthermore, a fraction of these tweets have corresponding location data, allowing us to perform regional analyses of language. In this dissertation, we use this Twitter dataset as a representation of a more colloquial form of language, closer to that of everyday speech than the edited language found in books. In various parts of our analysis we use (1) all tweets, (2) tweets tagged with the geographic coordinates of the location the tweet is sent from (‘geotagged’) when the coordinates are located in the United States to study American English, (3) tweets geotagged in the United Kingdom to study British English, and (4) tweets with user provided location information, entered as free text, matching towns in the United States to study regional variations in language across U.S. counties.

1.2 VERB REGULARIZATION

While many aspects of language variation and change are being investigated in the pursuit to further our collective understanding of language (Greenhill et al., 2017; Lieberman et al., 2007; Michel et al., 2011; Newberry et al., 2017; Ramiro et al., 2018; Reali et al., 2018), one particular area of study has been English verb regularization (Lieberman et al., 2007; Michel et al., 2011; Newberry et al., 2017). These studies have shown that English verbs are going through a process of regularization, where English speakers are using the regular form for the past tense of a verb, formed with the suffix -ed, rather than the original irregular past tense form.

For example, the irregular past tense of the verb ‘burn’ is ‘burnt’ and the regular past tense is ‘burned’. As another example, the verb ‘help’ had the irregular past tense ‘holp’, compared to the regular past ‘helped’. Although for some verbs, like ‘burn’, the irregular and regular past tense both seem reasonable, for others, like ‘help’, the original irregular

form is no longer recognizable to most people and has been forgotten to society. In general, across many English verbs, the regular past tense has become more popular, and for some verbs, like ‘help’, the regular form has overtaken the irregular form to become the popular past form used by speakers of the language.

In a 2007 study, Lieberman et al. explored the regularization of English verbs since Old English using the CELEX corpus, which gives word frequencies from several textual sources. They used a set of 177 verbs that were all irregular in Old English and examined how the rate of verb regularization relates to frequency of usage, finding that more common verbs regularize at a slower rate. They calculated half-lives for irregular verbs binned by frequency of usage and found that irregular verbs regularize with a half-life proportional to the square root of frequency.

In a 2011 paper, Michel et al. studied the regularization of verbs, along with other cultural and language trends, as an accompaniment to their introduction of the Google Books corpus. They found that most of the verb regularization over the last two centuries came from verbs using the suffix -t for the irregular form, like ‘burn’, and that British English texts were less likely than American English ones to use the regular form.

In a more recent study, Newberry et al. proposed a method for determining the underlying mechanisms driving language change, including the regularization of verbs (2017). Using the Corpus of Historical American English and inspired by ideas from evolution, the authors described a method to determine if language change is due to selection or random drift. They used a null hypothesis of stochastic drift and checked if selection would be strong enough to reject this null hypothesis. Of the 36 verbs Newberry et al. studied, only six showed statistical support for selection by their methods. They also claimed that rhyming patterns might be a driver of selection.

These prior studies of verb regularization have only focused on language data resulting from a formal editorial process, such as that of a published book. This editorial process will

tend to normalize language in some way, reflecting the linguistic opinions of authors and editors, rather than portray the language used by everyday people. For example, maybe the irregular form of a particular verb is considered proper by scholars, but a vast majority of the English-speaking population uses the regular form. While it is not a verb, one illustrative example is ‘whom’. Although ‘whom’ is the correct word to use in the objective case, it is common for everyday speakers to use ‘who’.

In Chapter 2, we take tweets to be a closer representation of everyday language. For the vast majority of accounts, tweets are authored by individuals without being edited. As such, the language used therein should more accurately represent average speakers than the language found in books.

We use data from both Twitter and Google Books, measuring the current state of verb regularization within each, to compare regularization between the more colloquial, unedited language of Twitter to the more formal, edited language of books. We also study the regional variation in regularization, both on a country scale, between the United States and the United Kingdom, and on a smaller scale, across counties within the United States.

1.3 STRETCHABLE WORDS

Watch a soccer match, and you are likely to hear the announcer shout ‘GOOOOOAAAAAAAAAL’. Stretched out words, also called elongated words (Wiktionary contributors, 2019a), are a fundamental part of spoken language, often used to modify the meaning of the base word in some way, such as to emphasize or exaggerate the meaning (e.g., ‘huuuuuge’), imply sarcasm (e.g., ‘suuuuure’), show excitement (e.g., ‘yeeeessss’), or communicate danger (e.g., ‘nooooooooooooo’).

Despite their being an integral part of spoken language, stretched words are much rarer in written language. They are not often found in literature or lexicons. The word ‘hahaha-hahahaha’ is not in the Oxford English Dictionary (Simpson and Weiner, 1989). However,

with the advent and rise of social media, stretched words have finally made it into written text on a large scale.

In Chapter 3, we study stretchable words on Twitter, collecting all stretchable words matching our criterion from September 9, 2008 through the end of 2016. We examine their frequency distributions and introduce two central parameters that quantify their independent properties of ‘balance’ and ‘stretch’. We develop ‘spelling trees’ as a way to visualize the many ways in which words involving two intermingling repeated characters stretch. We then discuss how the tools and methods we developed can be used to study the patterns of mistypings and misspellings.

The tools and methods we develop have many other potential applications, including the possible use by online dictionaries to finally include this natural part of language largely overlooked by lexicons. The online dictionary, Wiktionary, has already discussed the inclusion of some stretched words, and has made a policy on what to include (Wiktionary contributors, 2019b,c). An adoption of our methods may allow for a more inclusive policy and more informative entries. Other potential applications include improvements to language processing and the study of sequence construction generally.

1.4 UNLOCKING GOOGLE BOOKS

In the modern era, digitized text sources such as the Google Books corpus (Lin et al., 2012; Michel et al., 2011) and increased computational power provide profound opportunities for the study of linguistic evolution, and the study of cultural and societal changes inferable from language changes. As such, there have been a number of recent papers focusing on observations found in the Google Books n -grams corpus. For example, studies include analyses of Zipf’s and Heaps’ laws (Gerlach and Altmann, 2013), verb regularization (Gray et al., 2018; Michel et al., 2011), culture changes (Michel et al., 2011), new words in a

language (Petersen et al., 2012b), and word births, deaths, and general changes in word use over time (Pechenick et al., 2017; Petersen et al., 2012a).

Patterns in large scale corpora like Google Books would seem to be a reflection of culture and language, either through the collective voice of authors or the collective attention of readers. However, in Chapter 4 we find that by using token counts as a proxy for n -gram importance, both of these views are gravely distorted. Using token counts leads to nonsensical irregularities in terms of n -gram prevalence that do not accurately represent natural language.

We develop a revised method focused on book counts rather than token counts to reveal the collective author’s voice so that meaningful linguistic and cultural trends may be reliably discerned. Using this improved method, we are able to study the temporal evolution of the English Fiction corpus (The Google Ngram Viewer Team, 2013a).

We calculate the flux of n -grams across rank boundaries in Zipf distributions as a measure of ‘lexical turbulence’, summarizing the change in the distribution of n -gram ranks over time. We also perform a fine-grained analysis of the changes in language between time periods using wordshift plots and a rank divergence measure with a tunable parameter. Tuning this parameter allows us to resolve the changes from the most functional parts of language to those in n -grams that best represent cultural evolution and important historical events.

CHAPTER 2

ENGLISH VERB REGULARIZATION IN BOOKS AND TWEETS

The English language has evolved dramatically throughout its lifespan, to the extent that a modern speaker of Old English would be incomprehensible without translation. One concrete indicator of this process is the movement from irregular to regular (-ed) forms for the past tense of verbs. In this study we quantify the extent of verb regularization using two vastly disparate datasets: (1) Six years of published books scanned by Google (2003–2008), and (2) A decade of social media messages posted to Twitter (2008–2017). We find that the extent of verb regularization is greater on Twitter, taken as a whole, than in English Fiction books. Regularization is also greater for tweets geotagged in the United States relative to American English books, but the opposite is true for tweets geotagged in the United Kingdom relative to British English books. We also find interesting regional variations in regularization across counties in the United States. However, once differences in population are accounted for, we do not identify strong correlations with socio-demographic variables such as education or income.

2.1 INTRODUCTION

Human language reflects cultural, political, and social evolution. Words are the atoms of language. Their meanings and usage patterns reveal insight into the dynamical process by which society changes. Indeed, the increasing frequency with which electronic text is used as a means of communicating, e.g., through email, text messaging, and social media, offers us the opportunity to quantify previously unobserved mechanisms of linguistic development.

While there are many aspects of language being investigated towards an increased understanding of social and linguistic evolution [1–6], one particular area of focus has been on changes in past tense forms for English verbs [1–3]. These investigations have collectively demonstrated that English verbs are going through a process of regularization, where the original irregular past tense of a verb is replaced with the regular past tense, formed using the suffix *-ed*.

For example, the irregular past tense of the verb ‘burn’ is ‘burnt’ and the regular past tense is ‘burned’. Over time, the regular past tense has become more popular in general, and for some verbs has overtaken the irregular form. For example, in Fig. 2.1, we use the Google Ngram Online Viewer to compare the relative frequency of ‘burnt’ with that of ‘burned’ over the past 200 years. (As shown in an earlier paper involving two of the present authors [7], and expanded on below, the Google Ngram dataset is highly problematic but can serve as a useful barometer of lexical change.) In the first half of the 19th century, the irregular past tense ‘burnt’ was more popular. However, the regular past tense ‘burned’ gained in popularity and in the late 1800s became the more popular form, which has persisted through to today.

Looking at several examples like this, in a 2011 paper Michel et al. studied the regularization of verbs, along with other cultural and language trends, as an accompaniment to their introduction of the Google Books Ngram corpus (hereafter Ngrams) and the proto-

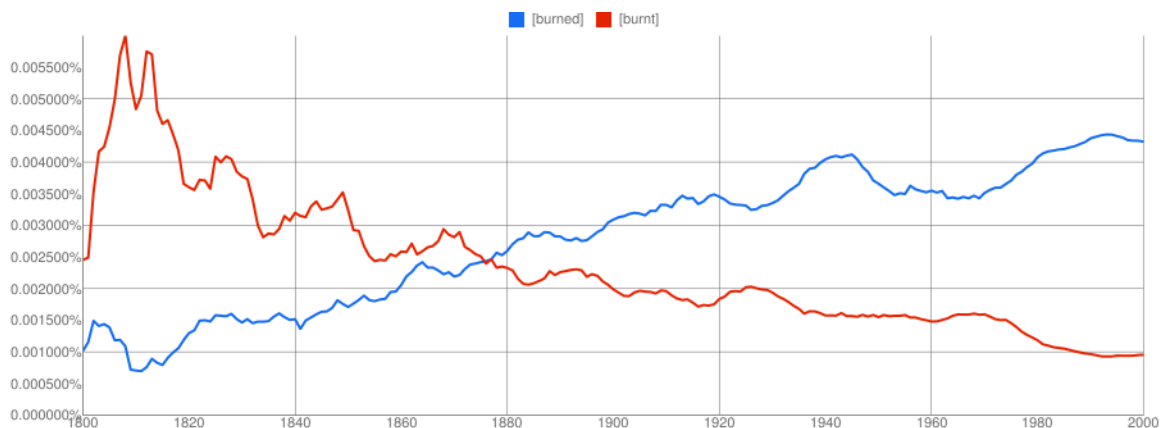


Figure 2.1: Relative word frequencies for the irregular and regular past verb forms for ‘burn’ during the 19th and 20th centuries, using the Google Ngram Online Viewer with the English Fiction 2012 corpus. Google Ngram trends can be misleading but capture basic shifts in a language’s lexicon [7, 8]. The irregular form ‘burnt’ was once more popular, but the regular form ‘burned’ overtook it in the late 19th century and its popularity has steadily increased ever since while that of ‘burnt’ has decreased. The dynamics of verb tense changes are rich, reflecting many processes at play in the Google Books Ngram data. An interactive version of this graphic can be found at https://books.google.com/ngrams/graph?content=burned%2Cburnt&year_start=1800&year_end=2000&corpus=16&smoothing=3.

field ‘Culturomics’ [2]. They found that most of the verb regularization over the last two centuries came from verbs using the suffix -t for the irregular form, and that British English texts were less likely than American English ones to move away from this irregular form.

In a 2007 study, Lieberman et al. explored the regularization of English verbs using the CELEX corpus, which gives word frequencies from several textual sources [1]. Focusing on a set of 177 verbs that were all irregular in Old English, they examined how the rate of verb regularization relates to frequency of usage, finding that more common verbs regularized at a slower rate. They calculated half-lives for irregular verbs binned by frequency, finding that irregular verbs regularize with a half-life proportional to the square root of frequency of usage.

In a more recent study, Newberry et al. proposed a method for determining the underlying mechanisms driving language change, including the regularization of verbs [3]. Using the Corpus of Historical American English and inspired by ideas from evolution, the authors

described a method to determine if language change is due to selection or drift, and applied this method to three areas of language change. They used a null hypothesis of stochastic drift and checked if selection would be strong enough to reject this null hypothesis. Of the 36 verbs Newberry et al. studied, only six demonstrated statistical support for selection. They also claimed that rhyming patterns might be a driver of selection.

Unfortunately, the corpora used in these studies have considerable limitations and corruptions. For example, early versions of the Ngrams data includes scientific literature, whose explosive growth through the 20th century is responsible for the decreasing trend in relative word usage frequency observed in many common search terms [7]. Moreover, the library-like nature of the corpus admits no accounting for popularity: Lord of the Rings and an unknown work contribute with equal weight to token counts.

Another general concern with large corpora of a global language like English is that language use varies tremendously with culture and geography. Ngrams allows only for the regional exploration of the English language with the British English corpus and the American English corpus. Twitter data enables us to focus on much smaller spatial regions (e.g., county or state).

Prior studies of verb regularization have also focused on data reflecting a formal editorial process, such as the one undergone by any published book. This editorial process will tend to normalize the language, reflecting the linguistic opinions of a small minority of canon gatekeepers, rather than portray the language used by everyday people. For example, maybe the irregular form of a particular verb is considered proper by scholars, but a vast majority of the English-speaking population uses the regular form. While it is not a verb form, one illustrative example is ‘whom’. Although ‘whom’ is the correct word to use in the objective case, it is common for everyday speakers to use ‘who’.

In the present study we take tweets to be a closer representation of everyday language. For the vast majority of accounts, tweets are authored by individuals without undergoing

a formal editing process. As such, the language therein should more accurately represent average speakers than what is found in books.

The demographic groups contributing to Twitter are by no means a carefully selected cross-section of society, but do offer natural language use by the roughly 20% of adult English speakers who use Twitter [9]. When exploring temporal changes in language use, the Ngrams and CELEX datasets evidently cover a much longer period than the decade for which social media is available. As a result, we are unable to infer anything about the temporal dimension of regularization looking at Twitter.

In this paper we use the Ngrams and Twitter datasets to establish estimates of the current state of English verb regularization. We structure our paper as follows: In Sec. 2.2, we describe the datasets we use. In Sec. 2.3, we present our results. We study verb regularization in English in general in Sec. 2.3.1. We compare verb regularization in American English (AE) and British English (BE) using both Ngrams and geotagged Twitter data in Sec. 2.3.2. In Sec. 2.3.3, we employ methods to study regional variation in verb usage, leveraging county level user location data in the United States. We also explore correlations between verb regularization and a number of socio-demographic and economic variables. Finally, in Sec. 2.4, we provide concluding remarks.

2.2 DESCRIPTION OF DATASETS

To be consistent with prior work, we chose the verb list for our project to match that of Michel et al. [2]. When comparing BE with AE, we use the subset of verbs that form the irregular past tense with the suffix *-t*. When calculating frequencies or token counts for the ‘past tense’ we use both the preterite and past participle of the verb. See Table 2.A1 in Appendix 2.7.1 for a complete tabulation of all verb forms.

The Ngrams data reflects relative frequency, providing, for a verb and a given year, the percentage of corpus tokens that are the given verb, where a token is an individual

occurrence of a word. The Google Ngram Online Viewer also has a smoothing parameter, s , which averages the relative frequency for the given year with that of each of the s years before and after the given year, if they exist. For example, Fig. 2.1 uses a smoothing of 3 years and shows that, averaged across the years 1997–2000 (the value displayed for the year 2000), the word ‘burned’ appeared with relative frequency 0.004321% (roughly once every 23,000 tokens), while ‘burnt’ appeared with relative frequency 0.000954% (roughly once every 105,000 tokens).

We downloaded the Ngrams verb data for the most recent 6-year period available (2003–2008) [10]. Specifically, we chose the 2008 values of relative frequency with a smoothing of 5 years, resulting in an average case insensitive¹ word frequency for the years 2003–2008. For general English, as suggested by [7], we queried the English Fiction 2012 corpus, which uses “books predominantly in the English language that a library or publisher identified as fiction.” For AE we used the American English 2012 corpus, which uses “books predominantly in the English language that were published in the United States.” For BE we used the British English 2012 corpus, which uses “books predominantly in the English language that were published in Great Britain” [11].

The Twitter messages for our project consist of a random sample of roughly 10% of all tweets posted between 9 September 2008 and 22 October 2017. This ‘decahose’ dataset comprises a total of more than 106 billion messages, sent by about 750 million unique accounts. From this larger set, we performed a case-insensitive search for verb forms of interest, also extracting geographic location when available in the meta-data associated with each tweet.

Tweets geotagged by mobile phone GPS with a U.S. location comprise about a 0.27% subset of the decahose dataset; United Kingdom locations comprise about a 0.05% subset. Many individuals provide location information, entered as free text, along with their bio-

¹When Ngrams computes a case insensitive word frequency it uses “the yearwise sum of the most common case-insensitive variants of the input query” [11].

graphical profile. We matched user specified locations of the form ‘city, state’ to a U.S. county when possible, comprising a 2.26% subset of the decahose dataset. Details on this matching process can be found in Appendix 2.7.2.

For general English, we counted the number of tokens in the decahose dataset for each verb. For AE, we used the tweets whose geotagged coordinates are located in the United States, and for BE we used the tweets whose geotagged coordinates are located in the United Kingdom. For the analysis of verbs by county, we used the tweets with the user entered location information. Table 2.1 summarizes the datasets used for both Ngrams and Twitter.

	Ngrams	Twitter
(I)	English Fiction 2012 corpus	All tweets
(II)	American English 2012 corpus	All tweets geolocated in the U.S.
(III)	British English 2012 corpus	All tweets geolocated in the U.K.
(IV)	N/A	All tweets with user entered location matching ‘city, state’

Table 2.1: A summary of the verb datasets.

The demographic data for U.S. counties comes from the 2015 American Community Survey 5-year estimates, tables DP02–Selected Social Characteristics, DP03–Selected Economic Characteristics, DP04–Selected Housing Characteristics, and DP05–Demographic and Housing Estimates, which can be found by searching online at <https://factfinder.census.gov/>. These tables comprise a total of 513 usable socio-demographic and economic variables.

We compute the *regularization fraction* for a verb as the proportion of instances in which the regular form was used for the past tense of the verb. More specifically, for Ngrams we divide the relative frequency for the regular past tense by the sum of the relative frequencies for the regular and irregular past tenses. Similarly, for Twitter we divide the token count

for the regular past tense by the sum of the token counts for both the regular and irregular past tenses. If the resulting regularization fraction is greater than 0.5, the regular past tense is more popular and we call the verb regular. Otherwise we call the verb irregular.

When calculating an average regularization across all verbs, we first compute the regularization fraction for each verb individually. Then we compute the average of the regularization fractions, with each verb contributing the same weight in the average, irrespective of frequency. We perform this ‘average of averages’ to avoid swamping the contribution of less frequent verbs.

2.3 METHODS AND RESULTS

2.3.1 VERB REGULARIZATION USING NGRAMS AND TWITTER

Using the datasets in row (I) of Table 2.1, we begin by comparing Ngrams and Twitter with respect to regularization of English verbs in Fig. 2.2, where we find that 21 verbs are more regular in Ngrams, and 85 are more regular on Twitter. A Wilcoxon signed rank test of the data has a p -value of 7.9×10^{-6} , demonstrating strong evidence that verbs on Twitter are more regular than verbs in Ngrams.

What mechanisms could be responsible for the observed increase in regularity on Twitter? One possibility is that authors of fiction published in the 2000s, along with their editors, being professional users of English, have a larger vocabulary than the typical user of Twitter. If so, their commitment to proper English would contribute to the appearance of relatively more irregular verbs in books. The average Twitter user may not know, or choose to use, the ‘correct’ past tense form of particular verbs, and thus use the default regular past tense.

Another driver may be that non-native English speakers writing English tweets may be more likely to use the default regular form. We will find quantitative support for this

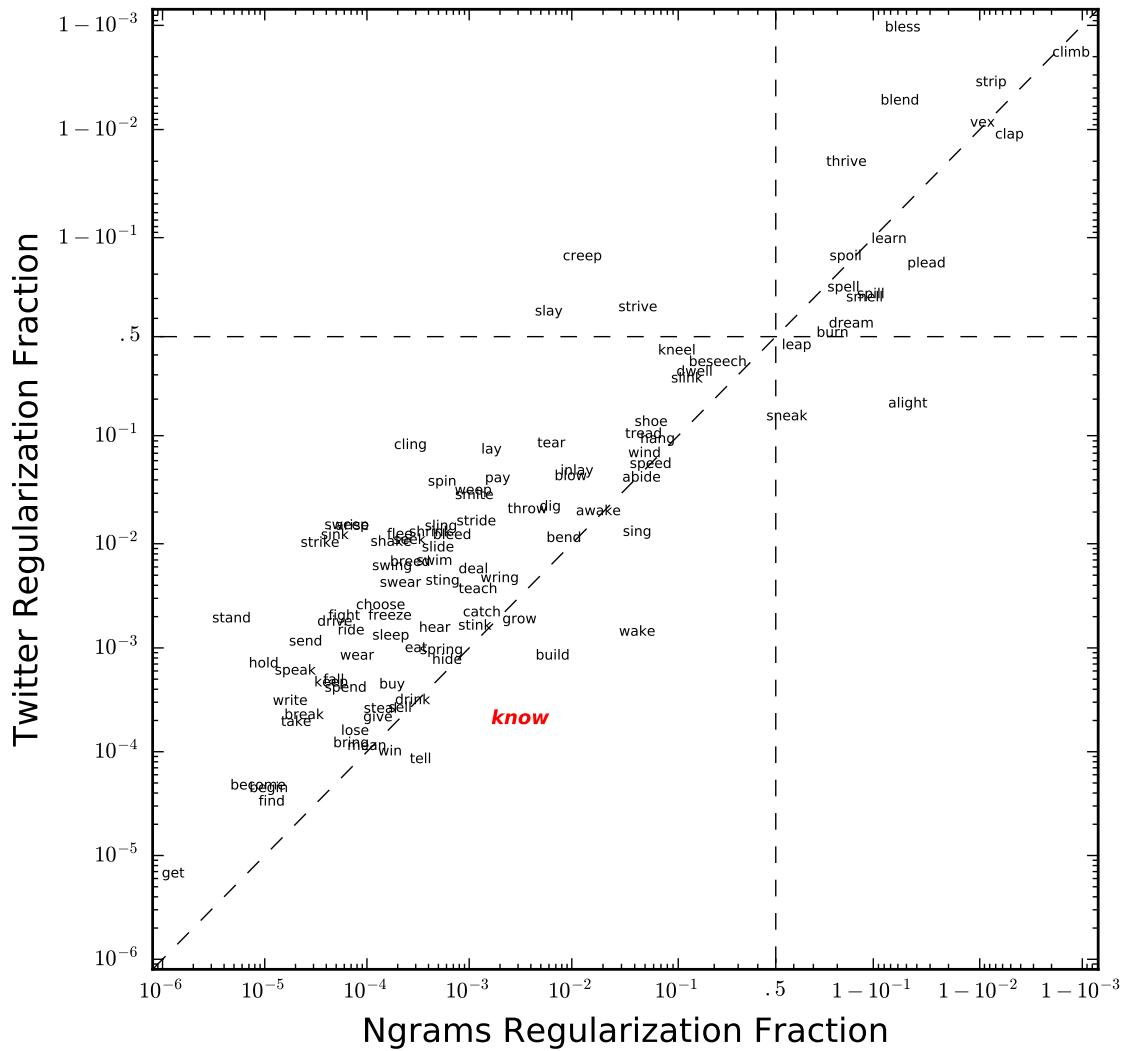


Figure 2.2: Comparison of verb regularization for Ngrams and Twitter. We calculate verb regularization fractions using the datasets in row (I) of Table 2.1. Verbs are centered at their regularization fraction in Ngrams (horizontal) and Twitter (vertical). Both axes are on a logit scale, which spreads out both extremes of the interval (0, 1). Verbs to the right of the vertical dashed line are regular in Ngrams; verbs above the horizontal dashed line are regular on Twitter. The diagonal dashed line separates verbs that are more regular on Twitter (those above and to the left of the line) from those that are more regular in Ngrams (those below and to the right of the line). For example, compared with ‘knew’, the word ‘knowed’ appears roughly 3 times in 1000 in Ngrams, and 2 times in 10,000 on Twitter, making ‘know’ irregular in both cases, but more than an order of magnitude more regular in Ngrams than on Twitter.

mechanism below. As a preview, we note that Fig. 2.2 shows that ‘burn’ is predominantly regular on Twitter globally, but we see later (Fig. 2.3B) that ‘burn’ is irregular on Twitter for both American English and British English. Thus, it is likely that non-native speakers are contributing to this difference.

2.3.2 AMERICAN AND BRITISH ENGLISH

We next study how verb regularization varies with geographic region. In this subsection we use the datasets in row (II) of Table 2.1 for AE and row (III) for BE and the subset of verbs that form the irregular past tense with the suffix *-t*.

In Fig. 2.3A, we compare American and British English in Ngrams. The average regularization fraction is 0.49 in AE and 0.42 in BE. For 17 out of 22 verbs, AE shows more regularization, with a Wilcoxon signed rank test p -value of 9.8×10^{-4} , giving statistical support that AE verbs are more regular on average in Ngrams than BE verbs.

As we show in the inset scatter plot of Fig. 2.3A, regularization in AE and BE are also strongly positively correlated with a Spearman correlation coefficient of 0.97 ($p = 2.3 \times 10^{-14}$). Verbs that are more regular in AE are also more regular in BE, just not to the same extent.

In Fig. 2.3B, we compare regularization in AE and BE on Twitter. For Twitter, the average regularization fraction is 0.54 for AE, higher than Ngrams, and 0.33 for BE, much lower than Ngrams. As with Ngrams, 17 verbs out of 22 show more regularization in AE than in BE. The Wilcoxon signed rank test gives a weaker but still significant p -value of 1.9×10^{-3} .

The inset in Fig. 2.3B also shows a positive correlation, although not as strong as Ngrams, with a Spearman correlation coefficient of 0.87 ($p = 1.1 \times 10^{-7}$). Generally, on Twitter, regular AE verbs are also regular in BE, but the difference in regularization fraction is much greater than for Ngrams.

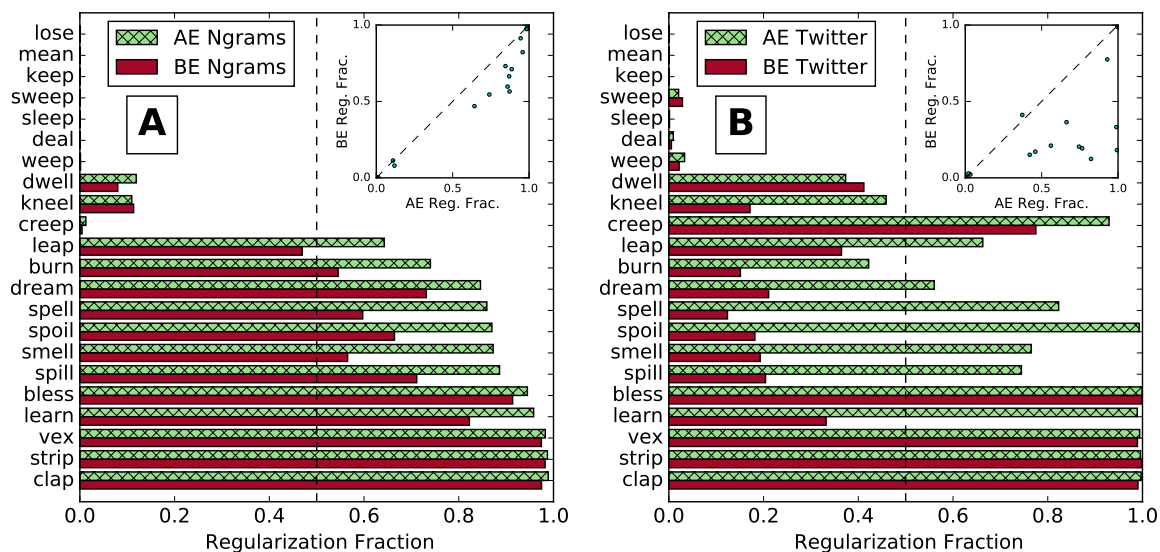


Figure 2.3: American and British English verb regularization fractions for (A) Ngrams and (B) Twitter. We use the subset of verbs that form the irregular past tense with the suffix *-t* and the datasets in rows (II) and (III) of Table 2.1. The inset scatter plot has a point for each verb. The dashed diagonal line separates verbs that are more regular in AE (below the line) from those that are more regular in BE (above the line).

In Fig. 2.4A, we demonstrate the difference in regularization between AE and BE for both Ngrams and Twitter. The values in this figure for Ngrams can be thought of as, for each verb in Fig. 2.3A, subtracting the value of the bottom bar from the top bar, and likewise for Twitter and Fig. 2.3B. Positive numbers imply greater regularization in AE, the more common scenario. When the difference is near zero for one corpus, it is usually close to zero for the other corpus as well. However, when Ngrams shows that AE is notably more regular than BE, Twitter tends to show a much larger difference.

The average difference in regularization fraction between AE and BE for Twitter is 0.21, whereas it is only 0.08 for Ngrams. Again, we find that these averages are significantly different with a Wilcoxon signed rank p -value of 1.9×10^{-2} .

The inset scatter plot tells a similar story, with a cluster of points near the origin. As the difference in regularization fraction between regions increases in Ngrams, it also tends

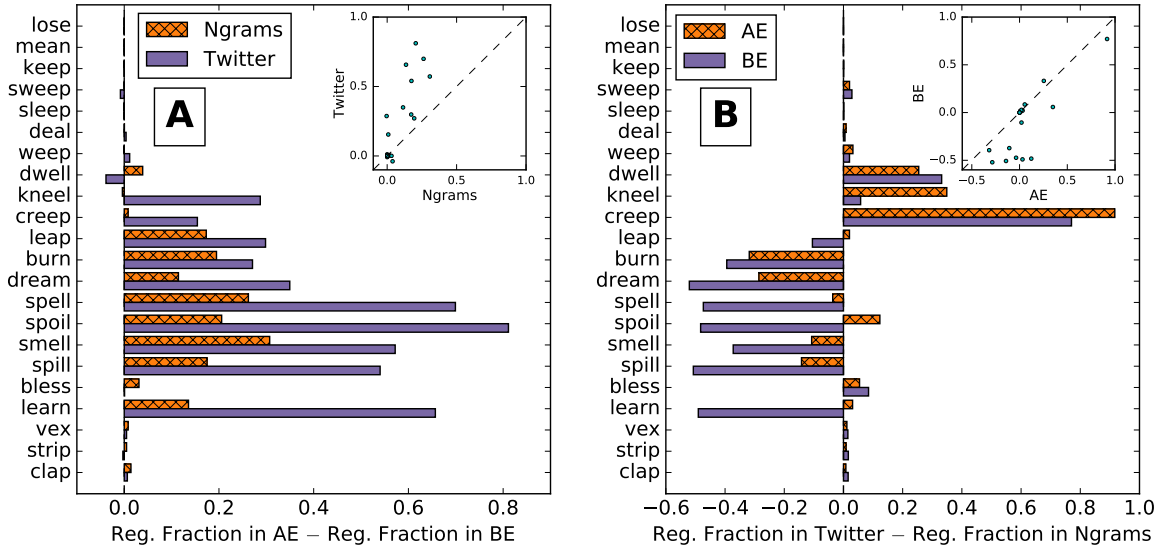


Figure 2.4: Differences in verb regularization fractions. The bar chart gives the difference for each verb in each corpus. The inset scatter plot has a point for each verb. (A) The difference between verb regularization fractions for AE and BE in Twitter and Ngrams. The dashed diagonal line of the inset scatter plot separates verbs for which this difference is greater in Ngrams (below the line) from those for which it is greater in Twitter (above the line). (B) The difference between verb regularization fraction for Twitter and Ngrams in AE and BE. The dashed diagonal line of the inset scatter plot separates verbs for which this difference is greater in AE (below the line) from those for which it is greater in BE (above the line).

to increase in Twitter, with Spearman correlation coefficient 0.65 and p -value 1.0×10^{-3} . The steep rise shows that the difference increases faster on Twitter than in Ngrams.

Fig. 2.4B returns to comparing Ngrams and Twitter, but now between AE and BE. For each verb, the bar chart shows the difference between the regularization fraction for Twitter and Ngrams in both AE and BE, with positive values showing that regularization for Twitter is greater. In this case, the values can be thought of as subtracting the values for the bars in Fig. 2.3A from the corresponding bars in Fig. 2.3B. As we find for English in general, regularization is greater on Twitter than in Ngrams for AE, with an average difference of 0.04. However, for BE, regularization is greater in Ngrams than on Twitter, with an average difference in regularization fraction of -0.09 .

	Twitter	Ngrams	Difference
AE	0.54	0.49	0.04
BE	0.33	0.42	-0.09
Difference	0.21	0.08	

Table 2.2: A summary of the average regularization fractions for AE and BE on Twitter and Ngrams. Note that the differences were taken prior to rounding.

We summarize our findings in Table 2.2. We found again that verbs on Twitter are more regular than in Ngrams for American English, likely for many of the same reasons that verbs on Twitter are more regular than Ngrams in general. However, we find that in British English the opposite is true: Verbs on Twitter are less regular than in Ngrams. In decreasing order by average regularization fraction, we have AE Twitter, then AE Ngrams, then BE Ngrams, and finally BE Twitter. Knowing that the general trend is towards regularization [1, 2], it seems that regularization is perhaps being led by everyday speakers of American English, with American published work following suit, but with a lag. Then, it may be that British English authors and editors are being influenced by American publications and the language used therein. Indeed, some studies have found a general ‘Americanization’ of English across the globe [12, 13], meaning that the various varieties of English used across the world are becoming more aligned with American English. Finally, it may be that average British users of Twitter are more resistant to the change. Indeed, from the figures in the study by Gonçalves et al., one can see that the ‘Americanization’ of British English is more pronounced in Ngrams than on Twitter [12], agreeing with what we have found here.

