N-Gram Model

Language Model

Vládia Pinheiro

FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA
ENSINANDO E APRENDENDO

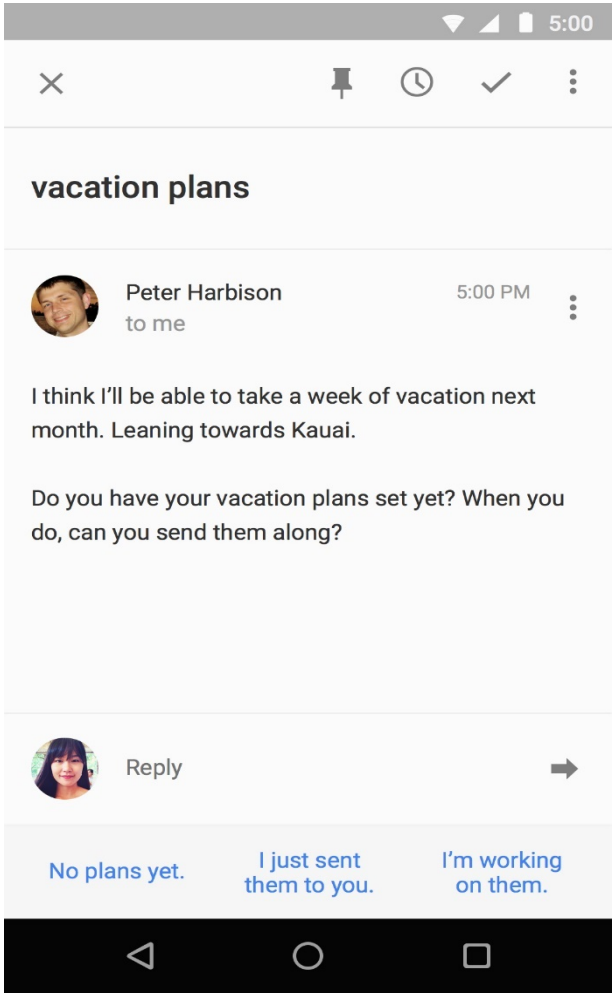
O que é um *language model*?

- Distribuição de probabilidade sobre sentenças (i.e. Sequência de palavras)
  - $P(W) = P(w_1,w_2,w_3,...,w_n)$
  - P("Today is Tuesday") > P("Today Tuesday is")
  - P("Today is Tuesday") > P("Today is Virginia")
- Probabilidades condicionais para descobrir a proxima palavra w_k
  - $P(w_k \mid w_1,w_2,w_3,...,w_{k-1})$

Aplicações de *Language Model*

Aplicações de *Language Model*

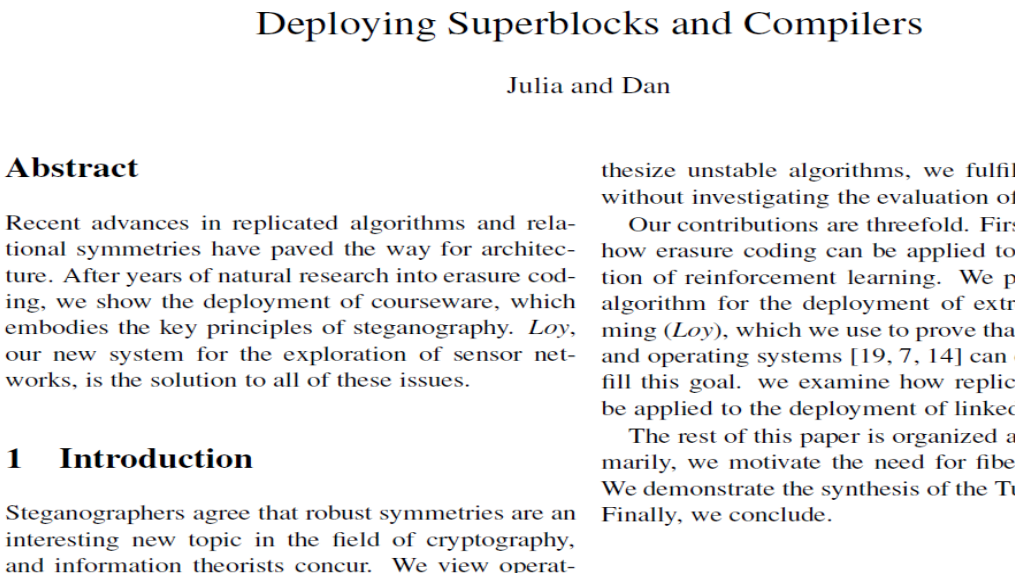
Chatbot



Aplicações de *Language Model*

Geração de texto

https://pdos.csail.mit.edu/archive/scigen/



Deploying Superblocks and Compilers

Julia and Dan

Abstract

Recent advances in replicated algorithms and relational symmetries have paved the way for architecture. After years of natural research into erasure coding, we show the deployment of courseware, which embodies the key principles of steganography. *Loy*, our new system for the exploration of sensor networks, is the solution to all of these issues.

