

What is a corpus?

Word type vs word token

The cat sat on the mat.

What is tokenization?

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

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

Spelling correction

The office is about fifteen minuets from my house.

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

Machine translation

Translating *The doctor recommended a cat scan,*

$P(\textit{La doctora recomendó una tomografía}) >$

$P(\textit{La doctora recomendó una exploración del gato})$

And more!

These are examples of computing

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

the probability of a sequence of words,

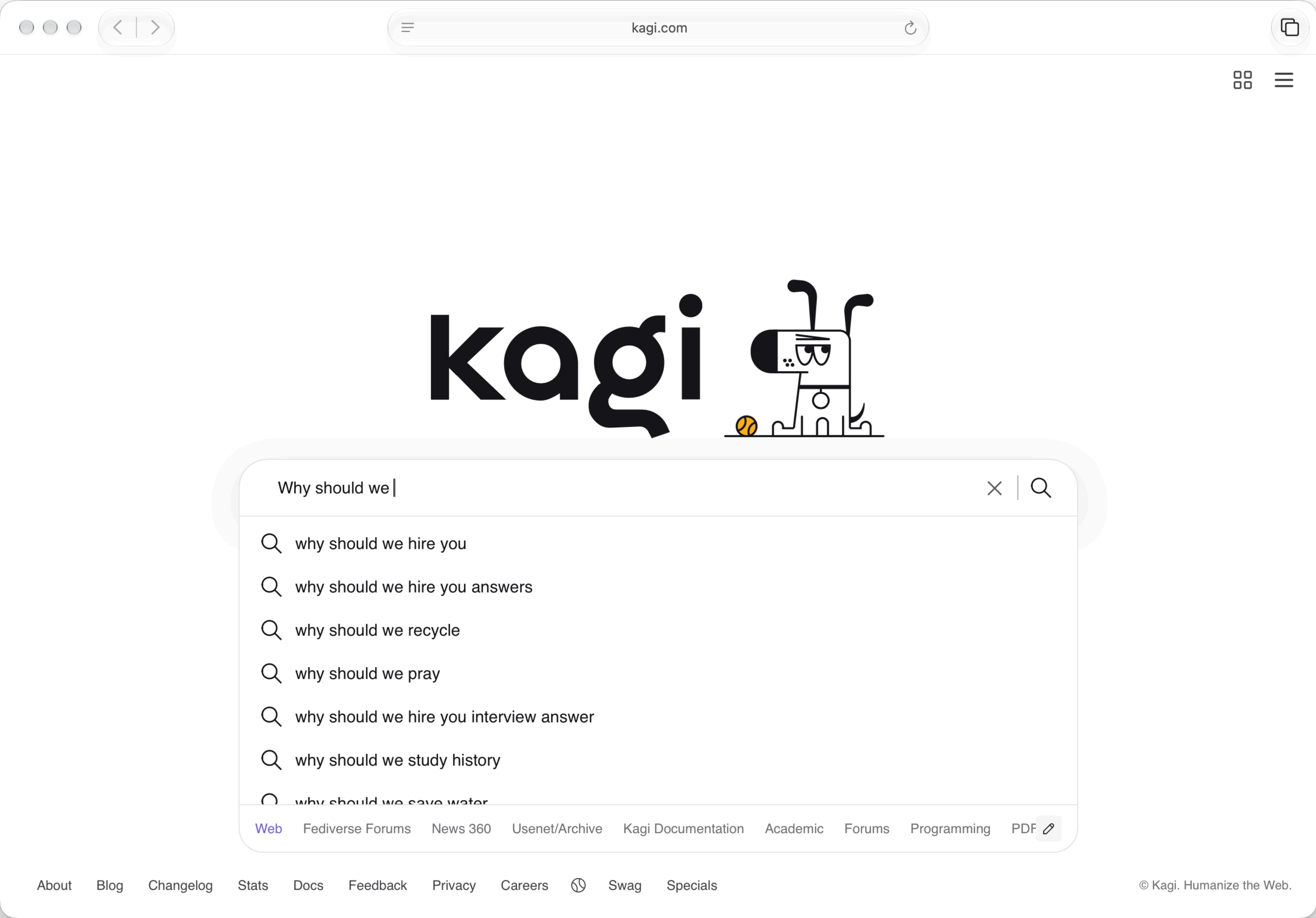
but language models also let us compute

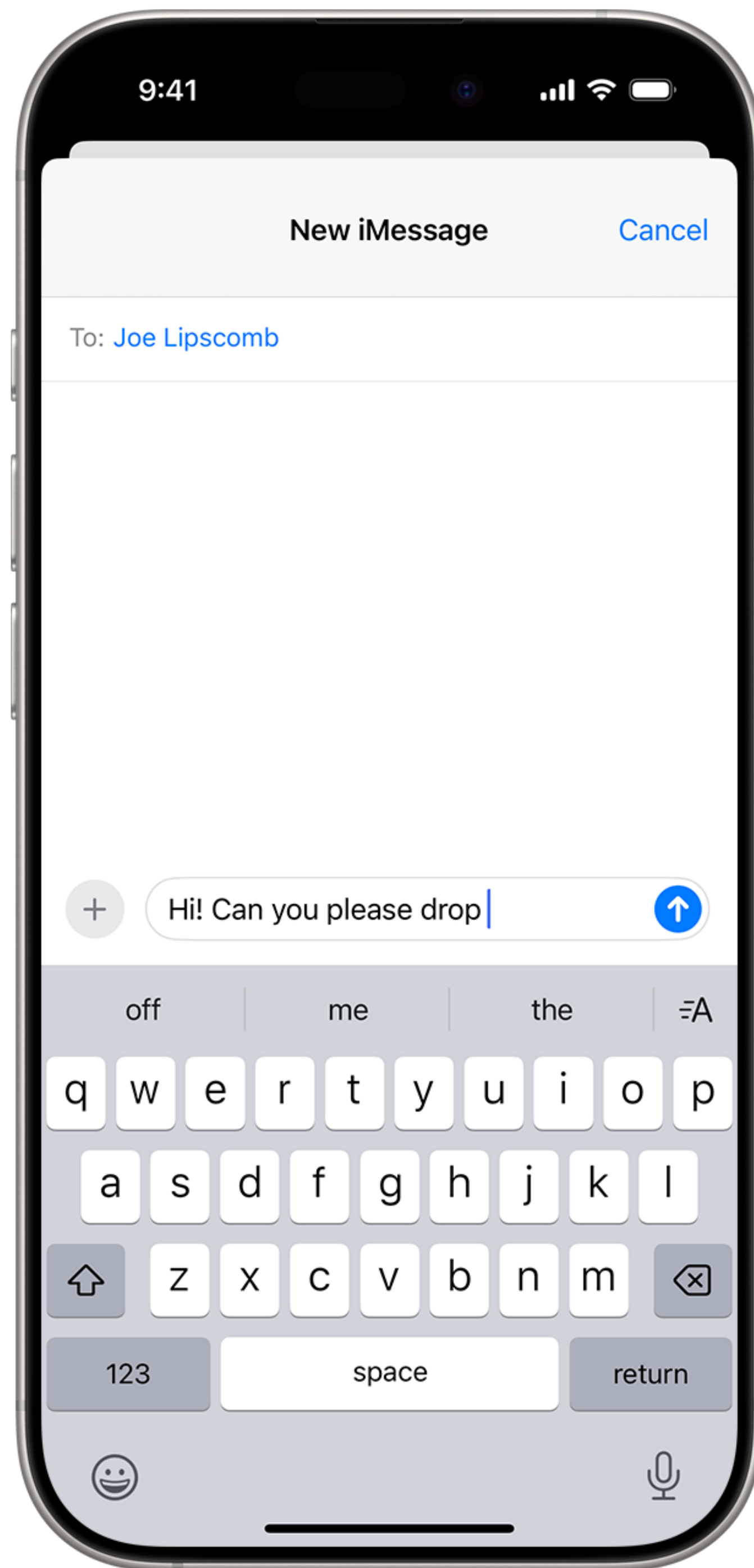
$$P(w_n \mid w_1, w_2, w_3, \dots, w_{n-1}),$$

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.





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.

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

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

$= \dots$

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