2.3.3 REGULARIZATION BY U.S. COUNTY

In Sec. 2.3.2, we demonstrated regional differences in verb regularization by comparing BE and AE. Here, we consider differences on a smaller spatial scale by quantifying regularization by county in the United States using the dataset in row (IV) of Table 2.1. We use methods inspired by Grieve et al. to study regional variation in language [14].

We only include counties that had at least 40 total tokens for the verbs under consideration. We plot the average regularization fraction for each county in the continental U.S. in Fig. 2.5A, where counties with not enough data are colored black. To control for the skewed distribution of samples associated with county population (see below for more details), we use residuals for this portion of the analysis. After regressing with the \log_{10} of data volume (total number of tokens) for each county, we compute the average regularization fraction residual, which is plotted in Fig. 2.5B.

That is, if we let d_i be the total number of tokens for verbs in tweets from county i ; α and β be the slope and intercept parameters computed from regression; and R_i be the average regularization fraction for county i , then we compute the average regularization fraction residual for county i , r_i^{reg} , as

$$r_i^{\text{reg}} = R_i - (\beta + \alpha \log_{10} d_i). \quad (2.1)$$

Using the average regularization residual at the county level as input, we measure local spatial autocorrelation using the Getis-Ord G_i^* z -score [15],

$$G_i^* = \frac{\sum_j w_{ij} r_j^{\text{reg}} - \bar{r}^{\text{reg}} \sum_j w_{ij}}{\sigma \sqrt{\left[n \sum_j w_{ij}^2 - \left(\sum_j w_{ij} \right)^2 \right] / (n-1)}}, \quad (2.2)$$

where

$$\sigma = \sqrt{\frac{\sum_j (r_j^{\text{reg}})^2}{n} - (\bar{r}^{\text{reg}})^2}, \quad (2.3)$$

$\bar{r}^{\text{reg}} = \frac{1}{n} \sum_i r_i^{\text{reg}}$, n is the number of counties, and w_{ij} is a weight matrix. To obtain the weight matrix used in this calculation, we first create a distance matrix, s_{ij} , where the distance between each pair of counties is the larger of the great circle distance, s_{ij}^{GC} , in

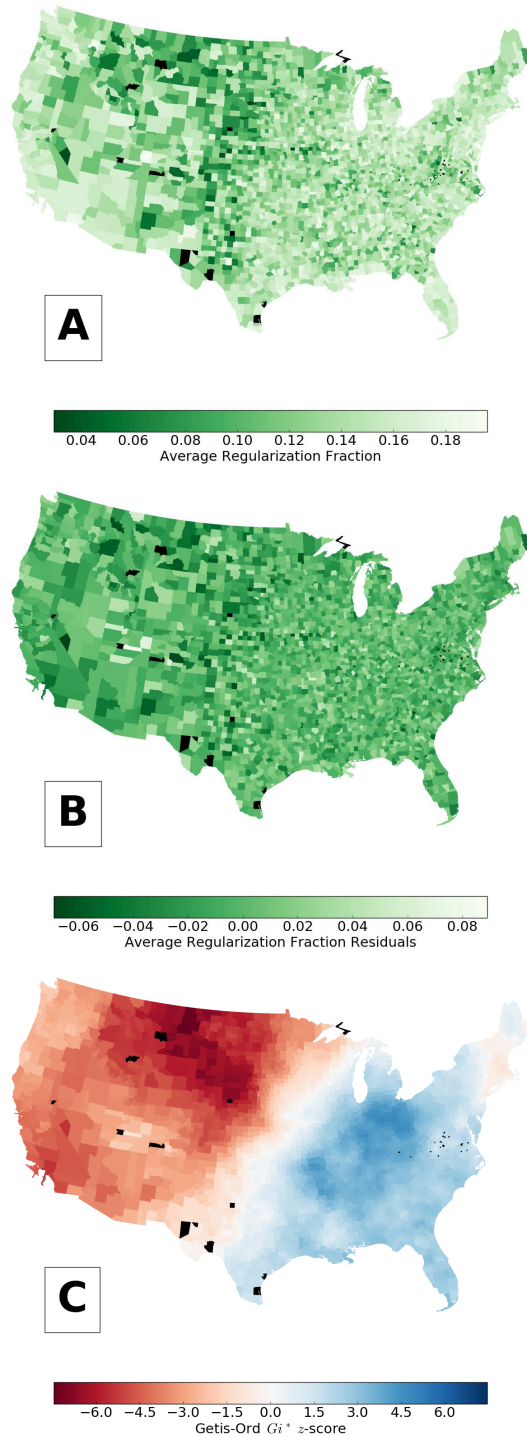


Figure 2.5: (A) The average verb regularization fraction by county for the lower 48 states, along with (B) residuals and (C) G_i^* z-score. A higher G_i^* z-score means a county has a greater regularization fraction than expected. Counties colored black did not have enough data. We used the dataset in row (IV) of Table 2.1.

miles between the centers of the bounding box for each county and 10 miles. That is,

$$s_{ij} = \max(s_{ij}^{\text{GC}}, 10). \quad (2.4)$$

We make the minimum value for s_{ij} 10 miles to prevent a county from having too large of a weight. We then compute the weight matrix as

$$w_{ij} = \frac{1}{\sqrt{s_{ij}}}. \quad (2.5)$$

Fig. 2.5C shows the results for the lower 48 states, where black represents counties left out because there was not enough data. For each county, the Gi^* z -score computes a local weighted sum of the residuals, r_j^{reg} , for the surrounding counties and compares that to the expected value of that weighted sum if all the counties had exactly the average residual, \bar{r}^{reg} , as their value, where the weighting is such that closer counties have a higher weight. Areas that are darker blue (positive z -score) belong to a cluster of counties that has higher regularization than average, and those that are darker red (negative z -score) belong to a cluster that has lower regularization than average. So, Fig. 2.5C shows that, in general, western counties show less regularization than average and eastern counties show more, except that the New England area is fairly neutral.

As usual, the z -score gives the number of standard deviations away from the mean. For this we would do a two-tail test for significance because we are looking for both high value and low value clusters. For example, a z -score greater in magnitude than 1.96 is significant at the .05 level. If we do a Bonferroni correction based on 3161 counties (the number included for this part of the analysis), then a z -score greater in magnitude than 4.32 is significant for a two-tail test at the $.05/3161 \approx 1.58 \times 10^{-5}$ level.

We do this same process looking at individual verbs as well. However, when looking at individual verbs, we use the regularization fraction rather than residuals, because the

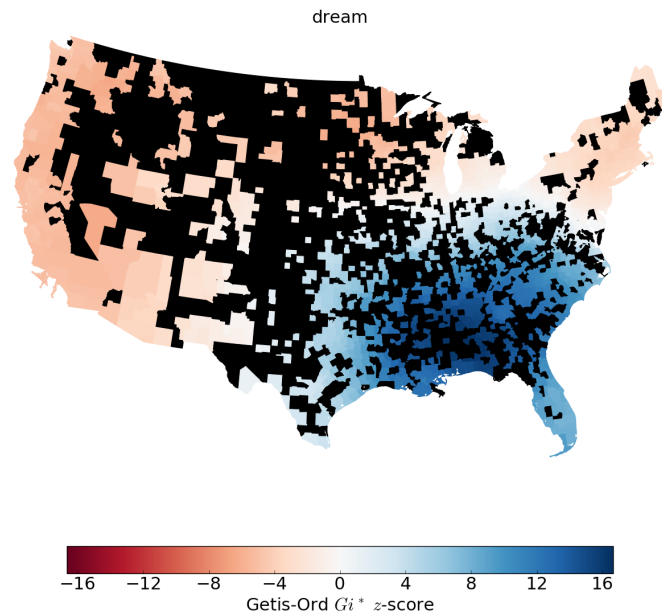


Figure 2.6: The G_i^* z -score for verb regularization by county for the verb ‘dream’ for the lower 48 states. Counties colored black did not have enough data. People tweet ‘dreamed’ rather than ‘dreamt’ more often than expected in the southeastern U.S.

data skew is not as problematic. This is because the main problem with data volume comes when averaging across verbs that have different frequencies of usage, as explained below. Also, here we include counties that have at least 10 tokens. Fig. 2.6 gives an example map showing the G_i^* z -scores for the verb ‘dream’. The maps showing local spatial autocorrelation for the complete list of verbs can be found in the Online Appendix A at <http://compstorylab.org/stretchablewords/>.

For many of the counties in the U.S., there is a small sample of Twitter data. We restrict our analysis to counties with a total token count of at least 40 for the verbs we consider. Even for the counties meeting this criterion, the volume of data varies, leading to drastically different sample sizes across counties.

More common verbs tend to have popular irregular forms (e.g., ‘found’ and ‘won’), and less common verbs tend to be regular (e.g., ‘blessed’ and ‘climbed’) [1]. As a result, samples

taken from populous counties are more likely to contain less common verbs. Our ‘average regularization’ is an average of averages, resulting in an underlying trend toward higher rates for more populous counties due to the increased presence of rarer regular verbs.

Fig. 2.7 demonstrates the relationship between data volume and regularization. To explore the connection further, we perform a synthetic experiment as follows.

To simulate sampling from counties with varying population sizes, we first combine all verb token counts (using the Twitter dataset from row (I) of Table 2.1) into a single collection. We then randomly sample a synthetic county worth of tokens from this collection. For a set of 1000 logarithmically spaced county sizes, we randomly draw five synthetic collections of verbs (each is a blue circle in Fig. 2.7). For each sample, we compute the average regularization fraction, as we did for U.S. counties. The goal is to infer the existences of any spurious trend introduced by the sampling of sparsely observed counties.

The resulting simulated curve is comparable to the trend observed for actual U.S. counties. As the data volume increases, the simulated version converges on roughly 0.17, which is the average regularization fraction for all of Twitter.

We also explored correlations between verb regularization and various demographic variables. Fig. 2.7 showed a strong relationship between data volume and verb regularization. It has been shown elsewhere that tweet density positively correlates with population density [16], and population size is correlated with many demographic variables. As a result, we use partial correlations as an attempt to control for the likely confounding effect of data volume.

For each demographic variable, we compute the regression line between the \log_{10} of data volume, d_i , and regularization, and compute the residuals as in Eq. 2.1. Then, if the demographic variable is an ‘Estimate’ variable, where the unit is number of people, we similarly compute the regression line between the \log_{10} of data volume and the \log_{10} of the

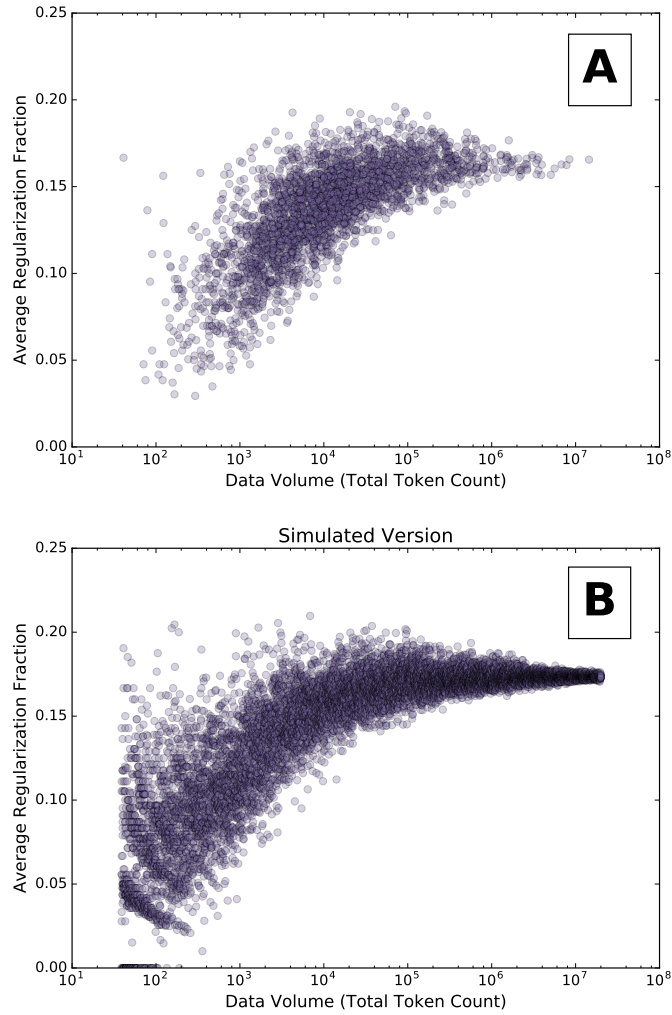


Figure 2.7: (A) Scatter plot of average verb regularization for counties. For each county, the horizontal coordinate is the total token count of verbs found in tweets from that county, and the vertical coordinate is that county’s average regularization fraction. For a version with verbs split into frequency bins, see Fig. 2.A1 in Appendix 2.7.3. (B) We created synthetic counties by sampling words from the collection of all occurrences of all verbs on Twitter (using the dataset from row (I) of Table 2.1). The point’s horizontal position is given by the total sample token count in a synthetic county; the vertical position is given by its average regularization fraction.

demographic variable² and compute the residuals, r_i^{dem} , as

$$r_i^{\text{dem}} = \log_{10}(D_i) - (\delta + \gamma \log_{10} d_i), \quad (2.6)$$

where D_i is the value of the demographic variable for county i , and γ and δ are the slope and intercept parameters calculated during regression.

Otherwise, the demographic variable is a ‘Percent’ variable, with units of percentage, and we compute the regression line between the \log_{10} of data volume and the demographic variable, and compute residuals as

$$r_i^{\text{dem}} = D_i - (\delta + \gamma \log_{10} d_i). \quad (2.7)$$

The correlation between residuals r_i^{reg} and r_i^{dem} gives the partial correlation between average regularization and the demographic variable.

Our findings suggest that data volume is a confounding variable in at least some of the cases because, after controlling for data volume, there is generally a large decrease in the correlation between verb regularization and the demographic variables. The largest in magnitude Pearson correlation between verb regularization and a demographic variable is 0.68, for the variable ‘Estimate; SCHOOL ENROLLMENT - Population 3 years and over enrolled in school’, whereas the largest in magnitude partial correlation is only -0.18 , for the variable ‘Percent; OCCUPATION - Civilian employed population 16 years and over - Management, business, science, and arts occupations’. Table 2.3 lists the 10 demographic variables with largest in magnitude partial correlation.

Fig. 2.8 shows an example for one of the demographic variables, the ‘Percent’ variable with largest simple correlation. Fig. 2.8A is the scatter plot of the demographic variable with average regularization, which corresponds to simple correlation. Fig. 2.8B is the scatter

²We do not include any county that has a value of zero for the demographic variable here to prevent errors when taking the \log_{10} .

Rank	Partial Correlation	Demographic Variable
1	-0.18	Percent; OCCUPATION - Civilian employed population 16 years and over - Management, business, science, and arts occupations
2	-0.16	Percent; UNITS IN STRUCTURE - Total housing units - 10 to 19 units
3	-0.16	Percent; CLASS OF WORKER - Civilian employed population 16 years and over - Self-employed in own not incorporated business workers
4	-0.16	Percent; UNITS IN STRUCTURE - Total housing units - 20 or more units
5	0.16	Percent; COMMUTING TO WORK - Workers 16 years and over - Car, truck, or van - drove alone
6	0.15	Percent; BEDROOMS - Total housing units - 3 bedrooms
7	-0.15	Percent; COMMUTING TO WORK - Workers 16 years and over - Worked at home
8	-0.15	Percent; INDUSTRY - Civilian employed population 16 years and over - Agriculture, forestry, fishing and hunting, and mining
9	-0.15	Percent; BEDROOMS - Total housing units - 1 bedroom
10	0.14	Percent; OCCUPATION - Civilian employed population 16 years and over - Production, transportation, and material moving occupations

Table 2.3: Top demographic variables sorted by the magnitude of their partial correlation with verb regularization in U.S. counties. For example, regularization is positively correlated with the percentage of workers driving alone to work, and anti-correlated with the percentage of individuals working from home. Statistics for all of the demographic variables can be found in the Online Appendix B at <http://compstorylab.org/stretchablewords/>.

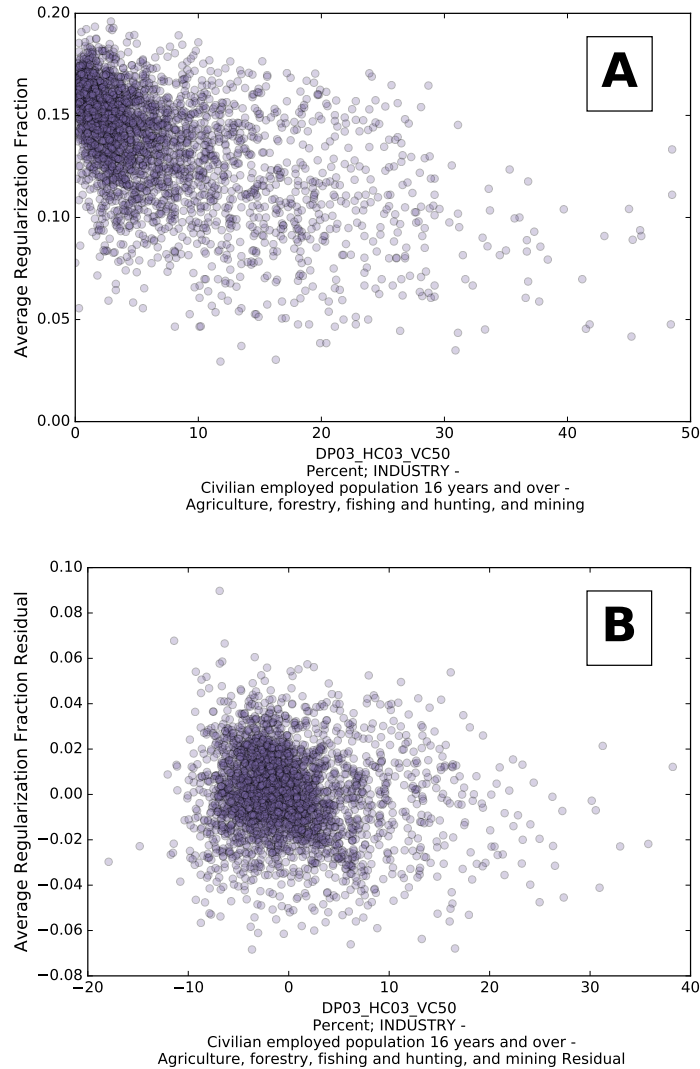


Figure 2.8: (A) Average verb regularization for counties as a function of the percentage of civilians employed in agriculture, forestry, fishing, hunting, and mining. Several hundred such plots are available in an interactive online appendix. (B) For each county, the horizontal coordinate is given by the residual left after regressing the demographic variable with the \log_{10} of data volume and the vertical coordinate is given by the residual left after regressing that county's average regularization fraction with the \log_{10} of data volume. Data volume, for a county, is the total token count of all verbs found in tweets from that county.

plot of the residuals, r_i^{dem} and r_i^{reg} , after regressing with the \log_{10} of data volume, and corresponds with partial correlation. We can see that there is a strong simple correlation (-0.52), but after accounting for data volume that correlation largely vanishes (-0.15). Similar plots for all of the demographic variables can be found in the Online Appendix B at <http://compstorylab.org/stretchablewords/>.

2.4 CONCLUDING REMARKS

Our findings suggest that, by and large, verb regularization patterns are similar when computed with Ngrams and Twitter. However, for some verbs, the extent of regularization can be quite different. If social media is an indicator of changing patterns in language use, Ngrams data ought to lag with a timescale not yet observable due to the recency of Twitter data. Very reasonably, Ngrams data may not yet be showing some of the regularization that is happening in everyday English.

We also found differences in verb regularization between American and British English, but found that this difference is much larger on Twitter than Ngrams. Overall, and in American English specifically, verbs are more regular on Twitter than in Ngrams, but the opposite is true for British English. In the U.S., we also find variation in average verb regularization across counties. Lastly, we showed that there are significant partial correlations between verb regularization and various demographic variables, but they tend to be weak.

Our findings do not account for the possible effects of spell checkers. Some people, when tweeting, may be using a spell checker to edit their tweet. If anything, this will likely skew the language on Twitter towards the ‘correct’ form used in edited textual sources. For example, in Fig. 2.2 we see that ‘stand’ is irregular for both Ngrams and Twitter, and likely most spell checkers would consider the regular ‘standed’ a mistake, but we see that ‘stand’ is still over 100 times more regular on Twitter than in Ngrams. So, the differences between

edited language and everyday language may be even larger than what we find here suggests. Future work should look into the effects of spell checkers.

Our study explored the idea that edited written language may not fully represent the language spoken by average speakers. However, tweets do not, of course, fully represent the English-speaking population. Even amongst users, our sampling is not uniform as it reflects the frequency with which different users tweet (see Fig. 2.A2 in Appendix 2.7.4). Furthermore, the language used on Twitter is not an unbiased sample of language even for people who use it frequently. The way someone spells a word and the way someone pronounces a word may be different, especially, for example, the verbs with an irregular form ending in -t, because -t and -ed are close phonetically. However, the fact that we found differences between the language of Ngrams and the language of Twitter suggests that the true language of everyday people is not fully represented by edited written language. We recommend that future studies should investigate speech data.

2.5 ACKNOWLEDGMENTS

We are thankful for the helpful reviews and discussions of earlier versions of this work by A. Albright and J. Bagrow, and for help with initial data collection from L. Gray. PSD & CMD were supported by NSF Grant No. IIS-1447634, and TJG, PSD, & CMD were supported by a gift from MassMutual.

2.6 REFERENCES

- [1] Erez Lieberman, Jean-Baptiste Michel, Joe Jackson, Tina Tang, and Martin A. Nowak. Quantifying the evolutionary dynamics of language. *Nature*, 449(7163):713–716, 10 2007.
- [2] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman

- Aiden. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.
- [3] Mitchell G. Newberry, Christopher A. Ahern, Robin Clark, and Joshua B. Plotkin. Detecting evolutionary forces in language change. *Nature*, 551(7679):223, 2017.
- [4] Simon J. Greenhill, Chieh-Hsi Wu, Xia Hua, Michael Dunn, Stephen C. Levinson, and Russell D. Gray. Evolutionary dynamics of language systems. *Proceedings of the National Academy of Sciences*, 114(42):E8822–E8829, 2017.
- [5] Florencia Reali, Nick Chater, and Morten H. Christiansen. Simpler grammar, larger vocabulary: How population size affects language. *Proceedings of the Royal Society of London B: Biological Sciences*, 285(1871), 2018.
- [6] Christian Ramiro, Mahesh Srinivasan, Barbara C. Malt, and Yang Xu. Algorithms in the historical emergence of word senses. *Proceedings of the National Academy of Sciences*, 115(10):2323–2328, 2018.
- [7] Eitan Adam Pechenick, Christopher M. Danforth, and Peter Sheridan Dodds. Characterizing the Google Books corpus: Strong limits to inferences of socio-cultural and linguistic evolution. *PLOS ONE*, 10(10):1–24, 10 2015.
- [8] Eitan Adam Pechenick, Christopher M. Danforth, and Peter Sheridan Dodds. Is language evolution grinding to a halt? The scaling of lexical turbulence in English fiction suggests it is not. *Journal of Computational Science*, 21:24–37, 2017.
- [9] Maeve Duggan. The demographics of social media users. <http://www.pewinternet.org/2015/08/19/the-demographics-of-social-media-users/>, 2015. Accessed: 2016-07-18.
- [10] The Google Ngram Viewer Team. Google Ngram Viewer. <https://books.google.com/ngrams/>, 2013.
- [11] The Google Ngram Viewer Team. About Ngram Viewer. <https://books.google.com/ngrams/info>, 2013.
- [12] Bruno Gonçalves, Lucía Loureiro-Porto, José J. Ramasco, and David Sánchez. Mapping the Americanization of English in space and time. *PLOS ONE*, 13(5):1–15, 05 2018.
- [13] Paul Baker. *American and British English: Divided by a Common Language?* Cambridge University Press, 2017.
- [14] Jack Grieve, Dirk Speelman, and Dirk Geeraerts. A statistical method for the identification and aggregation of regional linguistic variation. *Language Variation and Change*, 23(2):193–221, 2011.
- [15] J. K. Ord and Arthur Getis. Local spatial autocorrelation statistics: Distributional issues and an application. *Geographical Analysis*, 27(4):286–306, 1995.
- [16] Rudy Arthur and Hywel T. P. Williams. Scaling laws in geo-located Twitter data. *CoRR*, abs/1711.09700, 2017.

2.7 APPENDIX

2.7.1 TABLE OF VERB FORMS

Table 2.A1

Verb	Regular	Irregular		Token Count
	Preterit & Past Participle	Preterit	Past Participle	
abide	abided	abode	abode	146,566
alight	alighted	alit	alit	56,306
arise	arised	arose	arisen	164,134
awake	awaked	awoke	awoken, awoke	423,359
become	becomed	became	become	50,664,026
begin	beginned	began	begun	5,942,572
bend	bended	bent	bent	4,777,019
beseech	beseeched	besought	besought	3,390
bleed	bleeded	bled	bled	252,225
blend	blended	blent	blent	436,029
bless	blessed	blest	blest	22,547,387
blow	blowed	blew	blown	9,155,246
break	breaked	broke	broken	54,506,810
breed	breeded	bred	bred	1,040,854
bring	bringed	brought	brought	15,303,318
build	builded	built	built	8,521,553

Continued on next page

Table 2.A1: A tabulation of all verb forms used in this study. The Token Count column gives the sum of all the tokens for the past tense forms of the verb, both regular and irregular, in our Twitter dataset (see row (I) of Table 2.1 in Sec. 2.2).

Table 2.A1: (continued)

Verb	Regular	Irregular		Token Count
	Preterit & Past Participle	Preterit	Past Participle	
burn	burned	burnt	burnt	7,457,942
buy	buyed	bought	bought	24,841,526
catch	catched	caught	caught	24,891,188
choose	choosed	chose	chosen	10,290,205
clap	clapped	clapt	clapt	405,837
climb	climbed	clomb, clom	clomben	635,122
cling	clinged	clung	clung	49,742
creep	creeped	crept	crept	698,405
deal	dealed	dealt	dealt	1,181,974
dig	digged	dug	dug	941,656
dream	dreamed	dreamt	dreamt	2,794,060
drink	drinked	drank	drunk, drank	37,295,703
drive	drived	drove	driven	5,745,497
dwell	dwelled	dwelt	dwelt	25,725
eat	eated	ate	eaten	25,084,758
fall	fallen	fell	fallen	25,224,815
fight	fighded	fought	fought	3,625,297
find	finded	found	found	80,709,195
flee	fleed	fled	fled	405,295
freeze	freezed	froze	frozen	7,454,847

Continued on next page

Table 2.A1: A tabulation of all verb forms used in this study. The Token Count column gives the sum of all the tokens for the past tense forms of the verb, both regular and irregular, in our Twitter dataset (see row (I) of Table 2.1 in Sec. 2.2).

Table 2.A1: (continued)

Verb	Regular	Irregular		Token Count
	Preterit & Past Participle	Preterit	Past Participle	
get	getted	got	got, gotten	500,591,203
give	gived	gave	given	58,697,198
grow	growed	grew	grown	17,951,273
hang	hanged	hung	hung	3,991,956
hear	heard	heard	heard	52,605,822
hide	hided, hided	hid	hid, hidden	7,829,276
hold	holded	held	held	10,080,725
inlay	inlayed	inlaid	inlaid	44,811
keep	keeped	kept	kept	11,785,131
kneel	kneeled	knelt	knelt	83,765
know	knowed	knew	known	58,175,701
lay	layed	laid	laid	5,828,898
leap	leaped	leapt	leapt	91,620
learn	learned	learnt	learnt	18,134,586
lose	losed	lost	lost	72,695,892
mean	meaned	meant	meant	26,814,977
pay	payed	paid	paid	21,150,031
plead	pleaded	pled	pled	193,553
ride	rided	rode	ridden	1,710,109
seek	seeked	sought	sought	888,822

Continued on next page

Table 2.A1: A tabulation of all verb forms used in this study. The Token Count column gives the sum of all the tokens for the past tense forms of the verb, both regular and irregular, in our Twitter dataset (see row (I) of Table 2.1 in Sec. 2.2).

Table 2.A1: (continued)

Verb	Regular	Irregular		Token Count
	Preterit & Past Participle	Preterit	Past Participle	
sell	elled	sold	sold	14,251,612
send	sended	sent	sent	26,265,441
shake	shaked	shook	shaken	3,223,316
shoe	shoed	shod	shod	47,780
shrink	shrank	shrank, shrunk	shrunk, shrunken	296,188
sing	singed	sang, sung	sung	6,767,707
sink	sinked	sank, sunk	sunk, sunken	927,419
slay	slayed	slew	slain	2,153,981
sleep	sleeped	slept	slept	9,252,446
slide	slided	slid	slid	530,659
sling	slinged	slung	slung	38,320
slink	slinked	slunk	slunk	5,772
smell	smelled	smelt	smelt	1,089,814
smite	smitted, smited	smote	smitten, smote	176,768
sneak	sneaked	snuck	snuck	797,337
speak	speaked	spoke	spoken	8,502,050
speed	speeded	sped	sped	216,062
spell	spelled	spelt	spelt	3,812,137
spend	spended	spent	spent	17,603,781
spill	spilled	spilt	spilt	1,627,331

Continued on next page

Table 2.A1: A tabulation of all verb forms used in this study. The Token Count column gives the sum of all the tokens for the past tense forms of the verb, both regular and irregular, in our Twitter dataset (see row (I) of Table 2.1 in Sec. 2.2).

Table 2.A1: (continued)

Verb	Regular	Irregular		Token Count
	Preterit & Past Participle	Preterit	Past Participle	
spin	spinned	spun	spun	342,022
spoil	spoiled	spoilt	spoilt	3,891,576
spring	springed	sprang, sprung	sprung	626,400
stand	standed	stood	stood	3,942,812
steal	stealed	stole	stolen	11,884,934
sting	stinged	stung	stung	391,053
stink	stinked	stank, stunk	stunk	1,556,197
stride	strided	strode	stridden	17,811
strike	striked	struck	struck, stricken	2,167,165
strip	stripped	stript	stript	837,967
strive	strived	strove	striven	33,705
swear	swearred	swore	sworn	1,902,662
sweep	sweepred	swept	swept	931,245
swim	swimmed	swam	swum	356,842
swing	swinged	swung	swung	295,360
take	taked	took	taken	83,457,822
teach	teached	taught	taught	9,379,039
tear	teared	tore	torn	4,238,865
tell	telled	told	told	71,562,969
thrive	thrived	throve	thriven	43,612

Continued on next page

Table 2.A1: A tabulation of all verb forms used in this study. The Token Count column gives the sum of all the tokens for the past tense forms of the verb, both regular and irregular, in our Twitter dataset (see row (I) of Table 2.1 in Sec. 2.2).

Table 2.A1: (continued)

Verb	Regular	Irregular		Token Count
	Preterit & Past Participle	Preterit	Past Participle	
throw	threwed	threw	thrown	13,197,226
tread	treaded	trod	trodden	56,624
vex	vexed	vext	vext	139,411
wake	waked	woke	woken	30,796,918
wear	wearied	wore	worn	8,552,191
weep	weepied	wept	wept	200,690
win	winned	won	won	45,276,202
wind	winded	wound	wound	1,340,267
wring	wringed	wrung	wrung	29,141
write	writied	wrote	written, writ, wrote	23,926,025

2.7.2 DETAILS ON USER LOCATION MATCHING

To study regularization by county, we extracted location information from the user-provided location information, which was entered as free text in the user’s biographical profile. To do this, for each tweet we first checked if the location field was populated with text. If so, we then split the text on commas, and checked whether there were two tokens separated by a comma. If so, we made the assumption that it might be of the form ‘city, state’. Then we used a python package called uszipcode, which can be found here: pythonhosted.org/uszipcode/. We used the package’s method to search by city and state. If the package returned a location match, we used the returned latitude and longitude to determine which county the detected city belonged to.

The package allows for fuzzy matching, meaning the city and state do not have to be spelled correctly, and it allows for the state to be fully spelled out or be an abbreviation. In the source code of the package there was a hard-coded confidence level of 70 for the fuzzy matching. We modified the source code so that the confidence level was an input to the method, and running tests found we were satisfied with a confidence level of 91. We checked by hand the matches of 1000 tweets that this method returned a match for, 100 from each year in the dataset, and found the only potential error in these matches was when the user typed in ‘Long Island, NY’, or a similar variant. For this, the package returned Long Island City, NY, which is on Long Island, but there are multiple counties on Long Island, so the user may actually live in a different county. None of the other 1000 tweets were inappropriately or ambiguously assigned.

2.7.3 AVERAGE VERB REGULARIZATION FOR VERBS BINNED BY DATA VOLUME

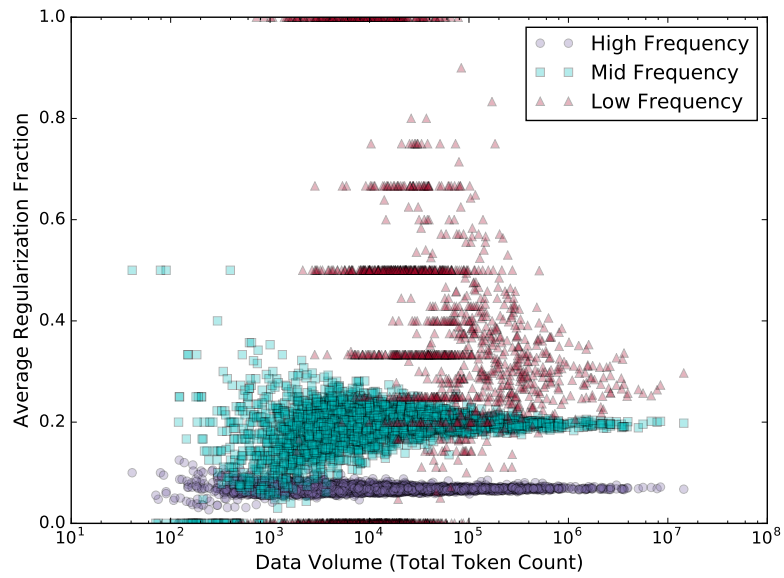


Figure 2.A1: The scatter plot of average binned verb regularization for counties. Verbs with a token count in the interval $[10^6, 10^8]$ in the Twitter dataset from row (IV) of Table 2.1 in Sec. 2.2 are considered ‘high frequency’, those in the interval $[10^4, 10^6]$ are ‘mid frequency’, and those in the interval $[10^2, 10^4)$ are ‘low frequency’. The bins contain 37, 55, and 14 verbs, respectively. For each county (with at least 40 total tokens), the average regularization fraction of the verbs in each of the three bins is calculated (if it is not empty) and plotted against the total token count for all verbs for that county.

