Model n-gram i generowanie tekstu

Agnieszka Ławrynowicz

Wydział Informatyki Politechniki Poznańskiej

21 marca 2018

Model języka: n-gram

Probabilistyczne modelowanie języka

Cel: przypisanie prawdopodobieństwa do zdania (sekwencji wyrazów):

$$P(W) = P(w_1, w_2, w_3, w_4, \ldots w_n)$$

Podobne zadanie: prawdopodobieństwo kolejnego słowa:

$$P(w_5|w_1, w_2, w_3, w_4)$$

Model języka oblicza jedno z dwóch:

$$P(W) \text{ lub } P(w_n|w_1, w_2 \ldots w_{n-1})$$

Czy ludzie potrafią przewidzieć następny wyraz?

Jak?

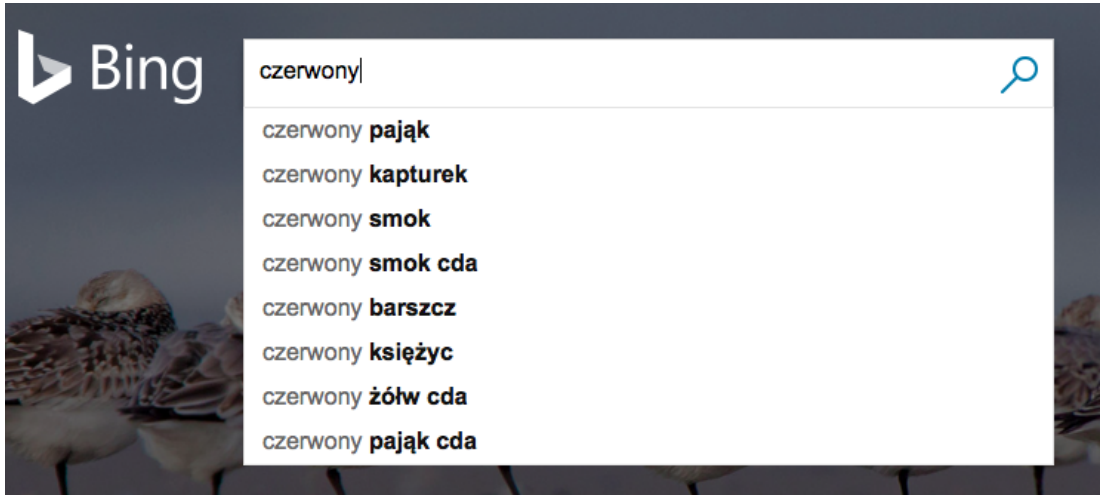
- Wiedza dziedzinowa: red blood vs. red hat
- Wiedza nt. składni: the. . . <adj|noun>
- Wiedza leksykalna: pieczony kurczak vs. placek

Probabilistyczne modele językowe

Cel: przypisanie prawdopodobieństwa do zdania
Zastosowania:

- Korekta pisowni
- Rozpoznawanie mowy
- Autouzupełnianie
- Podpowiedź odpowiedzi (np. na smsa)
- Generowanie języka
- Tłumaczenie maszynowe
- Streszczanie
- Odpowiadanie na pytania

Autouzupełnianie



Korekta pisowni

$$P(\text{ta reklama trwa 20 skeund}) < P(\text{ta reklama trwa 20 sekund})$$

Generowanie języka

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

SCIgen - An Automatic CS Paper Generator

About Generate Examples Talks Code Donations Related People Blog

About

SCIgen is a program that generates random Computer Science research papers, including graphs, figures, and citations. It uses a hand-written **context-free grammar** to form all elements of the papers. Our aim here is to maximize amusement, rather than coherence.

One useful purpose for such a program is to auto-generate submissions to conferences that you suspect might have very low submission standards. A prime example, which you may recognize from spam in your inbox, is SCI/IIIS and its dozens of co-located conferences (check out the very broad conference description on the **WMSCI 2005** website). There's also a list of **known bogus conferences**. Using SCIgen to generate submissions for conferences like this gives us pleasure to no end. In fact, one of our papers was accepted to SCI 2005! See **Examples** for more details.

We went to WMSCI 2005. Check out the **talks and video**. You can find more details in our **blog**.

Also, check out our 10th anniversary celebration project: **SCIpher**!

Generate a Random Paper

Want to generate a random CS paper of your own? Type in some optional author names below, and click "Generate".