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

$= \dots$ *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

$$P(x_1, x_2, x_3, \dots, x_n) = P(x_1) \cdot \\ P(x_2 \mid x_1) \cdot \\ P(x_3 \mid x_1, x_2) \cdots \\ P(x_n \mid x_1, \dots, x_{n-1})$$

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

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

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

...

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

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

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

$=$ 🤔 How do we compute this?

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

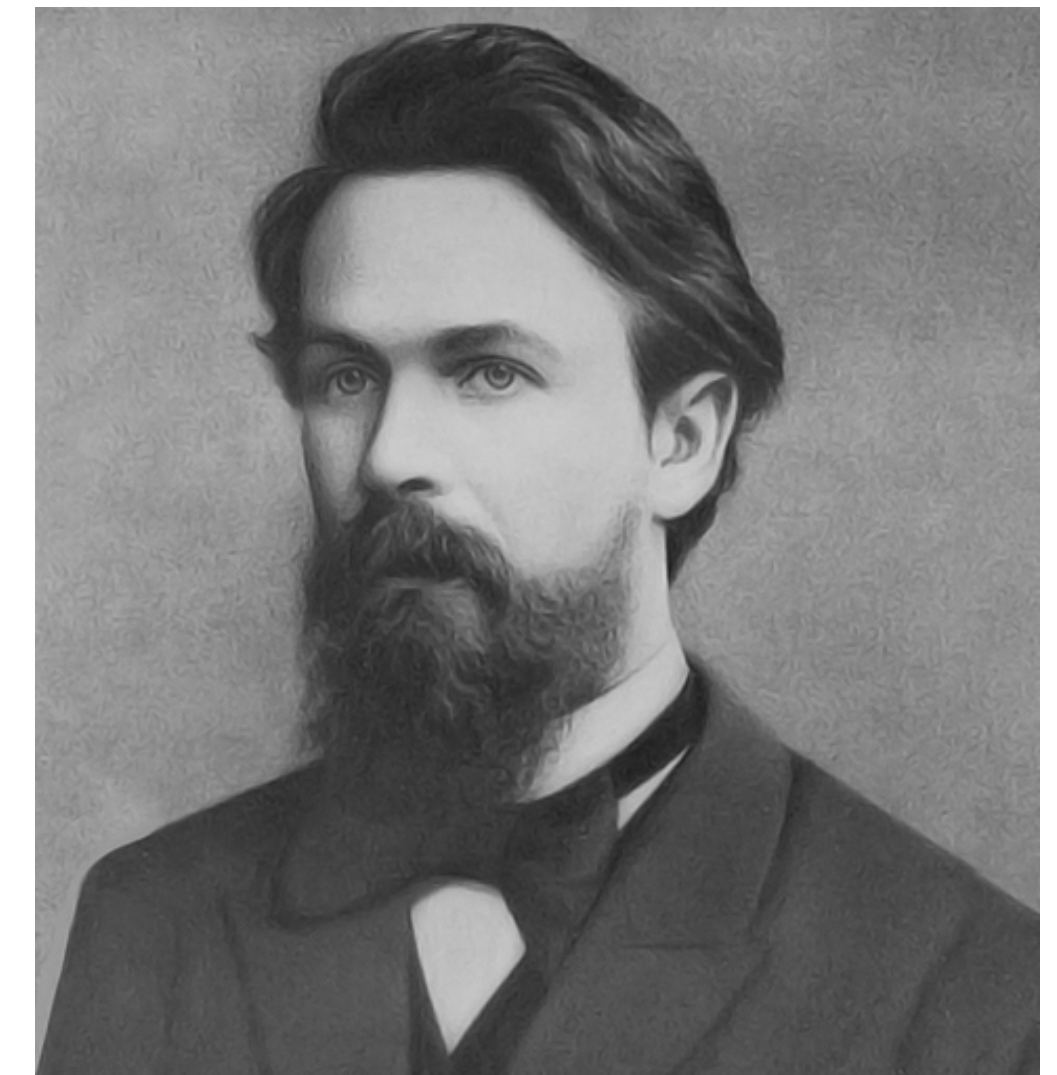
$$P(\textit{blue} \mid \textit{The water of Walden Pond is so beautifully}) \\ = \frac{C(\textit{The water of Walden Pond is so beautifully blue})}{C(\textit{The water of Walden Pond is so beautifully})}$$

$C(x)$ 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.,

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

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



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.