2.7.4 USER TWEET FREQUENCY COUNT

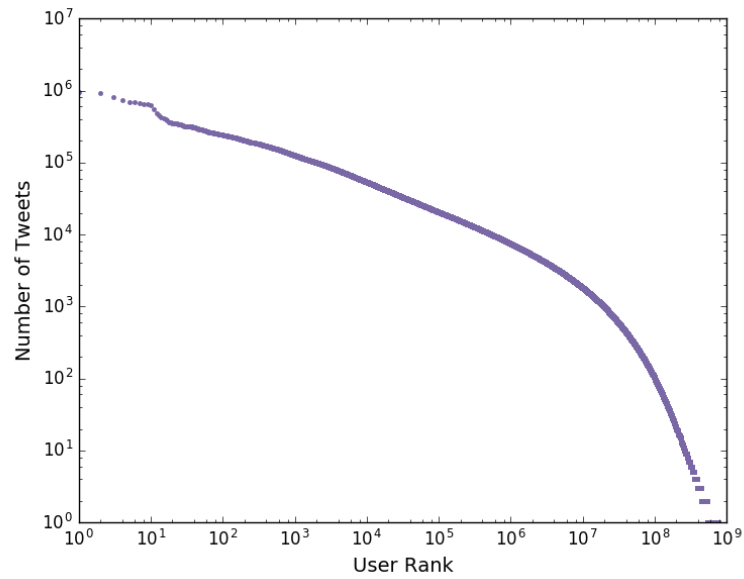


Figure 2.A2: The frequency counts of tweets by unique users in our Twitter decahose dataset (row (I) of Table 2.1 in Sec. 2.2). Users are ranked by their total number of tweets along the horizontal axis and the vertical axis gives the total number of tweets we have associated with each user's account.

CHAPTER 3

HAHAHAHAHA, DUUUUUUDE, YEEESSSS!: A TWO-PARAMETER CHARACTERIZATION OF STRETCHABLE WORDS AND THE DYNAMICS OF MISTYPINGS AND MISSPELLINGS

Stretched words like ‘heelllp’ or ‘heyyyyy’ are a regular feature of spoken language, often used to emphasize or exaggerate the underlying meaning of the root word. While stretched words are rarely found in formal written language and dictionaries, they are prevalent within social media. In this paper, we examine the frequency distributions of ‘stretchable words’ found in roughly 100 billion tweets authored over an 8 year period. We introduce two central parameters, ‘balance’ and ‘stretch’, that capture their main characteristics, and explore their dynamics by creating visual tools we call ‘balance plots’ and ‘spelling trees’. We discuss how the tools and methods we develop here could be used to study the statistical patterns of mistypings and misspellings, along with the potential applications in augmenting dictionaries, improving language processing, and in any area where sequence construction matters, such as genetics.

3.1 INTRODUCTION

Watch a soccer match, and you are likely to hear an announcer shout ‘GOOOOOOOOOAAAAAAL!!!!!!’. Stretched words, sometimes called elongated

words [1], are an integral part of spoken language, often used to modify the meaning of the base word in some way, such as to strengthen the meaning (e.g., ‘huuuuuge’), imply sarcasm (e.g., ‘suuuuure’), show excitement (e.g., ‘yeeeeesss’), or communicate danger (e.g., ‘nooooooooooooo’). We will refer to words that are amenable to such lengthening as ‘stretchable words’.

However, despite their being a fundamental part of spoken language, stretched words are rarely found in literature and lexicons: There is no ‘hahahahahaha’ in the Oxford English Dictionary [2]. With the advent and rise of social media, stretched words have finally found their way into large-scale written text.

With the increased use of social media comes rich datasets of a linguistic nature, granting science an unprecedented opportunity to study the everyday linguistic patterns of society. As such, in recent years there have been a number of papers published that have used data from social media platforms, such as Twitter, to study different aspects of language [3–9].

In this paper, we use an extensive set of social media messages collected from Twitter—tweets—to investigate the characteristics of stretchable words used in this particular form of written language. The tools and approach we introduce here have many potential applications, including the possible use by dictionaries to formally include this intrinsic part of language. The online dictionary Wiktionary has already discussed the inclusion of some stretched words and made a policy on what to include [10, 11]. Other potential applications include the use by natural language processing software and toolkits, and by Twitter to build better spam filters.

We structure our paper as follows: In Sec. 3.2, we detail our dataset and our method of collecting stretchable words and distilling them down to their ‘kernels’. In Sec. 3.3.1, we examine the frequency distributions for lengths of stretchable words. We quantify two independent properties of stretchable words: Their ‘balance’ in Sec. 3.3.2 and ‘stretch’ in Sec. 3.3.3. In Sec. 3.3.4, we develop an investigative tool, ‘spelling trees’, as a means of

visualizing stretchable words involving a two-character repeated element. We comment on mistypings and misspellings in Sec. 3.3.5. Finally, in Sec. 3.4, we provide concluding remarks.

3.2 DESCRIPTION OF THE DATASET AND METHOD FOR EXTRACTING STRETCHED WORDS

The Twitter dataset we use in this study comprises a random sample of approximately 10% of all tweets (the ‘gardenhose’ API) from 9 September 2008 to 31 December 2016. We limited our scope to tweets that either were flagged as an English tweet or not flagged for any language. All tweets in this time period have a maximum length of 140 characters. To collect stretchable words, we begin by making all text lowercase and collecting all tokens within our dataset from calendar year 2016 that match the Python regular expression `r'(\b\w*(\w)(?:\2|\3){28,}\w*\b)'`. This pattern will collect any token with at least 30 characters that has a single character repeated at least 29 times consecutively, or two different characters that are repeated in any order at least 28 times, for a total of at least 30 consecutive repeated occurrences of the two characters. The choice of 28 in the regular expression is a threshold we chose with the goal of limiting our collection to tokens of words that really do get stretched in practice.

After collecting these tokens, we remove any that contains a character that is not a letter ([a-z]), and distill each remaining token down to its ‘kernel’. Table 3.1 gives a few examples of this distillation process. Proceeding along the token from left to right, whenever any pair of distinct letters, l_1 and l_2 , occur in the token where 1. l_1 occurs followed by any sequence of l_1 and l_2 of total length at least three, and 2. such that l_1 and l_2 each occur at least twice in the sequence, we replace the sequence with the ‘two-letter element’ (l_1l_2). For example, see the first cell in Table 3.1.

1.	hahahahaahahaa → (ha)
2.	gooooooooaaaaal → g[o][a]l
3.	ggggooooooooaaaallllll → [g][o]aaaaallllll → [g][o][a]llll
4.	bbbbbaaaaaabbbbbbyyyyyyy → [b][a][b]yyyyyy → [b][a][b][y]
5.	awawawaaawwwwessssommmmmeeeeee → (aw)essssommmmmeeeeee → (aw)essso[m][e] → (aw)e[s]o[m][e]

Table 3.1: Examples of distilling tokens down to their kernels. The first line of each cell is the example token. The following lines show the result after every time a replacement of characters by the corresponding single letter element(s) or double letter element is made by the code, in order. The final line of each cell gives the resulting kernel for each example.

Exceptions to the preceding are: 1. The case where the sequence is a series of l_1 followed by a series of l_2 , which is replaced with the pair of ‘single letter elements’ $[l_1][l_2]$. For example, see the second cell in Table 3.1. And 2., the case where the sequence is a series of l_1 followed by a series of l_2 followed by a series of l_1 , which is replaced with $[l_1][l_2][l_1]$. For example, see the first step in the fourth cell of Table 3.1 where ‘bbbbbaaaaaabbbbb’ is replaced with $[b][a][b]$.

Following this process, whenever a single letter, l_3 , occurs two or more times in a row, we replace the sequence with the single letter element $[l_3]$. For example, see the last step of the fourth cell in Table 3.1 where ‘yyyyyy’ is replaced with $[y]$, or the last step in the fifth cell where ‘sssss’ is replaced with $[s]$.

We collected tokens in batches of seven consecutive days at a time throughout 2016 (with the last batch being only two days). If a kernel is not found in more than one batch, or within the same batch but from at least two distinct stretched words, then it is removed from consideration.

Different but related stretched words (that is, different stretched words, but both stretched out versions of the same base word) may distill to different kernels. We combine these into a single kernel for each word such that it covers all cases observed in the collected tokens. For example, the kernels $g[o]a[l]$ and $go[a][l]$ would be combined as $g[o][a][l]$. The kernels $h[a]$ and (ha) would be combined as (ha) .

After processing our dataset, we obtained a collection of 7,526 kernels. We then represented each kernel with a corresponding regular expression and collected all tokens in our entire gardenhose dataset that matched the regular expressions. To go from the kernel to the regular expression, we replaced $]$ with $]+$, replaced (l_1l_2) with $l_1[l_1l_2]^*l_2[l_1l_2]^*$, and we surrounded the kernel with word boundary characters $\backslash b$. So, for example, the kernel $g[o][a][l]$ goes to the Python regular expression `r'\bg[o]+[a]+[l]+\b'` and the kernel (ha) goes to the Python regular expression `r'\bh[ha]*a[ha]*\b'`.

Once we collected all tokens matching our kernels, we carried out a final round of thresholding on our kernel list, removing those with the least amount of data and least likely to represent a bona fide stretchable word. For each kernel, we calculated the token count as a function of token length (number of letters) for all tokens matching that kernel. For example, Fig. 3.1 gives the plot of the token count distribution for the kernel (to) . Then, with the token counts in order by increasing token length, as in Fig. 3.1, we found the location of the largest drop in the \log_{10} of token counts between two consecutive values within the first 10 values. We call the words with lengths coming before the location of the drop ‘unstretched’ versions of the kernel and those that come after ‘stretched’ versions. For most kernels, the largest drop will be between the first and second value. However, for some kernels this drop occurs later. For example, in Fig. 3.1 we see that for the kernel (to) , which covers both the common words ‘to’ and ‘too’, this drop is between the second and third value (between tokens of length three and four). Thus, the unstretched versions of (to)

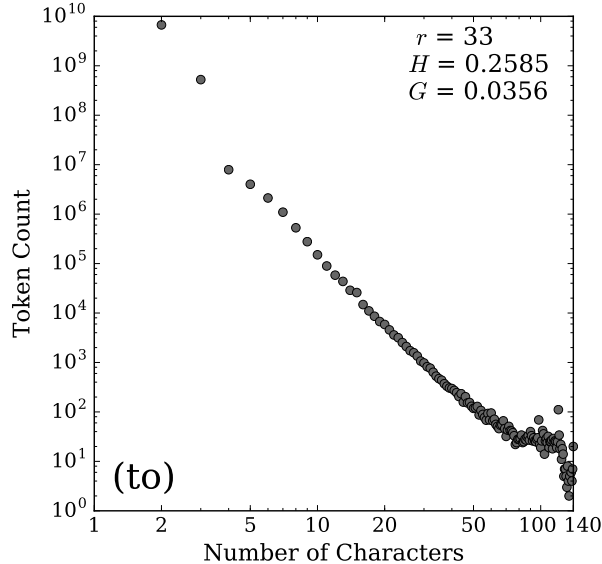


Figure 3.1: Token count distribution for the kernel (to). The horizontal axis represents the length (number of characters) of the token and the vertical axis gives the total number of tokens of a given length that match this kernel. The included statistics give the kernel rank, r (see Sec. 3.2), the value of the balance parameter (normalized entropy, H ; see Sec. 3.3.2), and the value of the stretch parameter (Gini coefficient, G ; see Sec. 3.3.3) for this kernel. The large drop between the second and third points denotes the change from ‘unstretched’ versions of (to), located to the left of this drop, to ‘stretched’ versions of (to), located to the right of this drop.

are represented by the first two points in Fig. 3.1, with the remaining points representing stretched versions of (to).

We then ranked the kernels by the sum of the token counts for their stretched versions. Fig. 3.2 shows this sum as a function of rank for each kernel. Inspired by the idea of a cutoff frequency [12], we estimate a cutoff rank for the kernels. Using the values between rank 10 and 10^3 , we found the regression line between the \log_{10} of the ranks and the \log_{10} of the summed token counts (straight line, Fig. 3.2). We calculated the cutoff as the first rank (after 10^3) where the summed token count is less than $1/10$ of the corresponding value of the regression line. This occurs at rank 5,164, which is shown by the vertical dashed line in Fig. 3.2. For the remainder of this study, we used the kernels with rank preceding this cutoff, giving us a total of 5,163 kernels, and, unless otherwise specified, a kernel’s ‘rank’, r , refers to

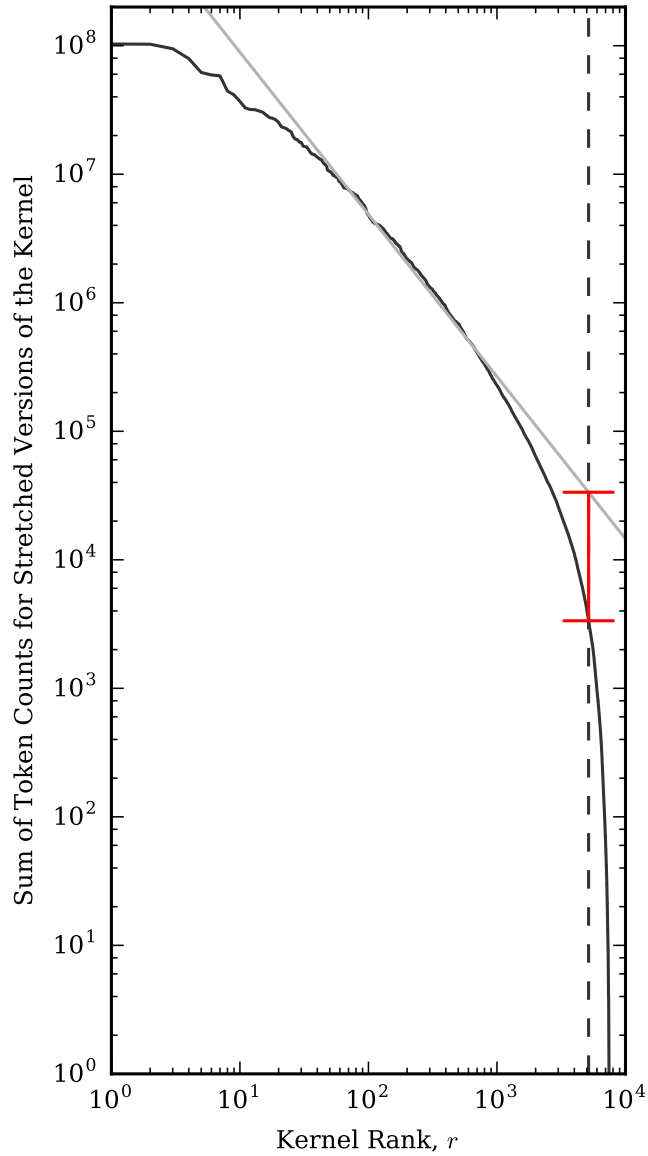


Figure 3.2: Total token counts for stretched versions of all kernels. Kernels are ranked by their descending total token count along the horizontal axis. The diagonal line gives the regression line calculated using the values between ranks 10 and 10^3 . The vertical dashed line denotes the first location after rank 10^3 where the distribution drops below $1/10$ of the corresponding value of the regression line, denoted by the red interval, giving the cutoff rank for the final threshold to decide which kernels to include in the study.

the rank found here. See Online Appendix A at <http://compstorylab.org/stretchablewords/> for a full list of kernels meeting our thresholds, along with their regular expressions and other statistics discussed throughout the remainder of this paper.

3.3 ANALYSIS AND RESULTS

3.3.1 DISTRIBUTIONS

For each kernel, we plotted the corresponding distribution of token counts as a function of token length. Most of the distributions largely follow a roughly power-law shape. For example, Fig. 3.3 gives the frequency distribution for the kernel `[g][o][a][l]`. From the elevated frequency of the first dot, we can see that the unstretched word ‘goal’ is used about two orders of magnitude more frequently than any stretched version. After the first point, we see a rollover in the distribution, showing that if users are going to stretch the word, they are more likely to include a few extra characters rather than just one. We also see that there are some users who indeed fill the 140-character limit with a stretched version of the word ‘goal’, and the elevated dot there suggests that if users get close to the character limit, they are more likely to fill the available space. The other dots elevated above the trend represent tokens that likely appear in tweets that have a small amount of other text at the beginning or end, such as a player name or team name, or, more generally, a link or a user handle.

In Fig. 3.4, we show the frequency distribution for the kernel `(ha)` as an example of a distribution for a two-character repeated element. For this distribution we observe an alternating up and down in frequency for even length tokens and odd length tokens. This behaviour is typical of distributions with a two-character repeated element, likely resulting from an intent for these tokens to be a perfect alternating repetition of ‘h’ and ‘a’, hahaha. . . , to represent laughter. Under this assumption, the correct versions will be even length. Then,

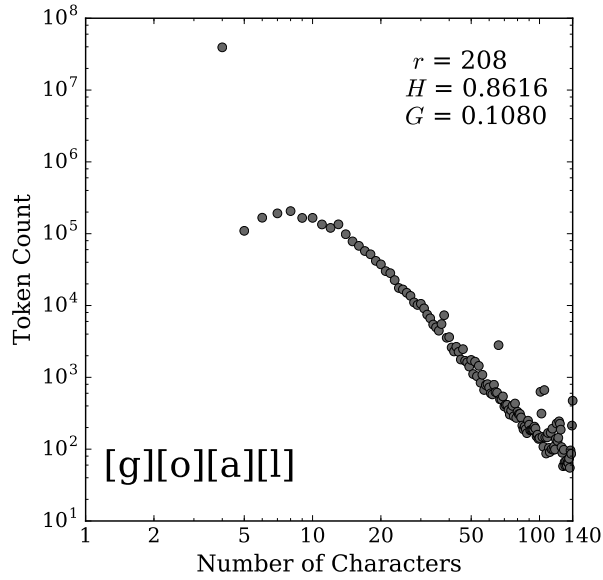


Figure 3.3: Token count distribution for the kernel `[g][o][a][l]`. The horizontal axis represents the length (number of characters) of the token and the vertical axis gives the total number of tokens of a given length that match this kernel. See Fig. 3.1 caption for details on the included statistics. The base version of the word appears roughly 100 times more frequently than the most common stretched version.

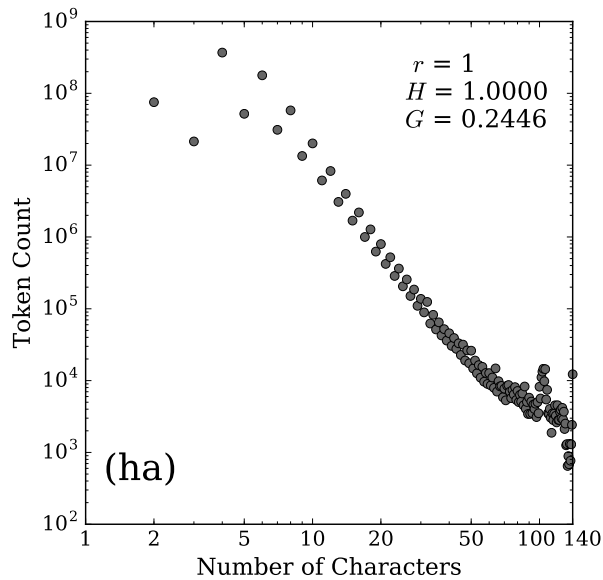


Figure 3.4: Token count distribution for the kernel `(ha)`. The horizontal axis represents the length (number of characters) of the token and the vertical axis gives the total number of tokens of a given length that match this kernel. See Fig. 3.1 caption for details on the included statistics.

any incorrect version could be odd or even length depending on the number of mistakes. We look at mistakes further in Sec. 3.3.5.

We note that there is also an initial rollover in this distribution, showing that the four-character token, with dominant contributor ‘haha’, is the most common version for this kernel. We also again see some elevated counts near the tail, including for 140 characters, along with some depressed counts just short of 140, which again suggests that when users approach the character limit with stretched versions of (ha), they will most likely fill the remaining space. We did not perform a detailed analysis of this area, but it is likely that the other elevated points near the end are again due to the inclusion of a link or user handle, etc. Similarly, the general flattening of the distribution’s right tail is likely a result of random lengths of short other text combined with a stretched word that fills the remaining space.

Similar distributions for each kernel can be found in Online Appendix B at <http://compstorylab.org/stretchablewords/>.

3.3.2 BALANCE

For each kernel, we measure two quantities: 1. The balance of the stretchiness across characters, and 2. the overall stretchiness of the kernel. To measure balance, we calculate the average stretch of each character in the kernel across all the tokens within a bin of token lengths. By average stretch of a character, we mean the average number of times that character appears. Fig. 3.5 shows the balance for the kernel [g][o][a][l] partitioned into bins of logarithmically increasing sizes of length. The horizontal dashed lines represent the bin edges. The distance between the solid diagonal lines represents the average stretch, or average number of times each character was repeated, and are plotted in the same order that they appear in the kernel. From this figure we see that ‘g’ is not stretched much on average, ‘o’ is stretched the most, and ‘a’ and ‘l’ are both stretched around 2/3 as much as ‘o’.

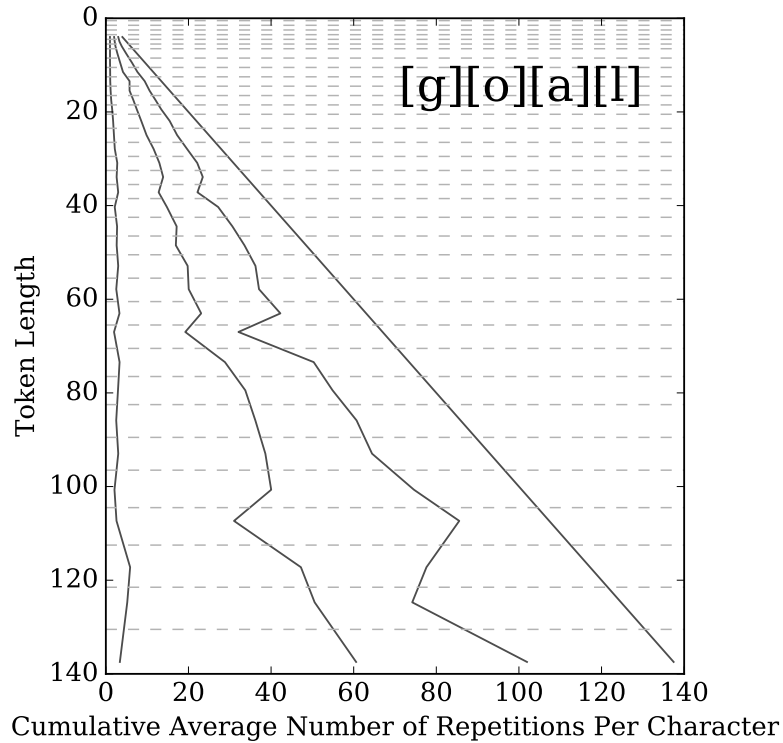


Figure 3.5: Balance plot for the kernel $[g][o][a][l]$. The vertical axis represents the length (number of characters) of tokens, and is broken into bins of lengths, with boundaries denoted by horizontal dashed lines, which increase in size logarithmically. For all the tokens that match the kernel and fall within a bin of lengths, the average number of times each character was stretched in those tokens was calculated, and is shown on the plot as the distance between two solid lines in the same order as in the kernel. Thus, for a given bin, the distance between the vertical axis and the first solid line is the average stretch for the letter ‘g’, the distance between that first line and the second line is the average stretch for the letter ‘o’, and so on. For example, the last bin contains tokens with lengths in the interval $[131, 140]$, with average length roughly 137. On average, tokens falling in this most celebratory bin contain roughly 3 ‘g’s, 57 ‘o’s, 41 ‘a’s, and 36 ‘l’s.

When part of the kernel is a two-letter element of the form (l_1l_2) , we still count the number of occurrences of l_1 and l_2 corresponding to this element in the kernel separately, even though the letters can be intermingled in the stretched word. When we display the results, we display it in the same order that the letters appear in the kernel. So in Fig. 3.6, which shows the results for the kernel (ha), the first space represents the average stretch for ‘h’ and the second space is for ‘a’. From this figure, we can see that the stretch is almost perfectly balanced between the two letters on average.

Similar balance plots can be found for each kernel in Online Appendix C at <http://compstoriylab.org/stretchablewords/>. In general, for these balance plots, we stop plotting at the first bin with no tokens, even if later bins may be nonempty.

For each kernel, we also calculate an overall measure of balance. To do this, we begin by binning the tokens by length. Then, for each bin (containing tokens longer than the kernel) we calculate the average stretch for each character across tokens within the bin as before. Then, we subtract one from each of these values (removing the contribution from each base character; counting just the number of times each character was repeated) and normalize the values so they sum to 1 and can be thought of like probabilities. We then average the probabilities across the bins, weighing each bin equally, and compute the normalized entropy, H , of the averaged probabilities as our overall measure of balance. This measure is such that if each character stretches the same on average, the normalized entropy is 1, and if only one character in the kernel stretches, the normalized entropy is 0. Thus, higher entropy corresponds with more balanced words. (For a comparison with an alternate entropy measure where tokens contribute equally rather than equally weighing each length bin, and an explanation of the different corresponding views, see Appendix 3.7.1.)

Fig. 3.7 shows two ‘jellyfish plots’ [13] for balance. Fig. 3.7A is the version containing all words and for Fig. 3.7B we remove the words that have a value of 0 for entropy. The top of the left plot in Fig. 3.7 shows the frequency histogram of the normalized entropy for each

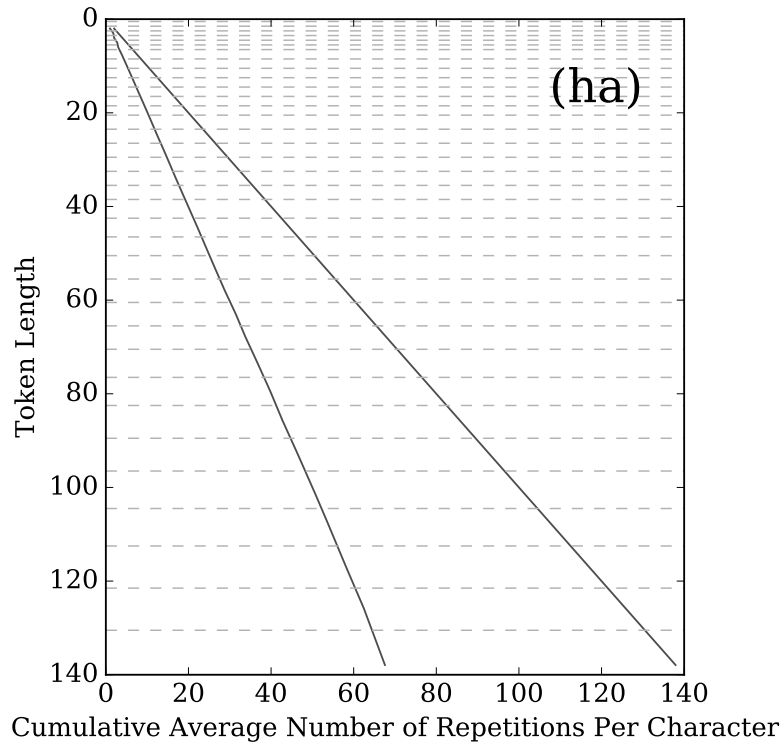


Figure 3.6: Balance plot for the kernel (ha). See the Fig. 3.5 caption for plot details. For two-letter elements, even though the letters can alternate within a given token, we still count the number of occurrences for each letter separately and display the average number of total repetitions in the same order as the letters appear in the kernel. Thus, for a given bin, the distance between the vertical axis and the first line is the average number of times the letter ‘h’ occurred in the tokens, and the distance between that first line and the second line is the average number of times the letter ‘a’ occurred in the token. This plot clearly shows that (ha) is well balanced across all bins of token lengths.

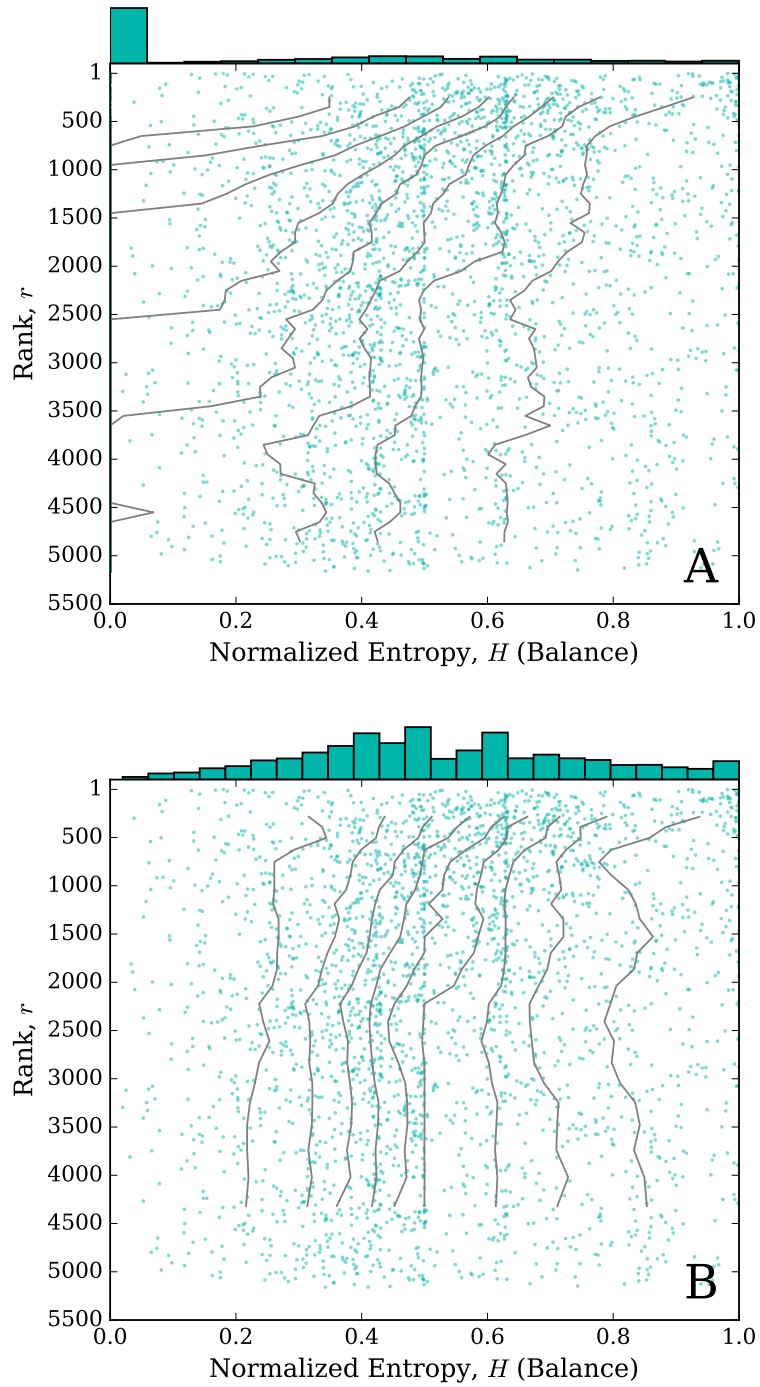


Figure 3.7: Jellyfish plots for kernel balance for (A) all kernels, and (B) excluding kernels with entropy exactly 0. Corresponding histograms are given at the top of each plot. Kernels are plotted vertically by their rank, r , and horizontally by their balance as given by normalized entropy, H , where larger entropy denotes increased balance. The deciles 0.1, 0.2, \dots , 0.9 are calculated for rolling bins of 500 kernels and are plotted as the ‘tentacles’.

kernel. The spike containing value 0 comes largely from kernels where only one character stretches, giving that kernel an entropy of exactly 0. The main plot shows the normalized entropy values as a function of word rank, where rank is given, as before, by the sum of stretched token counts. The ‘tentacles’ give rolling deciles. That is, for rolling bands of 500 words by rank, the deciles 0.1, 0.2, \dots , 0.9 are calculated for the entropy values, and are represented by the solid lines. These plots allow us to see how stable the distribution is across word ranks.

We can see from Fig. 3.7A that the distribution largely shifts towards smaller entropy values with increasing rank, mostly drawn in that direction by the kernels with only a single repeated letter and entropy exactly 0. For Fig. 3.7B, we remove all kernels with entropy 0. Everything else remains the same, including the rank of each kernel (we skip over ranks of removed kernels) and the rolling bands of 500 kernels for percentile calculations still have 500 kernels, and thus tend to be visually wider bands. In contrast to Fig. 3.7A, we now see a small left-shift in the earlier ranks, and then the distribution tends to stabilize for lower ranks. This shows that the highest ranked kernels tend to have a larger entropy, meaning the stretch of the kernel is more equally balanced across all characters. We also see that not many of the high ranked words stretch with just one character. It appears that these kernels that stretch in only a single character become more prevalent in the lower ranks.

Table 3.2 shows the kernels with the ten largest entropies and Table 3.3 shows those with the ten smallest nonzero entropies. We observe that the kernels with largest entropies are mostly of the form (l_1l_2) and are almost perfectly balanced. The least balanced kernels tend to be more recognizable English or Spanish words and names, with one exclamation also appearing in the bottom ten.

	H	Kernel	Example token
1	0.99998	(kd)	kdkdkdkdkdkdkd
2	0.99998	(ha)	hahahahaha
3	0.99997	[i][d]	iiiiiddddd
4	0.99997	(ui)	uiuiuiuiui
5	0.99997	(ml)	mlmlmlmlmlmlml
6	0.99995	(js)	jsjsjsjsjsj
7	0.99990	[e][t]	eeeeettttt
8	0.99988	(ox)	oxoxoxoxox
9	0.99980	(xq)	xqxqxqxqxqxqxq
10	0.99971	(xa)	xaxaxaxaxaxa

Table 3.2: Top 10 kernels by normalized entropy, H .

	H	Kernel	Example token
1	0.01990	[b][o][b]ies	boooooobies
2	0.02526	[d][o][d]e	dooooooode
3	0.03143	infini[t][y]	infinityyyyy
4	0.03342	che[l]se[a]	chelseaaaaaa
5	0.03587	tay[l]o[r]	taylrrrrrr
6	0.03803	f(re)	freeeeeeeeeee
7	0.03930	[f]ai[r]	fairrrrrrrr
8	0.05270	regr[e][s][e]	regreseeeee
9	0.05271	herm[a][n][a]	hermanaataaaaa
10	0.05323	sq[u][e]	squueeeeeee

Table 3.3: Bottom 10 kernels by normalized entropy, H .

3.3.3 STRETCH

To measure overall stretchiness for a kernel we calculated the Gini coefficient, G , of the kernel’s token length frequency distribution. (For a comparison with another possible measure of stretch, see Appendix 3.7.2.) If the distribution has most of its weight on the short versions and not much on stretched out versions, then the Gini coefficient will be closer to 0. If more tokens are long and the kernel is stretched longer more often, the Gini coefficient will be closer to 1. Fig. 3.8 gives the jellyfish plot for the Gini coefficient for each kernel. The horizontal axis has a logarithmic scale, and the histogram bins have logarithmic widths. From this plot, we see that the distribution for stretch is quite stable across word ranks, except for perhaps a slight shift towards higher Gini coefficient (more stretchiness) for the highest ranked kernels.

