Bài 4: Mô hình ngôn ngữ (Language Models)

AI Academy Vietnam

Giới thiệu về giảng viên

- TS. Nguyễn Minh Tiến
- Email: minhtienhy@gmail.com
- Mobile: 0983 860 318
- Research interest: natural language processing, information extraction.

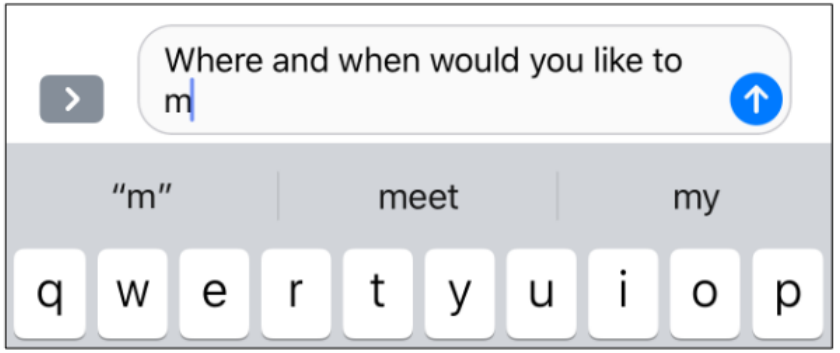
Nội dung

- Giới thiệu bài toán dự báo từ và sửa lỗi chính tả
- Mô hình hoá ngôn ngữ
  - Xác suất có điều kiện
  - Mô hình ngôn ngữ
  - Giả thuyết Markov
  - Mô hình ngôn ngữ *n*-grams
  - Ước lượng Likelihood
- Đánh giá mô hình
- Vấn đề số không và các phương pháp làm trơn
- Một số mô hình dựa trên mạng Neural
- Ứng dụng mô hình ngôn ngữ cho soát lỗi chính tả

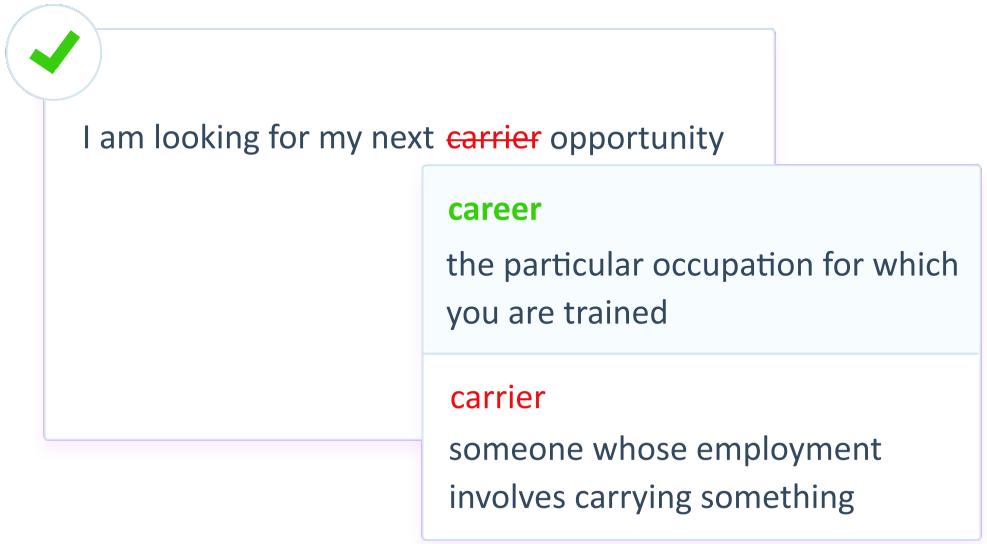
Bài toán dự báo từ và sửa lỗi chính tả

Giới thiệu bài toán dự báo từ và sửa lỗi chính tả

- **Dự báo từ**: Gợi ý từ/cụm từ tiếp theo trong quá trình soạn thảo.
- Ví dụ:
  - Soạn tin nhắn trên điện thoại di động