1 Introduction

Steganographers agree that robust symmetries are an interesting new topic in the field of cryptography, and information theorists concur. We view operat-

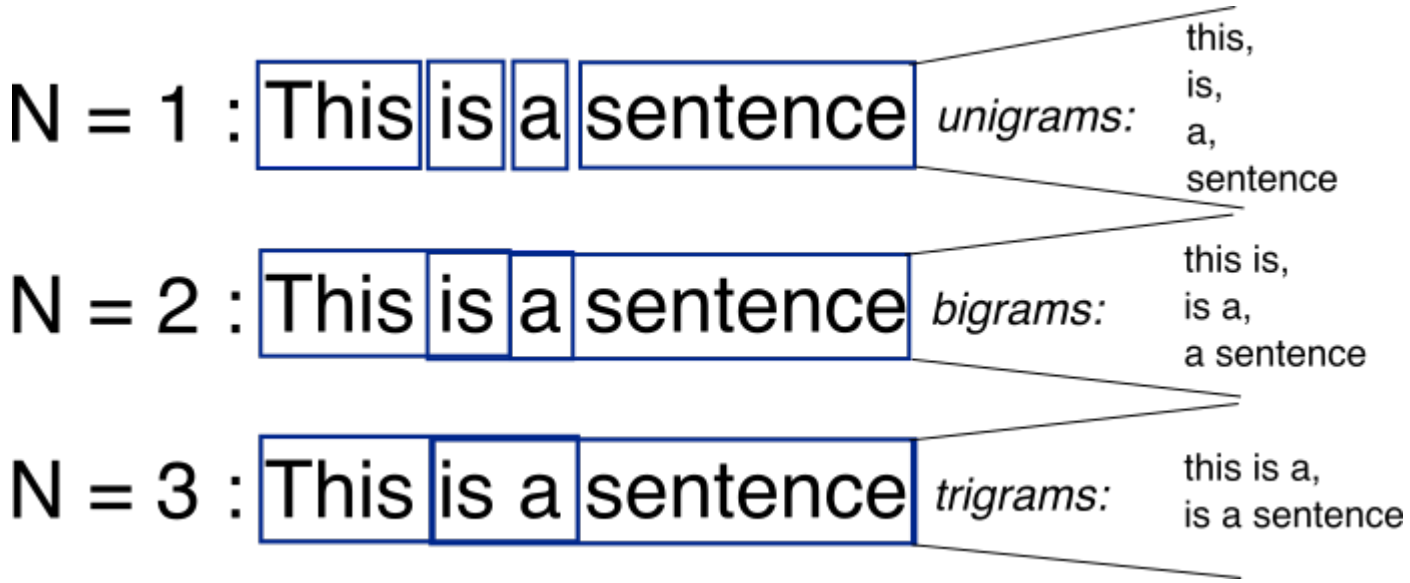
thesize unstable algorithms, we fulfil
without investigating the evaluation of

Our contributions are threefold. Firs
how erasure coding can be applied to
tion of reinforcement learning. We p
algorithm for the deployment of extr
ming (*Loy*), which we use to prove tha
and operating systems [19, 7, 14] can
fill this goal. we examine how replic
be applied to the deployment of linkec

The rest of this paper is organized a
marily, we motivate the need for fibe
We demonstrate the synthesis of the Tu
Finally, we conclude.