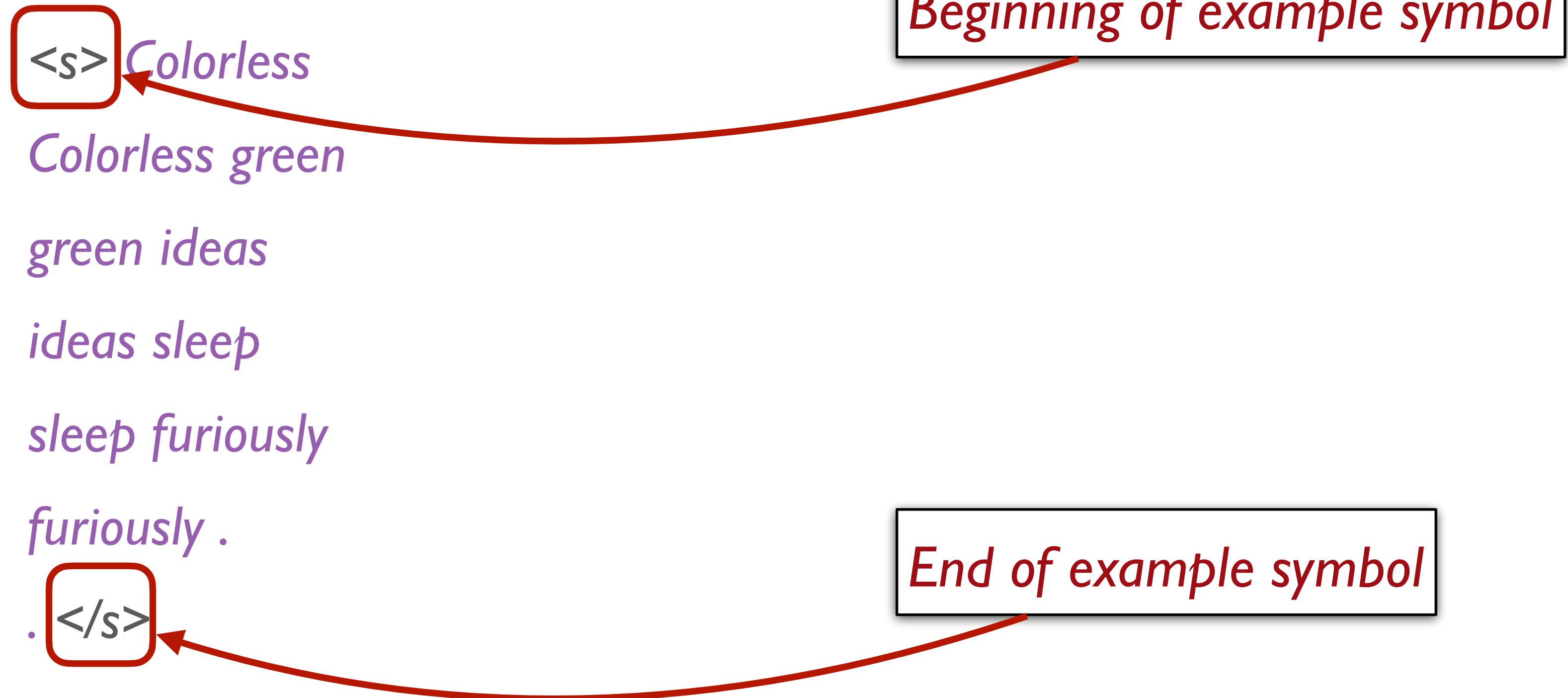
Bigrams:

<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>

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>

What's the best value of n ?

