CMPU 366 · Natural Language Processing

Language Models

15 September 2025

VASSAR COLLEGE
A·D·1861

Where are we?

What is a corpus?

Word type vs word token

The cat sat on the mat.

What is tokenization?

Language models

The idea of a statistical *language model* (LM) is to compute the probability of a sequence of words.

Why should we care about these probabilities?

Speech recognition

$P(\textit{I will be back soonish}) > P(\textit{I will be bassoon dish})$

Speech recognition

$P(\textit{I will be back soonish}) > P(\textit{I will be bassoon dish})$

Spelling correction

The office is about fifteen minuets from my house.

$P(\textit{about fifteen minutes from}) > P(\textit{about fifteen minuets from})$

Speech recognition

P(*I will be back soonish*) > *P*(*I will be bassoon dish*)

Spelling correction

The office is about fifteen minuets from my house.

P(*about fifteen minutes from*) > *P*(*about fifteen minuets from*)

Machine translation

Translating *The doctor recommended a cat scan*,

P(*La doctora recomendó una tomografía*) >
P(*La doctora recomendó una exploración del gato*)

And more!

These are examples of computing

$P(W) = P(w_1, w_2, w_3, \ldots, w_n)$,

the probability of a sequence of words,

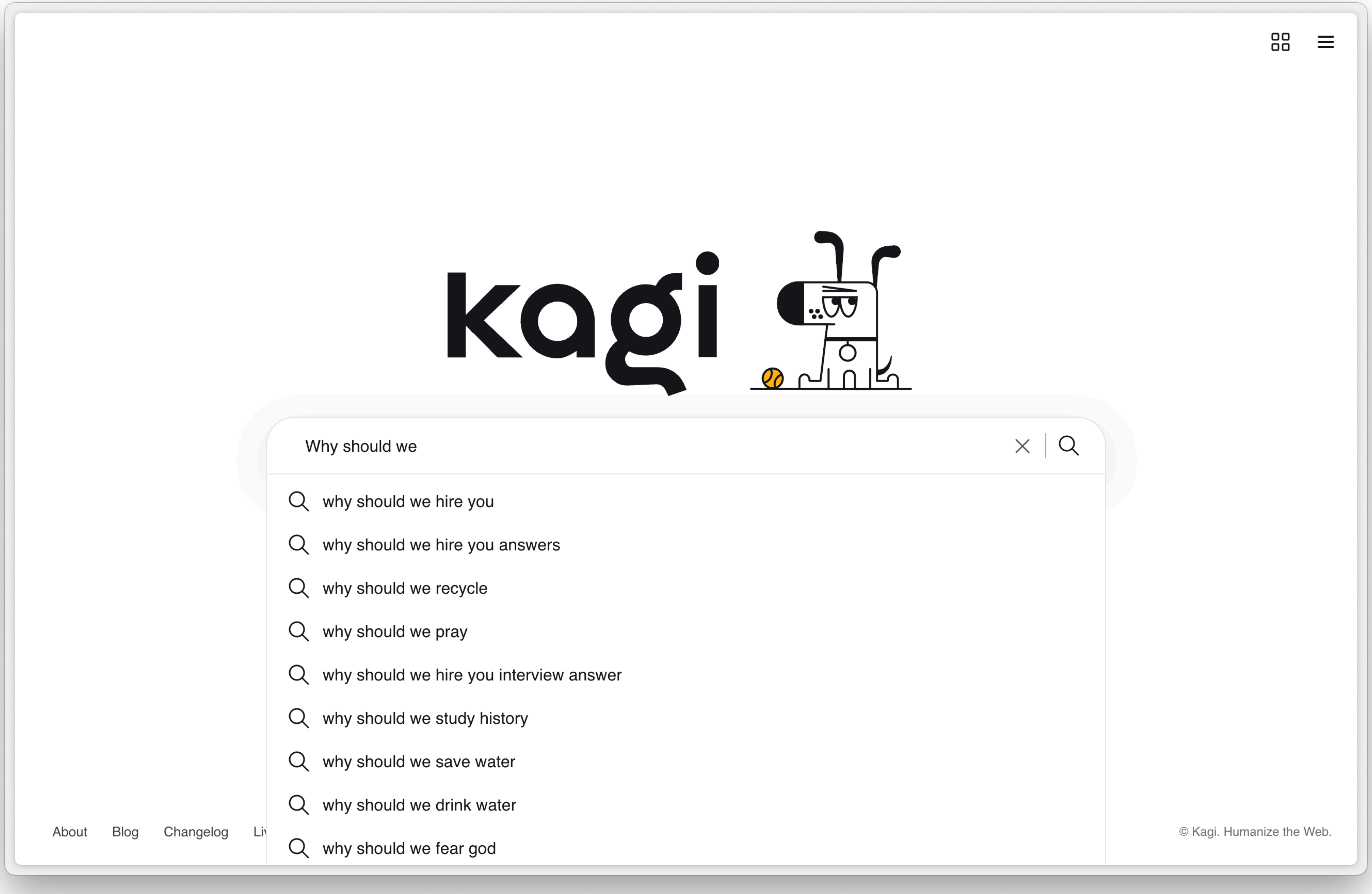
but language models also let us compute

$P(w_n \mid w_1, w_2, w_3, \ldots, w_{n-1})$,

the probability of a word given some previous words.

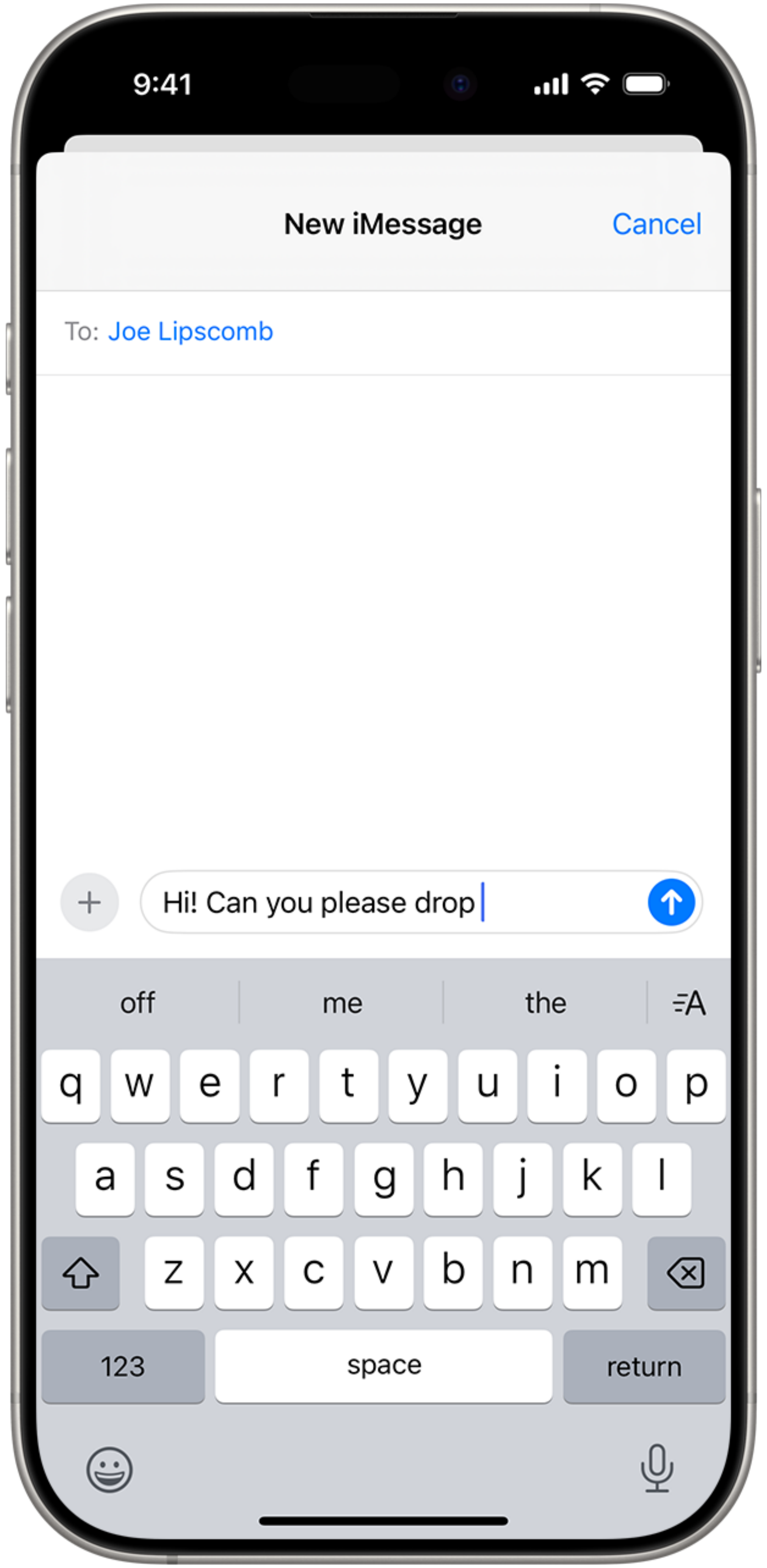
Why would that be useful?

For simplicity, we'll talk about "words", but these are really tokens.