Bag-of-Words com N-grams

- N-grams: uma sequência contínua de n tokens a partir de uma parte de texto



N-Gram Models

- Unigram model:
- Bigram model:
- Trigram model:

- N-gram model:

…

Random language via n-gram

- https://www.cs.jhu.edu/~jason/licl/PowerPoint/lect01b-ngram-text.pdf

- Atrás da cena:
  - Teoria da Probabilidade

Amostragem com Substituição

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

P(of) = 3/66
P(Alice) = 2/66
P(was) = 2/66
P(to) = 2/66
P(her) = 2/66
P(sister) = 2/66
P(,) = 4/66
P(') = 4/66

Probabilidade Condicional dada a palavra anterior

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

$P(w_{i+1}=\text{of} \mid w_i=\text{tired}) = 1$

$P(w_{i+1}=\text{of} \mid w_i=\text{use}) = 1$

$P(w_{i+1}=\text{sister} \mid w_i=\text{her}) = 1$

$P(w_{i+1}=\text{beginning} \mid w_i=\text{was}) = 1/2$

$P(w_{i+1}=\text{reading} \mid w_i=\text{was}) = 1/2$

$P(w_{i+1}=\text{bank} \mid w_i=\text{the}) = 1/3$

$P(w_{i+1}=\text{book} \mid w_i=\text{the}) = 1/3$

$P(w_{i+1}=\text{use} \mid w_i=\text{the}) = 1/3$

Probabilidade Condicional dada a palavra anterior

English

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

Word Salad

beginning by, very Alice but was and? reading no tired of to into sitting sister the, bank, and thought of without her nothing: having conversations Alice once do or on she it get the book her had peeped was conversation it pictures or sister in, 'what is the use had twice of a book''pictures or' to

Now, P(English) ≫ P(word salad)

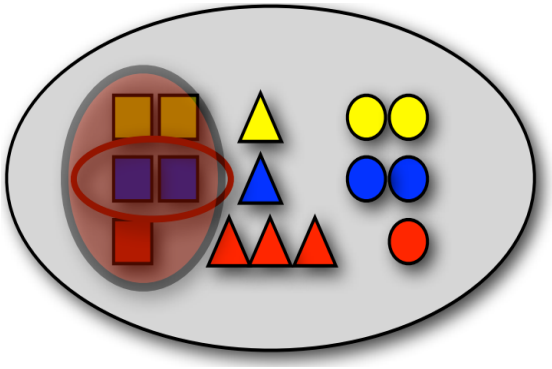
$P(w_{i+1}= \text{of} \mid w_i=\text{tired}) = 1$
$P(w_{i+1}= \text{of} \mid w_i=\text{use}) = 1$
$P(w_{i+1}= \text{sister} \mid w_i=\text{her}) = 1$
$P(w_{i+1}= \text{beginning} \mid w_i=\text{was}) = 1/2$
$P(w_{i+1}= \text{reading} \mid w_i=\text{was}) = 1/2$

$P(w_{i+1}= \text{bank} \mid w_i=\text{the}) = 1/3$
$P(w_{i+1}= \text{book} \mid w_i=\text{the}) = 1/3$
$P(w_{i+1}= \text{use} \mid w_i=\text{the}) = 1/3$

Recap: Teoria da Probabilidade

- Probabilidade: resultado entre 0 e 1, ou 0 e 100%
- P(evento impossível) = 0
- P(qualquer coisa) = 1 (ou 100%)
- P(A) ou P(B) = P(A) + P(B)
  - P(qualquer coisa) = P(uma coisa) + P(segunda coisa) + ... + P(enésima coisa)
- Probabilidade condicional $P(A|B) = P(A \cap B) / P(B)$
- $P(A \cap B) = P(B) * P(A|B) = P(A) * P(B|A)$
  - $P(A \cap B) = P(A) * P(B)$, se eventos independentes
  - $P(A_1 \cap ... \cap A_n) = P(A_1) * P(A_2|A_1) * P(A_3| A_1 \cap A_2) ... P(A_n| \cap_{i=1}^{n-1} A_i)$
- A e B são condicionalmente independentes se
  - $P(A \cap B|C) = P(A|C) * P(B|C)$

Recap: Teoria da Probabilidade



- Probabilidade Condicional
  - P(blue | □) = c(blue ∩ □)/c(□)
- Regra de Bayes:
  - P(B|A) = P(A|B) x P(B) / P(A)
  - Verificar: P(red | □) , P(□ | red), P(□), P(red)
- Independência Condicional:
  - P(B|A) = P(B)

Exemplo

- Sabe-se que catáforas são raras:
  - Só desejamos **isto**: **férias**!!!
- De todas as sentenças de um córpus, sabe-se que somente uma fração de 0.008 delas contêm catáforas
- Existe um sistema de PLN que diz se sentenças são ou não catafóricas
  - O sistema retorna sim – um verdadeiro positivo (as sentenças são catafóricas e o sistema diz que são) – em 98% dos casos
  - O sistema retorna não – um verdadeiro negativo (as sentenças não são catafóricas e o sistema diz que não são) – em 97% dos casos

- Uma sentença foi rotulada como catafórica pelo sistema. É possível afirmar que ela é catafórica? Qual a probabilidade de ela ser catafórica de fato?