Author 1:
Author 2:
Author 3:
Author 4:
Author 5:

Generate Wyzeruj

SCIgen currently supports Latin-1 characters, but not the full Unicode character set.

Obliczanie $P(w)$ (prawdopodobieństwa łącznego)

Jak obliczyć prawdopodobieństwo łączne?

P(mój, kot, jest, tak, zwinny, jak)

Reguła łańcuchowa (przypomnienie)

Prawdopodobieństwa warunkowe:

$$P(B|A) = P(A,B)/P(A)$$
$$P(A,B) = P(A)P(B|A)$$
$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

$$P(x_1, x_2, x_3, \ldots, x_n) =$$
$$P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)\ldots P(x_n|x_1, \ldots, x_{n-1})$$

Reguła łańcuchowa (obliczanie prawdopodobieństwa łącznego)

$$P(w_1 w_2 w_3 \ldots w_n) = \prod_i P(w_i | w_1 w_2 \ldots w_{i-1})$$

P("mój kot jest tak zwinny jak") = P(mój) · P(kot|mój) · P(jest | mój kot) · P(tak| mój kot jest)· P(zwinny|mój kot jest tak) · P(jak|mój kot jest tak zwinny)

Reguła łańcuchowa (obliczanie prawdopodobieństwa łącznego)

$$P(w_1 w_2 w_3 \ldots w_n) = \prod_i P(w_i | w_1 w_2 \ldots w_{i-1})$$

P("mój kot jest tak zwinny jak") = P(mój) · P(kot|mój) · P(jest | mój kot) · P(tak| mój kot jest)· P(zwinny|mój kot jest tak) · P(jak|mój kot jest tak zwinny)

Jak obliczyć prawdopodobieństwa?

Czy możemy po prostu zliczać wystąpienia zdań?

$$\text{P(jak|mój kot jest tak zwinny)} = \frac{count(\text{mój kot jest tak zwinny jak})}{count(\text{mój kot jest tak zwinny})}$$

× Zbyt dużo możliwości aby zgromadzić wystarczająco dużo danych do estymacji.
Musimy posłużyć się aproksymacją

Własność Markowa

Andriej Markow (1856-1922)

Upraszczające założenie:

$P(\text{jak}|\text{mój kot jest tak zwinny}) \approx P(\text{jak}|\text{zwinny})$

$P(\text{jak}|\text{mój kot jest tak zwinny}) \approx P(\text{jak}|\text{tak zwinny})$

Własność Markowa c.d.

$$P(w_1 w_2 w_3 \ldots w_n) = \prod_i P(w_i | w_1 w_2 \ldots w_{i-1})$$

Aproksymujemy każdy czynnik iloczynu:

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

Najprostszy przypadek: model unigram/1-gram

$$P(w_1 w_2 w_3 \ldots w_n) \approx \prod_i P(w_i)$$

Bigram/2-gram

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

Modele n-gram

Unigram (pojedynczy wyraz):

$$P(w_1 w_2 w_3 \dots w_n) \approx \prod_i P(w_i)$$

Bigram (para sąsiadujących wyrazów):