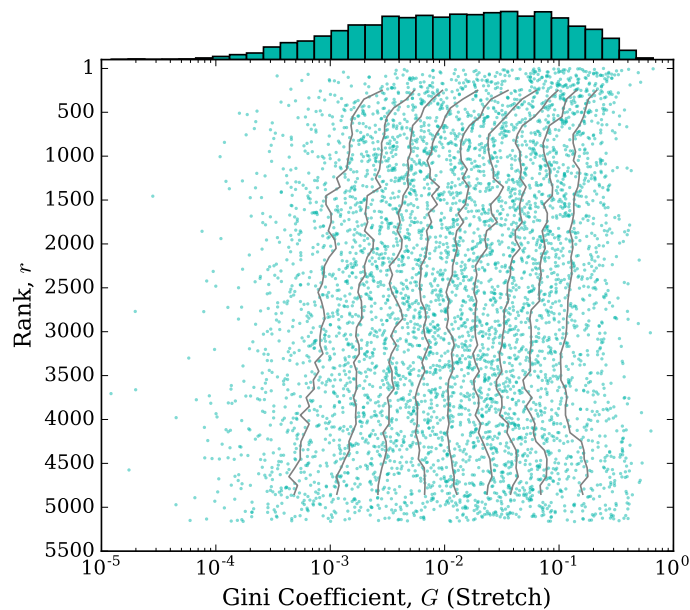


Figure 3.8: Jellyfish plots for kernel stretch as measured by the Gini coefficient, G , of its token count distribution, where higher Gini coefficient denotes increased stretch. The histogram is given at the top of the plot (with logarithmic width bins). Kernels are plotted vertically by their rank, r , and horizontally (on a logarithmic scale) by their stretch. The deciles 0.1, 0.2, \dots , 0.9 are calculated for rolling bins of 500 kernels and are plotted as the ‘tentacles’.

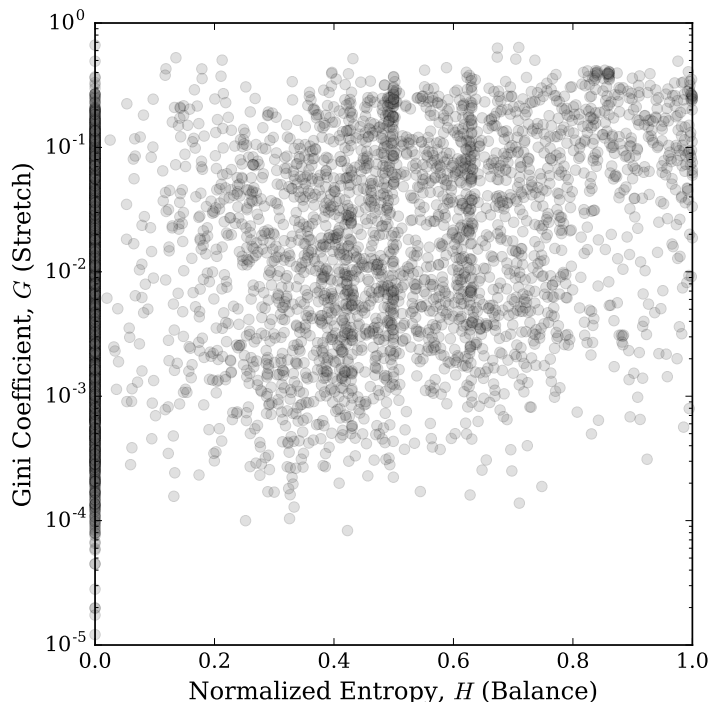


Figure 3.9: Kernels plotted in Balance-Stretch parameter space. Each kernel is plotted horizontally by the value of its balance parameter, given by normalized entropy, H , and vertically (on a logarithmic scale) by its stretch parameter, given by the Gini coefficient, G , of its token count distribution. Larger entropy implies greater balance and larger Gini coefficient implies greater stretch.

Table 3.4 shows the top 10 kernels ranked by Gini coefficient and Table 3.5 shows the bottom 10. The top kernel is [k], which represents laughter in Portuguese, similar to (ha) in English (and other languages). Containing a single letter, [k] is easier to repeat many times, and does not have an unstretched version that is a common word. We also see (go)[l] on the list, where ‘gol’ is Spanish and Portuguese for ‘goal’. Interestingly, (go)[l] has a much higher Gini coefficient ($G = 0.5171$) than [g][o][a][l] does ($G = 0.1080$). The kernels with lowest Gini coefficient all represent regular words and all allow just one letter to stretch, which does not get stretched much.

In Fig. 3.9, we show a scatter plot of each kernel where the horizontal axis is given by the measure of balance of the kernel using normalized entropy, and the vertical coordinate

	G	Kernel	Example token
1	0.66472	[k]	kkkkkkkkkkkkkkkk
2	0.63580	[w][v][w]	wwwvvvvvvvvvvvvvv
3	0.62843	[m][n][m]	mmmmmmmmmmmmmm
4	0.53241	[o][c][o]	oooooooooco
5	0.52577	wa(ki)	wakikikikikikik
6	0.51706	(go)[l]	gooooooooool
7	0.51273	[m][w][m]	mmmmmwmmmmmmmm
8	0.50301	galop[e]ir[a]	galopeeeaira
9	0.50193	[k][j][k]	kkkkkjkkkkkkkkkk
10	0.49318	[i][e][i]	iiiiieiiiiii

Table 3.4: Top 10 kernels by Gini coefficient, G .

	G	Kernel	Example token
1	0.00001	am[p]	ampppppppppp
2	0.00002	m[a]kes	maaaaaaaakes
3	0.00002	fr[o]m	frooooooooooom
4	0.00002	watch[i]ng	watchiiiiing
5	0.00003	w[i]th	wiiiiiiith
6	0.00004	pla[y]ed	playyyyyyed
7	0.00004	s[i]nce	siiiiince
8	0.00006	eve[r]y	everrrrrrrry
9	0.00006	manage[r]	managerrrr
10	0.00007	learnin[g]	learninggggg

Table 3.5: Bottom 10 kernels by Gini coefficient, G .

is given by the measure of stretch for the kernel using the Gini coefficient. Thus, this plot positions each kernel in the two-dimensional space of balance and stretch. We see that the kernels spread out across this space and that these two dimensions capture two independent characteristics of each kernel.

We do note that there are some structures visible in Fig. 3.9. There is some roughly vertical banding. In particular, the vertical band at $H = 0$ is from kernels that only allow one character to stretch and the vertical band near $H = 1$ is from kernels where all characters are allowed to stretch and do so roughly equally, which especially occurs with kernels that are a single two-letter element. Fainter banding around $H \approx .43$, $H \approx .5$, and $H \approx .63$ can also be seen. This largely comes from kernels of length 5, 4, and 3, respectively, that allow exactly two characters to stretch and those characters stretch roughly equally. If the stretch was perfectly equal, then the normalized entropy in each respective case would be $H = 1/\log_2(5) \approx .43$, $H = 1/\log_2(4) = .5$, and $H = 1/\log_2(3) \approx .63$.

3.3.4 SPELLING TREES

So far we have considered frequency distributions for kernels by token length, combining the token counts for all the different words of the same length matching the kernel. However, different tokens of the same length may of course be different words—different stretched versions—of the same kernel. For kernels that contain only single letter elements, these different versions may just have different amounts of the respective stretched letters, but all the letters are in the same order. However, for kernels that have two-letter elements, the letters can change order in myriad ways, and the possible number of different stretched versions of the same length becomes much larger and potentially more interesting.

In order to further investigate these intricacies, we introduce ‘spelling trees’ to give us a visual method of studying the ways in which kernels with two-letter elements are generally expanded. Fig. 3.10 gives the spelling tree for the kernel (ha). The root node is the first

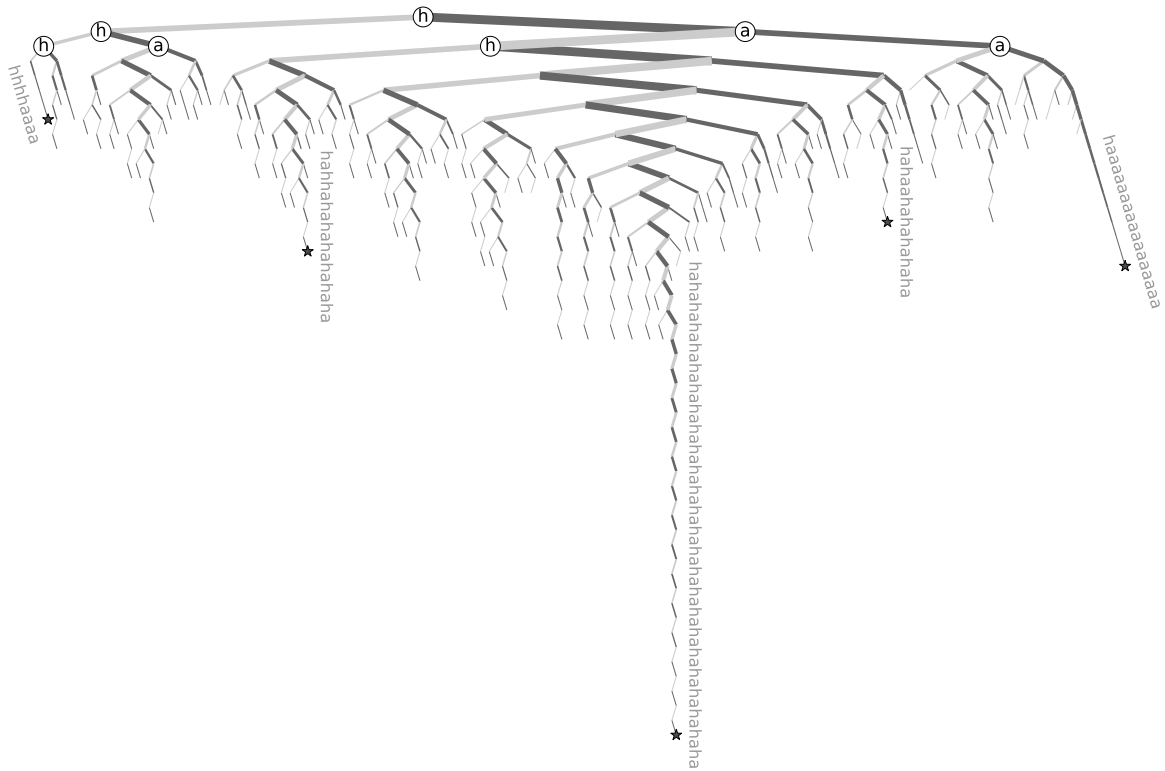


Figure 3.10: Spelling tree for the kernel (ha). The root node represents ‘h’. From there, branching to the left (light gray edge) is equivalent to appending an ‘h’. Branching to the right (dark gray edge) is equivalent to appending an ‘a’. The edge width is logarithmically related to the number of tokens that pass along that edge when spelled out. A few example words are annotated, and their corresponding nodes are denoted with a star. This tree was trimmed by only including words with a token count of at least 10,000. The code used to create the figures for these spelling trees is largely based on the algorithm presented by Wetherell and Shannon [14]. We note that Mill has written a more recent paper based largely on this earlier work specialized for Python [15], and an implementation for it as well [16], but they both contain algorithmic bugs (detailed in Appendix 3.7.3).

letter of the two-letter element, which in this case is ‘h’. Then, recursively, if the next letter in the word matches the first letter of the pair, it branches left, represented by a lighter gray edge, and if it matches the second letter of the pair then it branches right, represented by a darker gray edge. This branching continues until the word is finished. The first few nodes are highlighted with the letter corresponding to that point of the tree. The edge weights are logarithmically related to the number of tokens flowing through them. In Fig. 3.10, a few nodes, denoted by stars, are annotated with the exact word to which they correspond. The annotated nodes are all leaf nodes, but words can, and most do, stop at nodes that are not leaves. We also trimmed the tree by only including words that have a token count of at least 10,000. This threshold of pruning reveals the general pattern while avoiding making the spelling tree cluttered.

The spelling tree for (ha) has a number of interesting properties. Most notable among them is the self-similar, fractal-like structure. The main branch line dropping down just right of center represents the perfect alternating sequence ‘hahahahaha...’, as shown by the annotated example at the leaf of this line. There are also many similar looking subtrees that branch off from this main branch that each have their own similar looking main branch. These paths that follow the main branch, break off at one location, and then follow the main branch of a subtree represent words that are similar to the perfect alternating laugh, but either have one extra ‘h’ (if the branch veers left) or one extra ‘a’ (if the branch leads right). For example, the middle left annotation shows that the fourth letter was an extra ‘h’, and then the rest of the word retained an accurate alternating pattern. This word, ‘hahhahahahahahaha’, appeared 13,894 times in our dataset.

The tree also shows that ‘haaaaa...’ is a strong pattern, as can be seen farthest right in the (ha) spelling tree. The subtrees on the right show that users also start with the back and forth pattern for a stretch, and then finish the word with trailing ‘a’s. Many other patterns also appear in this tree, and additional patterns are occluded by our trimming of

the tree, but likely most of these come from users trying to follow one of the patterns we have already highlighted and introducing mistypings.

We made similar trees for every kernel that had a single occurrence of a two-letter element, where the tree represents just the section of word that matches the two-letter element. These trees are trimmed by only including words that have a token count of at least the fourth root of the total token count for the stretched tokens.

Fig. 3.11 gives eight more examples of these spelling trees. The trees for (ja) and (xo) have many of the same characteristics as the tree for (ha), as do most of the trees for kernels that are a two-letter element where tokens predominantly alternate letters back and forth. For the tree for (xo), the pattern where the first letter of the two-letter element is stretched, followed by the second letter being stretched, such as ‘xxxxxoooo’, is more apparent. This type of pattern is even more notable in the trees for (aw), and especially (fu). The tree for (mo) has stretched versions for both ‘mom’ and ‘moo’. Similarly, the tree for h(er) shows stretched versions of both ‘her’ and ‘here’, where we see that both ‘e’s and the ‘r’ all get stretched. In the tree for (to), the word ‘totoo’ has a much larger token count than words stretched beyond that (noticeable by the fact that the edges leaving that node are much smaller than the edge coming in). The word ‘totoo’ is Tagalog for ‘true’. Finally, every example tree here does show the back and forth pattern to at least some extent. All of the trees created are available for viewing in Online Appendix D at <http://compstorylab.org/stretchablewords/>.

3.3.5 MISTYPINGS AND MISSPELLINGS

Mistypings appear often in tweets and we see evidence of them in stretched words. For example, the kernel n[o](io) is likely a result of mistypings of n[o]. On at least some platforms, holding down the key for a letter does not make that letter repeat, so one must repeatedly press the same key. For the standard QWERTY keyboard layout, the letter ‘i’

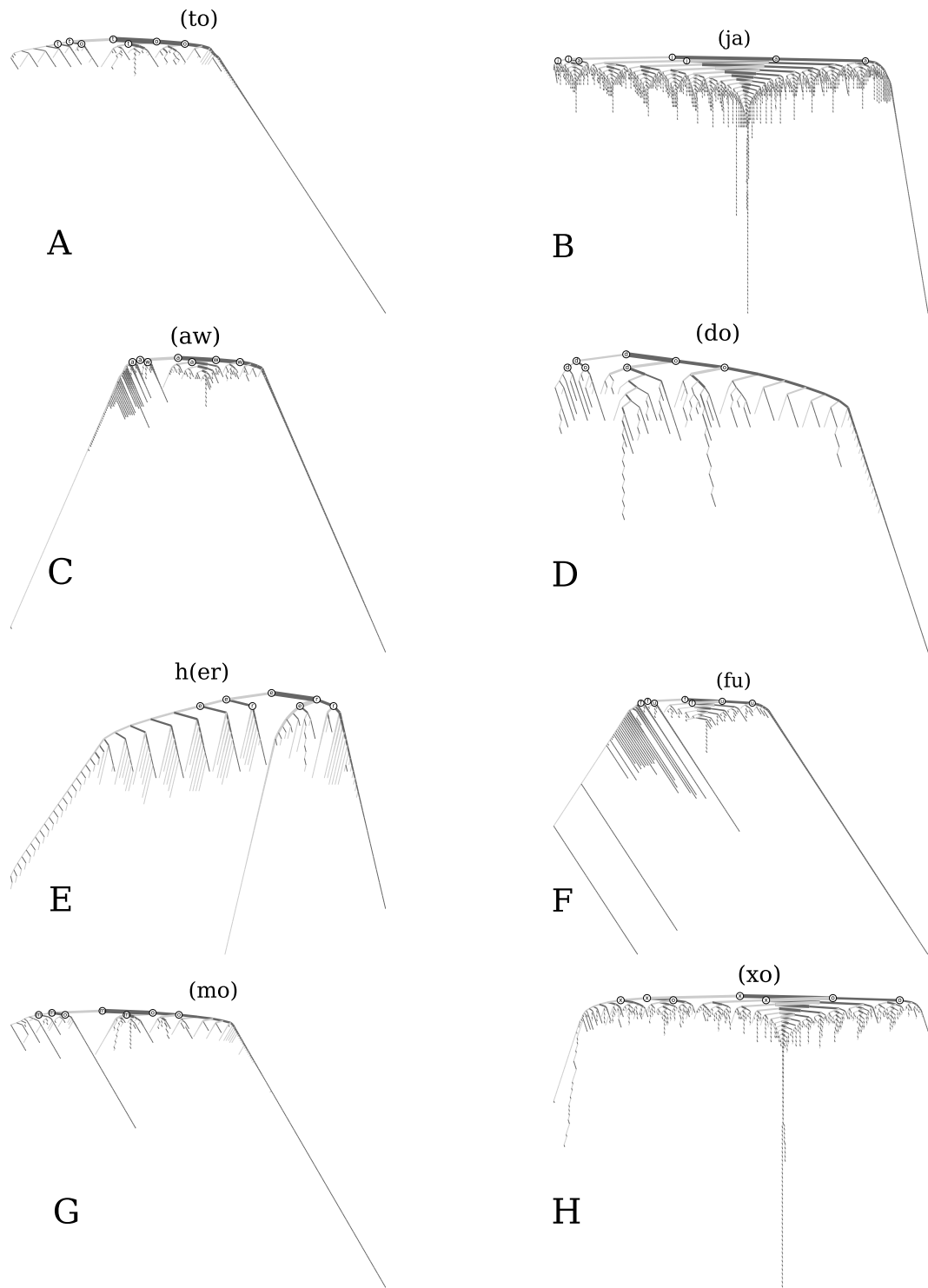


Figure 3.11: A collection of example spelling trees. From left to right, top to bottom, trees for the kernels (to), (ja), (aw), (do), h(er), (fu), (mo), and (xo).

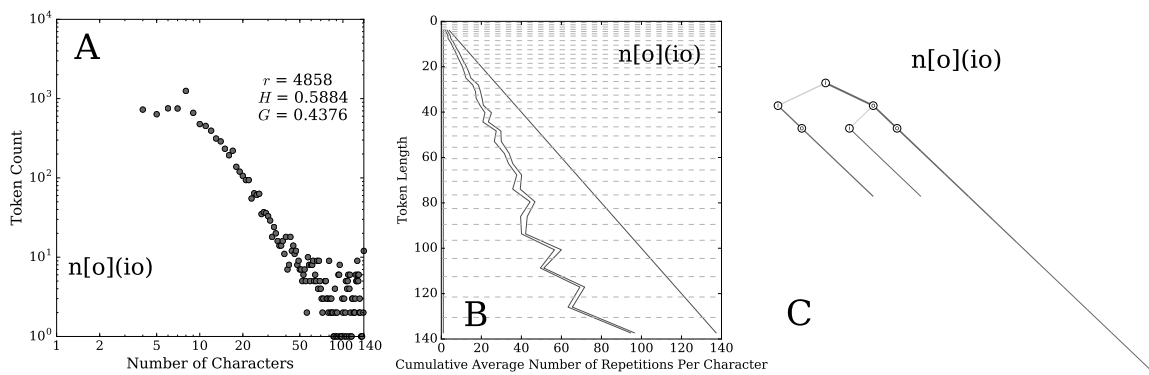


Figure 3.12: (A) Token count distribution, (B) balance plot, and (C) spelling tree for the kernel $n[o](io)$. In general, these types of plots offer diagnostic help when studying mistypings. In this case, they provide evidence towards the conclusion that the words that match this kernel were likely meant to be stretched versions of the word ‘no’ with a few mistaken ‘i’s included. Note that ‘i’ is next to ‘o’ on a standard QWERTY keyboard.

is next to the letter ‘o’, so it would be easy to accidentally press the letter ‘i’ occasionally instead of ‘o’ when trying to repeat it many times, especially on the small keyboards accompanying mobile phones. This sort of thing could lead to a kernel like $n[o](io)$ when users try to stretch the word ‘no’. Similarly, the letters ‘a’ and ‘s’ are next to each other on a QWERTY keyboard, so a kernel like $(ha)s(ha)(sh)(ah)$ likely comes from mistypings of the much simpler kernel (ha) .

However, it is not always clearly apparent if a kernel is from mistypings or on purpose, or perhaps comes as a result of both. For example, the letter ‘b’ is close to the letter ‘h’, so the kernel $(ha)b(ah)$ could come from mistypings of (ha) . But, this form could also be intentional, and meant to represent a different kind of laughter. For example, $(ba)(ha)$ is a highly ranked kernel (rank 211) representing a comedically sinister kind of laughter. Similarly, (ja) is a core component of laughter in Spanish, but ‘j’ is next to ‘h’ on the QWERTY keyboard, so it is not apparent if a kernel like $(ha)j(ah)(ja)(ha)$ comes from mistypings or from switching back and forth between English and Spanish as the word stretches.

3.4 CONCLUDING REMARKS

In this paper, we have studied stretched words, which are often used in spoken language. Until the advent of social media, stretched words were not prevalent in written language and largely absent from dictionaries. The area of stretchable language is rich, and we have discovered that these words span at least the two-dimensional parameter space of balance and stretch.

The tools we have developed not only help uncover the hidden dynamics of stretchable words, but can be further applied to study phenomena such as mistypings and misspellings, and possibly more. Online dictionaries, such as the Wiktionary [17], could use our kernels as a general entry for each type of stretchable word, and include the balance and stretch parameters as part of their structured word information, as they do, for example, with part of speech. Natural language processing software and toolkits could use the techniques we developed to help with processing stretched words, e.g., in their approach to stemming. Similarly, spell checking software may be able to use our methods to help prevent marking stretched words as misspellings. Our procedures could also be used to help prevent typosquatting [18]. Twitter could use our methods to help improve their spam filter, looking for slight variations of tweets. Also, spelling trees could more generally be used to analyze the construction of any sequence, such as genome sequences.

However, much more could be done. We have restricted our study to words containing only Latin letters. Future work could extend this to include all characters, including punctuation and emojis. We also limited the way we constructed kernels, focusing only on one and two-letter elements. This can be expanded to three-letter elements and possibly beyond to capture the characteristics of words like ‘omnomnomnom’. Furthermore, our methodology for creating kernels leads to situations where, for example, we have both (ha)g(ah) and

(ha)(ga)(ha) as kernels. Expanding to three-letter elements and beyond in the future could collapse these forms, and related kernels, into a kernel like (hag).

Along with more advanced kernels, similar but more advanced spelling trees could be developed. We only created spelling trees for kernels with a single two-letter element. Future work could explore kernels with more than two-letter elements. They could also be created for every kernel, where the branching of even the single letter elements is shown, where one branch would signify the repetition of that letter and the other branch would signify moving onto the next letter of the kernel. Furthermore, to go with three-letter elements, ternary trees could be developed. Among other things, this would reveal mistypings like (ha)(hs), for example, if this became a kernel with a three letter element like (has), and we assume that the ‘s’ is mostly a mistyping of the letter ‘a’ in the kernel (ha). This situation should be discernible from the case where the word ‘has’ is stretched.

Another interesting phenomenon to look at is the distinction between phonetic and visual stretching. When verbally stretching a word, only certain sounds can be stretched out, whereas when typing a word, any letter can be repeated. For example, compare ‘goooooaaal’ with ‘ggggggooooooaaaaalllll’. Both stretched words can be typed, but only the first can be said because of the plosive ‘g’. Relatedly, looking at what parts of words, such as the end of words, or which letters get stretched more could be interesting.

Finally, our methodology could be used to explore linguistic and behavioural responses to changes in Twitter’s protocol (e.g., character length restrictions) and platform (e.g., mobile vs. laptop). For example, what are the effects of auto-correct, auto-complete, and spell check technologies? And what linguistic changes result from platform restrictions such as when a single key cannot be held down anymore to repeat a character? Also, we only considered tweets before the shift from the 140- to 280-character limit on Twitter. Some initial work indicates that the doubling of tweet length has removed the edge effect that the character limit creates [19]. Further work could study how this change has affected

stretchable words, and in particular, the tail of their distributions.

3.5 ACKNOWLEDGMENTS

We thank Margaret Lima for her help, support, and the use of her classroom, which allowed the early stages of this collaboration and research to happen, and for just being a great teacher and lady. CMD and PSD were supported by NSF Grant No. IIS-1447634, and TJG, CMD, and PSD were supported by a gift from MassMutual.

3.6 REFERENCES

- [1] Appendix: Glossary — Wiktionary, the free dictionary. <https://en.wiktionary.org/w/index.php?title=Appendix:Glossary&oldid=51610328>. Accessed: 2019-03-24.
- [2] J. A. Simpson and E. S. C. Weiner, editors. *The Oxford English Dictionary*. Oxford University Press, Oxford, 2nd edition, 1989.
- [3] Yuan Huang, Diansheng Guo, Alice Kasakoff, and Jack Grieve. Understanding US regional linguistic variation with Twitter data analysis. *Computers, Environment and Urban Systems*, 59:244–255, 2016.
- [4] Tyler J. Gray, Andrew J. Reagan, Peter Sheridan Dodds, and Christopher M. Danforth. English verb regularization in books and tweets. *PLOS ONE*, 13(12):1–17, 12 2018.
- [5] Bruno Gonçalves and David Sánchez. Crowdsourcing dialect characterization through Twitter. *PLOS ONE*, 9(11):1–6, 11 2014.
- [6] Bruno Gonçalves, Lucía Loureiro-Porto, José J. Ramasco, and David Sánchez. Mapping the Americanization of English in space and time. *PLOS ONE*, 13(5):1–15, 05 2018.
- [7] Gonzalo Donoso and David Sanchez. Dialectometric analysis of language variation in Twitter. *CoRR*, abs/1702.06777, 2017.
- [8] Jacob Eisenstein, Brendan O’Connor, Noah A. Smith, and Eric P. Xing. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP ‘10*, pages 1277–1287, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.
- [9] Jacob Eisenstein, Brendan O’Connor, Noah A. Smith, and Eric P. Xing. Diffusion of lexical change in social media. *PLOS ONE*, 9(11):1–13, 11 2014.
- [10] Talk: cuuute — Wiktionary, the free dictionary. <https://en.wiktionary.org/w/index.php?title=Talk:cuuute&oldid=51216685>. Accessed: 2019-03-24.

- [11] Wiktionary: Criteria for inclusion — Wiktionary, the free dictionary. https://en.wiktionary.org/w/index.php?title=Wiktionary:Criteria_for_inclusion&oldid=52749064. Accessed: 2019-05-12.
- [12] Cutoff frequency — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Cutoff_frequency&oldid=873937426. Accessed: 2019-05-07.
- [13] Isabel M. Kloumann, Christopher M. Danforth, Kameron Decker Harris, Catherine A. Bliss, and Peter Sheridan Dodds. Positivity of the English language. *PLOS ONE*, 7(1):1–7, 01 2012.
- [14] Charles Wetherell and Alfred Shannon. Tidy drawings of trees. *IEEE Transactions on Software Engineering*, (5):514–520, 1979.
- [15] Bill Mill. Drawing presentable trees. *Python Magazine*, 2(8), 08 2008.
- [16] Bill Mill. Github — llimllib/pymag-trees: Code from the article “Drawing good-looking trees” in Python Magazine. <https://github.com/llimllib/pymag-trees/tree/9acfb8d52a09a495f25af91dcbf438499546748b>. Accessed: 2019-01-21.
- [17] Wiktionary, the free dictionary. https://en.wiktionary.org/wiki/Wiktionary:Main_Page. Accessed: 2019-05-12.
- [18] Typosquatting — Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/w/index.php?title=Typosquatting&oldid=884561229>. Accessed: 2019-05-12.
- [19] Kristina Gligorić, Ashton Anderson, and Robert West. How constraints affect content: The case of Twitter’s switch from 140 to 280 characters. *International AAAI Conference on Web and Social Media*, 06 2018.

3.7 APPENDIX

3.7.1 ALTERNATE BALANCE MEASURE

As a comparison to our normalized entropy measure for balance discussed in Sec. 3.3.2, we also compute an alternate normalized entropy measure, H_{alt} , that measures balance from a different view.

To compute H_{alt} , we first calculate the overall average stretch for each character as before, but now do so across all tokens at once. Then, we subtract one from each of these values and normalize them so they sum to 1 and can be thought of like probabilities. We then compute the normalized entropy, H_{alt} , of these values as a measure of overall balance. H_{alt} is similar to H in that if each character stretches the same on average, the normalized entropy is 1, and if only one character in the kernel stretches, the normalized entropy is 0. Again, higher entropy corresponds with more balanced words.

The difference is the view, and what is meant by ‘on average’. For H_{alt} , each token is weighted equally when calculating balance. Thus, this measure corresponds to the view that one randomly samples tokens and looks at how balanced they are on average.

By contrast, for H , as calculated in Sec. 3.3.2, tokens are grouped by length, and then each group gets an equal weight regardless of the group size. This view looks at how well balance is sustained across lengths, and corresponds to sampling tokens by first randomly picking a length, and then randomly picking a token from all tokens of that length, and then looking at how balanced the sampled tokens are on average.

For example, for the kernel (pa), $H_{\text{alt}} = 1.00000$, signifying nearly perfect balance. However, looking at the balance plot for (pa) in Fig. 3.A1, we see that perfect balance is not sustained across lengths. Because most of the tokens are short, and short stretched versions of (pa) are well balanced, all of the weight is on the well-balanced short ones when randomly picking tokens. However, as people create longer stretched versions of (pa), they

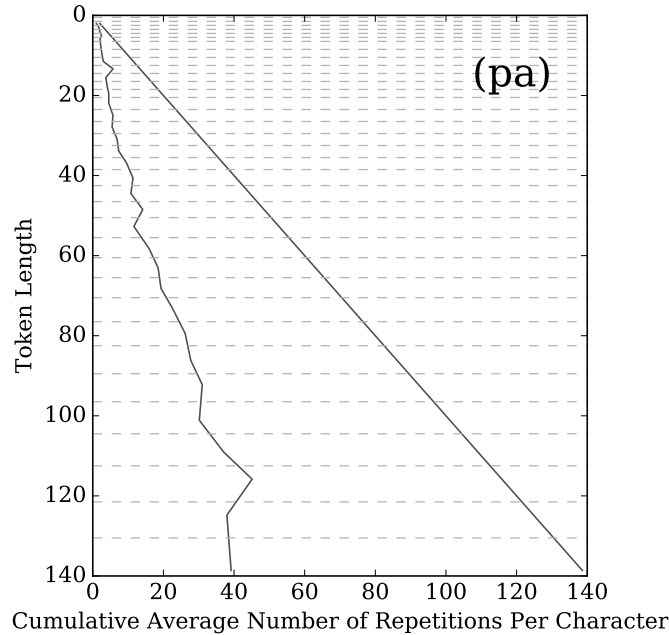


Figure 3.A1: Balance plot for the kernel (pa). See the Fig. 3.6 caption for plot details. Even though $H_{\text{alt}} = 1.00000$ for (pa), this plot clearly shows perfect balance is not sustained as tokens increase in length.

tend to use more ‘a’s than ‘p’s, and near perfect balance is not maintained. This is better captured by the measure $H = 0.80982$.

As our main measure of balance, we chose the view better representing how well-balanced tokens are as they are stretched, equally weighing lengths. This does have the limitation that groups of tokens with different lengths have different sizes, and some of them may contain a single token, possibly increasing the variance of the measure. It is possible this could be improved in the future by only including lengths that have a certain number of examples, or possibly creating larger bins of lengths for the longer tokens like we do in the balance plots.

We include the same plots and tables for H_{alt} as we did with H , and many of the observations are similar. Fig. 3.A2 shows the two jellyfish plots for H_{alt} . Similar to before, Fig. 3.A2A is the version containing all words and for Fig. 3.A2B we remove the words

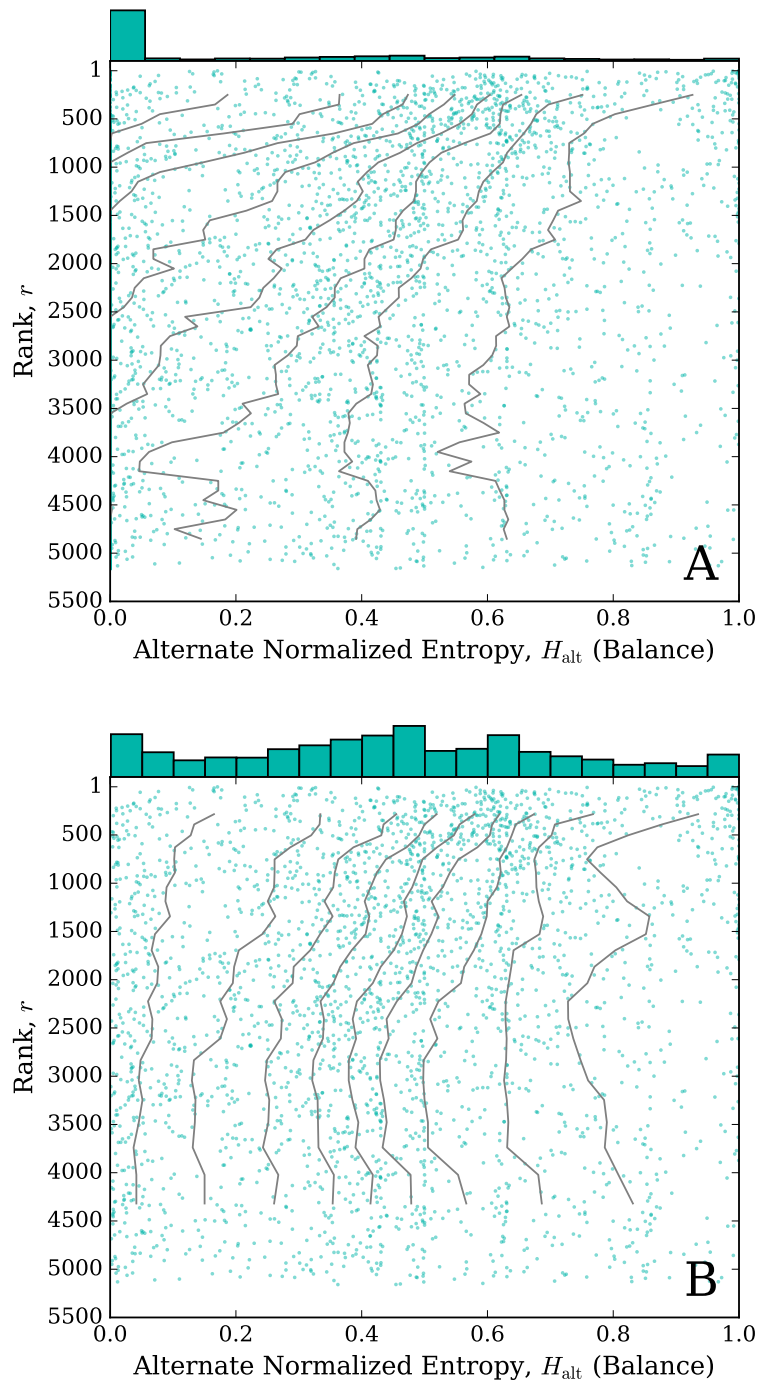


Figure 3.A2: Jellyfish plots for kernel balance based on an alternate entropy measure for (A) all kernels, and (B) excluding kernels with entropy exactly 0. Corresponding histograms are given at the top of each plot. Kernels are plotted vertically by their rank, r , and horizontally by their balance as given by an alternate normalized entropy, H_{alt} , where larger entropy denotes increased balance. The deciles 0.1, 0.2, \dots , 0.9 are calculated for rolling bins of 500 kernels and are plotted as the ‘tentacles’.