Exemplo

- Sabe-se que catáforas são raras: de todas as sentenças de um córpus, sabe-se que somente uma fração de 0.008 delas contêm catáforas

- Existe um sistema de PLN que diz se sentenças são ou não catafóricas
  - O sistema retorna sim – um verdadeiro positivo (as sentenças são catafóricas e o sistema diz que são) – em 98% dos casos
  - O sistema retorna não – um verdadeiro negativo (as sentenças não são catafóricas e o sistema diz que não são) – em 97% dos casos

- Um sentenças foi rotulada como catafórica pelo sistema. É possível afirmar que ela é catafórica? Qual a probabilidade de ela ser catafórica de fato?

Resolução: P(catáfora) = 0.008 P(sem catáfora) = 0.992

P(sim | catáfora) = 0.98 P(não | catáfora) = 0.02

P(sim | sem catáfora) = 0.03 P(não | sem catáfora) = 0.97

P(catáfora | sim) = P(sim | catáfora) * P(catáfora) = 0.98*0.008 = 0.0078

P(sem catáfora | sim) = P(sim | sem catáfora) * P(sem catáfora) = 0.03*0.992 = **0.0298**

Regra da Cadeia

- A probabilidade conjunta pode ser descrita em termos de probabilidade condicional

$$P(X,Y) = P(X \mid Y)\, P(Y)$$

- Mais variáveis:

$$P(X, Y, Z) = P(X \mid Y, Z)\, P(Y, Z)$$
$$= P(X \mid Y, Z)\, P(Y \mid Z)\, P(Z)$$

$$P(X_1, X_2, X_3, \ldots, X_n) = P(X_n \mid X_{n-1}, \ldots, X_3, X_2, X_1) \ldots$$
$$P(X_3 \mid X_2, X_1)\, P(X_2 \mid X_1)\, P(X_1)$$

Language model com N-gram

- Para fins de compactação do modelo:
  - Supõe que cada palavra depende apenas de suas (n-1) palavras antecessoras (suposição de Markov)
  - $P(X_n \mid X_{n-1}, \ldots, X_3, X_2, X_1) = P(X_n \mid X_{n-1}, X_{n-2})$

Andrei Markov

Language model com N-gram

- Exemplo considerando um modelo trigrama (3-gram)
  - $P(X_1, X_2, X_3, ..., X_n) = P(X_n \mid X_{n-1}, X_{n-2})\ P(X_{n-1} \mid X_{n-2}\ X_{n-3})\ \ ...\ P(X_3|X_2,X_1)\ P(X_2|X_1)\ P(X_1)$
  - P("Today is a sunny day") = ???

Um exemplo com modelo bigrama

<S> I am Sam </S>

<S> I am legend </S>

<S> Sam I am </S>

P(<S> | inicio) = ? P(I | <S>) = ? P(am | I) = ?

P(Sam | am) = ? P(</S> | Sam) = ?

P(<S>I am Sam</S> | bigram model) = ?

P(<S>Sam I am</S> | bigram model) = ?

Um exemplo com modelo bigrama

<S> I am Sam </S>

<S> I am legend </S>

<S> Sam I am </S>

P(<S> | inicio) = 3/3 P(I | <S>) = 2/3 P(am | I) = 3/3

P(Sam | am) = 1/3 P(</S> | Sam) = 1/2

P(<S> | inicio) = 3/3

P(<S>I am Sam</S> | bigram model) = 1/2 * 1/3 * 3/3 * 2/3 * 3/3 = **0,1249**

P(<S>Sam I am </S> | bigram model) = 1/3 * 3/3 * 1/2 * 1/3 * 3/3 = 0,0555

Unigram model

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i)$$

- To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
- Every enter now severally so, let
- Hill he late speaks; or! a more to leg less first you enter
- Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like

Bigram model

- $P(w_i \mid w_1 w_2 \ldots w_{i-1}) \approx P(w_i \mid w_{i-1})$

- What means, sir. I confess she? then all sorts, he is trim, captain.
- Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.
- What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?
- Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt

Ngram model

$$P(w_i \mid w_1 w_2 \ldots w_{i-1}) \approx P(w_i \mid w_{i-k} \ldots w_{i-1})$$

Trigram

- Sweet prince, Falstaff shall die. Harry of Monmouth's grave.
- This shall forbid it should be branded, if renown made it empty.
- Indeed the duke; and had a very good friend.
- Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

Quadrigram

- King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;
- Will you not tell me who I am?
- It cannot be but so.
- Indeed the short and the long. Marry, 'tis a noble Lepidus.