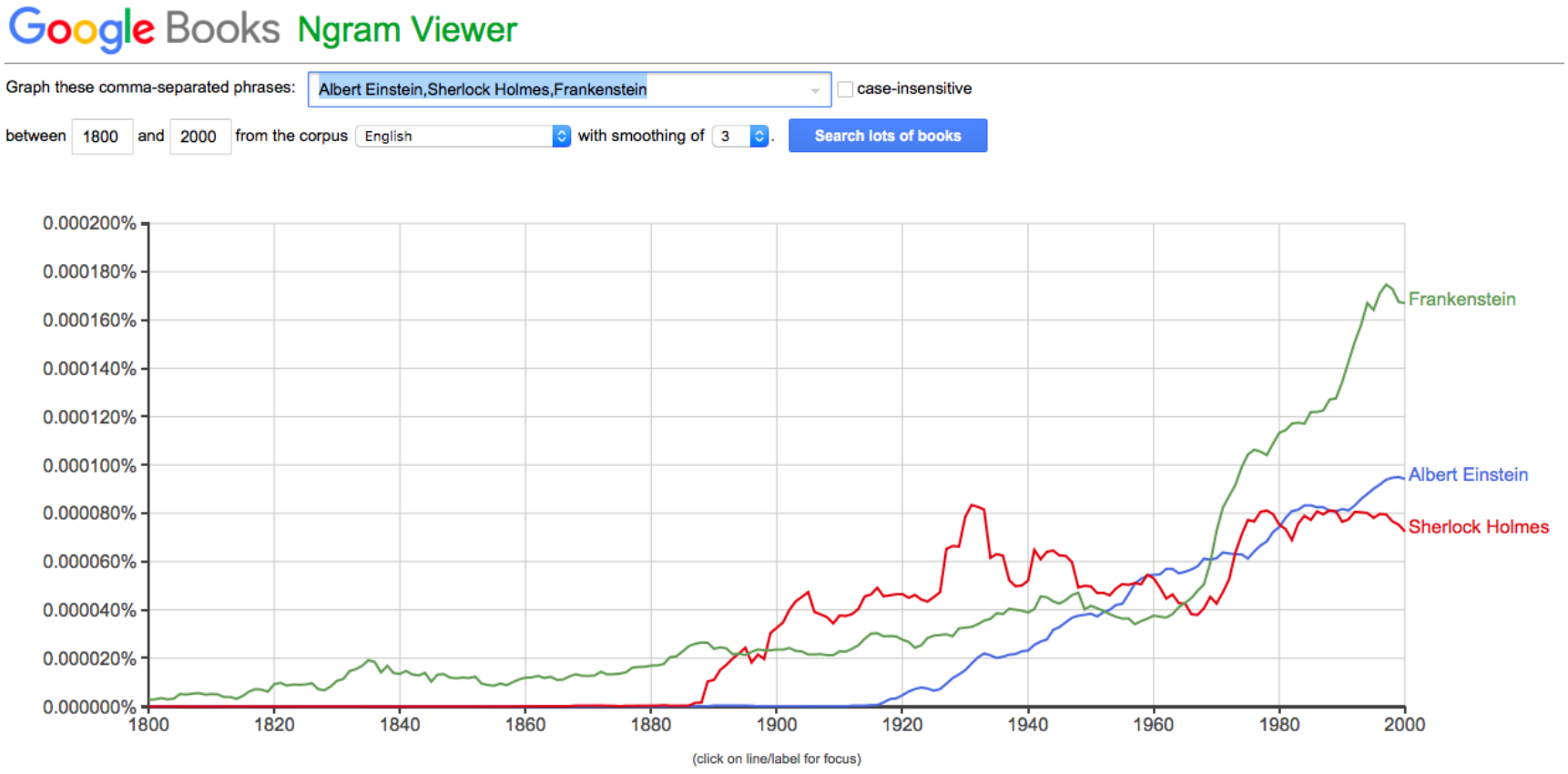
$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-1})$$

N-gram (w ogólności n sąsiadujących wyrazów):

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

Google N-Gram

https://books.google.com/ngrams



Estymacja prawdopodobieństwa bigramów

$$P(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

Specjalne tokeny

Możemy reprezentować początek i koniec zdania za pomocą specjalnych tokenów:

<s> *To jest zdanie* </s>

Przykład

$$P(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

<s>Wlazł kotek na płotek i mruga</s>
<s>Wlazł kotek w płotek </s>
<s>kotek sobie mruczy i mruga </s>

$$P(\text{kotek} \mid \text{Wlazł}) = \frac{2}{2}$$
$$P(\text{mruga} \mid \text{i}) = \frac{2}{2}$$
$$P(\text{płotek} \mid \text{w}) = \frac{1}{1}$$
$$P(\text{sobie} \mid \text{kotek}) = \frac{1}{3}$$
$$P(\text{Wlazł} \mid \text{<s>}) = \frac{2}{3}$$

Przykład: Berkeley Restaurant Project c.d.

Dane z systemu dialogowego, który odpowiadał na pytania dotyczące bazy danych restauracji w Berkeley w Kalifornii (Jurafsky i inni, 1994).
Próbka z 9332 zdań:

- 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

Przykład z "*Speech and Language Processing (3rd ed. draft)*", Dan Jurafsky and James H. Martin. Draft chapters in progress, August 28, 2017, `https://web.stanford.edu/~jurafsky/slp3/`

Przykład: Berkeley Restaurant Project c.d.

Liczba bigramów dla 8 wyrazów (z $|V| = 1446$).