	H_{alt}	Kernel	Example token
1	1.00000	(ba)	baaaaaaaaaaaa
2	1.00000	(pa)	pppppppppppppa
3	1.00000	(uo)	uouuuuuuuuuuuuu
4	0.99998	(pr)	prrrrrrrrrrr
5	0.99998	(du)	duduudduudududuu
6	0.99995	(xa)	xaxaxaxaxxa
7	0.99995	(ai)	aaaaaaaaaaaaaaaaai
8	0.99993	(he)	hehehheheheh
9	0.99986	(bi)	biiiiiiiiii
10	0.99985	(wq)	wqwqwqwqwqw

Table 3.A1: Top 10 kernels by an alternate normalized entropy, H_{alt} .

that have a value of 0 for entropy. The top of the plots in Fig. 3.A2 shows the frequency histograms in each case. As before, after removing kernels with an entropy of 0, we see a small left-shift in the highest ranked kernels, and then the distribution largely stabilizes. Again, the highest ranked kernels tend to be more equally balanced, and kernels only stretching a single character tend to be lower ranked.

Table 3.A1 shows the kernels with the ten largest entropies and Table 3.A2 shows those with the ten smallest nonzero entropies as measured in this alternate way. We observe that the kernels with largest entropies are all of the form (l_1l_2) and are almost perfectly balanced given the view of equally weighing all tokens. The kernels with lowest entropies all expand to regular words that when spelled in the standard way contain a letter that is repeated, plus these kernels allow other letters to stretch.

Finally, Fig. 3.A3 shows the scatter plot of each kernel where the horizontal axis is given by this alternate measure of balance, H_{alt} , and the vertical coordinate is again given by the measure of stretch for the kernel using the Gini coefficient, G . We again see that the kernels span the two-dimensional space.

We still get the same kind of rough vertical banding that we saw in Fig. 3.9 for the same reason, but we also see a curved dense band at lower entropy values, which seems to mostly

	H_{alt}	Kernel	Example token
1	0.00115	[t][e][t]h	teeeeeeeeth
2	0.00119	f[e]l[i]ng	feeeeeling
3	0.00170	c[a]l[l]ing	callllling
4	0.00196	a[c]ep[t]	acceptttttt
5	0.00197	fa[l]i[ng]	fallllling
6	0.00217	hi[l]ar[y]	hilllllaryy
7	0.00227	m[i]s[i]ng	misssssssssing
8	0.00271	ba[n]e[d]	banedddddddd
9	0.00277	t[h]r[e]	threeeeeeee
10	0.00302	th(er)	therrreeeee

Table 3.A2: Bottom 10 (nonzero) kernels by an alternate normalized entropy, H_{alt} .

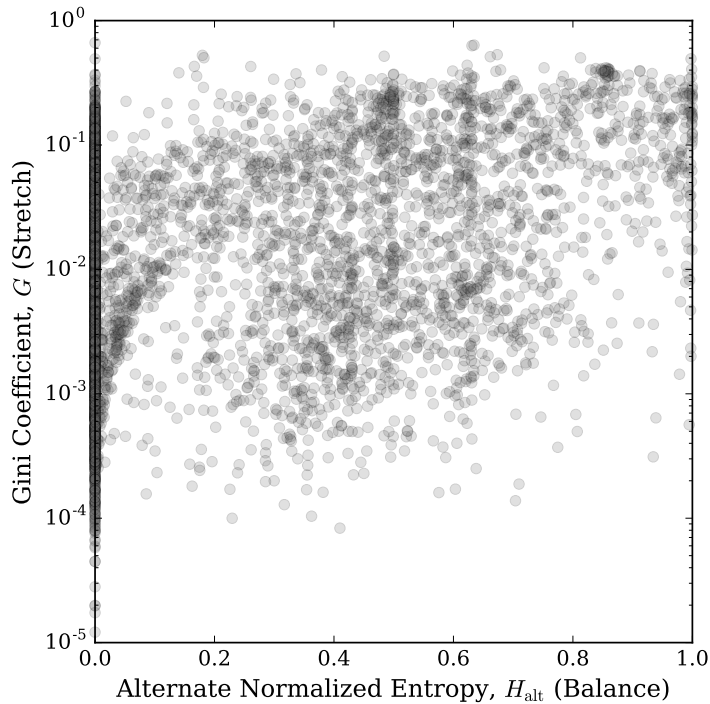


Figure 3.A3: Kernels plotted in Balance-Stretch parameter space using an alternate measure of normalized entropy for balance. Each kernel is plotted horizontally by the value of its balance parameter, given by an alternate normalized entropy, H_{alt} , and vertically (on a logarithmic scale) by its stretch parameter, given by the Gini coefficient, G , of its token count distribution. Larger entropy implies greater balance and larger Gini coefficient implies greater stretch.

contain kernels whose base word is spelled with a double letter, like ‘summer’ (with kernel [s][u][m][e][r]).

3.7.2 STRETCH RATIO

For each kernel, we also measure a ‘stretch ratio’, ρ . This is simply the ratio of the total number of stretched tokens, n_s , to the total number of unstretched tokens, n_u , for that kernel. That is,

$$\rho = \frac{n_s}{n_u}. \quad (3.1)$$

Fig. 3.A4 gives the jellyfish plot for the stretch ratio. Like Fig. 3.8, the horizontal axis has a logarithmic scale and the histogram bins have logarithmic widths. The stretch ratio

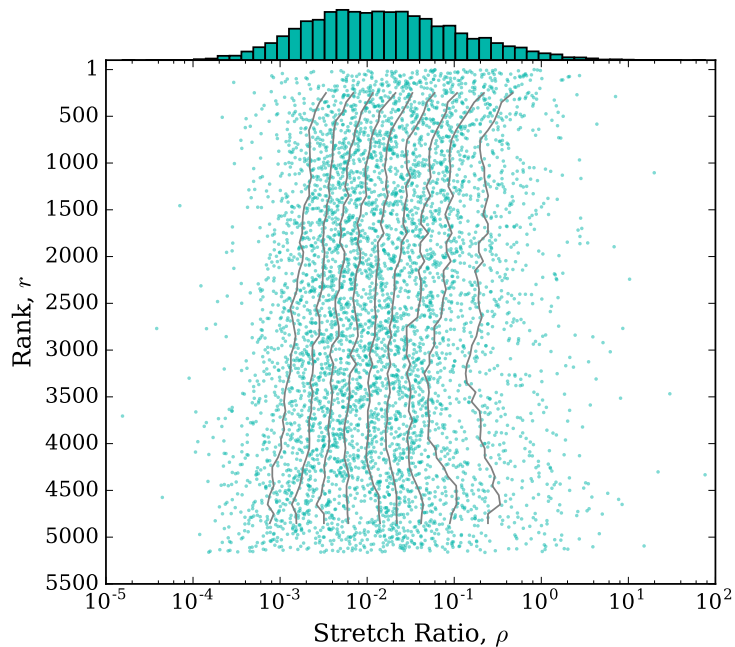


Figure 3.A4: Jellyfish plots for kernel stretch ratio, ρ , as given by the ratio of the sum of the kernel’s stretched tokens to the sum of its unstretched tokens. The histogram is given at the top of the plot (with logarithmic width bins). Kernels are plotted vertically by their rank and horizontally (on a logarithmic scale) by their stretch ratio. The deciles 0.1, 0.2, \dots , 0.9 are calculated for rolling bins of 500 kernels and are plotted as the ‘tentacles’.

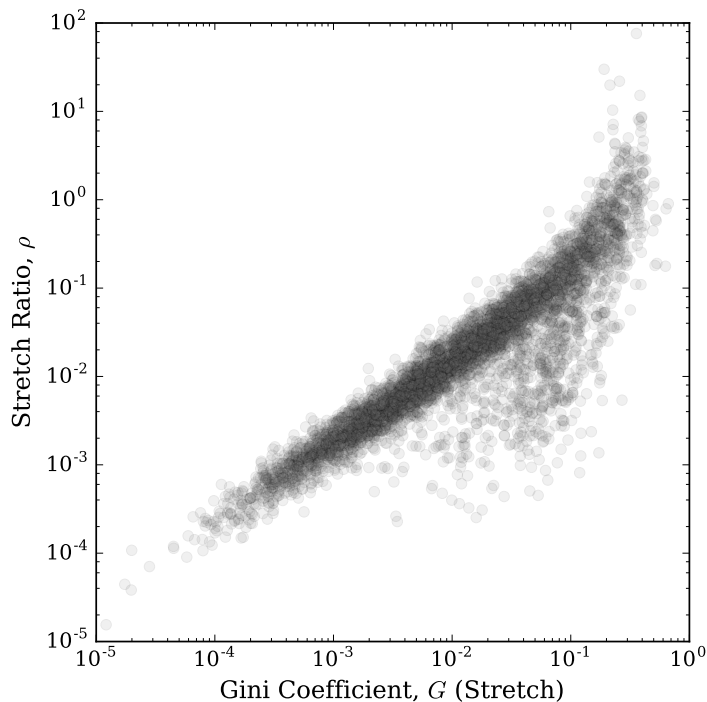


Figure 3.A5: Scatter plot comparing two measures of stretch for each kernel. For each kernel, the horizontal axis gives its stretch as measured by the Gini coefficient, G , of its token count distribution and the vertical axis gives its stretch ratio, ρ . Both axes have a logarithmic scale.

distribution stays fairly stable across ranks, except for the highest ranked kernels, which tend to have a larger ratio.

This stretch ratio can be thought of as a simple measure for the stretchiness of a kernel, with a larger ratio representing stretchier words. As stretched versions of the word are used more, the numerator increases and the ratio value increases. Conversely, as unstretched versions of the kernel are used more, the denominator increases, and the ratio value decreases. However, this simpler measure uses less information from the full distribution than a measure like the Gini coefficient does, so we would expect some differences between the two. Indeed, Fig. 3.A5 shows that there are some kernels for which the two measures seem to disagree. Yet, Fig. 3.A5 shows that the stretch ratio and Gini coefficient are quite well correlated, with Pearson correlation coefficient 0.89 ($p < 10^{-100}$), so there

	ρ	Kernel	Example token
1	76.04717	s[o][c][o][r][o][k]	socorrokkkkkk
2	29.94863	mou(ha)	mouhahahaha
3	21.93369	p[f](ha)	pffhahahaha
4	19.82821	bu(ha)	buhahahahaha
5	15.15702	(ha)j(ah)(ja)(ha)	hahahahajahajaha
6	10.32701	pu(ha)	puhahahahaa
7	8.63055	(ha)(ba)(ha)	habahahahaha
8	8.47429	(ha)b(ha)	hahahhahabha
9	8.13269	(ah)j(ah)	ahahahjahah
10	7.72953	a[e]h[o]	aehoooooooooooo

Table 3.A3: Top 10 kernels by stretch ratio, ρ .

	ρ	Kernel	Example token
1	0.00002	am[p]	ampppppppppp
2	0.00004	fr[o]m	froooooooooom
3	0.00004	m[a]kes	maaaaaaakes
4	0.00007	w[i]th	wiiiiiiiiith
5	0.00009	eve[r]y	everrrrrrrry
6	0.00011	p[r]a	prrrrrrrrrra
7	0.00011	watch[i]ng	watchiiiiing
8	0.00011	s[i]nce	siiiiince
9	0.00012	pla[y]ed	playyyyyyyed
10	0.00012	vi[a]	viaaaaaaaaaaaaa

Table 3.A4: Bottom 10 kernels by stretch ratio, ρ .

is not much gained by including both. We choose to use the Gini coefficient as our main measure of stretchiness both because of its wide usage and because of the fact that it uses more information from the full distribution than the simpler stretch ratio.

Table 3.A3 shows the top 10 kernels by stretch ratio and Table 3.A4 gives the bottom 10. The correlation between stretch ratio and Gini coefficient, at least for the least stretchy kernels, can be seen further when comparing this to Table 3.5. Many of the kernels that show up as the least stretchy words (lowest Gini coefficients) also show up here in the list of kernels with smallest stretch ratio.

3.7.3 “DRAWING PRESENTABLE TREES” ALGORITHMIC BUGS

Wetherell and Shannon presented an algorithm for drawing large trees in a nice way in their paper “Tidy drawing of trees” [14]. The article “Drawing presentable trees” [15] by Mill and related code [16], based largely on the earlier work of Wetherell and Shannon, provide a version of the algorithm written in the Python syntax, but both the article and the code contain algorithmic bugs. In the following, we present the bugs we found.

We will discuss Listing 5 in Mill’s paper [15], as that is the version that most closely resembles Algorithm 3 of Wetherell’s and Shannon’s paper [14], which is what our code to create the spelling trees is based off of.

In Listing 5, the definition of `setup` contains the code:

```
elif len(tree.children) == 1:
    place = tree.children[0].x - 1
```

This needs to be split into a left case and a right case. If the only child node is a left child, then the parent should be placed to the right by one, and if the only child node is a right child, then the parent should be placed to the left by one. The `DrawTree` class needs a way to tell if a node has a left or right child. Let us assume the class `DrawTree` has an attribute `left` properly implemented that is set to `True` iff the node has a left child. Then the code should be something more like the following:

```
elif len(tree.children) == 1:
    if tree.left:
        place = tree.children[0].x + 1
    else:
        place = tree.children[0].x - 1
```

Compare the above fix to the corresponding code in the `right_visit` case in the first `while` loop in Algorithm 3 in “Tidy drawing of trees” [14]:

```
elseif current↑.left_son = nil
  then place := current↑.right_son↑.x - 1;
elseif current↑.right_son = nil
  then place := current↑.left_son↑.x + 1;
```

Later in Listing 5 in the definition of `setup` is the following line:

```
nexts[depth] += 2
```

However, we want the next available spot, recorded in `nexts`, to be two spots to the right of the current placement, and the current placement is sometimes different from the current next available spot. Thus, the line should look something like the following:

```
nexts[depth] = tree.x + 2
```

Again, compare this to the corresponding code found near the end of the `right_visit` case of the first `while` loop of Algorithm 3 in “Tidy drawing of trees”:

```
next_pos[h] := current↑.x + 2;
```

The final bug in Listing 5 is not an algorithmic bug, but merely a typo. In the definition of `addmods` is the line of code:

```
modsum += tree.offset
```

However, `tree` does not have the attribute `offset`. Instead the `mod` attribute should be added to the accumulated sum as follows:

```
modsum += tree.mod
```

CHAPTER 4

CULTUROMICS REDUX: REBUILDING THE GOOGLE BOOKS n -GRAM DATASET TO REVEAL THE HIDDEN COLLECTIVE VOICE OF ENGLISH FICTION AUTHORS

The large-scale quantification of the dynamics of words and phrases in books written across centuries should provide an extraordinary platform for a rich study of language evolution and changes in culture and society. The Google Books n -grams corpora for a range of languages has promised as much but has proved unreliable due to (1) An increasing prevalence of scientific material through the 20th Century, and (2) An inherent failure to represent the popularity of language and culture from the perspective of authors or readers. In this paper, we show how the specific Google Books n -grams corpus for English Fiction can be reconstructed to afford a view into the mind of the ‘collective author’ (but not that of the ‘collective reader’). We are then able to properly explore, at the resolution of years, the topics authors and editors have historically deemed to be of interest to readers, along with the language they favored to do so. We first explain why raw n -gram counts are a fundamentally flawed proxy for word and phrase importance within the Google Books n -gram framework. We then devise a method using book-appearance count of n -grams to rebuild

the Google Books corpus so that meaningful linguistic and cultural trends, according to the standpoint of the collective author, may be reliably discerned. We measure and examine the dynamics of ‘lexical turbulence’, a summary statistic for churn in word rank, by looking at the flux of n -grams across rank boundaries in Zipf distributions. We show that lexical turbulence decreases for the original Google Books version but is roughly constant for our revised version—there is no apparent slowing down or speeding up. We then examine linguistic and sociocultural changes between time periods using wordshift plots and a rank divergence measure with a tunable parameter. We show how tuning our rank divergence parameter allows us to systematically explore n -gram usage changes ranging from n -grams used in functional parts of language to n -grams that best represent cultural evolution and important historical events. In parallel with this paper, we make our revised version of the Google Books English Fiction n -gram dataset freely available, along with an interactive online viewer.

4.1 INTRODUCTION

Language is in a constant state of evolution, driven in part by both random drift [1] and sociocultural factors. Snapshots of the current linguistic state are created when words are printed in books. The continuous recording and aggregation of text over time allows us to examine the linguistic and cultural trends of the past, and provides insight into the future trajectories of such trends.

In the modern era, digitized text sources such as the Google Books corpus [2, 3] and increased computational power provide profound opportunities for the study of linguistic evolution over the last two hundred years in a number of major languages. Indeed, a number of recent studies have focused on an array of observed trends in the Google Books corpus, including analyses of Zipf’s and Heaps’ laws [4], verb regularization [2, 5], culture changes

[2], new words in a language [6], and word births, deaths, and general changes in word use over time [7, 8].

In this paper, our goal is to carefully examine the Zipf distributions [9] of 1-grams, 2-grams, and 3-grams in the Google Books corpus and the trends of n -grams over time. Through a proper statistical treatment of n -gram frequency changes over time, we reveal the topics common to authors over the last 200 years, and gain insight into linguistic and cultural changes.

Unfortunately, studies have identified troubling issues with the Google Books corpus [5, 10]. There is a large prevalence of scientific works in both the 2009 and 2012 versions of the English corpus and in the 2009 version of the English Fiction corpus [10]. Furthermore, due to its library-esque nature, where each book contributes roughly one time (not counting reprints and different editions), the data fails to reflect information about the popularity of words based on readership views [10]. Finally, it has been shown that the edited form of language found in books does not necessarily match the colloquial language of everyday speakers [5].

Despite these setbacks, the Google Books corpus still contains a wealth of information from which we are able to extract meaningful results if we keep the preceding limitations in mind. In this paper, we use the 2012 English Fiction corpus [11] as suggested by [10]. We begin by showing that the straightforward method of constructing the Zipf distributions using token counts, as is used by Google’s Ngram Viewer [12], reveals nonsensical irregularities in terms of n -gram prevalence. We then develop an improved method, focused on book counts rather than token counts, that affords a more meaningful output. Using this improved method, we are then able to study the temporal evolution of the English Fiction corpus.

We structure our paper as follows: In Sec. 4.2, we describe the data. In Sec. 4.3, we explain the problems associated with using token counts to study language. We provide a

revised method for using the Google Books corpus to study language in Sec. 4.4, explaining how we use book counts as the primary unit to rank tokens in Sec. 4.4.1, and developing methods of subsampling and smoothing in Sec. 4.4.2. We provide our analysis of language in Sec. 4.5. We begin our analysis by studying the Zipf distributions in Sec. 4.5.1 and rank flux in Sec. 4.5.2. Then, in Sec. 4.5.3, we use wordshift plots to provide fine-grained analysis of the changes in language between two time periods. Finally, in Sec. 4.6, we present our concluding remarks.

4.2 DESCRIPTION OF THE DATASET

In this paper, we use the English Fiction 2012 Google Books corpus [2, 3, 11, 13]. This data consists of “books predominantly in the English language that a library or publisher identified as fiction” [11] that have been split into ‘ n -grams’. A 1-gram is often, but not always, the same as a word. The set of 1-grams also includes lexical structures such as punctuation and numbers. An n -gram is made up of n 1-grams. For example, ‘the sun.’ is a 3-gram made up of the three 1-grams ‘the’, ‘sun’, and the period, ‘.’. A token is an individual occurrence of an n -gram.

For each n -gram that occurs within the corpus at least 40 times overall, the total number of times that n -gram is found in each year (token count) and the total number of books it was found in (book count) are recorded [13]. Within the raw data, there are also n -grams that are annotated with information like part of speech, and some that contain wildcards [3, 11]. From our inspection, these seem to be either subsets or supersets of the pure n -grams, so we only use the pure n -gram data.

Also provided in the data, for each year, are the total token counts for 1-grams, the total number of sentences, and the total number of books. Google does not provide total token counts for n -grams with n greater than 1. We calculate these counts ourselves and confirm by comparison with the online viewer.

To do so, we must handle a complication in that Google does not permit n -grams to cross sentence boundaries. For n -grams with n greater than 1, Google artificially inserts a special n -gram denoting the start of each sentence, and one denoting the end of each sentence [3].

Totals may then be calculated as follows: The number of 2-grams in a particular year should be: the number of 1-grams minus the number of sentences (because a sentence splits into 1 less 2-gram than it does 1-grams) plus two times the number of sentences (because Google added two special 1-grams to each sentence). The resulting total count for 2-grams is then equal to the number of 1-grams plus the number of sentences. For each increase in n , the total number of n -grams should be the number of sentences less than the previous number of n -grams. So, for example, the number of 3-grams is then the same as the number of 1-grams.

We checked our calculations by retrieving the time series of relative token frequency values for several 2-grams and 3-grams from the online viewer. The yearly token frequency values from the online viewer agree with the values obtained by taking the counts from the raw data divided by our theorized yearly totals.

Fig. 4.1 gives the total number of books used to generate the dataset for each year from 1800-2008. The plot for total token counts has a nearly identical shape, but is vertically multiplied by about 10^5 , representing an average of roughly 10^5 n -grams per book. Fig. 4.1 also shows that the data volume is increasing exponentially over time.

4.3 THE PROBLEM WITH USING TOKEN COUNTS

We began by creating Zipf distributions, or relative frequency distributions where the horizontal position gives the rank of the n -gram, and the n -gram with the greatest frequency has the highest rank of 1. To construct Zipf distributions for n -grams, we started with the most natural thing, which is to use the token counts that underlie Google's Ngram viewer.

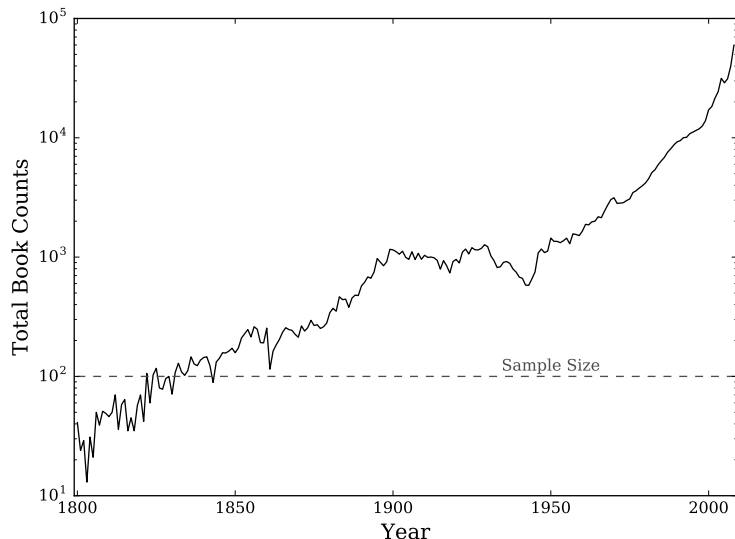


Figure 4.1: Yearly book counts for the Google Books English Fiction 2012 corpus. Overall, the number of books being sampled by Google has grown exponentially. However, during the first half of the 20th century the number of books stayed relatively level, and even experienced a significant dip during the Great Depression Era into the time of World War II. The dashed horizontal line represents the number of books used during our subsampling process (see Sec. 4.4.2).

Fig. 4.2 shows the Zipf distributions for 2-grams and 3-grams with each decade-initial year from 1800–2000 plotted on the same axes. The way the distributions overlay each other suggests a general overall consistency in the distributions over time. The distributions for 3-grams is flatter with a longer right tail because of the combinatorial increase in possibilities for 3-grams (see Sec. 4.5.1 for more information).

We explore language churn further with the Jensen-Shannon divergence (JSD). JSD allows us to compare distributions and rank n -grams by their largest contribution to JSD. As we will show, visual inspection of the JSD contributions reveals several examples indicating that the natural use of token counts is extremely problematic. This straightforward construction of Zipf distributions based on token counts leads to nonsensical irregularities in terms of the prevalence of n -grams.

In Fig. 4.3 we give the JSD wordshift plot for subsampled (see more about subsampling in Sec. 4.4.2) 2-grams comparing the distribution from combining the 10 years starting in

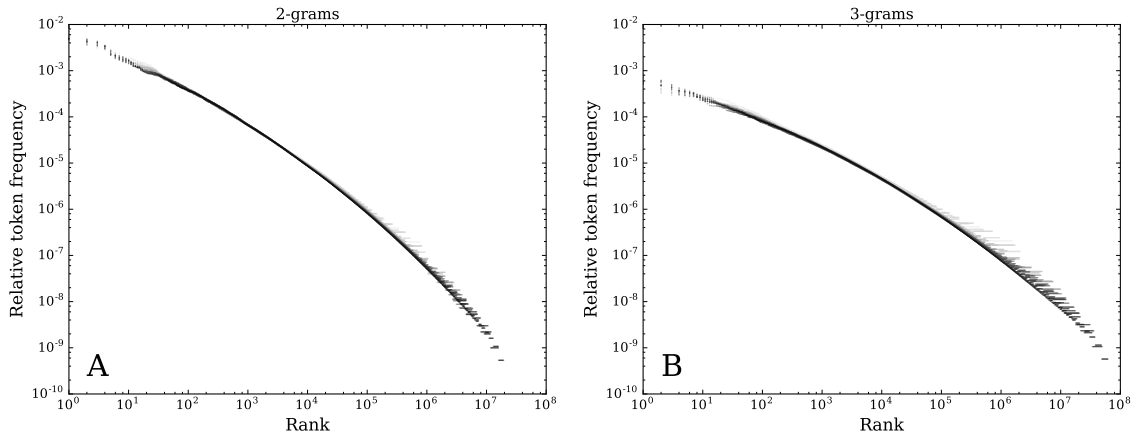


Figure 4.2: Zipf distributions for (A) 2-grams and (B) 3-grams based on token counts in the Google Books English Fiction 2012 corpus. We rank n -grams based on their relative token frequency. We overlay Zipf distributions for the initial year of each decade, 1800, 1810, 1820, ..., 2000, with more recent years being plotted in darker gray.

1934 to that from combining the 10 years starting in 1944. The wordshift plot indicates which n -grams were the largest contributors to the JSD between these time periods. The years we examine here cover the Great Depression, World War II, the start of the Post War era, and the Korean War. However, any influence these events had on language is largely unapparent. For example, one of the 2-grams contributing the most to the JSD is ‘Raintree County’, which is simply the title of a single book, and one that does not explicitly involve any of these crucial world events. The relative token frequency time series for ‘Raintree County’ (Fig. 4.4) shows a giant spike in 1948, the first year this roughly 1000-page book was published. The spike represents a total of 3430 ‘Raintree County’ tokens counted in only five books. This anomalous spike is likely a result of Google scanning the same book more than once and the title being printed on the top of every page (or every other page) and is not at all representative of a jump in the use of this bigram in natural language.

We find these kinds of anomalous n -gram patterns across all time periods. In the JSD wordshift plot for 3-grams comparing the distributions from 1979 and 1980 (Fig. 4.5), we see the trigrams ‘on copyright page’, ‘No statement of’, and ‘of printing on’. The 3-gram

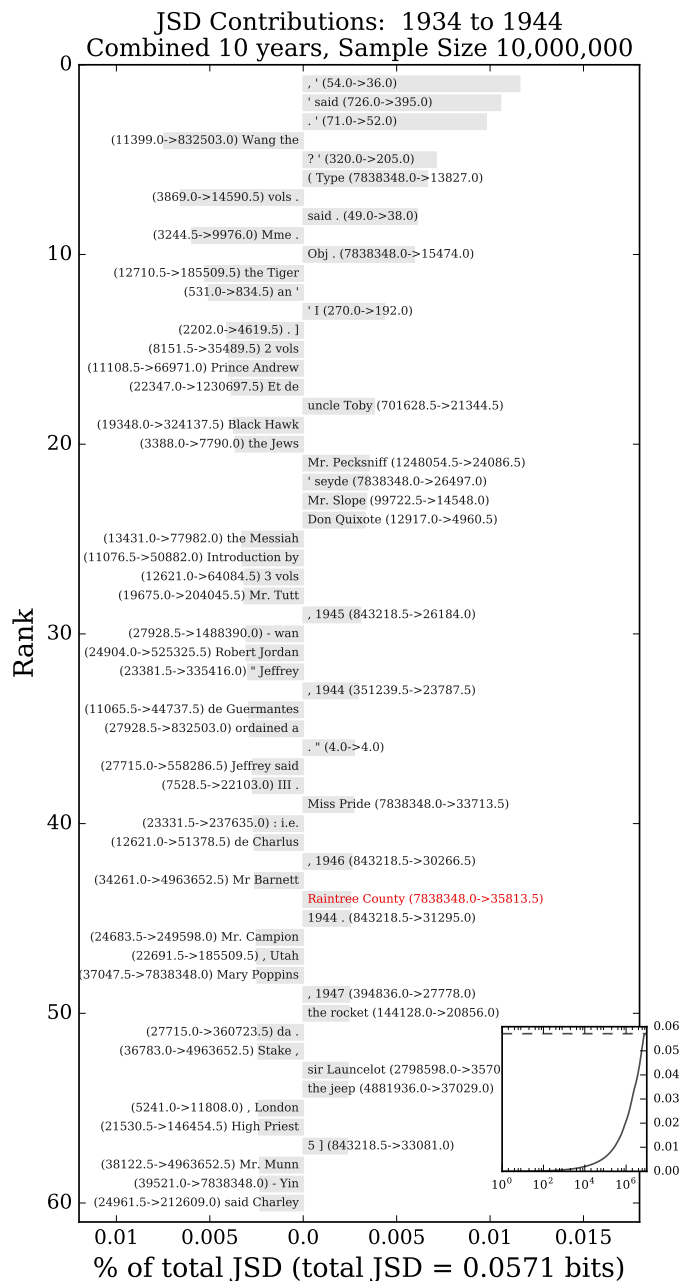


Figure 4.3: The top contributors to the Jensen-Shannon divergence (JSD) between subsampled frequency distributions of 2-grams for the periods 1934–1943 and 1944–1953. The bars to the right of zero represent the percentage contribution to the JSD of n -grams that have a higher relative frequency in the more recent time period. Bars to the left are n -grams with a higher relative frequency in the earlier time period. In parentheses next to each n -gram is its earlier rank based on token counts followed its more recent ranking. The inset plot shows the cumulative sum of JSD by summing the decreasing contributions of n -grams. ‘Raintree County’ (highlighted in red), the title of a book, is the 44th largest contributor to the JSD between these two decades.

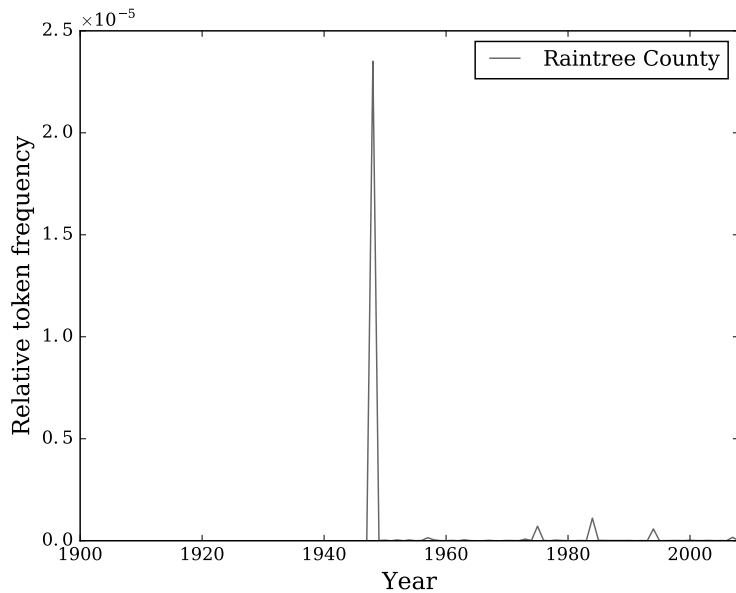


Figure 4.4: Relative token frequency time series for the bigram ‘Raintree County’. The spike in 1948 represents 3430 ‘Raintree County’ tokens in only five books.

‘on copyright page’ appeared an enormous 5990 times in just one book in 1979. Similarly, in 1979 ‘No Statement of’ shows up 2359 times in a single book and ‘of printing on’ is found 2231 times in three books. We show the spikes in 1979 for these trigrams in Fig. 4.6. We also include the 3-grams ‘the house and’ and ‘thought it was’; both of these 3-grams appear about the same number of times in 1979 as ‘on copyright page’, with ‘the house and’ showing up 6016 times and ‘thought it was’ showing up 6027. However, they are found in many more books: 1979¹ for ‘the house and’ and 2189 for ‘thought it was’, giving an average of about three occurrences per book rather than thousands. Furthermore, their time series look more like what one might expect, with an overall trend and minor fluctuations from year to year rather than an anomalous spike in a single year.

The examples we have illustrated above are not isolated. They are a small sample of many spurious time series that became apparent during our initial analysis, and are representative of many more similar anomalies existing within the data. The anomalous

¹Yes, 1979 is not only the year, but also happens to be the book count for ‘the house and’ in 1979.