Why should we

- why should we hire you
- why should we hire you answers
- why should we recycle
- why should we pray
- why should we hire you interview answer
- why should we study history
- why should we save water
- why should we drink water
- why should we fear god



To: Joe Lipscomb

Hi! Can you please drop

Word prediction is also the basis for how large language models (LLMs) work!

We'll return to these systems in a few weeks; we're building up the foundations they're built on.

$$
\begin{aligned}
& P(\textit{The water of Walden Pond is so beautifully blue})? \\
= \; & P(\textit{The}, \textit{water}, \textit{of}, \textit{Walden}, \textit{Pond}, \textit{is}, \textit{so}, \textit{beautifully}, \textit{blue}) \\
= \; & \ldots
\end{aligned}
$$

$$P(\textit{The water of Walden Pond is so beautifully blue})?$$

$$= P(\textit{The}, \textit{water}, \textit{of}, \textit{Walden}, \textit{Pond}, \textit{is}, \textit{so}, \textit{beautifully}, \textit{blue})$$

$$= \ldots$$

Chain rule of probability!

The chain rule of probability

Recall the definition of conditional probabilities,

$$P(B|A) = P(A, B) \,/\, P(A)$$

which we can rewrite to get

$$P(A, B) = P(A)\, P(B|A).$$

The chain rule of probability

If we have more variables, we get more terms, e.g.,

$$P(A, B, C, D) = P(A)\ P(B \mid A)\ P(C \mid A, B)\ P(D \mid A, B, C)$$

In general, the chain rule says

$$\begin{aligned} P(x_1, x_2, x_3, \ldots, x_n) = \ & P(x_1) \cdot \\ & P(x_2 \mid x_1) \cdot \\ & P(x_3 \mid x_1, x_2) \cdots \\ & P(x_n \mid x_1, \ldots, x_{n-1}) \end{aligned}$$

$$
\begin{aligned}
& P(\textit{The water of Walden Pond is so beautifully blue})?\\
={}& P(\textit{The}, \textit{water}, \textit{of}, \textit{Walden}, \textit{Pond}, \textit{is}, \textit{so}, \textit{beautifully}, \textit{blue})\\
={}& P(\textit{The})\, P(\textit{water} \mid \textit{The})\, P(\textit{of} \mid \textit{The water})\, P(\textit{Walden} \mid \textit{The water of}) \cdots
\end{aligned}
$$

$$P(\textit{The water of Walden Pond is so beautifully blue})?$$

$$= P(\textit{The}, \textit{water}, \textit{of}, \textit{Walden}, \textit{Pond}, \textit{is}, \textit{so}, \textit{beautifully}, \textit{blue})$$

$$= P(\textit{The})\, P(\textit{water} \mid \textit{The})\, P(\textit{of} \mid \textit{The water})\, P(\textit{Walden} \mid \textit{The water of}) \cdots$$

= 🤔 How do we compute this?

To estimate conditional probabilities, we use a text corpus that we've tokenized, and we do some counting!

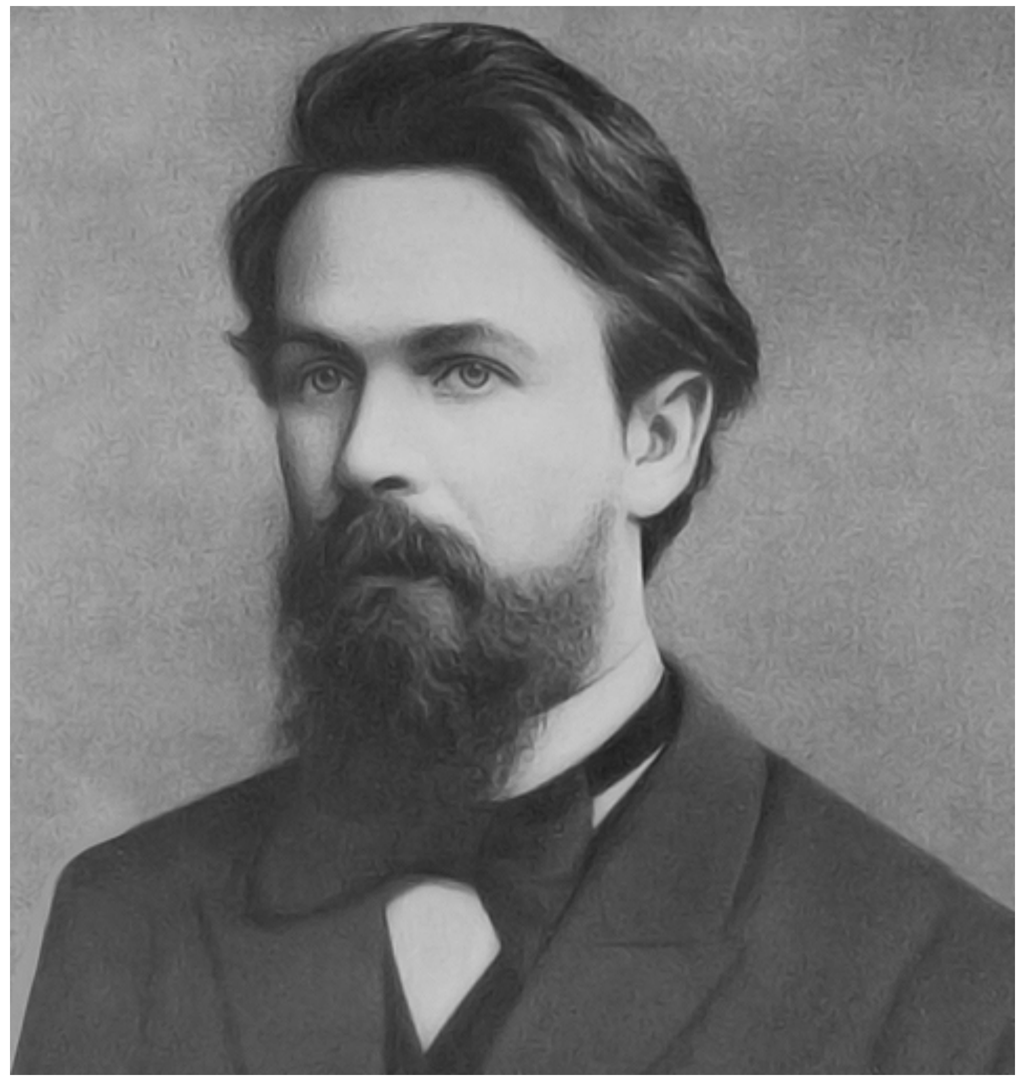
$$P(\textit{blue} \mid \textit{The water of Walden Pond is so beautifully}) = \frac{C(\textit{The water of Walden Pond is so beautifully blue})}{C(\textit{The water of Walden Pond is so beautifully})}$$

$C(x)$ is the count of how many times x occurs in the corpus

In practice, we make a simplifying ***Markov assumption*** that we can predict the probability of a future event without looking too far into the past, e.g.,

$$P(\textit{blue} \mid \textit{The}, \textit{water}, \textit{of}, \textit{Walden}, \textit{Pond}, \textit{is}, \textit{so}, \textit{beautifully})$$

$$\approx P(\textit{blue} \mid \textit{so}, \textit{beautifully})$$



Andrei Markov

We can estimate the true probabilities using ***n-grams*** – sequences of text that are always *n* tokens long.

Colorless green ideas sleep furiously.

Colorless green ideas sleep furiously.

Unigrams:

Colorless

green

ideas

sleep

furiously

.

Colorless green ideas sleep furiously.

Bigrams:

<s> *Colorless*

Colorless green

green ideas

ideas sleep

sleep furiously

furiously .

. </s>

Colorless green ideas sleep furiously.

Bigrams:

<s> *Colorless*

Colorless green

green ideas

ideas sleep

sleep furiously

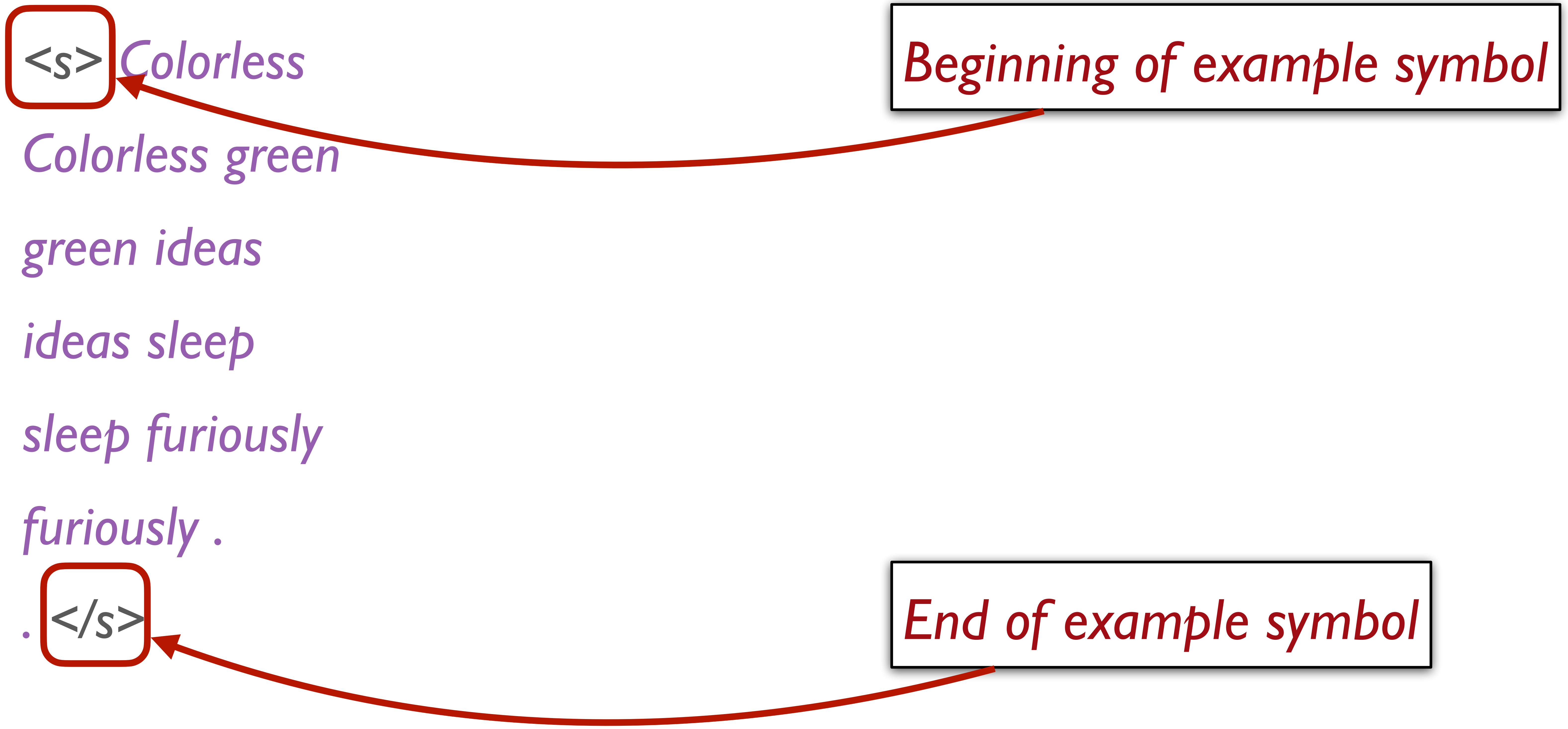
furiously .

. </s>

Beginning of example symbol

Colorless green ideas sleep furiously.

Bigrams:



<s> *Colorless*

Colorless green

green ideas

ideas sleep

sleep furiously

furiously .

. </s>

Beginning of example symbol

End of example symbol

Colorless green ideas sleep furiously.

Trigrams:

<s> <s> *Colorless*

<s> *Colorless green*

Colorless green ideas

green ideas sleep

ideas sleep furiously

sleep furiously .

furiously . </s>

. </s> </s>

Colorless green ideas sleep furiously.

4-grams:

<s> <s> <s> *Colorless*

<s> <s> *Colorless green*

<s> *Colorless green ideas*

Colorless green ideas sleep

green ideas sleep furiously

ideas sleep furiously .

sleep furiously . </s>

furiously . </s> </s>

. </s> </s> </s>

What’s the best value of *n*?

That is, how many previous words do we need?

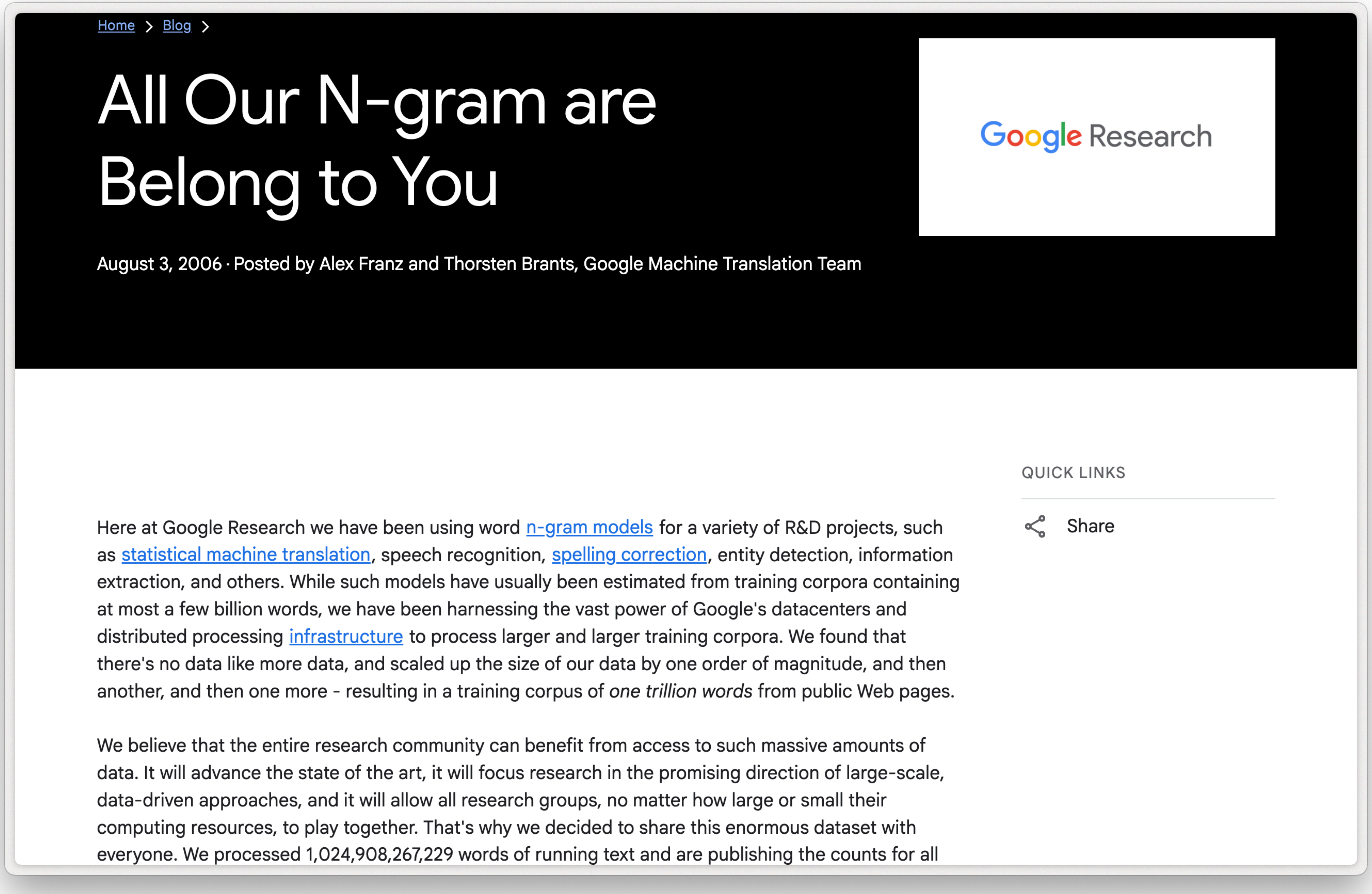
Given any choice of *n*, are *n*-grams a sufficient model of language?

Language has *long-distance dependencies*:

The computer / computers which I had just put into the machine room on the fifth floor is / are crashing.

But we can often get away with *n*-gram models.

Corpora and *n*-grams



Home > Blog >

All Our N-gram are Belong to You

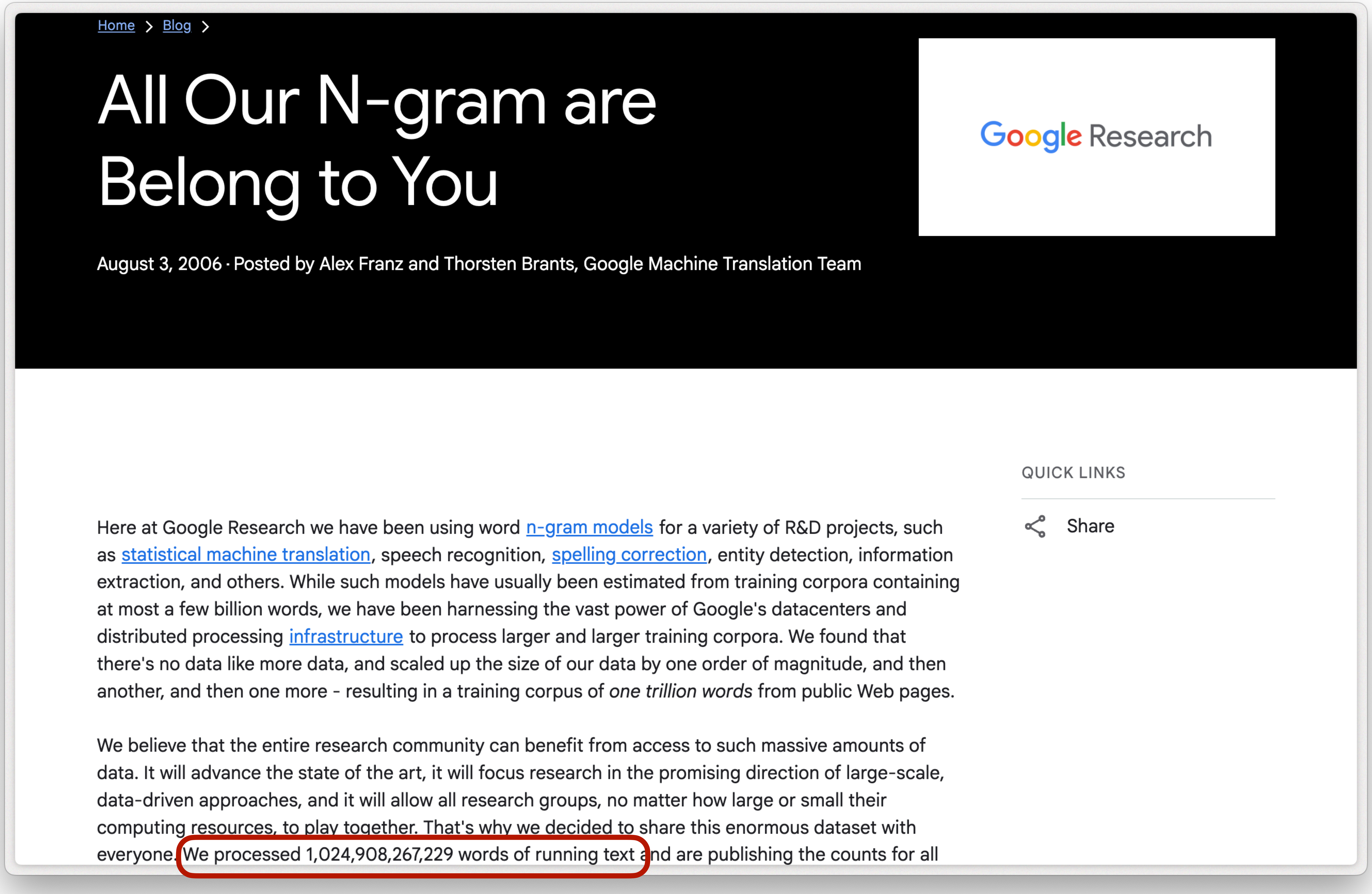
August 3, 2006 · Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