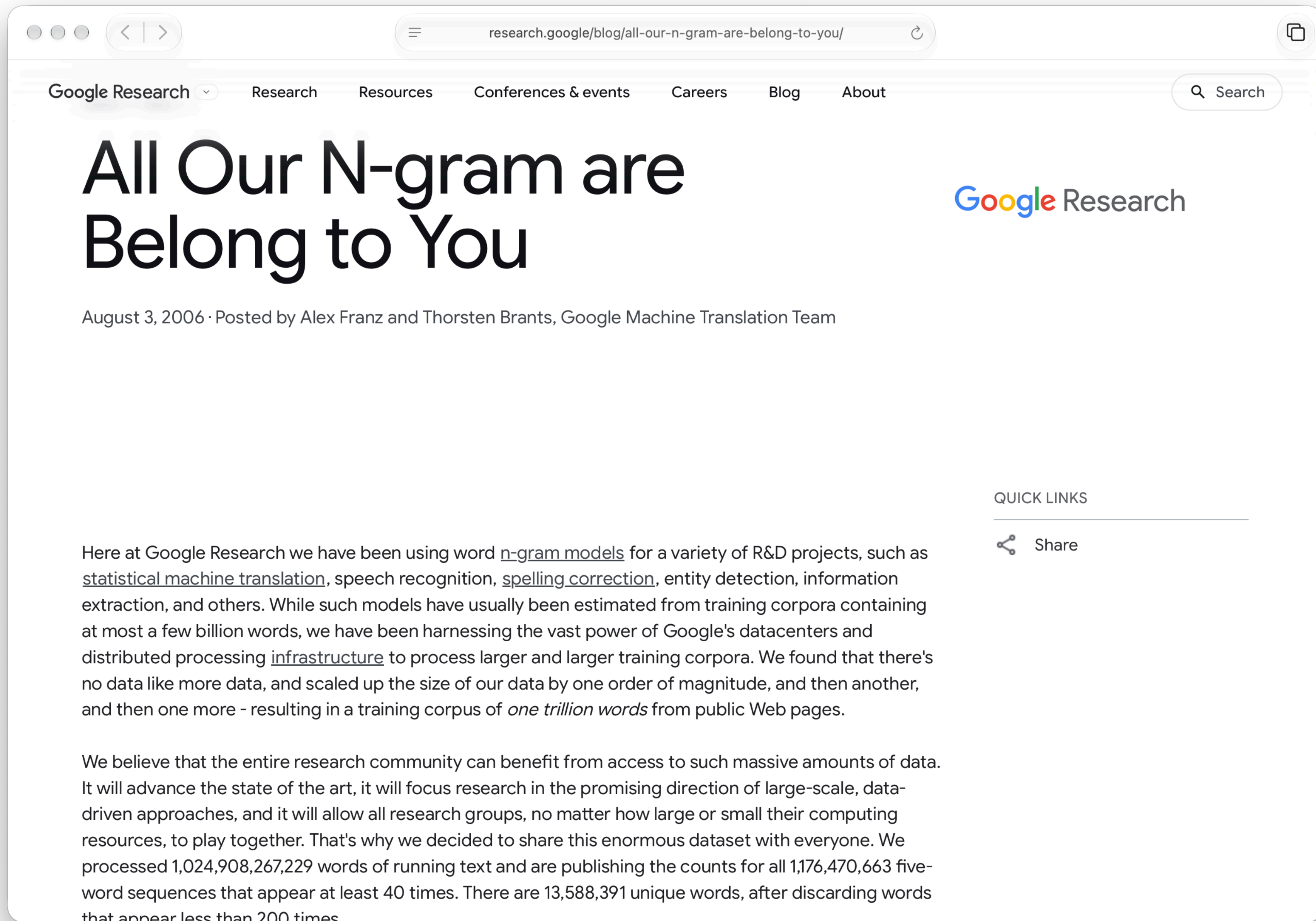
That is, how many previous words do we need?

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.



Google Research

QUICK LINKS

 Share

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 1,176,470,663 five-word sequences that appear at least 40 times. There are 13,588,391 unique words, after discarding words that appear less than 200 times.

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 indispensable 40
serve as the individual 234

<

|

>

storage.googleapis.com/books/ngrams/books/datasetsv3.html

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.

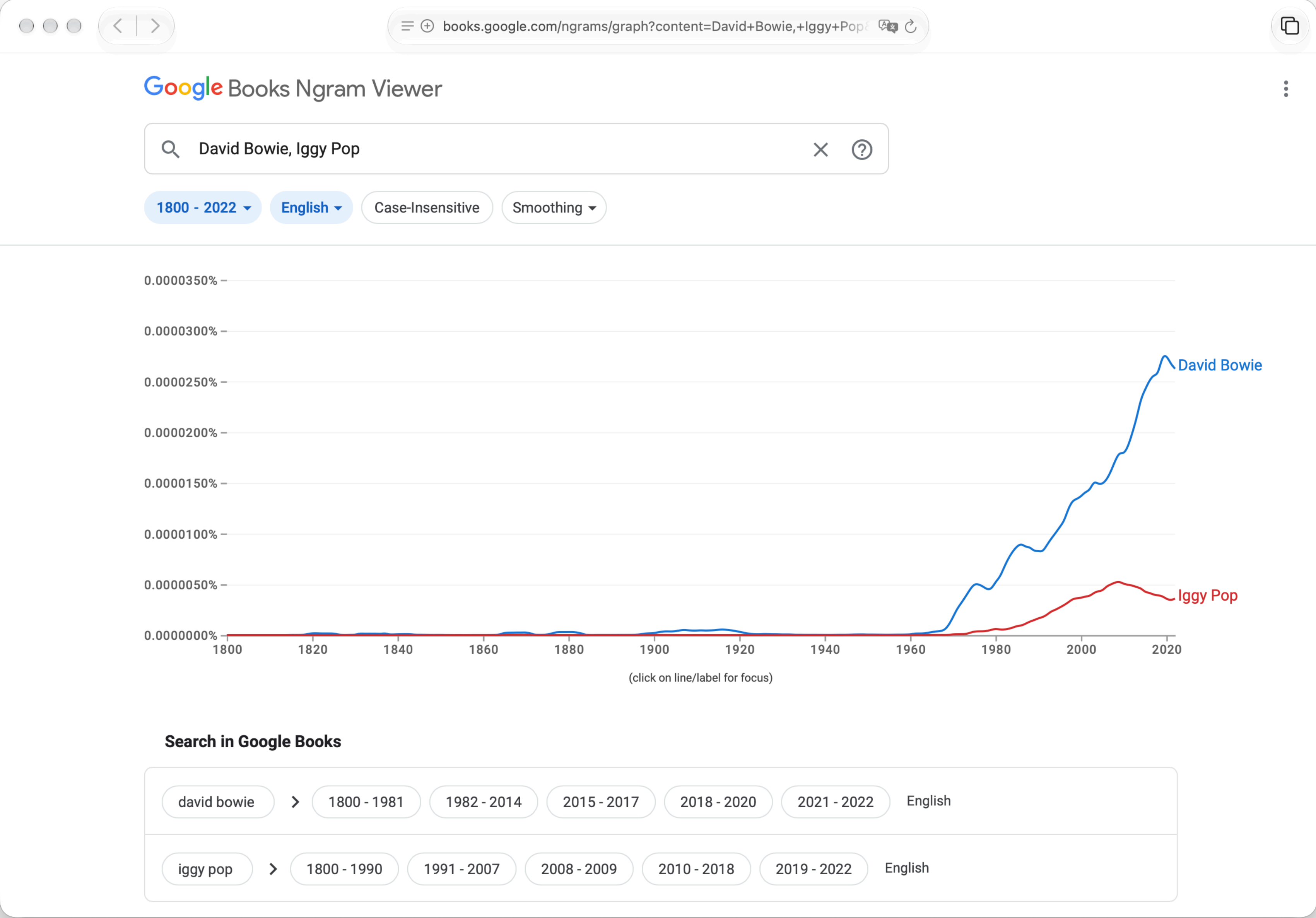
The files vary widely in size because some patterns of letters are more common than others: the "na" file will be larger than the "ng" file since so many more words begin with "na" than "ng". Files with a letter followed by an underscore (e.g., `s_`) contain ngrams that begin with the first letter, but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