	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
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

Przykład z "*Speech and Language Processing (3rd ed. draft)*", Dan Jurafsky and James H. Martin. Draft chapters in progress, August 28, 2017, `https://web.stanford.edu/~jurafsky/slp3/`

Przykład: Berkeley Restaurant Project c.d

Normalizacja za pomocą unigramów (podzielenie każdego wiersza liczb przez odpowiednią liczbę unigramów dla w_{n-1}):

- $\frac{\text{count}(I,want)}{\text{count}(I\text{wszystkie})}$
- $p(I|want) = \frac{827}{2533} = 0,3264$

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

Przykład z "*Speech and Language Processing (3rd ed. draft)*", Dan Jurafsky and James H. Martin. Draft chapters in progress, August 28, 2017, `https://web.stanford.edu/~jurafsky/slp3/`

Przykład: Berkeley Restaurant Project

Prawdopodobieństwa dla 8 wyrazów (ze słownika $|V| = 1446$):

	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
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

Przykład z "*Speech and Language Processing (3rd ed. draft)*", Dan Jurafsky and James H. Martin. Draft chapters in progress, August 28, 2017, `https://web.stanford.edu/~jurafsky/slp3/`

Przykład: Berkeley Restaurant Project c.d

Dodatkowe prawdopodobieństwa:
P(i|<s>) = 0,25
P(english|want) = 0,0011
P(food|english) = 0,5
P(</s>|food) = 0,68

Wyliczmy prawdopodobieństwo zdania: *I want English food*:
P(<s> i want english food </s>)
= P(i|<s>) · P(want|i)· P(english|want)·P(food|english)·
P(</s>|food)
= 0,25·0,33·0,011·0,5·0,68
= 0,000031

Przykład z "*Speech and Language Processing (3rd ed. draft)*", Dan Jurafsky and James H. Martin. Draft chapters in progress, August 28, 2017, `https://web.stanford.edu/~jurafsky/slp3/`

Zadanie

Ala idzie biegać
Tomek idzie biegać
Tomek idzie pływać
Tomek idzie biegać

$P(\text{Tomek}) = ?$
$P(\text{Ala}) = ?$
$P(\text{idzie}|\text{Tomek}) = ?$
$P(\text{biegać}|\text{idzie}) = ?$
$P(\text{biegać}|\text{Tomek idzie}) = ?$
$P_{\text{unigramy}}(\text{Tomek idzie biegać}) = ?$
$P_{\text{bigramy}}(\text{Tomek idzie biegać}) = ?$ (wykorzystaj <s>)

Zadanie - rozwiązanie

Ala idzie biegać
Tomek idzie biegać
Tomek idzie pływać
Tomek idzie biegać

$P(\text{Tomek}) = \frac{3}{12} = 0,25$
$P(\text{Ala}) = \frac{1}{12} = 0,08$
$P(\text{idzie}|\text{Tomek}) = \frac{3}{3} = 1$
$P(\text{biegać}|\text{idzie}) = \frac{3}{4} = 0,75$
$P(\text{biegać}|\text{Tomek idzie}) = \frac{2}{3} = 0,66$
$P_{\text{unigramy}}(\text{Tomek idzie biegać}) = \frac{3}{12} \cdot \frac{4}{12} \cdot \frac{3}{12} = 0,02$
$P_{\text{bigramy}}(\text{Tomek idzie biegać}) = \frac{3}{4} \cdot \frac{3}{3} \cdot \frac{3}{4} = 0,56$

Ewaluacja jakości modelu

Ewaluacja modeli językowych

Czy nasz model językowy szacuje większe prawdopodobieństwo dla "dobrych" zdań ("prawdziwych" lub "często obserwowalnych") w porównaniu do prawdopodobieństwa "złych"?
Metodologia (analogiczna do problemu uczenia nadzorowanego):

1. wytrenuj model na zbiorze trenującym
2. przetestuj model na zbiorze testowym (uprzednio nieuwzględnianych danych) za pomocą miary ewaluacji

Zewnętrzna ewaluacja modeli językowych

Najlepsza ewaluacja w celu porównania modeli językowych MJ_1 i MJ_2 to:

- włączenie ich do rozwiązania konkretnego zadania (korekta pisma, rozpoznawanie mowy itd.)
- wykonanie zadania, obliczenie wartości miary jakości (np. trafności klasyfikacji) dla MJ_1 i MJ_2
  - Jak wiele błędnie zapissanych wyrazów zostało prawidłowo poprawionych?
  - ...
- porównaj trafność dla MJ_1 i MJ_2

Zewnętrzna ewaluacja modeli językowych

A jeśli nie mamy na to czasu lub nie interesuje nas przydatność modelu w danym zadaniu a ewaluacja samego modelu?

Gra Shannona

Jak dobrze możemy przewidzieć następny wyraz?