QUICK LINKS

Share

Here at Google Research we have been using word n-gram models for a variety of R&D projects, such as statistical machine translation, speech recognition, spelling correction, entity detection, information extraction, and others. While such models have usually been estimated from training corpora containing at most a few billion words, we have been harnessing the vast power of Google's datacenters and distributed processing infrastructure to process larger and larger training corpora. We found that there's no data like more data, and scaled up the size of our data by one order of magnitude, and then another, and then one more - resulting in a training corpus of *one trillion words* from public Web pages.

We believe that the entire research community can benefit from access to such massive amounts of data. It will advance the state of the art, it will focus research in the promising direction of large-scale, data-driven approaches, and it will allow all research groups, no matter how large or small their computing resources, to play together. That's why we decided to share this enormous dataset with everyone. We processed 1,024,908,267,229 words of running text and are publishing the counts for all



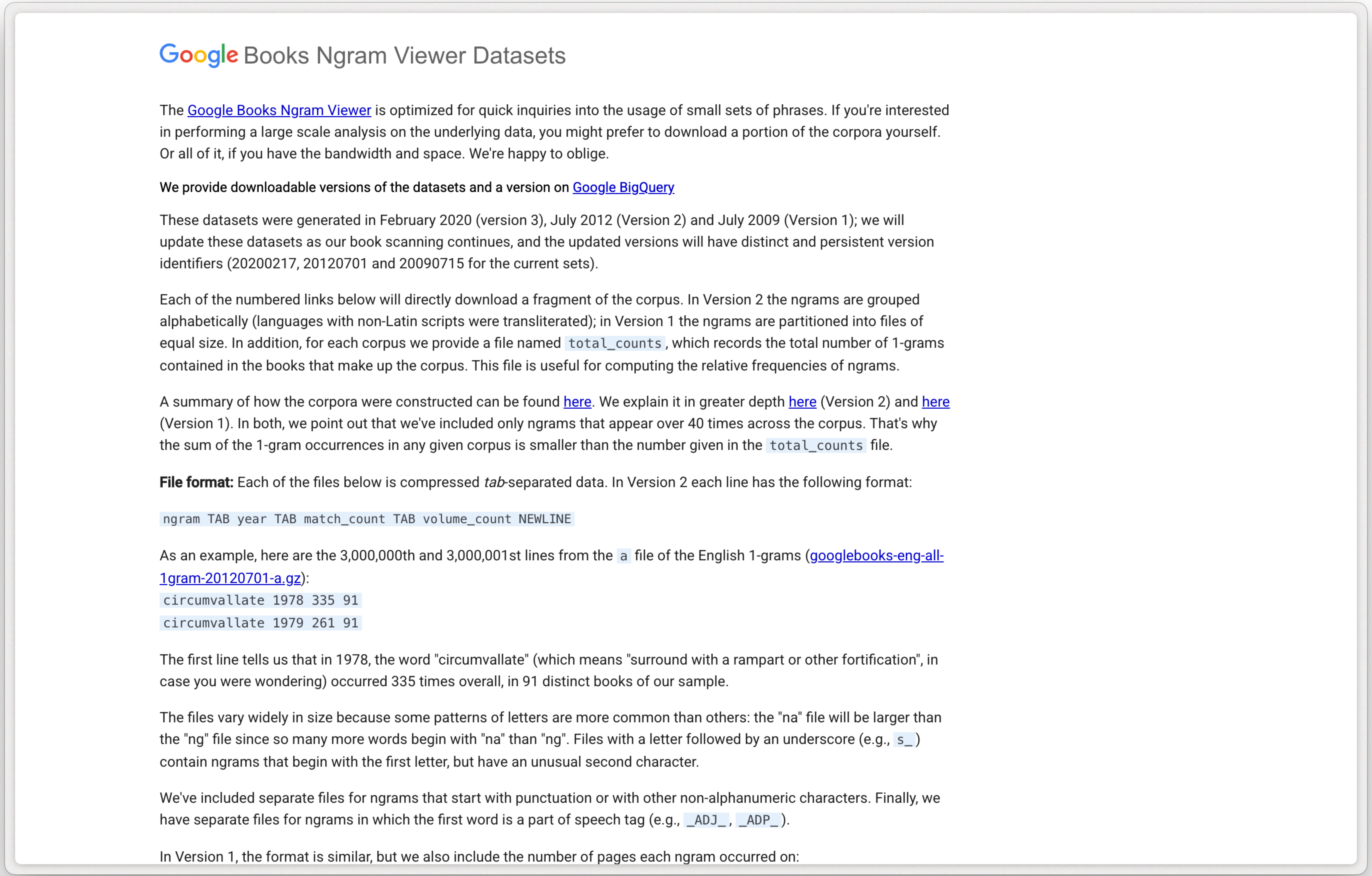
All Our N-gram are Belong to You

August 3, 2006 · Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

Here at Google Research we have been using word n-gram models for a variety of R&D projects, such as statistical machine translation, speech recognition, spelling correction, entity detection, information extraction, and others. While such models have usually been estimated from training corpora containing at most a few billion words, we have been harnessing the vast power of Google's datacenters and distributed processing infrastructure to process larger and larger training corpora. We found that there's no data like more data, and scaled up the size of our data by one order of magnitude, and then another, and then one more - resulting in a training corpus of *one trillion words* from public Web pages.

We believe that the entire research community can benefit from access to such massive amounts of data. It will advance the state of the art, it will focus research in the promising direction of large-scale, data-driven approaches, and it will allow all research groups, no matter how large or small their computing resources, to play together. That's why we decided to share this enormous dataset with everyone. We processed 1,024,908,267,229 words of running text and are publishing the counts for all

```
serve as the incoming 92
serve as the incubator 99
serve as the independent 794
serve as the index 223
serve as the indication 72
serve as the indicator 120
serve as the indicators 45
serve as the indispensable 111
serve as the indispensible 40
serve as the individual 234
```



Google Books Ngram Viewer Datasets

The Google Books Ngram Viewer is optimized for quick inquiries into the usage of small sets of phrases. If you're interested in performing a large scale analysis on the underlying data, you might prefer to download a portion of the corpora yourself. Or all of it, if you have the bandwidth and space. We're happy to oblige.

We provide downloadable versions of the datasets and a version on Google BigQuery

These datasets were generated in February 2020 (version 3), July 2012 (Version 2) and July 2009 (Version 1); we will update these datasets as our book scanning continues, and the updated versions will have distinct and persistent version identifiers (20200217, 20120701 and 20090715 for the current sets).

Each of the numbered links below will directly download a fragment of the corpus. In Version 2 the ngrams are grouped alphabetically (languages with non-Latin scripts were transliterated); in Version 1 the ngrams are partitioned into files of equal size. In addition, for each corpus we provide a file named `total_counts`, which records the total number of 1-grams contained in the books that make up the corpus. This file is useful for computing the relative frequencies of ngrams.

A summary of how the corpora were constructed can be found here. We explain it in greater depth here (Version 2) and here (Version 1). In both, we point out that we've included only ngrams that appear over 40 times across the corpus. That's why the sum of the 1-gram occurrences in any given corpus is smaller than the number given in the `total_counts` file.