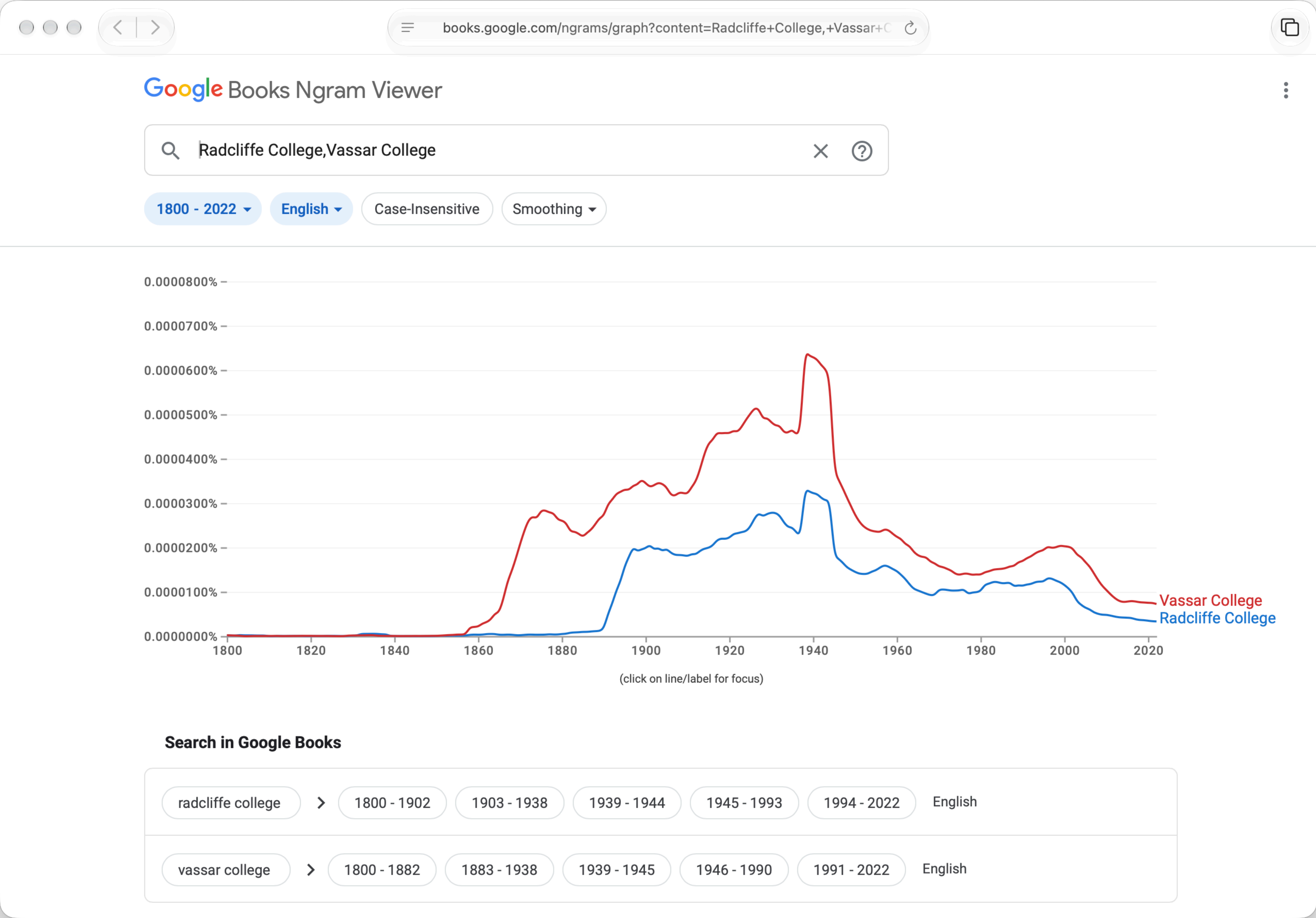
We've included separate files for ngrams that start with punctuation or with other non-alphanumeric characters. Finally, we have separate files for ngrams in which the first word is a part of speech tag (e.g., `_ADJ_`, `_ADP_`).