Recursos

- Google n-gram:
  - https://books.google.com/ngrams

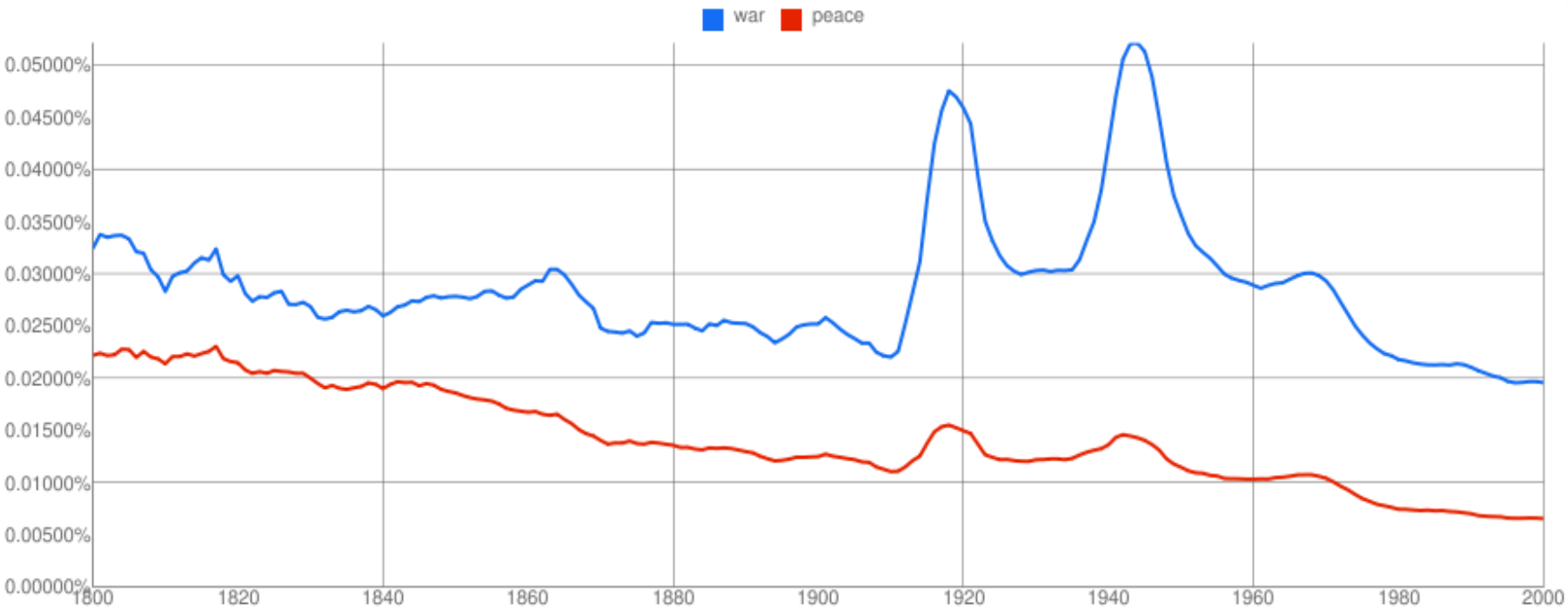
```
File sizes: approx. 24 GB compressed (gzip'ed) text files

Number of tokens: 1,024,908,267,229
Number of sentences: 95,119,665,584
Number of unigrams: 13,588,391
Number of bigrams: 314,843,401
Number of trigrams: 977,069,902
Number of fourgrams: 1,313,818,354
Number of fivegrams: 1,176,470,663
```

Graph these **case-sensitive** comma-separated phrases: war,peace

between 1800 and 2000 from the corpus English with smoothing of 3.