File format: Each of the files below is compressed *tab*-separated data. In Version 2 each line has the following format:

```
ngram TAB year TAB match_count TAB volume_count NEWLINE
```

As an example, here are the 3,000,000th and 3,000,001st lines from the `a` file of the English 1-grams (googlebooks-eng-all-1gram-20120701-a.gz):

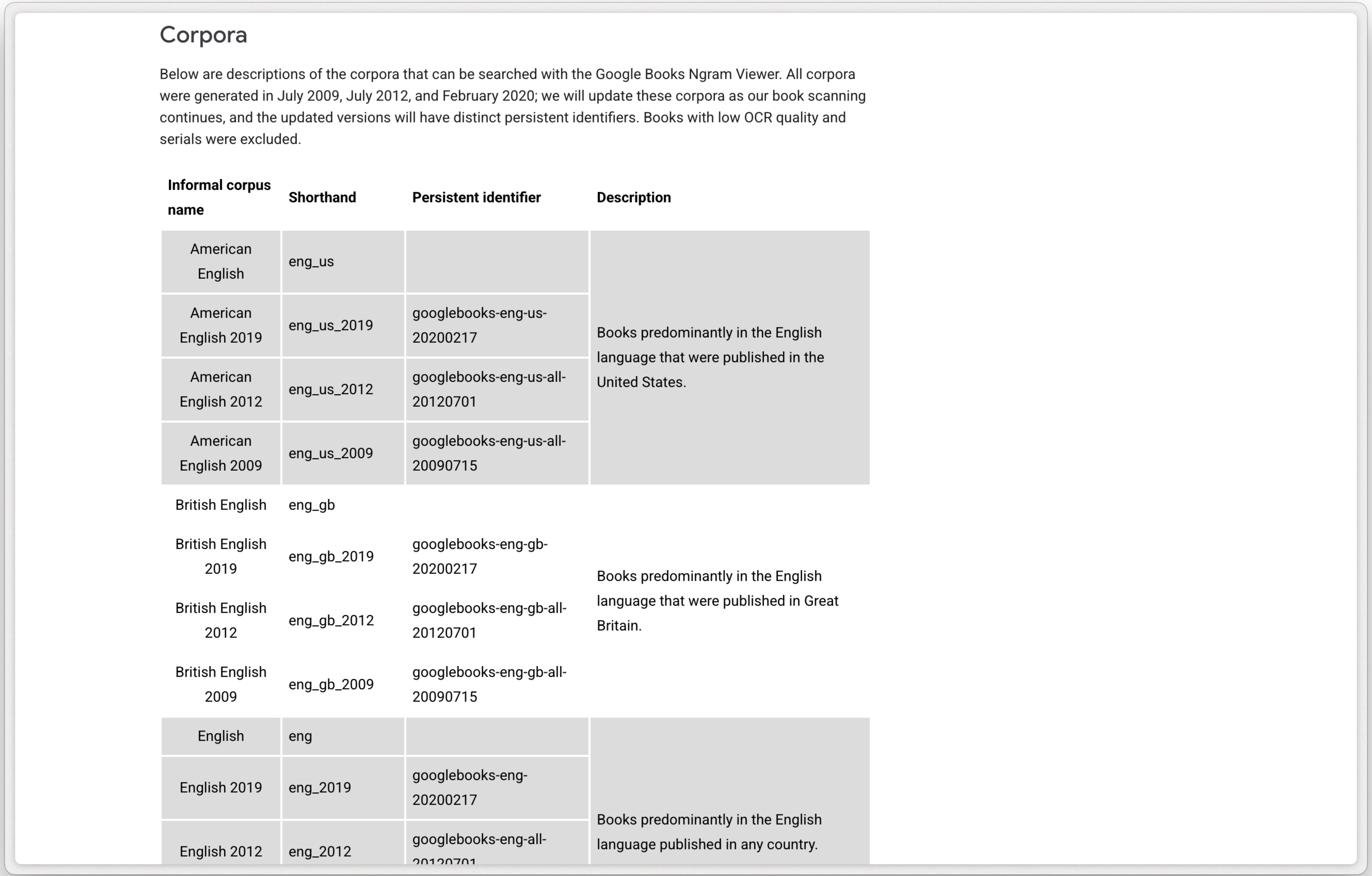
```
circumvallate 1978 335 91
circumvallate 1979 261 91
```

The first line tells us that in 1978, the word "circumvallate" (which means "surround with a rampart or other fortification", in case you were wondering) occurred 335 times overall, in 91 distinct books of our sample.

The files vary widely in size because some patterns of letters are more common than others: the "na" file will be larger than the "ng" file since so many more words begin with "na" than "ng". Files with a letter followed by an underscore (e.g., `s_`) contain ngrams that begin with the first letter, but have an unusual second character.

We've included separate files for ngrams that start with punctuation or with other non-alphanumeric characters. Finally, we have separate files for ngrams in which the first word is a part of speech tag (e.g., `_ADJ_`, `_ADP_`).

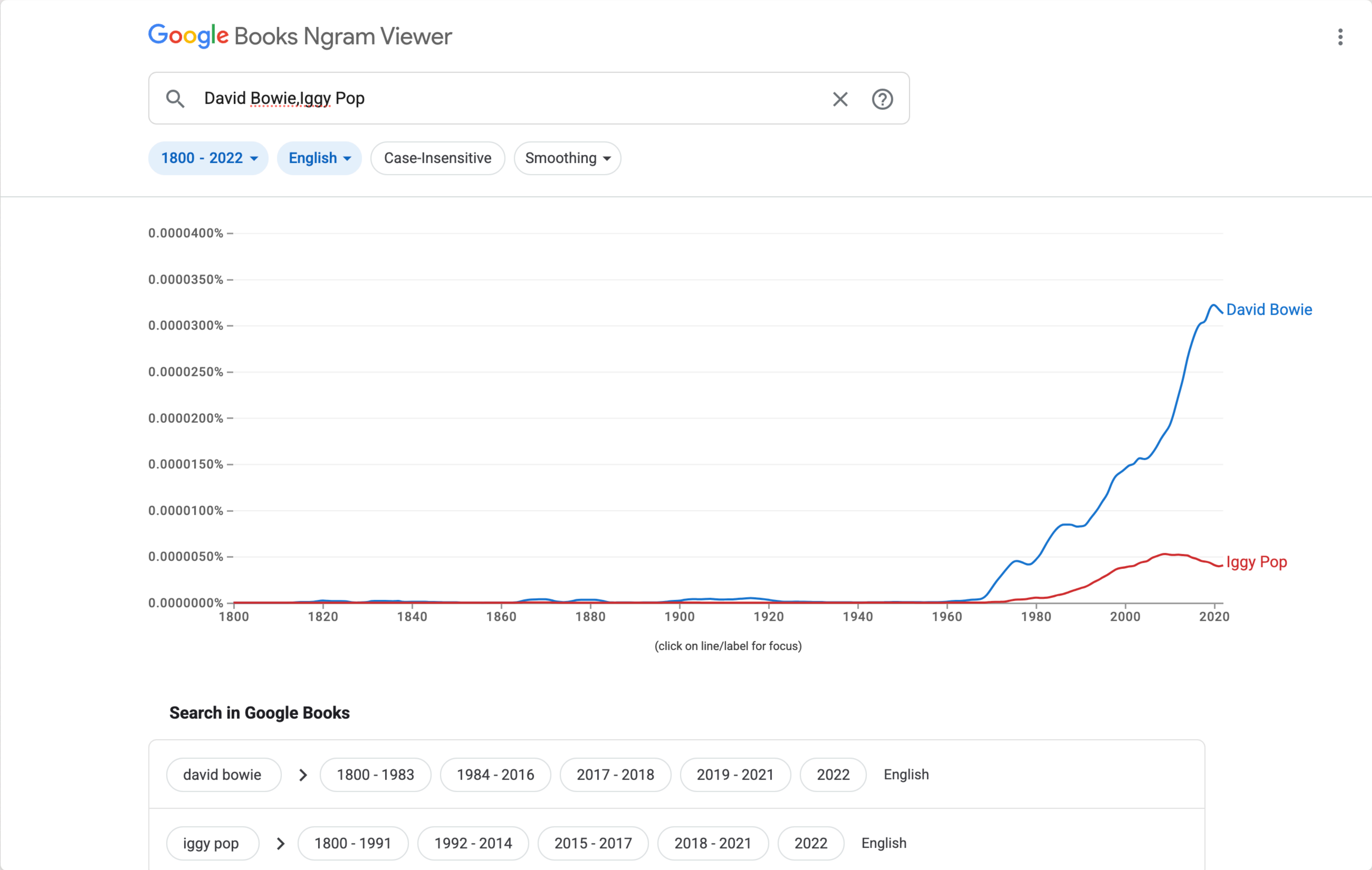
In Version 1, the format is similar, but we also include the number of pages each ngram occurred on:



Corpora

Below are descriptions of the corpora that can be searched with the Google Books Ngram Viewer. All corpora were generated in July 2009, July 2012, and February 2020; we will update these corpora as our book scanning continues, and the updated versions will have distinct persistent identifiers. Books with low OCR quality and serials were excluded.