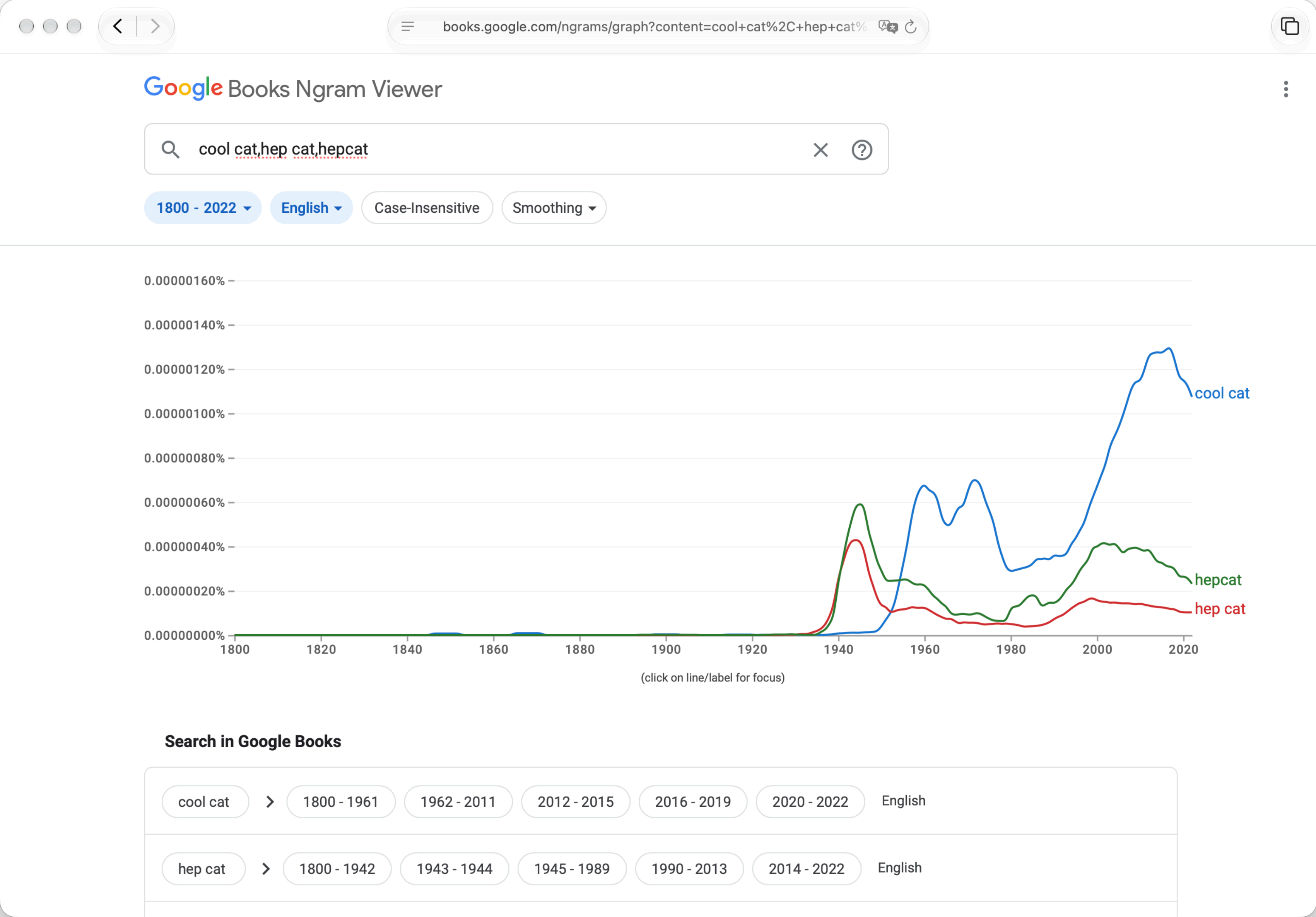
Corpora

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	eng_gb_2012	googlebooks-eng-gb-all-	