Ja zawsze zamawiam pizzę z serem i ...

Gra Shannona

Jak dobrze możemy przewidzieć następny wyraz?

Ja zawsze zamawiam pizzę z serem i ...

grzybami 0.1

pepperoni 0.1

brokułem 0.01

...

prażonym ryżem 0.0001

...

i 1e-100

Lepszy model to ten, który przyznaje wyższe prawdopodobieństwo wyrazom jakie rzeczywiście występują jako następny

Czy unigramy dobrze sprawdzą się w tym zadaniu?

Gra Shannona

Jak dobrze możemy przewidzieć następny wyraz?

Ja zawsze zamawiam pizzę z serem i ...

grzybami 0.1

pepperoni 0.1

brokułem 0.01

...

prażonym ryżem 0.0001

...

i 1e-100

Lepszy model to ten, który przyznaje wyższe prawdopodobieństwo wyrazom jakie rzeczywiście występują jako następny

Czy unigramy dobrze sprawdzą się w tym zadaniu?

Gra Shannona

Jak dobrze możemy przewidzieć następny wyraz?

Ja zawsze zamawiam pizzę z serem i ...

grzybami 0.1

pepperoni 0.1

brokułem 0.01

...

prażonym ryżem 0.0001

...

i 1e-100

Lepszy model to ten, który przyznaje wyższe prawdopodobieństwo wyrazom jakie rzeczywiście występują jako następny

Czy unigramy dobrze sprawdzą się w tym zadaniu?

Wewnętrzna ewaluacja modeli językowych

Jaką miarą ewaluacji mierzyć jak dobrze model języka modeluje język (naturalny) lub dany korpus?

Nieokreśloność (*perplexity*)

Mając dane zdanie W o długości N, miarą jakości modelu języka dla tego zdania jest odwrotność prawdopodobieństwa zbioru testowego, znormalizowana przez długość N:

$$PP(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

Perplexity pokazuje stopień niepewności naszego modelu co do tego, że dane słowo pochodzi z modelowanego języka. Im mniejsza wartość, tym model jest lepszy.

Miara nieokreśloności (*perplexity*)

Reguła łańcuchowa $$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \ldots w_{i-1})}}$$

Dla bigramów $$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}}$$

Problem przeuczenia

- model n-gram sprawdza się gdy korpus testowy przypomina korpus trenujący
- często tak nie jest
- żeby wytrenować odporny model musimy uwzględnić rzeczy, które występują w zbiorze testowym, ale nie ma ich w zbiorze trenującym

Problem przeuczenia: zerowe prawdopodobieństwa

Zbiór trenujący:

- denied the allegations: 5
- denied the speculation: 2
- denied the rumors: 1
- denied the report: 1

Zbiór testowy:

- denied the offer
- denied the loan

$P(\text{offer}|\text{denied the})$ wynosi 0!

Problem przeuczenia: zerowe prawdopodobieństwa

Dwa problemy:

- niedoszacowujemy prawdopodobieństw wyrazów, które mogą wystąpić
- jeśli prawdopodobieństwo wyrazu w zbiorze testowym wynosi 0 to nie możemy policzyć wartości miary nieokreśloności (*perplexity*) (dzielenie przez 0)

Nieznane wyrazy

- słownik zamknięty: w zbiorze testowym umieszczamy wyrazy tylko ze słownika
- słownik otwarty: może wystąpić problem wyrazów spoza słownika: *Out of vocabulary (OOV)*, które możemy modelować za pomocą pseudo-wyrazu <UNK>

Nieznane wyrazy c.d.

Dwa sposoby radzenia sobie z nimi:

- przekształcenie problemu w taki ze słownikiem zamkniętym, wybierając a priori ustalony zasób wyrazów:
  1. wybierz słownik z ustaloną a priori listą wyrazów
  2. przekonwertuj w zbiórze trenującym wyrazy OOV na <UNK>
  3. oszacuj prawdopodobieństwa <UNK> z ich liczności tak jak dla regularnego wyrazu
- jeśli nie mamy zdefiniowanego słownika, można stworzyć go niejawnie poprzez zamianę wyrazów w zbiorze trenującym na <UNK> na podstawie ich częstości

Wygładzanie Laplace'a

Dodajemy 1 do liczby wystąpień, tak jak byśmy widzieli każdy wyraz jeden raz więcej

$$P_{Add-1}(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i) + 1}{count(w_{i-1}) + |V|}$$

Wygładzanie Laplace'a

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Dziękuję za uwagę!