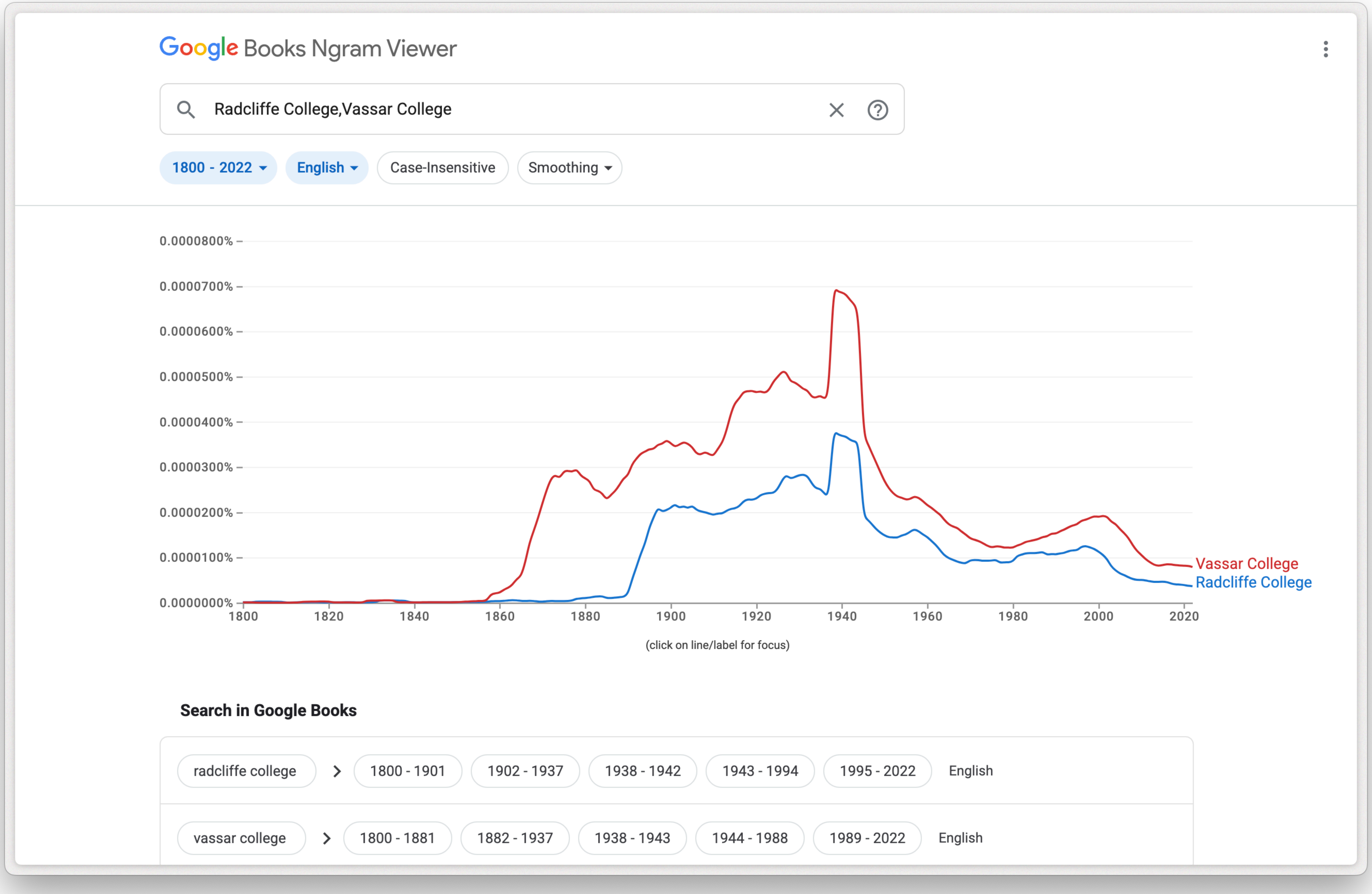
Informal corpus name	Shorthand	Persistent identifier	Description
American English	eng_us		Books predominantly in the English language that were published in the United States.
American English 2019	eng_us_2019	googlebooks-eng-us-20200217	
American English 2012	eng_us_2012	googlebooks-eng-us-all-20120701	
American English 2009	eng_us_2009	googlebooks-eng-us-all-20090715	
British English	eng_gb		Books predominantly in the English language that were published in Great Britain.
British English 2019	eng_gb_2019	googlebooks-eng-gb-20200217	
British English 2012	eng_gb_2012	googlebooks-eng-gb-all-20120701	
British English 2009	eng_gb_2009	googlebooks-eng-gb-all-20090715	
English	eng		Books predominantly in the English language published in any country.
English 2019	eng_2019	googlebooks-eng-20200217	
English 2012	eng_2012	googlebooks-eng-all-20120701	



Search in Google Books

david bowie > 1800 - 1983 | 1984 - 2016 | 2017 - 2018 | 2019 - 2021 | 2022 | English

iggy pop > 1800 - 1991 | 1992 - 2014 | 2015 - 2017 | 2018 - 2021 | 2022 | English



Google Books Ngram Viewer

Radcliffe College,Vassar College

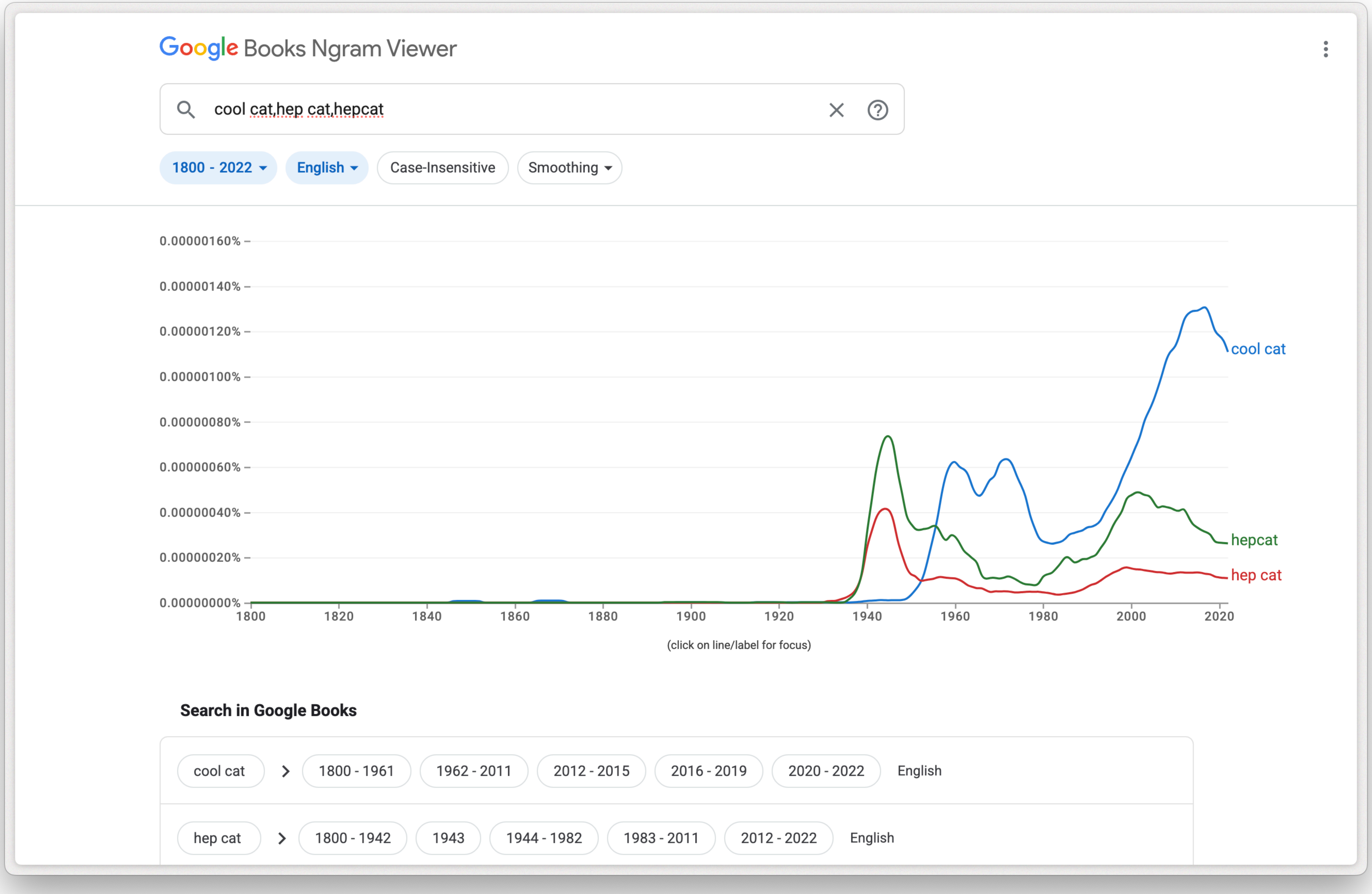
1800 - 2022 · English · Case-Insensitive · Smoothing

(click on line/label for focus)

Search in Google Books

radcliffe college > 1800 - 1901 · 1902 - 1937 · 1938 - 1942 · 1943 - 1994 · 1995 - 2022 · English

vassar college > 1800 - 1881 · 1882 - 1937 · 1938 - 1943 · 1944 - 1988 · 1989 - 2022 · English