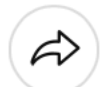




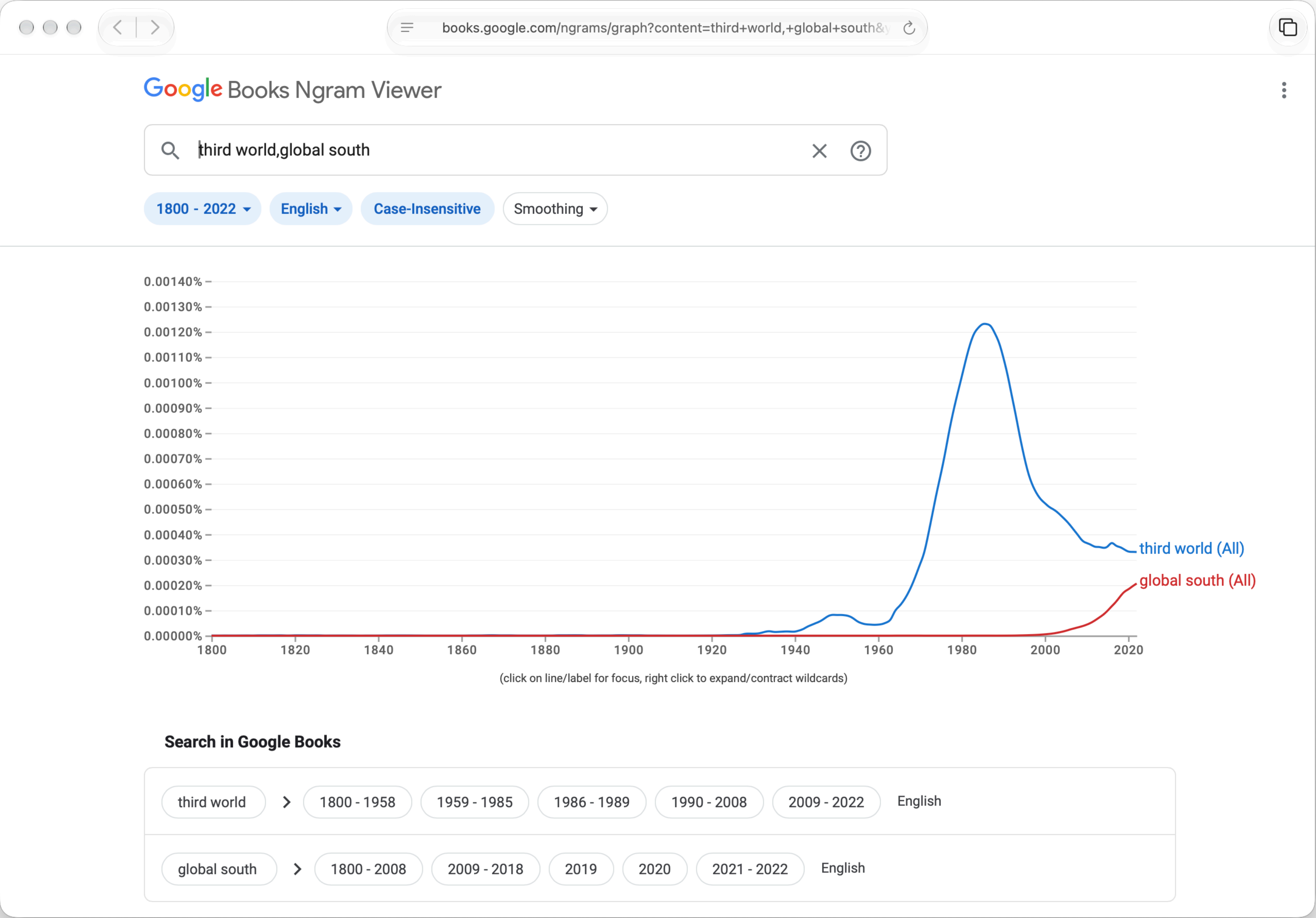


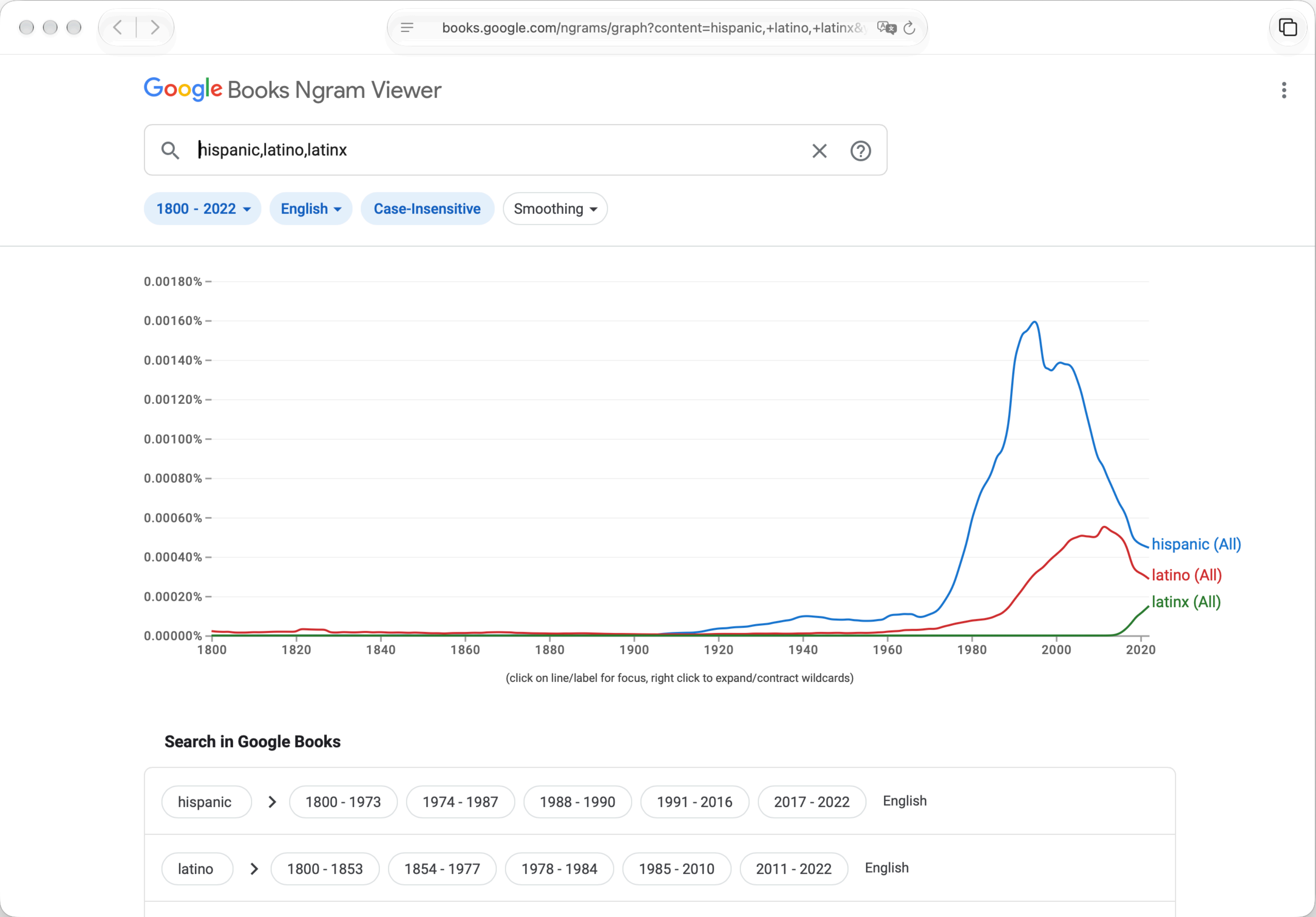
WOULD YOU USE THE FOLLOWING WORDS OR TERMS?

●



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





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:

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

Think–pair–share

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

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

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

Calculate the probabilities of the bigrams

$$P(\textit{I} \mid \textit{<s>}) =$$

$$P(\textit{Sam} \mid \textit{<s>}) =$$

$$P(\textit{</s>} \mid \textit{Sam}) =$$

$$P(\textit{Sam} \mid \textit{am}) =$$

in the toy corpus above, using the formula $P(w_i \mid w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$

In maximum likelihood estimation, 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*.

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>i</i>	<i>want</i>	<i>to</i>	<i>eat</i>	<i>chinese</i>	<i>food</i>	<i>lunch</i>	<i>spend</i>
W_1	<i>i</i>	5	827	0	9	0	0	0	2
	<i>want</i>	2	0	608	1	6	6	5	1
	<i>to</i>	2	0	4	686	2	0	6	211
	<i>eat</i>	0	0	2	0	16	2	42	0
	<i>chinese</i>	1	0	0	0	0	82	1	0
	<i>food</i>	15	0	15	0	1	4	0	0
	<i>lunch</i>	2	0	0	0	0	1	0	0
	<i>spend</i>	1	0	1	0	0	0	0	0