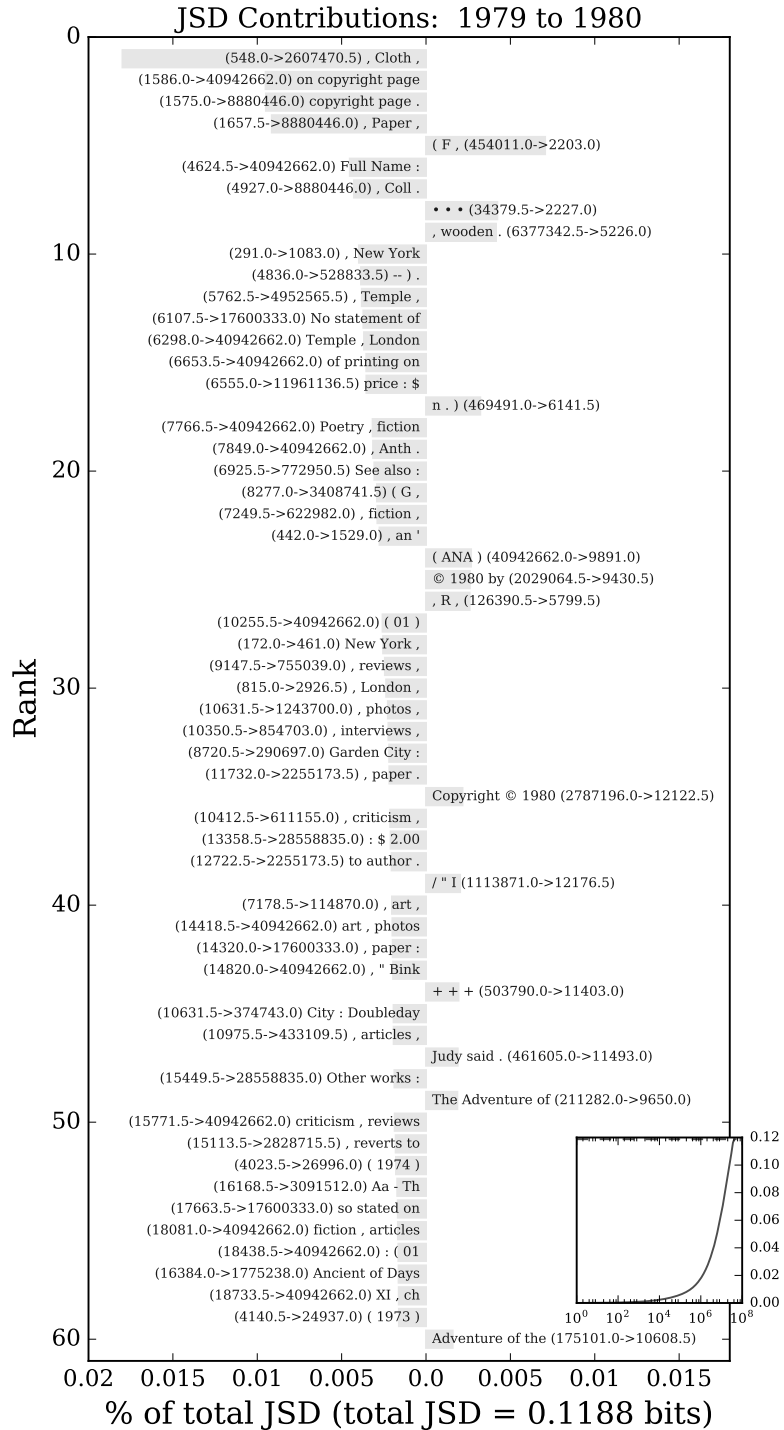


Figure 4.5: The top contributors to the JSD between the relative token frequency distributions for 3-grams in 1979 and 1980. See Fig. 4.3 for details. The 2nd largest contributing 3-gram, ‘on copyright page’, appeared 5990 times in a single book in 1979.

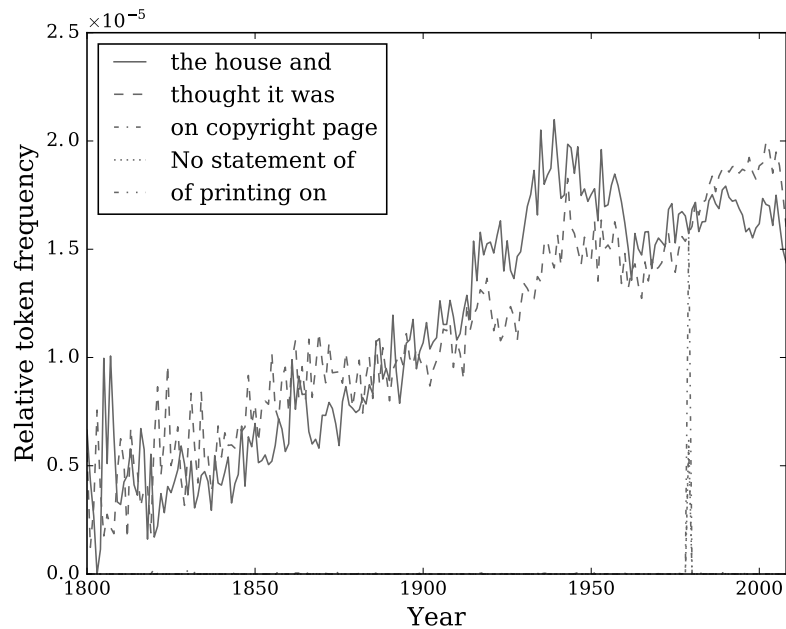


Figure 4.6: Relative token frequency time series for a select collection of 3-grams. The 3-grams ‘on copyright page’, ‘No statement of’, and ‘of printing on’ all spike anomalously in 1979, whereas the 3-grams ‘the house and’ and ‘thought it was’ have more representative, healthy looking time series.

spikes shown here can be thought of as examples of data acquisition errors of a certain kind. They are likely a result of an individual n -gram (e.g., the book title, or copyright information) being printed at the top or bottom of every page in a book. Or perhaps an error occurred during data collection, where a single page was reread multiple times during the optical character recognition (OCR) process. These examples do not reflect statistical or linguistic intuition when looking at time series of n -grams. The spikes in the time series are in no way representative of a spike in the use of these n -grams in the collective language of English speakers, nor are they representative of any shift in culture.

We also see that, overall, many of the n -grams at the top of these JSD wordshifts are names of characters in books. Again, the use of a single name many times within a single book or a few books does not represent a shift in the popularity of that name in society. Yet,

by using raw token counts as a proxy for the importance of an n -gram, names collectively appear as the largest change in language over time.

Of course, an increased usage of an individual name (or any other n -gram) by several authors could represent a legitimate shift in that n -gram’s importance. That is, if the number of unique books an n -gram appears in is changing over time, then it seems more likely that the societal usage of that n -gram in language is changing (e.g., ‘Frankenstein’). This realization inspires a different method of measuring language change over time.

4.4 UNLOCKING THE SECRETS

4.4.1 USING BOOK COUNTS TO RANK n -GRAMS

In the preceding section we showed that an analysis of language change in Google Books using JSD and token counts is fundamentally flawed. However, the Google Books corpus still contains a wealth of information from which we aim to extract goodness. Our solution to the issues described in Sec. 4.3 is to use unique book counts as the primary measure for the importance of an n -gram, and token counts as a secondary measure. That is, we use a composite ranking where we rank an n -gram first based on the number of books it appears in, and then break ties with the token count for the n -gram.

For example, there is a set of function 1-grams (e.g., ‘the’, ‘a’, ‘.', etc.) that will likely appear in every book for a given year. If we only ranked according to book counts, these function 1-grams would all be tied for the top rank. In this case we then split the tie based on token counts, giving higher ranks to those with higher token counts.

In Fig. 4.7, we compare the rank for 2-grams in 1948 based on token counts to their rank based on our composite ranking. The points appearing farther to the right from the diagonal axis are those that have a disproportionately high token count given the number of

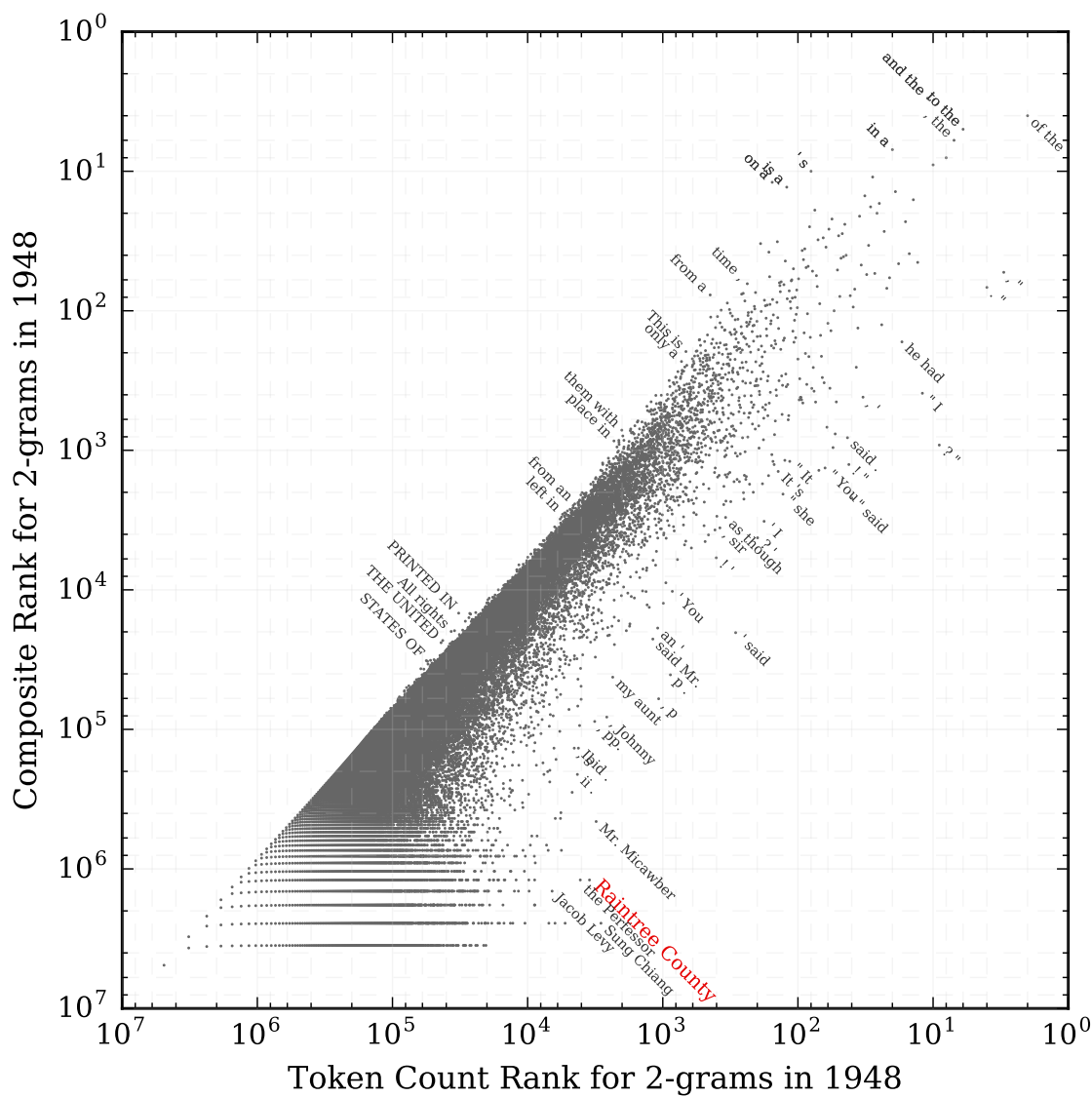


Figure 4.7: A comparison of the token count rank and composite rank for bigrams in 1948. Bigrams farther right from the diagonal have a larger token count than their corresponding book count suggests. For example, ‘Raintree County’, highlighted in red, appears an average of 686 times per book, which is far more frequent than most bigrams with a similar book count.

books they appear in. Because this plot has logarithmic axes, the imbalance farther down the diagonal is much larger than a visually comparable distance farther up the diagonal.

The point corresponding to the bigram ‘Raintree County’ is annotated in Fig. 4.7. Based on token counts, ‘Raintree County’ has a rank in the thousands. Our composite ranking, based primarily on book counts, drops the rank down into the millions, which corresponds closer to the relative importance of this n -gram in 1948. The time series for this n -gram based on its book count (not shown) no longer exhibits the spike found in its token count time series (Fig. 4.4).

In Fig. 4.8, we show the token count time series for the 1-grams ‘Lanny’ and ‘Hitler’. Lanny Budd is a character in a series of 11 books authored by Upton Sinclair in the 1940s and early 1950s. The plot shows that even though World War II and Hitler unarguably dominated the history of the 1940s, a single character from a few books by a single author ranks higher in importance when ranking with token counts. However, when using book counts, the rank of ‘Lanny’ becomes inconsequential while ‘Hitler’ is clearly ranked much higher, as we show in Fig. 4.9. Our reranking thereby more accurately represents the relative historical importance of these two 1-grams during that time period.

While we cannot infer reader popularity from Google Books [10], we can approximate the collective attention of authors on individual topics. Using book counts, we find a measure of n -gram popularity among authors, where each book gets one vote, so that the importance of a single n -gram is not artificially inflated by its overrepresentation from a single author. Linguistic and cultural shifts, or important events like World War II, will affect authors generally, and our method will capture this shift while filtering out inconsequential spikes in token counts generated by the choices of single authors or from scanning-related errors. Thus, this method of composite ranking provides a more meaningful output when studying language using the Google Books data.

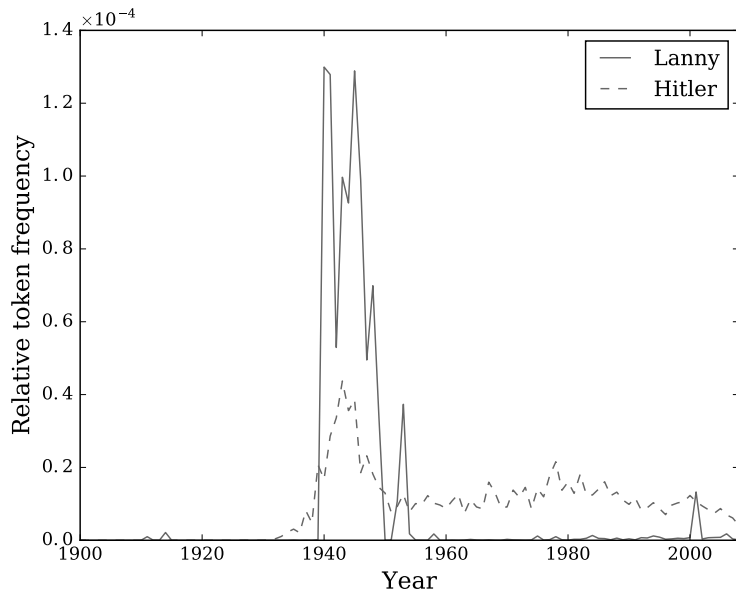


Figure 4.8: Relative token frequency time series for the 1-grams ‘Lanny’ and ‘Hitler’. Although Lanny was only a story book character, the token frequency for Lanny is much greater than that of Hitler during the WWII era.

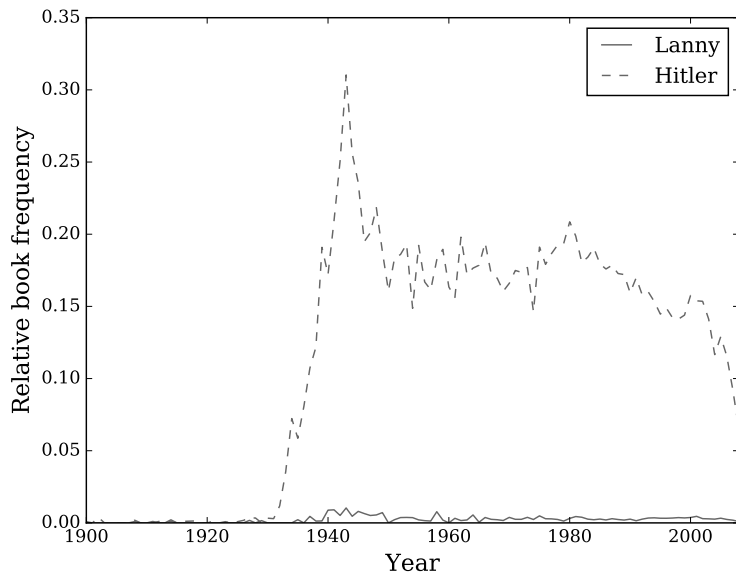


Figure 4.9: Relative book frequency time series for the 1-grams ‘Lanny’ and ‘Hitler’. Using book counts, the relative frequency for ‘Lanny’ drops to near zero while ‘Hitler’ appears in nearly 20% of books sampled by Google since the 1940s, more accurately representing the relative importance of these two figures throughout modern history.

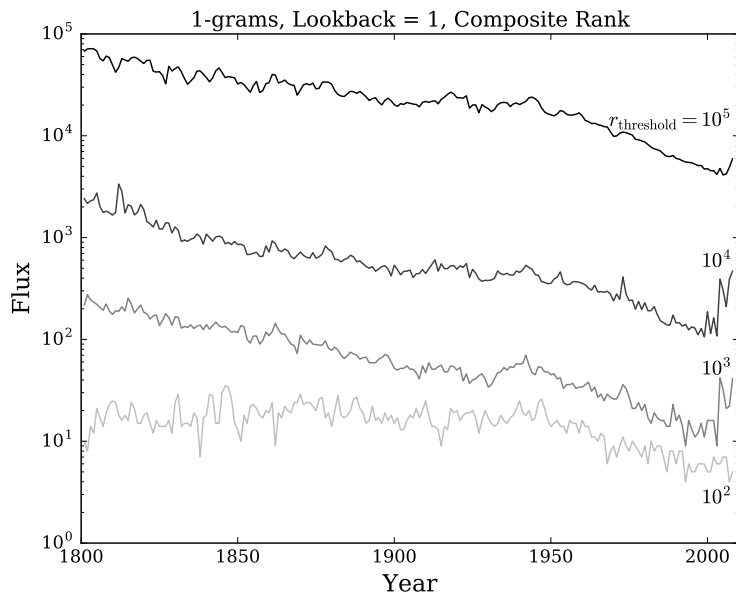


Figure 4.10: Flux time series for 1-grams. For each of the composite rank thresholds, $r_{\text{threshold}} = 10^2$, 10^3 , 10^4 , and 10^5 , the number of 1-grams moving up across that threshold (which is the same as the number moving down across the threshold) between consecutive years is plotted at the point for the more recent year.

4.4.2 SUBSAMPLING AND SMOOTHING

Although our composite ranking for n -grams based primarily on book counts corrects many irregularities, the exponential growth in data volume of Google Books over time presents additional challenges for statistical inference.

Fig. 4.10 shows the flux across select composite rank thresholds, $r_{\text{threshold}}$, for 1-grams over time. It shows that the number of 1-grams crossing the various rank thresholds is decreasing with time, exhibiting less rank churn, and representing less change in language in more modern years. By this measure, the English language appears to be cooling down [6].

However, the size of the underlying dataset is largely increasing over time. As we saw in Fig. 4.1, the total number of books in the Google Books dataset for each year from 1800

onward has followed a roughly exponential growth. We must account for how the increasing sample size greatly decreases the uncertainty in the distribution for n -grams.

Let us imagine that there is a single underlying probability distribution for n -grams from which every user of the language draws, and each book is a sampling of this distribution. As more books are included in the sample for a year, the sample distribution will become more solidified, and better match the underlying distribution from which samples are drawn. The probability estimates for n -grams, especially the more popular ones, become more precise, and the variance of these estimates decreases. Looking at two years with a small number of sampled books will require a comparison of distributions where the n -grams are not necessarily converged onto their ‘true’ underlying ranks, and some of the flux could simply be a result of n -grams moving around because of the small sample size. When the sample size increases, the ranks settle down, and comparison between years reflect changes in the underlying distribution rather than mainly changes due to sampling, and therefore the flux will be less.

Comparing Figs. 4.1 and 4.10, we see a general flattening during the first half of the 20th century in both figures. More recent years demonstrate the fastest increases in book counts and correspondingly the fastest decreases in flux, providing additional evidence towards this connection between flux and data volume.

To overcome the non-stationary sample, we attempt to normalize the sample size across years. We chose a subsample size of 100 books for each year, represented by the dashed horizontal line in Fig. 4.1. Note that there really are not as many English Fiction books sampled by Google as one might think, and this subsample size is actually larger than the true sample size in the earlier years (roughly before 1830). Thus, we picked 100 as a balance between being a true subsample for most of the years and not being excessively small.

We created the yearly subsample in the following way: For a given year, y , for n -gram i , let its raw book count be $n_{b,i}$ and raw token count be $n_{t,i}$. Then, if $N_{b,y}$ is the total

number of books for year y , we obtain the sampled book count, $n'_{b,i}$, where $0 \leq n'_{b,i} \leq 100$, from a binomial distribution with 100 trials and probability of success $n_{b,i}/N_{b,y}$. That is,

$$P\left(n'_{b,i}; 100, \frac{n_{b,i}}{N_{b,y}}\right) = \binom{100}{n'_{b,i}} \left(\frac{n_{b,i}}{N_{b,y}}\right)^{n'_{b,i}} \left(1 - \frac{n_{b,i}}{N_{b,y}}\right)^{100-n'_{b,i}}. \quad (4.1)$$

Then, the sampled token count, $n'_{t,i}$, is

$$n'_{t,i} = n_{t,i} \left(\frac{n'_{b,i}}{n_{b,i}}\right). \quad (4.2)$$

That is, we are effectively sampling 100 books from all the books used for the given year, with each book equally likely to appear in our subsample. Each n -gram appears in some number of books in the full sample. We use the binomial distribution to estimate how many of these books for n -gram i are in our 100-book subsample. To find the sampled token count, we make the simplifying assumption that the tokens for the n -gram are distributed equally across the books it shows up in.

Fig. 4.11 shows the flux across the same rank thresholds for 1-grams using the sampled version of the data. Indeed, we now see that flux in the later years has largely increased compared to Fig. 4.10, and flux has largely flattened out over time, and even now seems to be increasing some in the highest ranked 1-grams.

We are unable to subsample in a way that would fully undo the effects of Google's sampling, seeing as we do not have the n -gram distribution for individual books. Furthermore, due to the very low book counts in the earlier years of Google's sample, we do not trust the representativeness of the data before 1850, where the true sample is smaller than our subsample. However, our sampling method does seem to largely undo the effects of the increasing sample size, especially in the later years, and gets us much closer to having a true normalized sample across years. Thus, we will use the sampled version of the data for

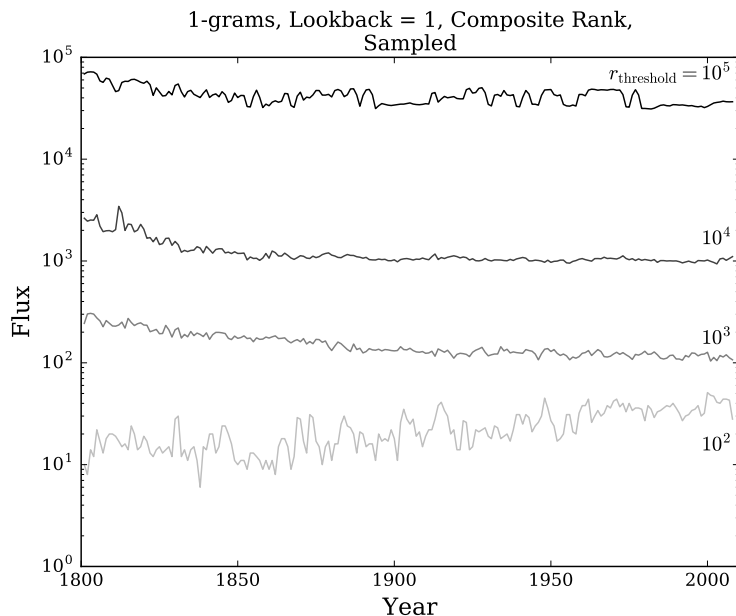


Figure 4.11: Flux for 1-grams between consecutive years using the sampled version of the data. See Fig. 4.10 caption for details. Using the sampled data largely flattens the flux time series.

the remainder of the language analysis, and will focus on the data after 1900 where we feel more confident in the results given the larger sample size.

Our subsampling allows us to normalize the data across years, and enables comparison of the Zipf distributions between years. However, because they still come from samples, there is uncertainty in the distributions, and some of the deviations between years are likely a result of this uncertainty. To deal with this, we smooth out some of the uncertainty by aggregating the distribution over some number of years. Because the subsample size is the same for all years and we are using ranks, we do not average, but just add the token counts and book counts for n -grams across the years, but the effect is the same. For the remainder of the study we will often use the subsampled data combined across 10 years for our analyses.

In Fig. 4.12 we again show the flux for 1-grams over time, but now using the subsampled data combined across 10 years. A point on the plot represents the flux across rank thresholds

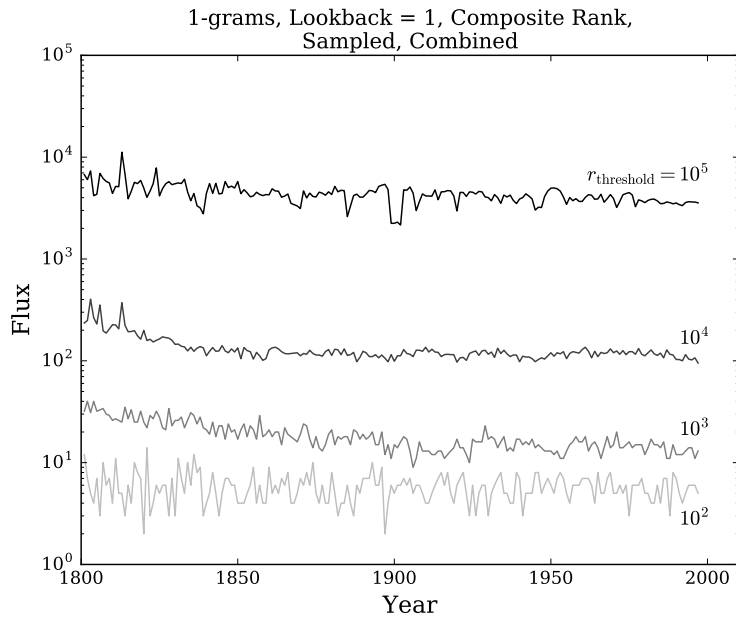


Figure 4.12: Flux for 1-grams using the sampled, combined data. A point plotted at year y measures flux crossing composite rank boundaries going from the distribution resulted from combining the ten years starting at year $y - 1$ to that from combining the ten years starting at year y .

comparing the distributions resulting from combining the subsampled data for 10 years. That is, for a point at year y , we measure the flux between the distribution resulting from combining the subsampled data for the 10 years y to $y + 9$ and the distribution resulting from combining the subsampled data for the 10 years $y - 1$ to $y + 8$. Like the version using subsampled data in Fig. 4.11, the flux is relatively constant over time. However, we see that flux has largely decreased by an order of magnitude, both due to smoothing out some of the uncertainty in the distribution and the fact that these smoothed versions now overlap in many of the years making up the distribution. We study flux more in Sec. 4.5.2, both looking at other values of n , and other values of the lookback—the number of years separating the first year of each distribution.

4.5 LEXICAL TURBULENCE

Now that we have a method to extract meaningful information from the Google Books dataset, we are able to study the linguistic and cultural evolution of language over the past two centuries.

4.5.1 ZIPF DISTRIBUTIONS

In Fig. 4.13 we provide the relative book frequency Zipf distributions for 1-grams, 2-grams, and 3-grams. For each plot, the distribution for the decade initial year for each decade from 1800–2000 is plotted in increasingly darker gray as time increases. All of the distributions have a flat plateau for the highest ranked n -grams. This plateau results from functional n -grams that appear in all or almost all of books, such as ‘the’, ‘and’, ‘of the’, ‘:’, ‘;’, ‘”’, ‘,’, and the’, etc. The distributions then roll over into a roughly power-law shaped tail.

For smaller n , the distributions have a flatter, longer plateau area, rolling over to a steeper tail. As n increases, the number of options for n -grams grows exponentially. Even among the most functional n -grams, few show up in every book like ‘the’ does, hence the shorter, less flat plateau. Also, with more options, there are more choices for roughly equally frequent n -grams, for example, ‘of the’, ‘in the’, ‘at the’, ‘on the’, etc., or ‘he said’, ‘she said’, ‘they said’, ‘we said’, ‘I said’, etc. The extra options extend the tail and reduce its slope.

If we think of the limiting case of these distributions as n increases, e.g., for 100-grams or 1000-grams, every n -gram will be unique, and the front part of the distribution will drop to 1 divided by the number of books, and the tail will extend straight out so the distribution is just a flat line at 1 divided by the number of books.

For a given n , looking at the Zipf distributions across years in Fig. 4.13, we see that they stay fairly consistent. The more recent ones appear as though they may generally be

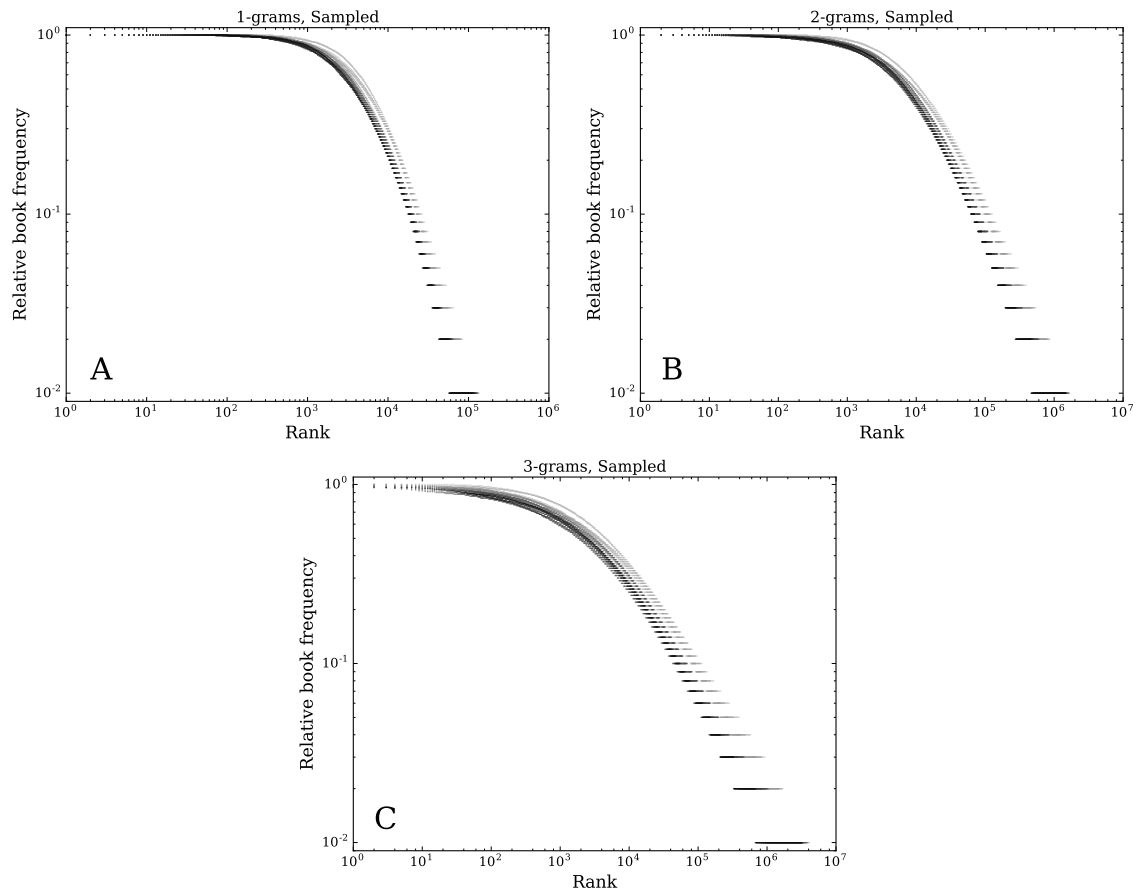


Figure 4.13: Zipf distributions for (A) 1-grams, (B) 2-grams, and (C) 3-grams based on book counts in our subsampled version of the data. The n -grams are ranked based on their relative book frequencies. Overlaying each other are distributions for every decade-initial year from 1800–2000, with more recent years plotted in darker gray.

shifted down relative to the older years, but this may be an artifact still of the increased volume of books in more recent years. With more books in the sample, and as the sample distribution approaches the true underlying distribution, it is more likely that a particular n -gram will not appear in all of the books, and the relative book frequency for the n -gram will decrease.

Thinking backwards and to the limiting case, if we reduce the number of books in our sample down to a single book, every n -gram will then have a relative frequency of 1, and we get a flat line at 1. So, as we increase the sample size, the Zipf distribution will curve down and approach the true underlying distribution, with some amount of variance that will decrease with an increased sample size. Also, the tail of the distribution will extend as more novel n -grams are encountered with each book added to the sample.

The distributions in Fig. 4.13 are based on our subsampled version of the data. However, as mentioned previously, even with the subsampling we are not able to completely undo all of the effects resulting from the changes in the underlying data size. Also, note that it is because of our subsampling down to a more effectively equal data size of 100 books that we do not see the tails for the more recent years extend much farther than the earlier years.

4.5.2 RANK FLUX

Even though we saw that the Zipf distribution for n -grams is fairly stable over the years, we would like to measure how much turbulence there is under the surface. As one measure of turbulence, we consider the ‘rank flux’ [8]. That is, when comparing the ranks of n -grams from one time period to another, how many n -grams cross a given rank threshold.