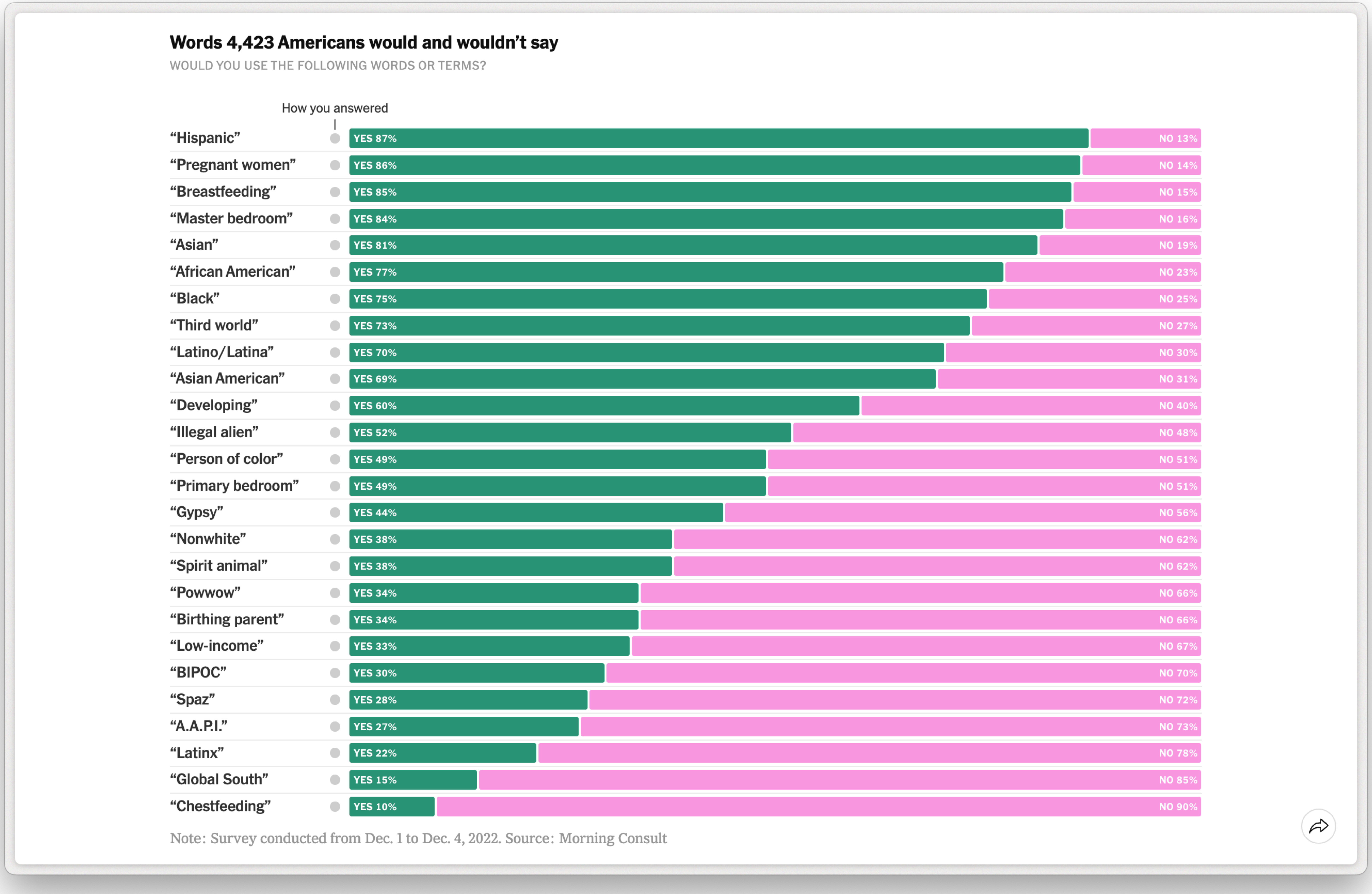
cool cat,hep cat,hepcat

1800 - 2022 English Case-Insensitive Smoothing

Search in Google Books

cool cat > 1800 - 1961 1962 - 2011 2012 - 2015 2016 - 2019 2020 - 2022 English

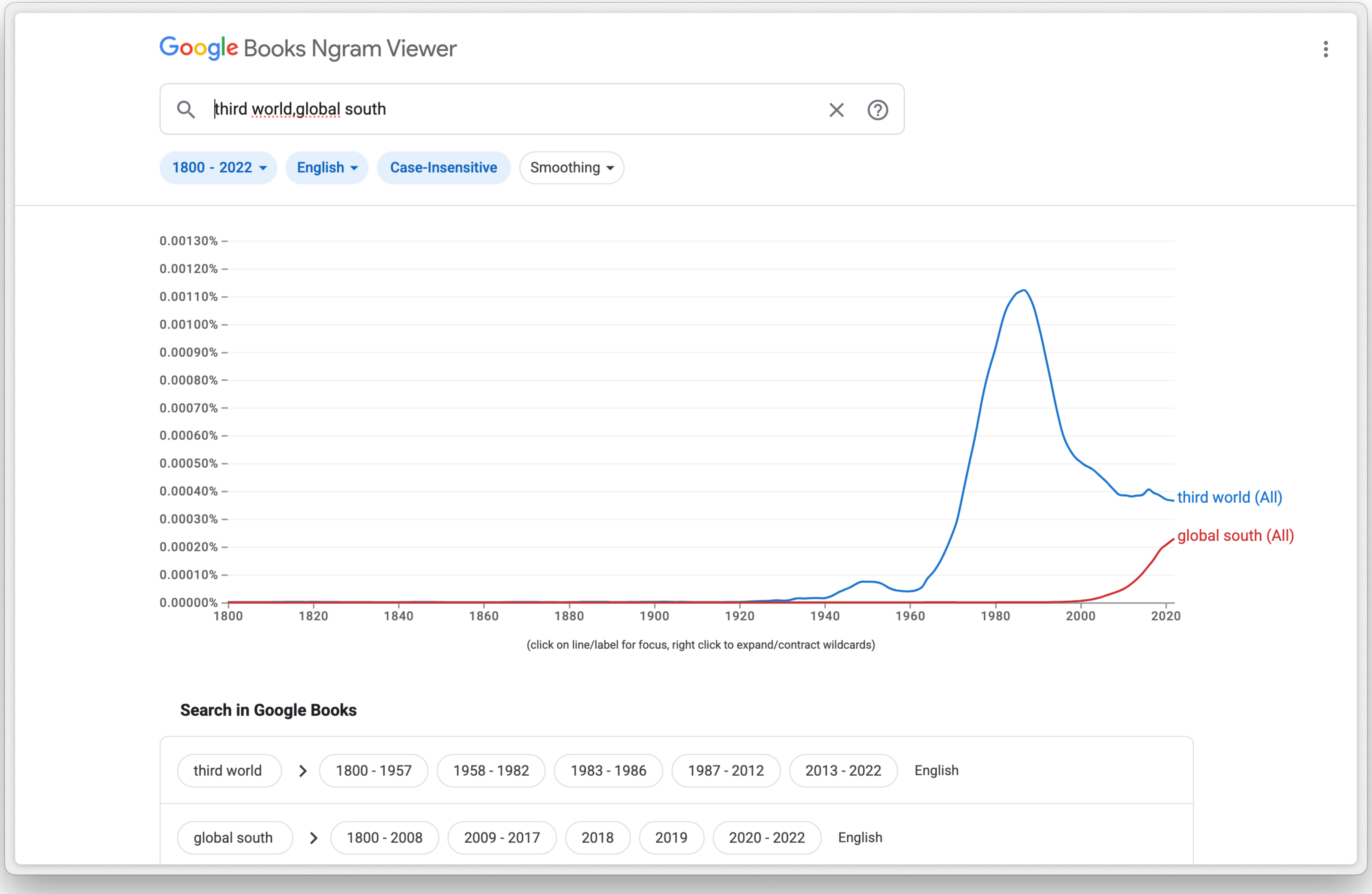
hep cat > 1800 - 1942 1943 1944 - 1982 1983 - 2011 2012 - 2022 English



Words 4,423 Americans would and wouldn’t say

WOULD YOU USE THE FOLLOWING WORDS OR TERMS?