Search lots of books



Search in Google Books:

1800 - 1817	1818 - 1939	1940 - 1952	1953 - 1971	1972 - 2000	war (English)
1800 - 1812	1813 - 1825	1826 - 1872	1873 - 1963	1964 - 2000	peace (English)

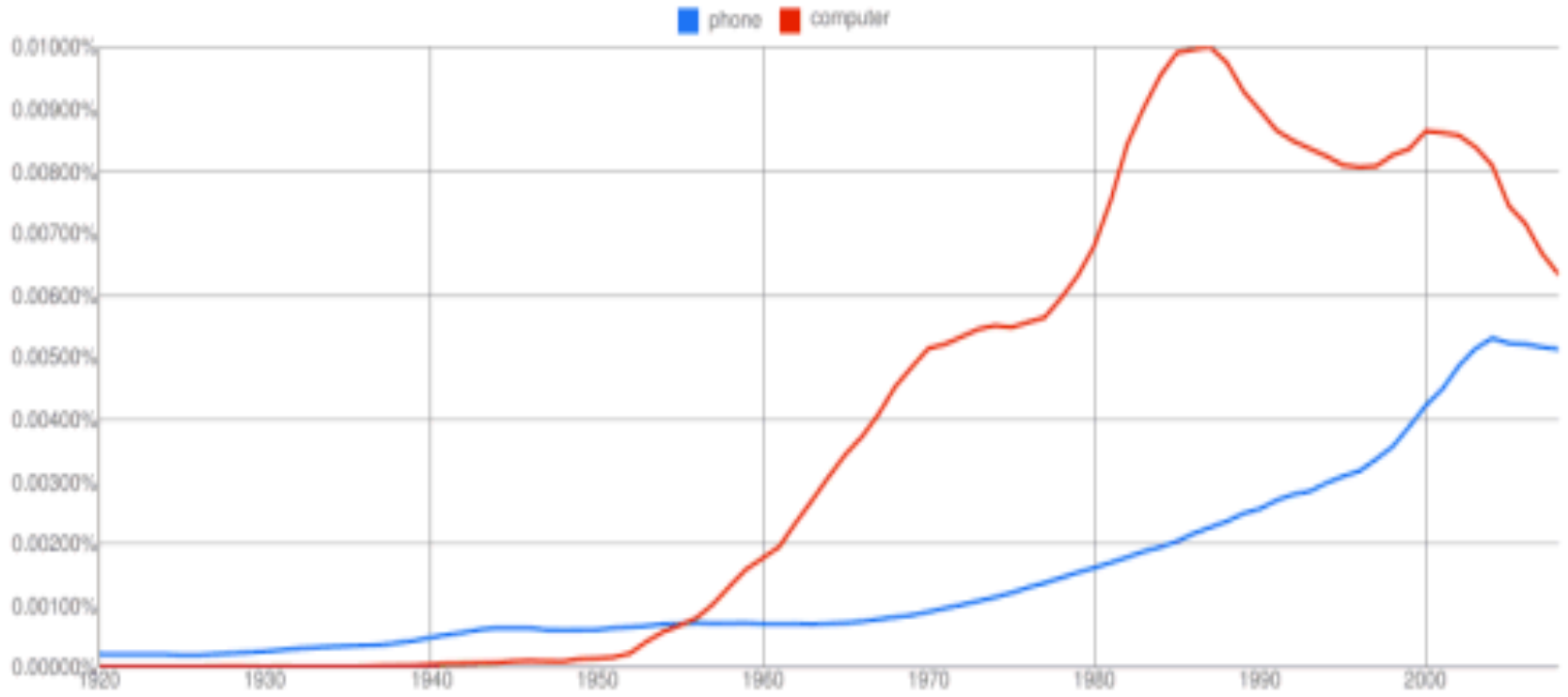
Run your own experiment! Raw data is available for download here.

Google labs Books Ngram Viewer

Graph these **case-sensitive** comma-separated phrases: phone,computer

between 1920 and 2008 from the corpus English with smoothing of 3 .