Normalize
by unigram
counts

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

W_2

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

W_1

Berkeley Restaurant Project: Bigram probabilities

		W ₂							
		<i>i</i>	<i>want</i>	<i>to</i>	<i>eat</i>	<i>chinese</i>	<i>food</i>	<i>lunch</i>	<i>spend</i>
W ₁	<i>i</i>	0.002	0.33	○	0.0036	○	○	○	0.00079
	<i>want</i>	0.0022	○	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
	<i>to</i>	0.00083	○	0.0017	0.28	0.00083	○	0.0025	0.087
	<i>eat</i>	○	○	0.0027	○	0.021	0.0027	0.056	○
	<i>chinese</i>	0.0063	○	○	○	○	0.52	0.0063	○
	<i>food</i>	0.014	○	0.014	○	0.00092	0.0037	○	○
	<i>lunch</i>	0.0059	○	○	○	○	0.0029	○	○
	<i>spend</i>	0.0036	○	0.0036	○	○	○	○	○

We use the bigram model to compute sentence probabilities:

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

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

$$P(\text{english} \mid \text{want}) = 0.0011$$

$$P(\text{chinese} \mid \text{want}) = 0.0065$$

World knowledge; culture

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

$$P(\text{english} \mid \text{want}) = 0.0011$$

$$P(\text{chinese} \mid \text{want}) = 0.0065$$

World knowledge; culture

$$P(\text{to} \mid \text{want}) = 0.66$$

$$P(\text{eat} \mid \text{to}) = 0.28$$

$$P(\text{food} \mid \text{to}) = 0$$

$$P(\text{want} \mid \text{spend}) = 0$$

Syntactic preferences

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

$$P(\text{english} \mid \text{want}) = 0.0011$$

$$P(\text{chinese} \mid \text{want}) = 0.0065$$

World knowledge; culture

$$P(\text{to} \mid \text{want}) = 0.66$$

$$P(\text{eat} \mid \text{to}) = 0.28$$

$$P(\text{food} \mid \text{to}) = 0$$

$$P(\text{want} \mid \text{spend}) = 0$$

Syntactic preferences

$$P(i \mid \langle s \rangle) = 0.25$$

Discourse

Generate random sentences:

Choose a random bigram ($\langle s \rangle$, w) according to its probability.

Now choose a random bigram (w , x) according to its probability.

And so on until we randomly choose $\langle /s \rangle$.

$\langle s \rangle$ /

Generate random sentences:

Choose a random bigram ($\langle s \rangle$, w) according to its probability.

Now choose a random bigram (w , x) according to its probability.

And so on until we randomly choose $\langle /s \rangle$.

$\langle s \rangle$ *I want*

Generate random sentences:

Choose a random bigram ($\langle s \rangle$, w) according to its probability.

Now choose a random bigram (w , x) according to its probability.

And so on until we randomly choose $\langle /s \rangle$.

$\langle s \rangle$ / *want to*

Generate random sentences:

Choose a random bigram ($\langle s \rangle$, w) according to its probability.

Now choose a random bigram (w , x) according to its probability.

And so on until we randomly choose $\langle /s \rangle$.

$\langle s \rangle$ *I want to eat*

Generate random sentences:

Choose a random bigram ($\langle s \rangle$, w) according to its probability.

Now choose a random bigram (w , x) according to its probability.

And so on until we randomly choose $\langle /s \rangle$.

$\langle s \rangle$ *I want to eat food*

Generate random sentences:

Choose a random bigram ($\langle s \rangle$, w) according to its probability.

Now choose a random bigram (w , x) according to its probability.

And so on until we randomly choose $\langle /s \rangle$.

$\langle s \rangle$ *I want to eat food* $\langle /s \rangle$

Think–pair–share

PROMPT: Generate a question in the style of William Shakespeare, starting with the words *Hath in the*.

Hath in the world of men, a truth that doth endure, or is all but fleeting, as the winds of change do lure?

Hath in the world the promise been written, whilst cherubs danced and fawns lay smitten?

Hath in the tables of his law commanded that thou shalt know, the treason that my haste forbids me show?

Which one was written...

...a 4-gram LM trained on Shakespeare?

...ChatGPT?

...a human?

Think–pair–share

PROMPT: Generate a question in the style of William Shakespeare, starting with the words *Hath in the*.

Hath in the world of men, a truth that doth endure, or is all but fleeting, as the winds of change do lure?

ChatGPT

Hath in the world the promise been written, whilst cherubs danced and fawns lay smitten?

a human

Hath in the tables of his law commanded that thou shalt know, the treason that my haste forbids me show?

a 4-gram LM trained on Shakespeare

Approximating Shakespeare

Approximating Shakespeare

1-gram

*To him swallowed confess hear both. Which. Of
save on trail for are ay device and rote life have*

*Hill he late speaks; or! a more to leg less first you
enter*

Approximating Shakespeare

1-gram

*To him swallowed confess hear both. Which. Of
save on trail for are ay device and rote life have*

*Hill he late speaks; or! a more to leg less first you
enter*

2-gram

*Why dost stand forth thy canopy, forsooth; he is
this palpable hit the King Henry. Live king. Follow.*

*What means, sir? I confess she? then all sorts, he
is trim, captain.*

Approximating Shakespeare

1-gram

*To him swallowed confess hear both. Which. Of
save on trail for are ay device and rote life have*

*Hill he late speaks; or! a more to leg less first you
enter*

2-gram

*Why dost stand forth thy canopy, forsooth; he is
this palpable hit the King Henry. Live king. Follow.*

*What means, sir? I confess she? then all sorts, he
is trim, captain.*

3-gram

*Fly, and will rid me these news of price. Therefore
the sadness of parting, as they say, 'tis done.*

*This shall forbid it should be branded, if renown
made it empty.*

Approximating Shakespeare

1-gram

*To him swallowed confess hear both. Which. Of
save on trail for are ay device and rote life have*

*Hill he late speaks; or! a more to leg less first you
enter*

2-gram

*Why dost stand forth thy canopy, forsooth; he is
this palpable hit the King Henry. Live king. Follow.*

*What means, sir? I confess she? then all sorts, he
is trim, captain.*

3-gram

*Fly, and will rid me these news of price. Therefore
the sadness of parting, as they say, 'tis done.*

*This shall forbid it should be branded, if renown
made it empty.*

4-gram

*King Henry. What! I will go seek the traitor
Gloucester. Exeunt some of the watch.*

It cannot be but so.