Note: Survey conducted from Dec. 1 to Dec. 4, 2022. Source: Morning Consult



Google Books Ngram Viewer

third world,global south

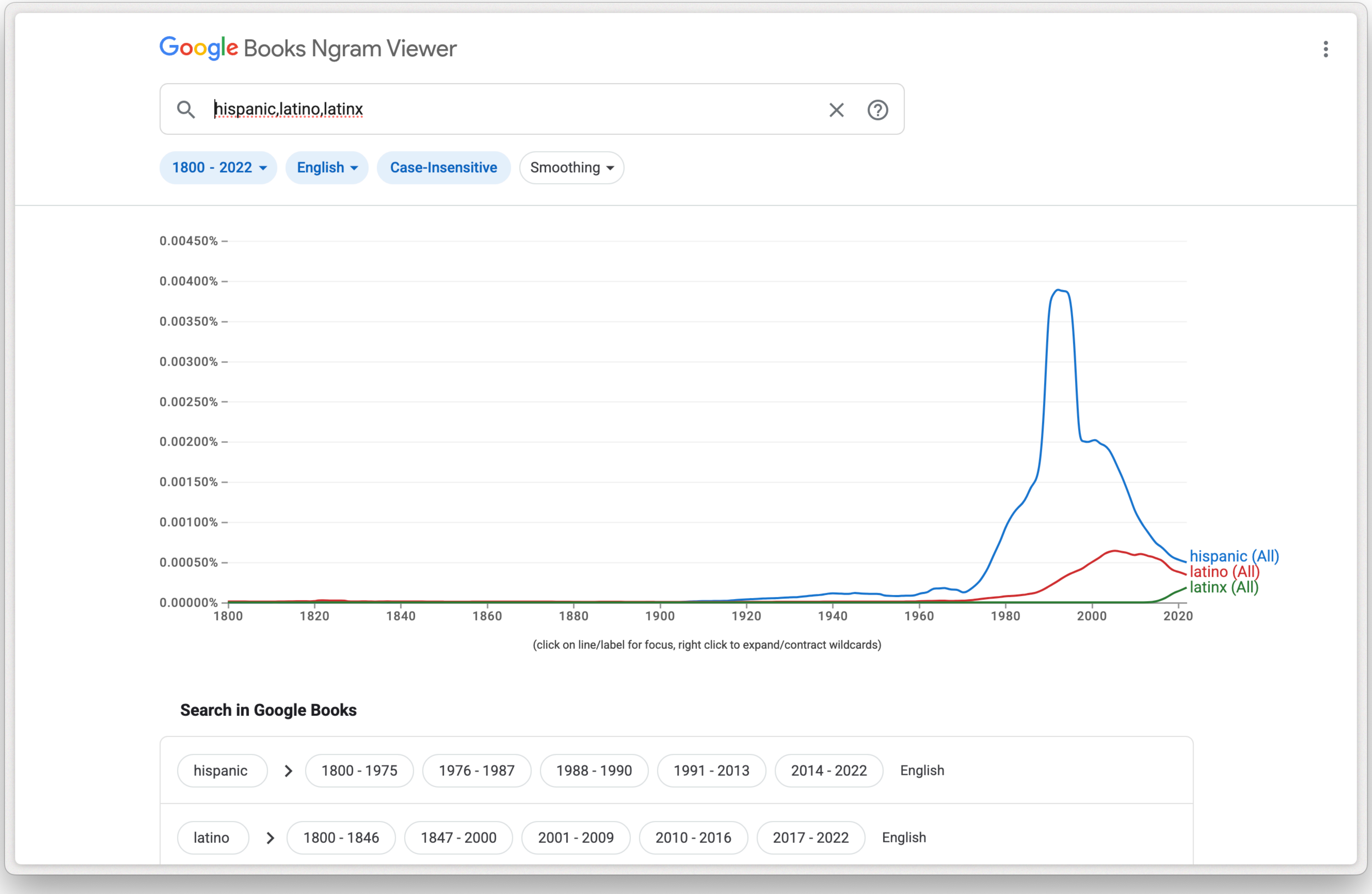
1800 - 2022 English Case-Insensitive Smoothing

(click on line/label for focus, right click to expand/contract wildcards)

Search in Google Books

third world > 1800 - 1957 1958 - 1982 1983 - 1986 1987 - 2012 2013 - 2022 English

global south > 1800 - 2008 2009 - 2017 2018 2019 2020 - 2022 English



Google Books Ngram Viewer

hispanic,latino,latinx

1800 - 2022 ▾ English ▾ Case-Insensitive Smoothing ▾

(click on line/label for focus, right click to expand/contract wildcards)

Search in Google Books

hispanic › 1800 - 1975 1976 - 1987 1988 - 1990 1991 - 2013 2014 - 2022 English

latino › 1800 - 1846 1847 - 2000 2001 - 2009 2010 - 2016 2017 - 2022 English

Estimating *n*-gram probabilities

We estimate the probabilities of *n*-grams using the ***maximum likelihood estimate*** (MLE) – the estimate that maximizes the likelihood of the training data given the model.

For unigram probabilities,

that's the fraction of times the word occurs in the corpus:

$$P(w_i) = \frac{C(w_i)}{|V|}$$

For bigram probabilities,

that's the number of times that word follows the other word divided by the number of times the other word occurs in the corpus:

$$P(w_i \mid w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$$

For example, given the corpus

<s> *I am Sam* </s>

<s> *Sam I am* </s>

<s> *I do not like green eggs and ham* </s>

we can compute $P(w_i \mid w_{i-1}) = \dfrac{C(w_{i-1}, w_i)}{C(w_{i-1})}$ and get these probabilities:

P(*I* | <s>) = 2/3 = 0.67

P(</s> | *Sam*) = 1/2 = 0.50

P(*Sam* | <s>) = 1/3 = 0.33

P(*Sam* | *am*) = 1/2 = 0.50

P(*am* | *I*) = 2/3 = 0.67

P(*do* | *I*) = 1/3 = 0.33

Probability is assigned ***exactly*** based on the *n*-gram count in the training corpus

Anything not found in the training corpus gets probability 0.

Downside of MLE

Suppose you toss a coin 10 times and get 8 heads.

The MLE is that this coin comes down heads 8 times out of 10.

Would you agree?

Downside of MLE

Suppose you toss a coin 10 times and get 8 heads.

The MLE is that this coin comes down heads 8 times out of 10.

Would you agree?

This is the *prior belief* that influences beliefs even in the face of contradicting evidence

Bayesian statistics measure degrees of belief:

Start with prior beliefs and update them in the face of evidence using *Bayes Theorem* – more on this next week!

Berkeley Restaurant Project: Sentences

can you tell me about any good cantonese restaurants close by

mid priced thai food is what i'm looking for

tell me about chez panisse

can you give me a listing of the kinds of food that are available

i'm looking for a good place to eat breakfast

when is caffe venezia open during the day

Berkeley Restaurant Project: Bigram counts

From 9222 sentences

w_2

		i	*want*	*to*	*eat*	*chinese*	*food*	*lunch*	*spend*
	i	5	827	0	9	0	0	0	2
	want	2	0	608	1	6	6	5	1
	to	2	0	4	686	2	0	6	211
w_1	*eat*	0	0	2	0	16	2	42	0
	chinese	1	0	0	0	0	82	1	0
	food	15	0	15	0	1	4	0	0
	lunch	2	0	0	0	0	1	0	0
	spend	1	0	1	0	0	0	0	0

Normalize by unigram counts

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

w_2

		i	want	to	eat	chinese	food	lunch	spend
	i	5	827	0	9	0	0	0	2
	want	2	0	608	1	6	6	5	1
	to	2	0	4	686	2	0	6	211
w_1	eat	0	0	2	0	16	2	42	0
	chinese	1	0	0	0	0	82	1	0
	food	15	0	15	0	1	4	0	0
	lunch	2	0	0	0	0	1	0	0
	spend	1	0	1	0	0	0	0	0

Berkeley Restaurant Project: Bigram probabilities

		w_2							
		i	*want*	*to*	*eat*	*chinese*	*food*	*lunch*	*spend*
	i	0.002	0.33	0	0.0036	0	0	0	0.00079
	want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
	to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
w_1	*eat*	0	0	0.0027	0	0.021	0.0027	0.056	0
	chinese	0.0063	0	0	0	0	0.52	0.0063	0
	food	0.014	0	0.014	0	0.00092	0.0037	0	0
	lunch	0.0059	0	0	0	0	0.0029	0	0
	spend	0.0036	0	0.0036	0	0	0	0	0

We use the bigram model to compute sentence probabilities:

$$
\begin{aligned}
& P(\text{<s>}\ \textit{I want english food}\ \text{</s>}) \\
= \ & P(\textit{I} \mid \text{<s>}) \cdot \\
& P(\textit{want} \mid \textit{I}) \cdot \\
& P(\textit{english} \mid \textit{want}) \cdot \\
& P(\textit{food} \mid \textit{english}) \cdot \\
& P(\text{</s>} \mid \textit{food}) \\
= \ & 0.00031
\end{aligned}
$$

As simple as they are, *n*-gram probabilities capture a range of interesting facts about language:

P(*english* | *want*) = 0.0011

P(*chinese* | *want*) = 0.0065

World knowledge; culture

As simple as they are, *n*-gram probabilities capture a range of interesting facts about language:

P(*english* | *want*) = 0.0011

P(*chinese* | *want*) = 0.0065

World knowledge; culture

P(*to* | *want*) = 0.66

P(*eat* | *to*) = 0.28

P(*food* | *to*) = 0

P(*want* | *spend*) = 0

Syntactic preferences

As simple as they are, *n*-gram probabilities capture a range of interesting facts about language:

P(*english* | *want*) = 0.0011

P(*chinese* | *want*) = 0.0065

World knowledge; culture

P(*to* | *want*) = 0.66

P(*eat* | *to*) = 0.28

P(*food* | *to*) = 0

P(*want* | *spend*) = 0

Syntactic preferences

P(*i* | <*s*>) = 0.25

Discourse

A practical concern

When programming, we handle probabilities in log space:

$\log(p_1 \cdot p_2 \cdot p_3 \cdot p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$

It's nice that adding is faster than multiplying, but the main reason is that it avoids underflow.

This will be true for the rest of the class!

Numeric underflow:

```
a = 1e-10
b = 1e-90
c = 1e-30
d = 5e-130
e = 1e-40
f = 1e-100
a * b * c * d * e *f
→ 0.0
```

But, using log-space math:

```
from math import log
log(a) + log(b) + log(c) + log(d) + log(e) +
log(f)
→-919.4245992851843
```

Next time

Smoothing and generalization

How do we know if a language model is good?

Text generation using language models

Bring a computer!

Acknowledgments

This class incorporates material from:

Carolyn Anderson, Wellesley College

Nancy Ide, Vassar College

Katie Keith, Williams College

Jurafsky & Martin, *Speech and Language Processing*, 3rd ed. draft