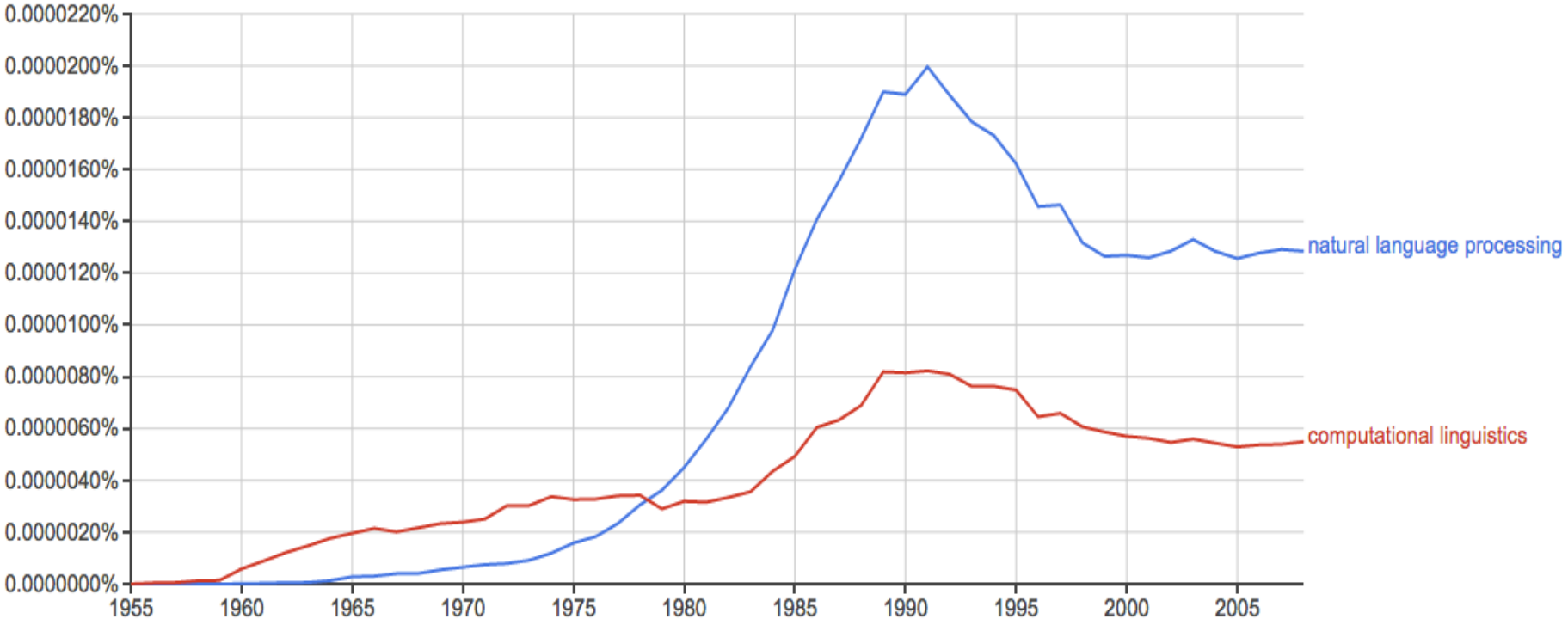
Search lots of books



Google Books Ngram Viewer

Graph these comma-separated phrases: natural language processing,computational linguistics case-insensitive

between 1955 and 2008 from the corpus English with smoothing of 3. Search lots of books



Google Books Ngram Viewer

Graph these comma-separated phrases: covid,coronavirus case-insensitive

between 1800 and 2019 from the corpus English (2012) with smoothing of 3 . Search lots of books

N-Gram Tagging

- POS Tagging baseado em unigramas
  - Usamos somente a tag mais provavel para a palavra, isolada do contexto em que esta ocorre.
    - “wind” receberá a mesma tag . Independente se ela ocorre em “***the wind***” ou “***to wind***”
- Um *N-Gram Tagger* é uma generalização de um *Unigram Tagger*, cujo contexto é a palavra atual (a ser classificada) junto com as POS tags das n-1 palavras precedentes.
  - Por exemplo, um *Bigram Tagger* considera a tag da palavra precedente em adição à palavra atual.
  - P(tag(“wind”) = “VB” | tag_anterior=“PRP”), como em “*I wind it all the time*”

O que dizer de palavras e expressões “nao vistas”?

- Shakespeare corpus
  - N=884,647 tokens
  - V=29,066 tokens distintos (vocabulário)
  - Somente 0,04% dos possíveis bigramas ocorrem neste *corpus*

- Tagging palavras ou expressoes desconhecidas (UNK)
  - Limitando o vocabulário para as ***n*** palavras mais frequentes, e substituicao das palavras desconhecida por ”UNK”.
  - Um bigram-tagger poderá detectar que UNK precedido da particula “TO” deve ser classificado como um verbo