For this part of the analysis, we use the sampled, 10-year combined version of the data. Fig. 4.14 gives the rank flux for 1-grams, 2-grams, and 3-grams for selected rank thresholds with a lookback of 10 years. So, a point on the plot at year y gives the rank flux between

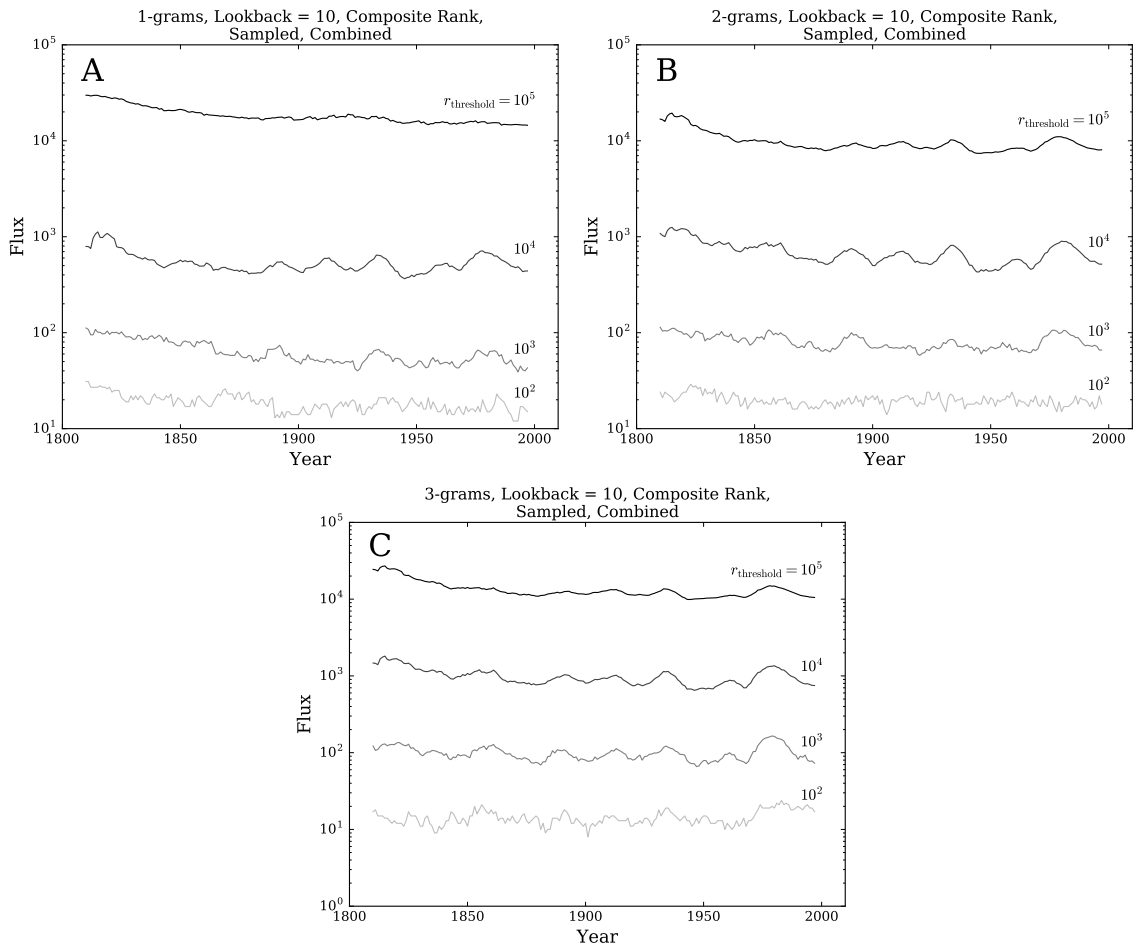


Figure 4.14: Rank flux for (A) 1-grams, (B) 2-grams, and (C) 3-grams using the sampled, combined data comparing distributions combined over 10 years and separated by 10 years. A point plotted at year y for a composite rank threshold gives the number of n -grams whose composite rank crosses that threshold (in one direction) when comparing the distribution from combining the 10 years starting at year y to the 10 years starting at year $y - 10$.

the sampled data combined across the 10 years starting with year y to the sampled data combined across the 10 years starting with year $y - 10$.

In Fig. 4.14 we see that, generally, the flux has remained constant over time. That is, language is churning at the same rate. Locally, however, we do see time periods of increased flux. We also see that the patterns of flux are quite similar across the values of n , with the largest exception being that we do not see the local bumps in flux in the lower ranked 1-grams.

In comparing Figs. 4.12 and 4.14, the local bumps in flux are not visible in 1-grams when the lookback is 1 year. Fig. 4.12 looked at rank flux with the 10-year combined sampled data with a lookback of only 1 year, so, for a given point, the two time spans being compared overlap by 9 years. So, in that case, we are comparing two dates that are ten years apart, but dampened by the overlap of the 9 years in between. However, the lack of spikes in Fig. 4.12, and also in Fig. 4.11, show that the local increases of flux in Fig. 4.14 are not due to large 1-year changes, but more so due to a gradual extra change (compared to the long-run average) over many years that add up to make one decade extra different from the previous decade. The figures are not included, but the 1-year lookback figures and comparison with 10-year ones are similar for 2-grams and 3-grams.

4.5.3 DETAILED ANALYSIS OF LANGUAGE CHANGES WITH RANK DIVERGENCE WORDSHIFT PLOTS

Now we begin a more detailed analysis of the specific changes between a few selected time periods. To measure the change in the n -gram rank distribution between two time periods we use a rank divergence measure. Let $r_i^{(1)}$ be the rank of n -gram i in the first time period and $r_i^{(2)}$ be the rank of n -gram i in the second time period. Then, we compute the rank

divergence,

$$D_{\alpha}^{\text{R}} = \sum_i \left| \frac{1}{(r_i^{(1)})^{\alpha}} - \frac{1}{(r_i^{(2)})^{\alpha}} \right|^{1/\alpha}, \quad (4.3)$$

where α is a tunable parameter.

D_{α}^{R} is a sum of contributions from each n -gram, allowing us to sort n -grams by decreasing contribution to the total rank divergence. Fig. 4.15 is an example wordshift plot that shows the top 60 contributors to the total rank divergence with $\alpha = 0.3$ for 1-grams between the sampled, combined 10 years starting in 1925 to those starting in 1935. The n -grams to the right of 0 had a higher rank in the second time period and those to the left of 0 had a higher rank in the first time period. The inset plot shows the cumulative sum of rank divergence when summing in decreasing order of contribution.

Similarly, Figs. 4.16 and 4.17 show contributions using $\alpha = 0.2$ and $\alpha = 0.1$ respectively. Comparing these three plots, we can see how larger α emphasizes changes in higher ranked, more functional n -grams such as ‘was’, ‘he’, and ‘but’ for example. Smaller α puts more emphasis on n -grams making large jumps in ranks, but from farther down the distribution, and tends to surface n -grams more representative of cultural changes or important events.

For example, when $\alpha = 0.2$ we see n -grams related to the Great Depression, like ‘C.I.O’ and ‘WPA’, and n -grams related to WWII, like ‘Gestapo’, ‘Fuhrer’, and ‘Fiihrer’ (likely an OCR mistaken translation of Führer). When $\alpha = 0.1$, even more n -grams related to these events appear in the top 60, and n -grams that showed up in the $\alpha = 0.2$ list, like ‘Gestapo’, move up even higher. We see other n -grams representing cultural and societal changes besides those related to the Great Depression and World War II as well. For example, ‘stooge’ shows up, perhaps a result of The Three Stooges, which began as a comedy group in the 1930s [14]. We also get n -grams like ‘screwball’, a type of film that became popular in the Great Depression Era [15], ‘checkroom’, ‘receptionist’ and ‘baloney’. However, even with $\alpha = 0.1$, we still see some of the more functional n -grams that made the largest jumps in rank, like ‘was’.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.3$

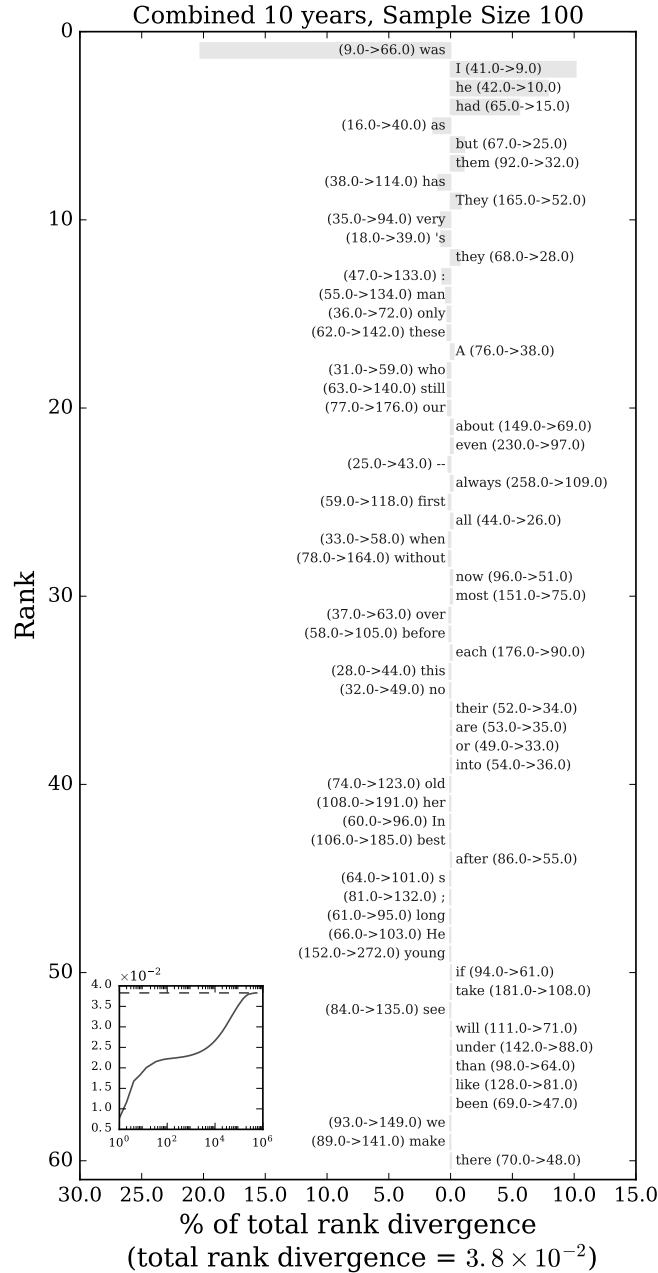


Figure 4.15: The top 60 1-grams contributing to rank divergence, Eq. 4.3, with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. The n -grams to the right of zero represent those with a larger composite rank in the more recent time period and those to the left of zero had a larger composite rank in the earlier time period. In parentheses next to each n -gram is its earlier composite rank followed by its newer composite rank. The inset plot gives the cumulative sum of rank divergence, summed in order of decreasing contribution.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.2$

Combined 10 years, Sample Size 100

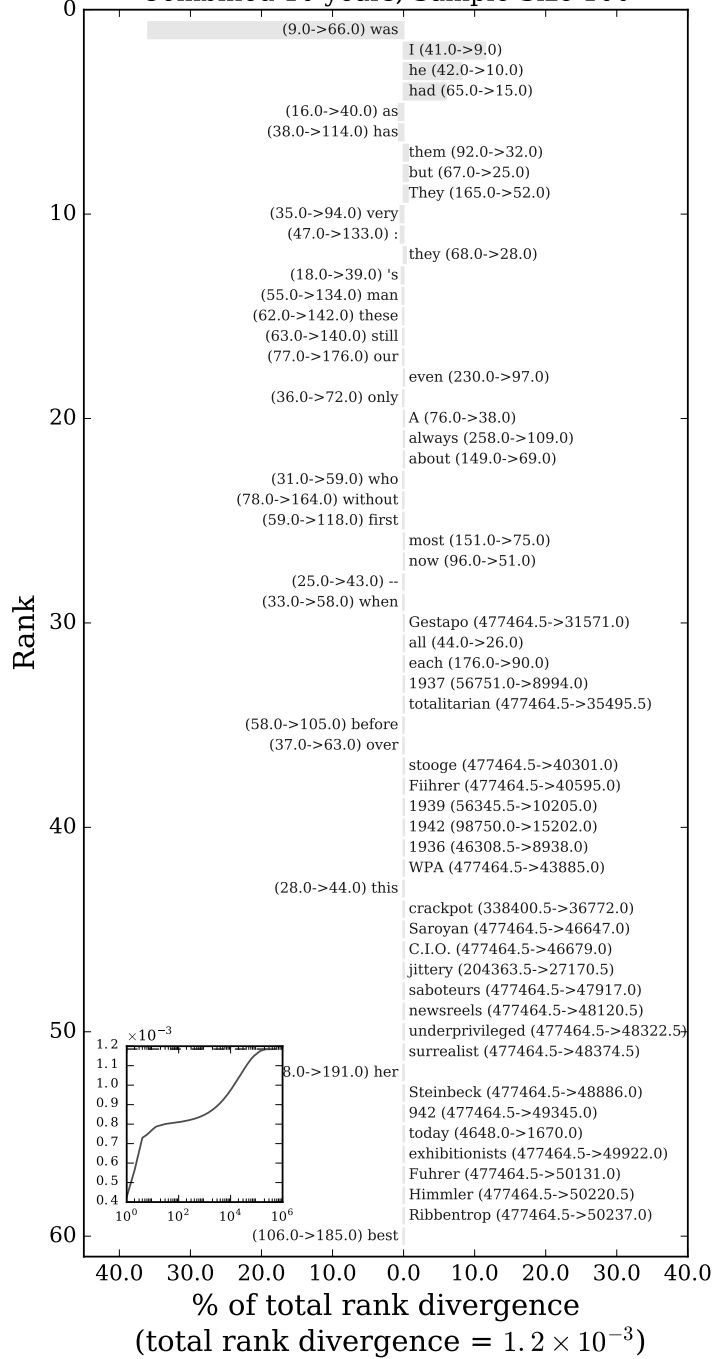


Figure 4.16: The top 60 1-grams contributing to rank divergence with $\alpha = 0.2$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.1$

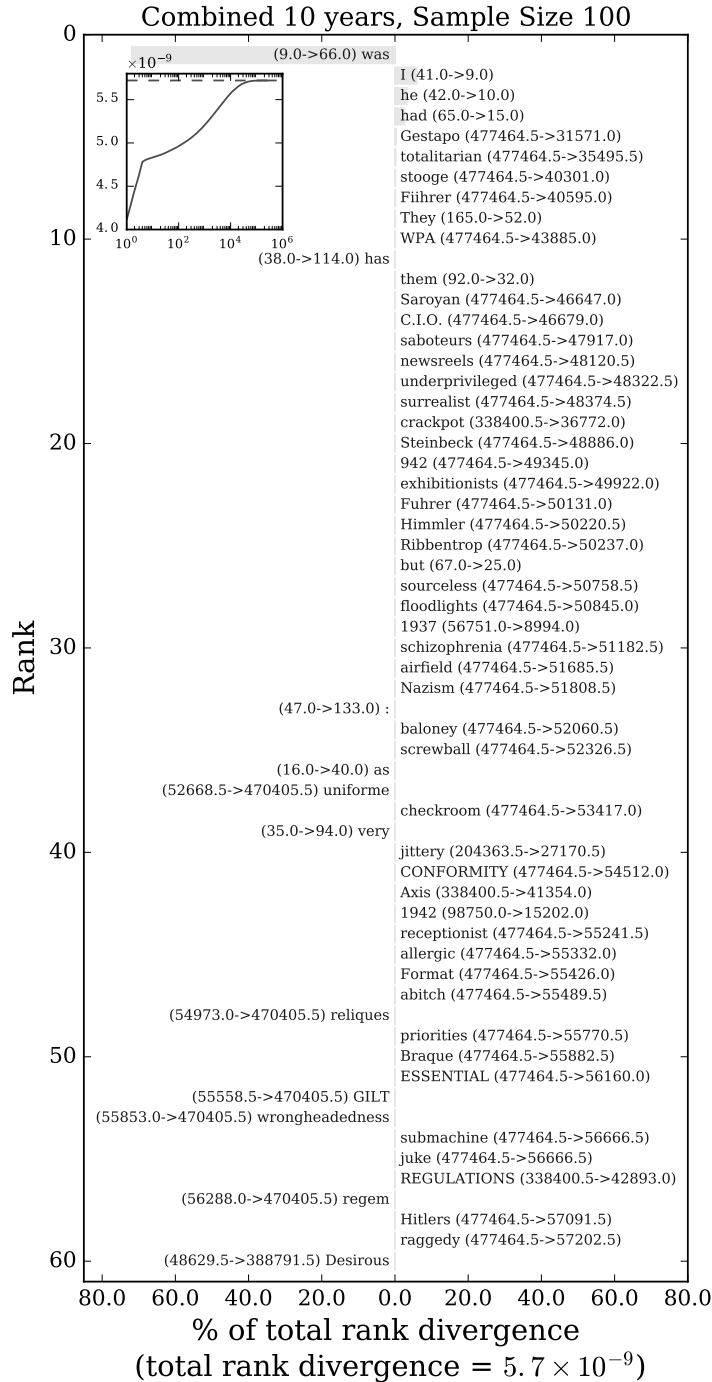


Figure 4.17: The top 60 1-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. See Fig. 4.15's caption for details.

Figs. 4.18 and 4.19 give the wordshift plots for the same comparison years for bigrams with $\alpha = 0.3$ and $\alpha = 0.1$, respectively. Similarly, Figs. 4.20 and 4.21 are for trigrams. Here, we still generally see the same stories emerging. Namely, for $\alpha = 0.3$ the largest contributors to rank divergence are more functional n -grams, and for $\alpha = 0.1$ we get many more n -grams related to cultural changes and current events. ‘New Deal’ and many Hitler and Nazi related bigrams show up for $\alpha = 0.1$, but so does ‘slacks .’, perhaps representative of a change in popular clothing style. Besides the abundance of World War II related 3-grams in Fig. 4.21, we also see 3-grams related to drugstores, ‘air - conditioned’, ‘first World War’, which prior to World War II was called the Great War, and copyright related information.

Fig. 4.20 shows an interesting linguistic shift with $\alpha = 0.3$. We see that 3-grams like ‘to - day’, ‘to - morrow’, and ‘to - night’ are decreasing in rank. Correspondingly, in Fig. 4.16 with $\alpha = 0.2$ we see the increase in the 1-gram ‘today’. We see that the early 1930s are a transition point in English where hyphenating these words loses popularity. A word like ‘today’ would have started its life as two separate words, but because it was such a common construction, the two words begin to appear hyphenated, indicating two words that are becoming more of a single unit of meaning. Then, over time, as this connection becomes stronger, the hyphen is dropped and the two words fully morph into a new single word, and the original way of thinking about the concept in relation to two other independent concepts is completely lost.

This is a repeated process through the history and evolution of language, with a more recent example being the transition from ‘electronic mail’ to ‘e-mail’ to ‘email’. Email, being a new concept, started by relating back to a known concept. It was thought of as being an electronic form of regular mail, and its two-word name reflected that. Then, over time, it developed its own identity, and no longer required reference to an older concept.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.3$

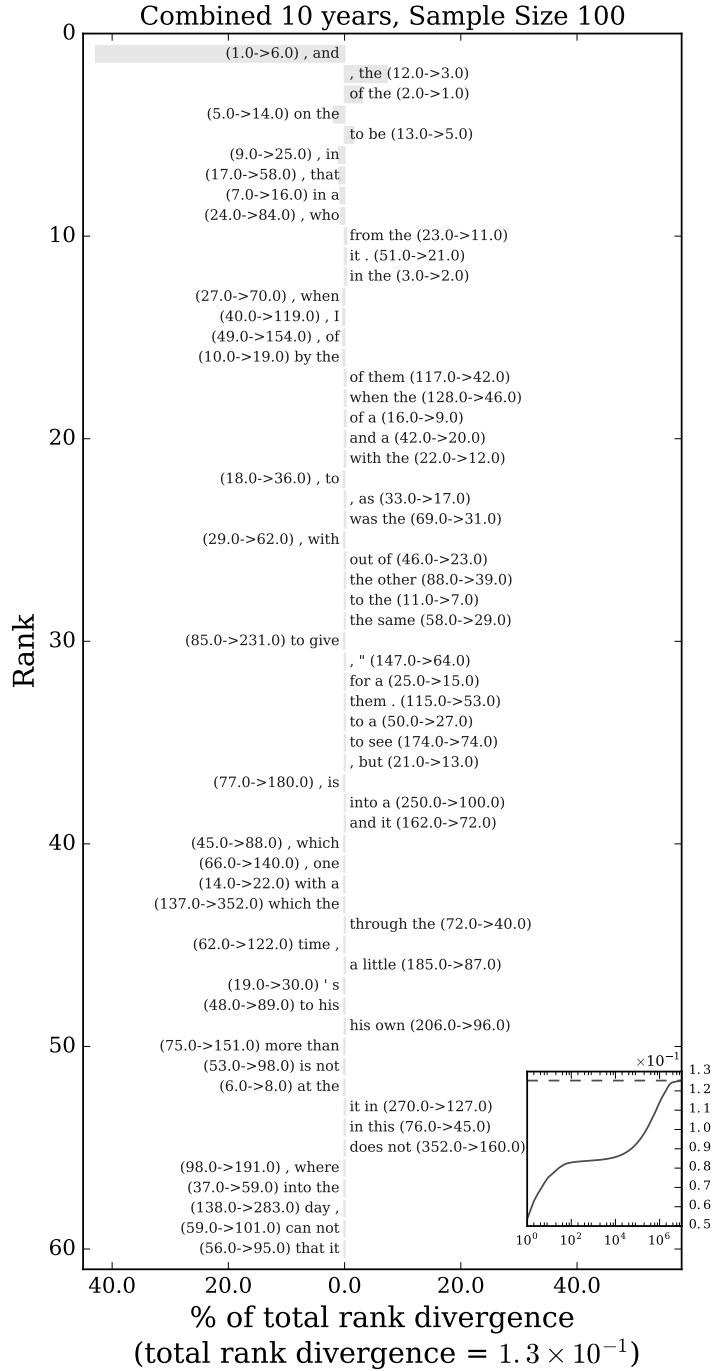


Figure 4.18: The top 60 2-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.1$

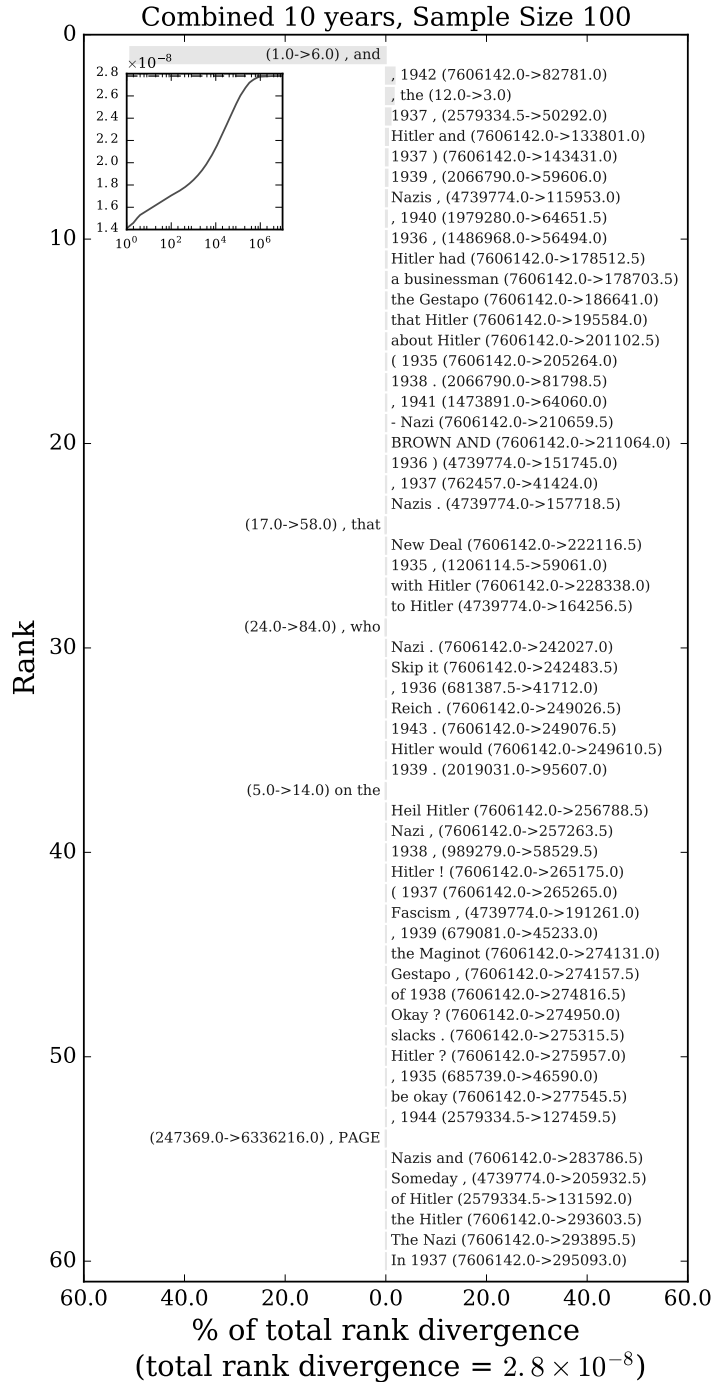


Figure 4.19: The top 60 2-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.3$

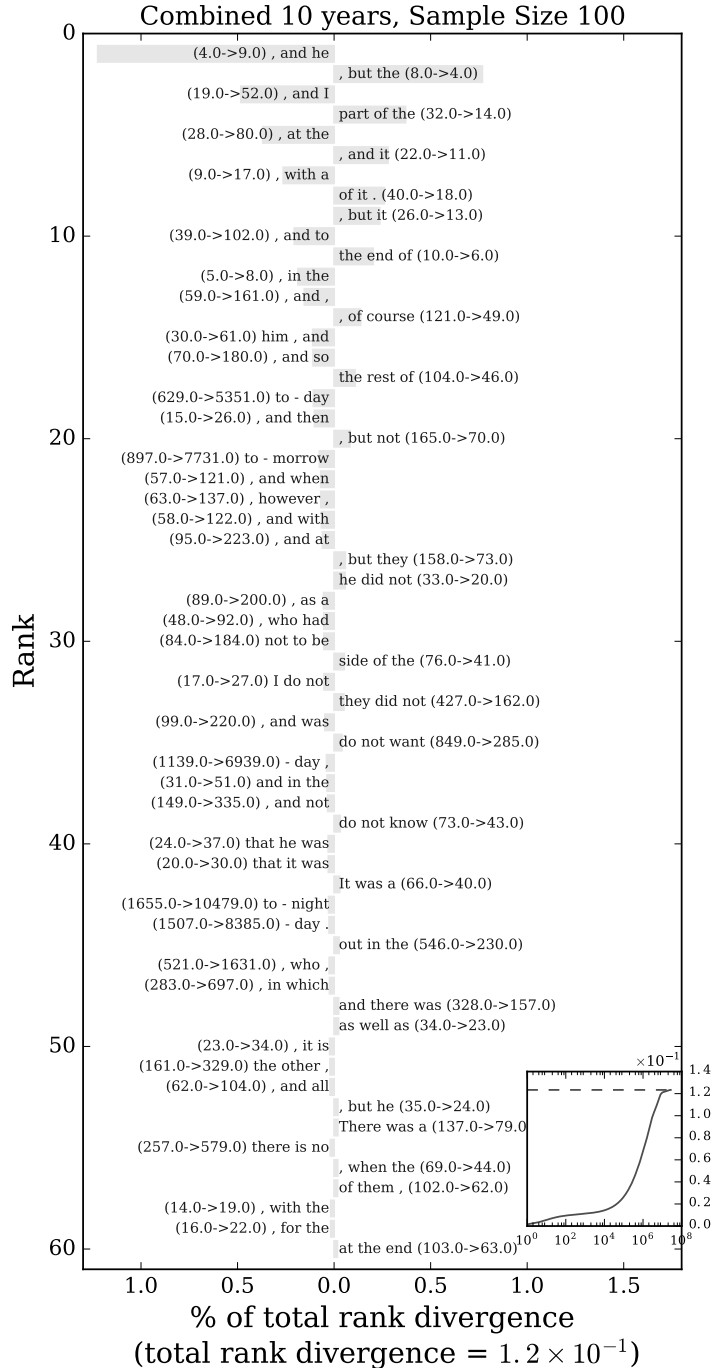


Figure 4.20: The top 60 3-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1925 to 1935
 $\alpha = 0.1$

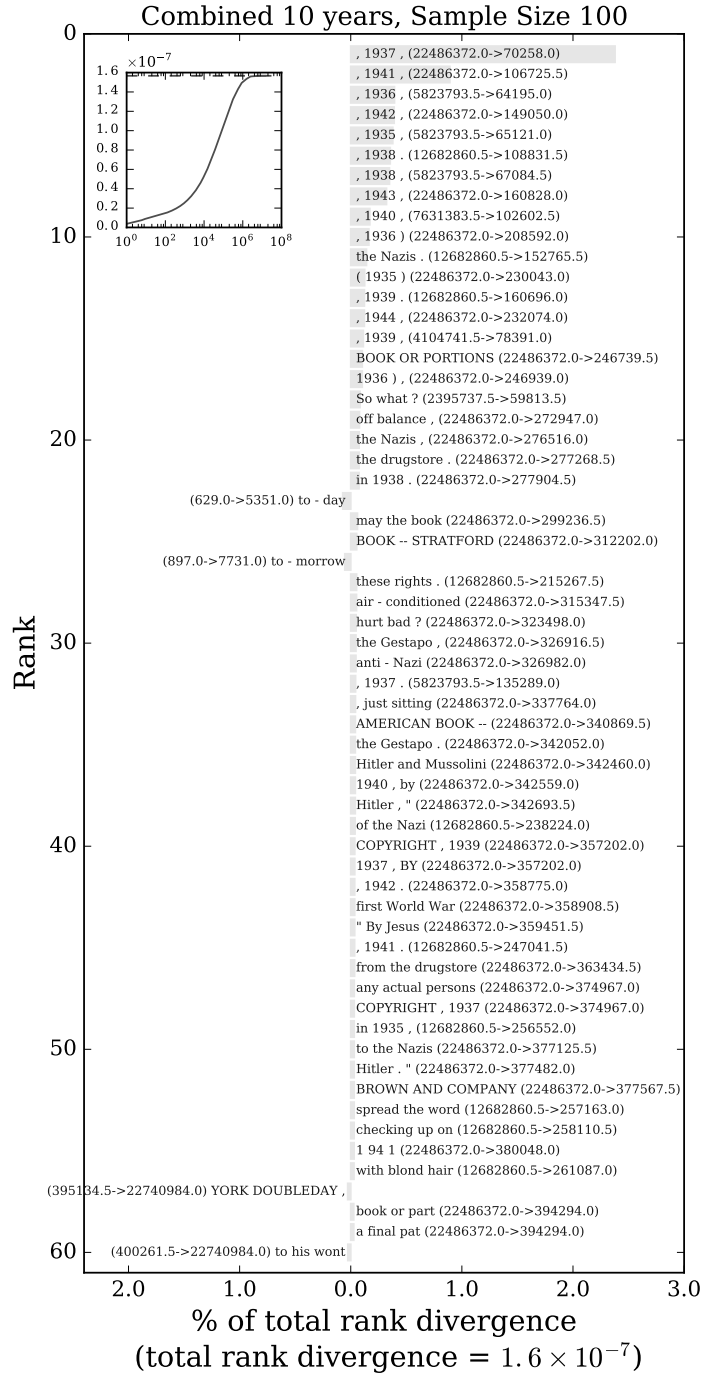


Figure 4.21: The top 60 3-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1925 to that starting in 1935. See Fig. 4.15's caption for details.

Next, we look at changes over the 20th century by comparing the sampled, combined data for the first decade of the 1900s to that of the last decade. Figs. 4.22 and 4.23 give the 1-gram wordshift plots for $\alpha = 0.3$ and $\alpha = 0.1$, respectively. For changes in the more functional parts of language, we see ‘that’ rose in ranks while ‘which’ fell; the functionality of this pair of words is often confused. Also, we see the decrease of ‘morrow’, being part of ‘to - morrow’, and the complementary increase of ‘today’ again. The less formal words ‘Yeah’ and ‘okay’ jumped up in ranks over the century. We also see ‘parked’ showing up in the top 60 for $\alpha = 0.3$, representative of the dramatic change in personal transportation over the century.

Fig. 4.23 is rich with words whose rank fluxes reflect our expectations regarding cultural and societal changes. The list contains new technology including electronics (e.g., ‘computer’ and ‘television’) as well as transportation words (e.g., ‘airport’, ‘windshield’, ‘parking’, and ‘motel’) and other technological changes (e.g., ‘fridge’). There is also an increase in profane language. The newer concept of the teenage phase of life is apparent with the increase of n -grams like ‘teenage’, ‘boyfriend’, and ‘girlfriend’. We see changes in attire with words like ‘sunglasses’ and ‘slacks’. We even see food changes including a large rank increase for ‘pizza’. There are more n -grams here reflecting different aspects of societal evolution that we have not highlighted, and there would be even more if we went farther along the list.

Figs. 4.24 and 4.25 compare the first and last decades of the 20th century for bigrams with $\alpha = 0.3$ and $\alpha = 0.1$, respectively. They tell much the same story. We see the increase of more colloquial, less formal conversational language with the increased use of ‘okay’ and ‘yeah’ showing up in multiple bigrams on the list. We also see increased appearance of phone related bigrams, radio related bigrams, the concept of a weekend, and ‘World War’ (‘Vietnam’ appeared in the 1-grams).

Composite Rank Divergence Contributions:
1900 to 1990
 $\alpha = 0.3$

Combined 10 years, Sample Size 100

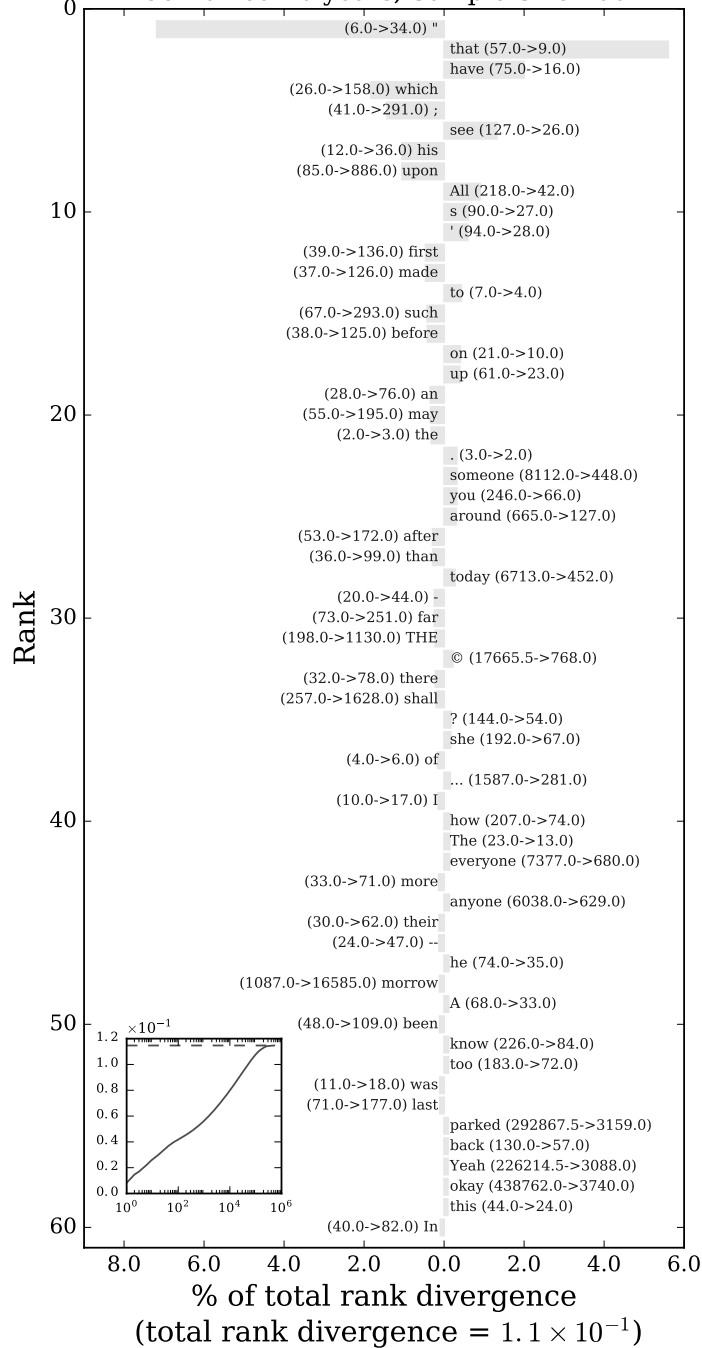


Figure 4.22: The top 60 1-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1900 to 1990
 $\alpha = 0.1$

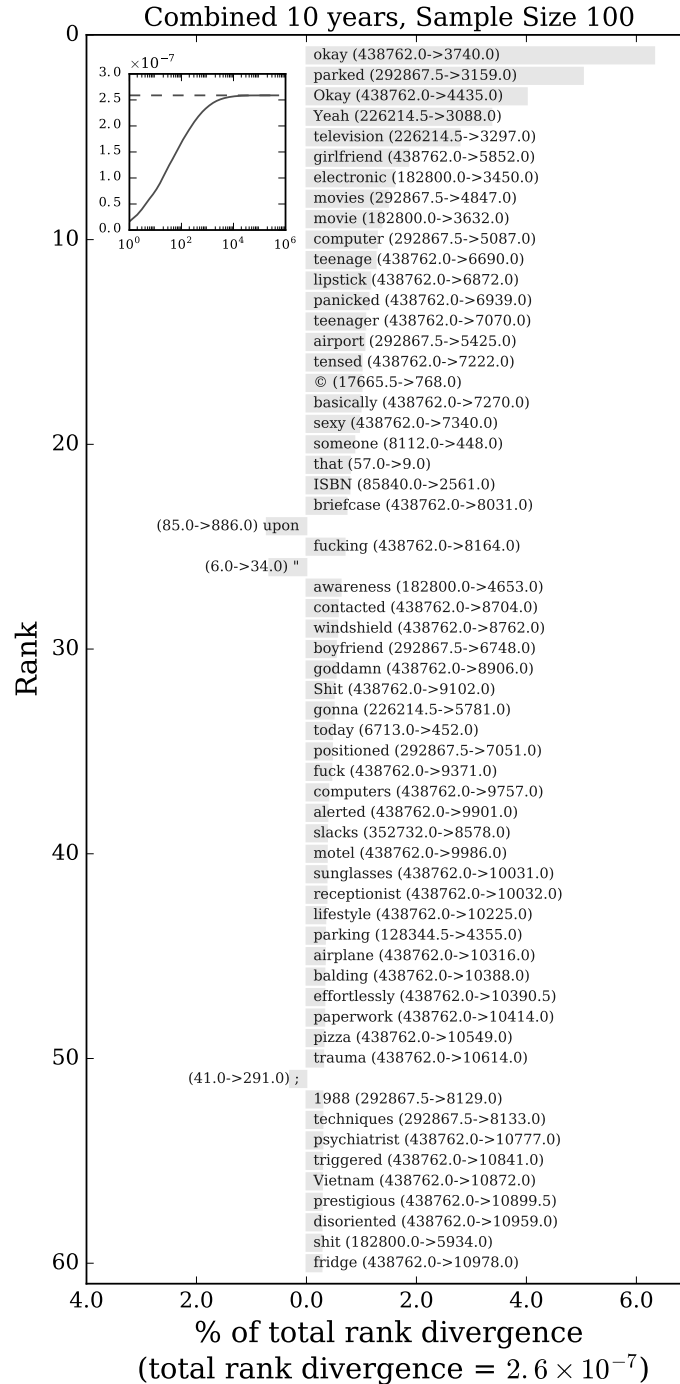


Figure 4.23: The top 60 1-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1900 to 1990
 $\alpha = 0.3$

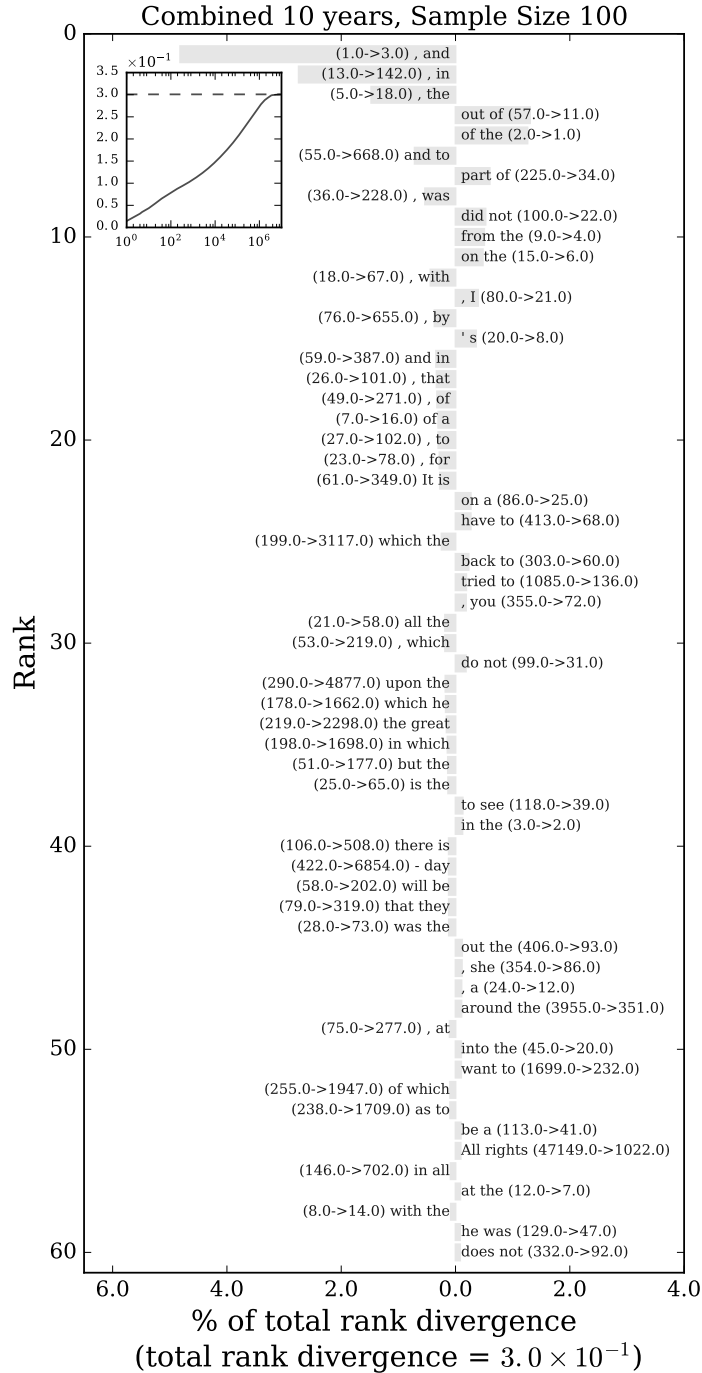


Figure 4.24: The top 60 2-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1900 to 1990
 $\alpha = 0.1$

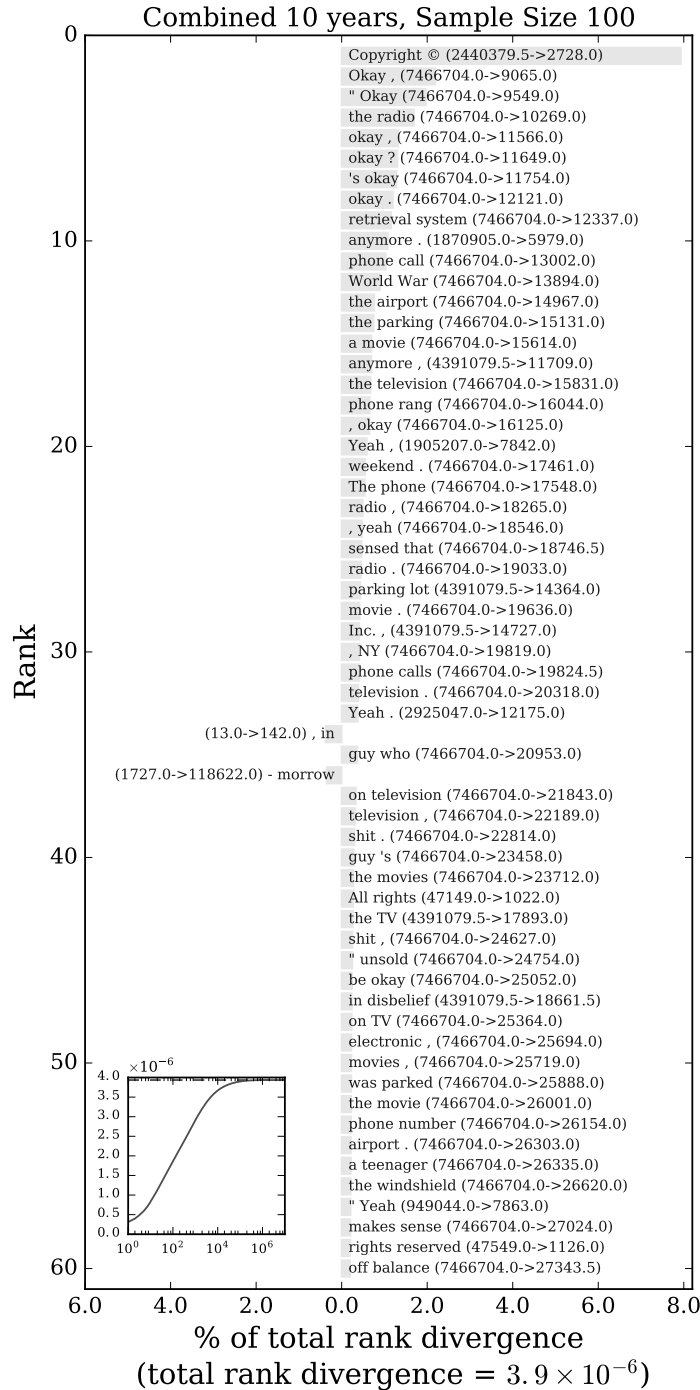


Figure 4.25: The top 60 2-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990. See Fig. 4.15's caption for details.

Figs. 4.26 and 4.27 make the same comparison across the century for trigrams for $\alpha = 0.3$ and $\alpha = 0.1$. We again see the change in words like ‘today’ and ‘tomorrow’. There is an increase in trigrams with ‘not’ in them. The increased use of ‘okay’ and ‘yeah’ is also quite prevalent in these trigrams. The increased popularity of T-shirts is apparent. Technology terms are present, but they are not as prevalent for trigrams as they were for bigrams and unigrams.

Many of the trigrams appearing in Fig. 4.27 come from the copyright page of books. Some are obvious, like ‘All rights reserved’, but even ones like ‘retrieval system ,’ and ‘recording , or’ are from the copyright page. Also, the warning about buying ‘stripped books’ [16] that appears on copyright pages is responsible for many of the trigrams here.

The prevalence of these copyright related trigrams is exemplary of a problem that still exists within the Google Books corpus that our improved method does not sort out. Namely, n -grams that occur in many books, but only about once per book, which do not occur in natural language, but are a result of book features like standardized front matter in books that was scanned in with the rest of the book. We call these n -grams ‘pathological hapax legomena’.

Fig. 4.28 gives one other example where the problem of n -grams appearing only once per book but in many books is apparent. Besides the copyright related n -grams in this wordshift we also see many n -grams specifically related to what library the book comes from and library checkout information, such as ‘CALIFORNIA LIBRARY BERKELEY’, ‘book is DUE’, and ‘Return to desk’.

4.6 CONCLUDING REMARKS

In this paper, we have shown that using token counts in the Google Books corpus to study language is fundamentally flawed. In an effort to resurrect the dataset, we have provided

Composite Rank Divergence Contributions:
1900 to 1990
 $\alpha = 0.3$

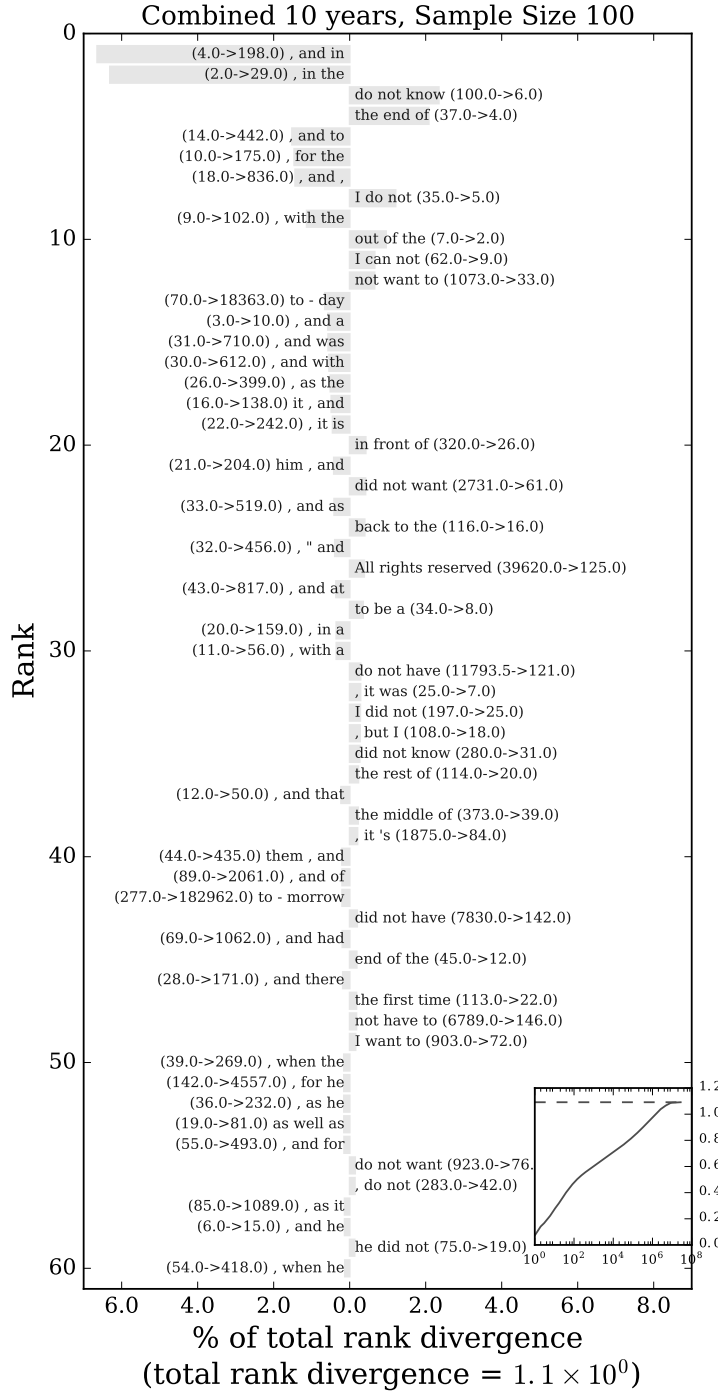


Figure 4.26: The top 60 3-grams contributing to rank divergence with $\alpha = 0.3$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1900 to 1990

$\alpha = 0.1$

Combined 10 years, Sample Size 100

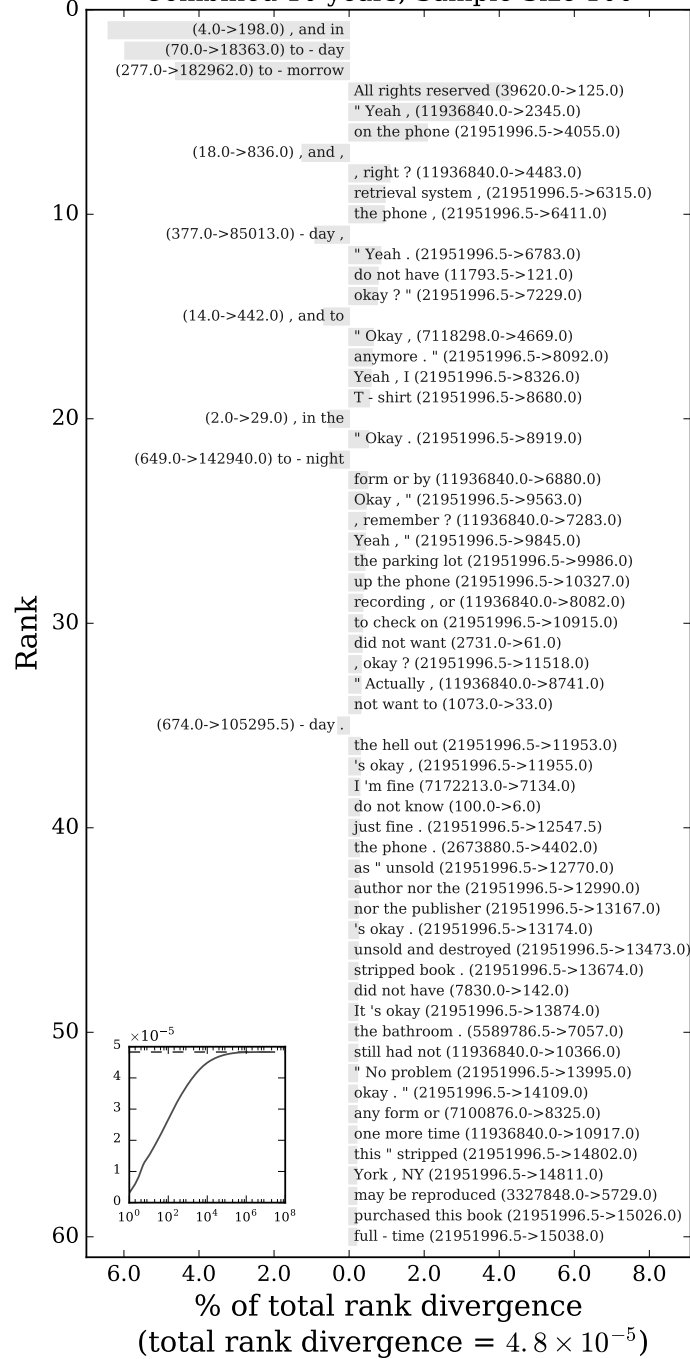


Figure 4.27: The top 60 3-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1900 to that starting in 1990. See Fig. 4.15's caption for details.

Composite Rank Divergence Contributions:
1947 to 1957

$\alpha = 0.1$

Combined 10 years, Sample Size 100

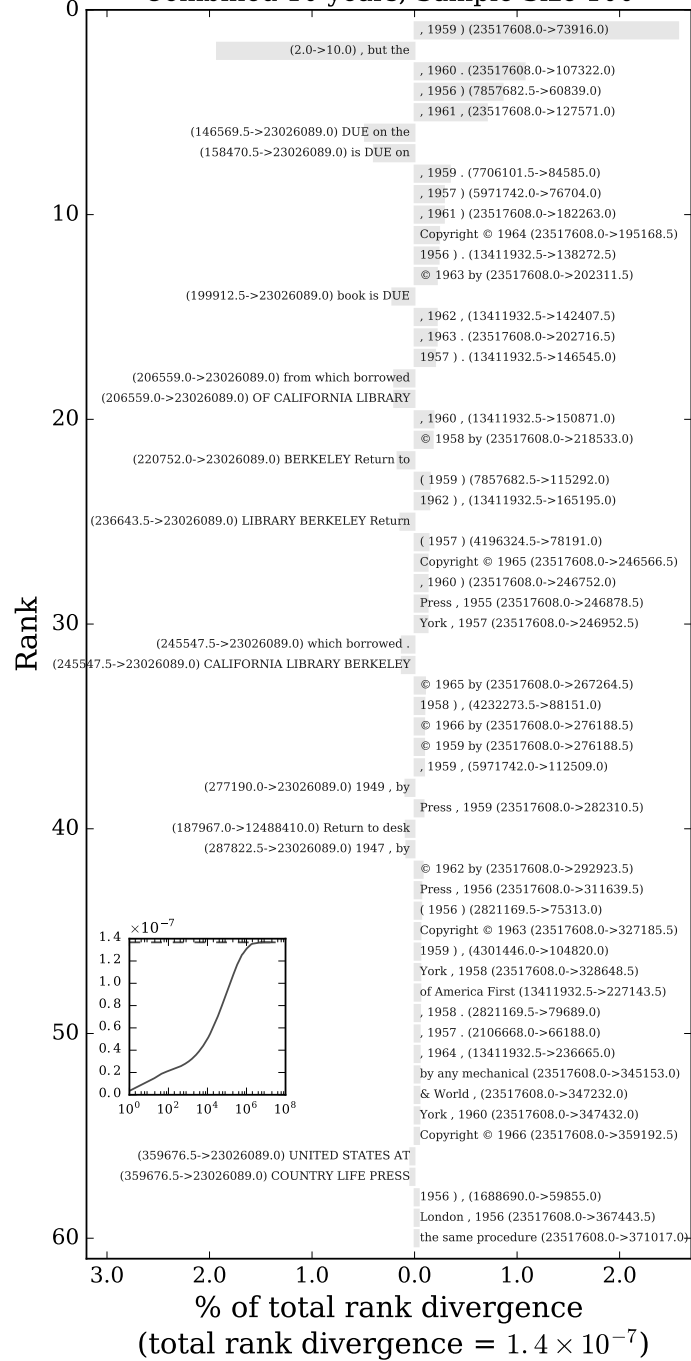


Figure 4.28: The top 60 3-grams contributing to rank divergence with $\alpha = 0.1$ between the sampled, 10-year combined distribution starting in 1947 to that starting in 1957. See Fig. 4.15's caption for details.

a method that produces more meaningful results, and used this method to study linguistic and cultural trends, reflective of a ‘collective author’s voice’.

With our revised method, due to the construction of the corpus there are still some results that do not reflect natural language. For example, we find an increase in copyright and publishing related n -grams in later years. We have also observed n -grams showing up related to library information. However, even with these remaining issues, our revised method does clear up many of the problems present within the Google Books corpus, and future work can likely result in a further improved method that discounts pathological hapax legomena, those n -grams that appear in many books, but only roughly once per book.

Future work might include looking at time series for n -grams and seeing which ones change the most based on the composite ranking, rather than a token count based time series. It would also be interesting to see in what ways studying language using the token counts has deceived us and what has been missed by using token counts.

4.7 ACKNOWLEDGMENTS

CMD and PSD were supported by NSF Grant No. IIS-1447634, and TJG, CMD, and PSD were supported by a gift from MassMutual.

4.8 REFERENCES

- [1] Mitchell G. Newberry, Christopher A. Ahern, Robin Clark, and Joshua B. Plotkin. Detecting evolutionary forces in language change. *Nature*, 551(7679):223, 2017.
- [2] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.
- [3] Yuri Lin, Jean-Baptiste Michel, Erez Lieberman Aiden, Jon Orwant, Will Brockman, and Slav Petrov. Syntactic annotations for the Google Books Ngram corpus. In *Pro-*

- ceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 169–174, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.
- [4] Martin Gerlach and Eduardo G. Altmann. Stochastic model for the vocabulary growth in natural languages. *Physical Review X*, 3(2), 2013.
 - [5] Tyler J. Gray, Andrew J. Reagan, Peter Sheridan Dodds, and Christopher M. Danforth. English verb regularization in books and tweets. *PLOS ONE*, 13(12):1–17, 12 2018.
 - [6] Alexander M. Petersen, Joel N. Tenenbaum, Shlomo Havlin, H. Eugene Stanley, and Matjaž Perc. Languages cool as they expand: Allometric scaling and the decreasing need for new words. *Scientific Reports*, 2:943, 12 2012.
 - [7] Alexander M. Petersen, Joel Tenenbaum, Shlomo Havlin, and H. Eugene Stanley. Statistical laws governing fluctuations in word use from word birth to word death. *Scientific Reports*, 2:313, 03 2012.
 - [8] Eitan Adam Pechenick, Christopher M. Danforth, and Peter Sheridan Dodds. Is language evolution grinding to a halt? The scaling of lexical turbulence in English fiction suggests it is not. *Journal of Computational Science*, 21:24–37, 2017.
 - [9] George Kingsley Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley Press, 1949.
 - [10] Eitan Adam Pechenick, Christopher M. Danforth, and Peter Sheridan Dodds. Characterizing the Google Books corpus: Strong limits to inferences of socio-cultural and linguistic evolution. *PLOS ONE*, 10(10):1–24, 10 2015.
 - [11] The Google Ngram Viewer Team. About Ngram Viewer. <https://books.google.com/ngrams/info>, 2013.
 - [12] The Google Ngram Viewer Team. Google Ngram Viewer. <https://books.google.com/ngrams/>, 2013.
 - [13] The Google Ngram Viewer Team. Ngram datasets. <http://storage.googleapis.com/books/ngrams/books/datasetsv2.html>, 2013.
 - [14] The Three Stooges — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=The_Three_Stooges&oldid=899277965. Accessed: 2019-05-29.
 - [15] Screwball — Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/w/index.php?title=Screwball&oldid=896733307>. Accessed: 2019-05-29.
 - [16] Stripped book — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Stripped_book&oldid=810859504. Accessed: 2019-05-29.

CHAPTER 5

CONCLUDING REMARKS

In this dissertation we have used two large text-based datasets to study the patterns and evolution of language, culture, and society. Both of these datasets, the Google Books corpus and the collection of messages, or ‘tweets’, from Twitter, provide a view of language for a given time period. However, they give slightly different views in that the data of Google Books is from a more formal edited form of language and the data from Twitter is a closer representation of the colloquial language used by everyday speakers. Furthermore, the Google Books dataset covers a longer period of time allowing for the study of temporal trends in ways that cannot be done with Twitter, and similarly, Twitter provides the possibility of more fine-grained location information, allowing for regional analyses that cannot be done using Google books. When choosing any dataset to study language, it is important to identify and keep in mind both the benefits and the limitations of each corpus.

For example, with our study on verb regularization, we found that the current state of verb regularization differed between Google Books and Twitter, showing that the edited text of books does not necessarily reflect the colloquial language of everyday speakers. Also, the advent of social media has allowed for the study of stretchable words, which are prevalent in natural spoken language, but are rarely found in books, and thus cannot be studied with a corpus like Google Books. The ability to study stretchable words has now led to methods and tools that have potential applications in further areas of language analysis, natural language processing, and beyond.

Finally, our restructuring of the Google Books corpus, focusing on the collective authorial voice provided by the utilization of book counts as the main proxy for word rank,

has overcome many of the problems and irregularities associated with this dataset. This is an unprecedented dataset, rich with information, used by many researchers in the study of language and culture (Gerlach and Altmann, 2013; Gray et al., 2018; Michel et al., 2011; Pechenick et al., 2017; Petersen et al., 2012a,b). However, it is important when studying language in this way that the results correspond to the questions asked. It is important that we, as scientists, are not misled by spurious results due to irregularities that exist within our datasets. It is for these reasons that the Google Books corpus required an in-depth examination to make sure it is used appropriately and that the results derived from it are justified. This examination led to our revised methods for analyzing the Google Books dataset, and these revised methods now finally allow for a more meaningful output, better reflective of natural language, and the ways in which language has been affected by cultural and social factors as well as important historical events.

BIBLIOGRAPHY

- Arthur, R. and Williams, H. T. P. (2017). Scaling laws in geo-located Twitter data. *CoRR*, abs/1711.09700.
- Baker, P. (2017). *American and British English: Divided by a Common Language?* Cambridge University Press.
- Donoso, G. and Sanchez, D. (2017). Dialectometric analysis of language variation in Twitter. *CoRR*, abs/1702.06777.
- Duggan, M. (2015). The demographics of social media users. <http://www.pewinternet.org/2015/08/19/the-demographics-of-social-media-users/>. Accessed: 2016-07-18.
- Eisenstein, J., O’Connor, B., Smith, N. A., and Xing, E. P. (2010). A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP ‘10*, pages 1277–1287, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Eisenstein, J., O’Connor, B., Smith, N. A., and Xing, E. P. (2014). Diffusion of lexical change in social media. *PLOS ONE*, 9(11):1–13.
- Gerlach, M. and Altmann, E. G. (2013). Stochastic model for the vocabulary growth in natural languages. *Physical Review X*, 3(2).
- Gligorić, K., Anderson, A., and West, R. (2018). How constraints affect content: The case of Twitter’s switch from 140 to 280 characters. *International AAAI Conference on Web and Social Media*.
- Gonçalves, B., Loureiro-Porto, L., Ramasco, J. J., and Sánchez, D. (2018). Mapping the Americanization of English in space and time. *PLOS ONE*, 13(5):1–15.
- Gonçalves, B. and Sánchez, D. (2014). Crowdsourcing dialect characterization through Twitter. *PLOS ONE*, 9(11):1–6.
- Gray, T. J., Danforth, C. M., and Dodds, P. S. (2019). Hahahahaha, Duuuuude, Yeeessss!: A two-parameter characterization of stretchable words and the dynamics of mistypings and misspellings. Manuscript submitted for publication.
- Gray, T. J., Reagan, A. J., Dodds, P. S., and Danforth, C. M. (2018). English verb regularization in books and tweets. *PLOS ONE*, 13(12):1–17.
- Greenhill, S. J., Wu, C.-H., Hua, X., Dunn, M., Levinson, S. C., and Gray, R. D. (2017). Evolutionary dynamics of language systems. *Proceedings of the National Academy of Sciences*, 114(42):E8822–E8829.

- Grieve, J., Speelman, D., and Geeraerts, D. (2011). A statistical method for the identification and aggregation of regional linguistic variation. *Language Variation and Change*, 23(2):193–221.
- Huang, Y., Guo, D., Kasakoff, A., and Grieve, J. (2016). Understanding US regional linguistic variation with Twitter data analysis. *Computers, Environment and Urban Systems*, 59:244–255.
- Kloumann, I. M., Danforth, C. M., Harris, K. D., Bliss, C. A., and Dodds, P. S. (2012). Positivity of the English language. *PLOS ONE*, 7(1):1–7.
- Lieberman, E., Michel, J.-B., Jackson, J., Tang, T., and Nowak, M. A. (2007). Quantifying the evolutionary dynamics of language. *Nature*, 449(7163):713–716.
- Lin, Y., Michel, J.-B., Aiden, E. L., Orwant, J., Brockman, W., and Petrov, S. (2012). Syntactic annotations for the Google Books Ngram corpus. In *Proceedings of the ACL 2012 System Demonstrations*, ACL ‘12, pages 169–174, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., The Google Books Team, Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., Pinker, S., Nowak, M. A., and Aiden, E. L. (2011). Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182.
- Mill, B. (2008). Drawing presentable trees. *Python Magazine*, 2(8).
- Mill, B. (2013). Github — llimllib/pymag-trees: Code from the article “Drawing good-looking trees” in Python Magazine. <https://github.com/llimllib/pymag-trees/tree/9acfb8d52a09a495f25af91dcbf438499546748b>. Accessed: 2019-01-21.
- Newberry, M. G., Ahern, C. A., Clark, R., and Plotkin, J. B. (2017). Detecting evolutionary forces in language change. *Nature*, 551(7679):223.
- Ord, J. K. and Getis, A. (1995). Local spatial autocorrelation statistics: Distributional issues and an application. *Geographical Analysis*, 27(4):286–306.
- Pechenick, E. A., Danforth, C. M., and Dodds, P. S. (2015). Characterizing the Google Books corpus: Strong limits to inferences of socio-cultural and linguistic evolution. *PLOS ONE*, 10(10):1–24.
- Pechenick, E. A., Danforth, C. M., and Dodds, P. S. (2017). Is language evolution grinding to a halt? The scaling of lexical turbulence in English fiction suggests it is not. *Journal of Computational Science*, 21:24–37.
- Petersen, A. M., Tenenbaum, J., Havlin, S., and Stanley, H. E. (2012a). Statistical laws governing fluctuations in word use from word birth to word death. *Scientific Reports*, 2:313.
- Petersen, A. M., Tenenbaum, J. N., Havlin, S., Stanley, H. E., and Perc, M. (2012b). Languages cool as they expand: Allometric scaling and the decreasing need for new words. *Scientific Reports*, 2:943.

- Ramiro, C., Srinivasan, M., Malt, B. C., and Xu, Y. (2018). Algorithms in the historical emergence of word senses. *Proceedings of the National Academy of Sciences*, 115(10):2323–2328.
- Real, F., Chater, N., and Christiansen, M. H. (2018). Simpler grammar, larger vocabulary: How population size affects language. *Proceedings of the Royal Society of London B: Biological Sciences*, 285(1871).
- Simpson, J. A. and Weiner, E. S. C., editors (1989). *The Oxford English Dictionary*. Oxford University Press, Oxford, 2nd edition.
- The Google Ngram Viewer Team (2013a). About Ngram Viewer. <https://books.google.com/ngrams/info>.
- The Google Ngram Viewer Team (2013b). Google Ngram Viewer. <https://books.google.com/ngrams/>.
- The Google Ngram Viewer Team (2013c). Ngram datasets. <http://storage.googleapis.com/books/ngrams/books/datasetsv2.html>.
- Wetherell, C. and Shannon, A. (1979). Tidy drawings of trees. *IEEE Transactions on Software Engineering*, SE-5(5):514–520.
- Wikipedia contributors (2019a). Cutoff frequency — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Cutoff_frequency&oldid=873937426. Accessed: 2019-05-07.
- Wikipedia contributors (2019b). Screwball — Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/w/index.php?title=Screwball&oldid=896733307>. Accessed: 2019-05-29.
- Wikipedia contributors (2019c). Stripped book — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Stripped_book&oldid=810859504. Accessed: 2019-05-29.
- Wikipedia contributors (2019d). The Three Stooges — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=The_Three_Stooges&oldid=899277965. Accessed: 2019-05-29.
- Wikipedia contributors (2019e). Typosquatting — Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/w/index.php?title=Typosquatting&oldid=884561229>. Accessed: 2019-05-12.
- Wiktionary contributors (2019a). Appendix: Glossary — Wiktionary, the free dictionary. <https://en.wiktionary.org/w/index.php?title=Appendix:Glossary&oldid=51610328>. Accessed: 2019-03-24.
- Wiktionary contributors (2019b). Talk: cuuute — Wiktionary, the free dictionary. <https://en.wiktionary.org/w/index.php?title=Talk:cuuute&oldid=51216685>. Accessed: 2019-03-24.

- Wiktionary contributors (2019c). Wiktionary: Criteria for inclusion — Wiktionary, the free dictionary. https://en.wiktionary.org/w/index.php?title=Wiktionary:Criteria_for_inclusion&oldid=52749064. Accessed: 2019-05-12.
- Wiktionary contributors (2019d). Wiktionary, the free dictionary. https://en.wiktionary.org/wiki/Wiktionary:Main_Page. Accessed: 2019-05-12.
- Zipf, G. K. (1949). *Human Behavior and the Principle of Least Effort*. Addison-Wesley Press.