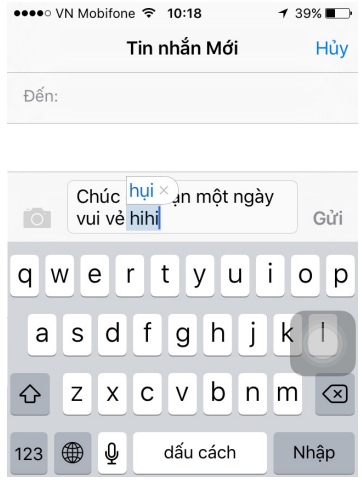


- **Sửa lỗi chính tả**: Tìm từ có thể sai chính tả và gợi ý từ thay thế cho đúng

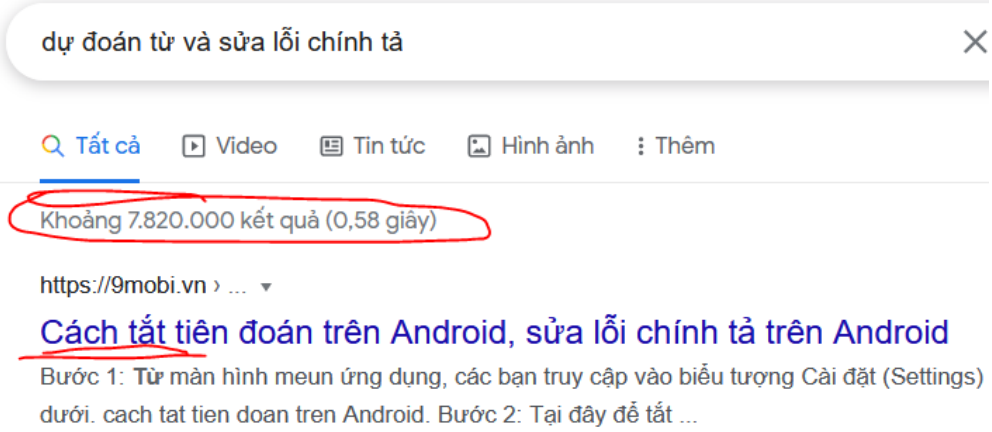


Ứng dụng

- Soạn thảo văn bản **nhanh** và **chính xác**
  - Người khuyết tật
  - Người không quen thao tác với các thiết bị điện tử
  - Người không thành thạo ngôn ngữ, tránh lỗi ngữ pháp
  - Tăng giá trị của sản phẩm soạn thảo văn bản.
- Tuy nhiên, gợi ý không chính xác có thể làm giảm niềm tin của người dùng



Trải nghiệm của người dùng là yếu tố quan trọng



Mô hình ngôn ngữ

Mô hình hoá ngôn ngữ

- Language modeling
- Mô hình ngôn ngữ là mô hình dự đoán xác xuất của một chuỗi các từ.
  - $P(W) = P(w_1, w_2, \ldots, w_n)$
  - Ví dụ
    - S_1 = “con mèo nhảy qua con chó”, $P(S_1) \sim 1$
    - S_2 = “qua con mèo con chó nhảy”, $P(S_2) \sim 0$

Ứng dụng

- Dịch máy
  - P(high winds tonight) > P (large winds tonight)
- Sửa lỗi văn bản
  - The office is about fifteen ***minuets*** from my house
  - P (“about fifteen ***minutes*** from”) > P(about fifteen minuets from)
- Nhận dạng giọng nói
  - P(I saw a van) > P(eyes awe of an)
- Nhận dạng chữ viết
  - P(Act naturally) > P(Abt naturally)
- Tóm tắt, hỏi – đáp, ..

Xác suất có điều kiện

$$P(A|B) = \frac{P(A \cap B)}{P(A)}$$

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

$$P(A, B, C, D) = P(A) \cdot P(B|A) \cdot P(C|A, B) \cdot P(D|A, B, C)$$

Xác suất có điều kiện

$$P(S) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_1, w_2) \dots P(w_n|w_1, w_2, , w_3, \dots, , w_n)$$

$$P(S) = \prod_{i}^{n} P(w_i|w_1, w_2, \dots, w_{i-1})$$

P(Computer,can,recognize,speech) = P(Computer)·
P(can|Computer)·
P(recognize|Computer can)·
P(speech|Computer can recognize)

Mô hình ngôn ngữ

- Mô hình ngôn ngữ = mô hình dự đoán từ
- Quy tắc dây chuyền

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

- Ví dụ
  - P("its water is so transparent") = P(its) × P(water|its) × P(is|its water) × P(so|its water is) × P(transparent|its water is so)

- Mô hình ngôn ngữ là mô hình dự đoán từ dựa trên các từ phía trước

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

Xác suất của từ

- Tính xác suất của từ trong ngữ cảnh dựa trên N-gram

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

- P(speech|Computer can recognize) = ?

$$P(speech | Computer\ can\ recognize) = \frac{\#(Computer\ can\ recognize\ speech)}{\#(Computer\ can\ recognize)}$$

Có vấn đề gì với cách tính trên?

Giả thuyết Markov

$$P(S) = \prod_{i-1}^{n} P(w_i | w_1, w_2, \dots, w_{i-1})$$

Andrei Markov

$$P(S) = \prod_{i-1}^{n} P(w_i | w_{i-1})$$

Giả thuyết Markov

P(Computer,can,recognize,speech) = P(Computer)·
P(can|Computer)·
P(recognize|Computer can)·
P(speech|Computer can recognize)

P(Computer,can,recognize,speech) = P(Computer)·
P(can|Computer)·
P(recognize|can)·
P(speech|recognize)

$$P\left(speech|recognize\right)=\frac{\#\left(recognize\ speech\right)}{\#\left(recognize\right)}$$

Mô hình n-gram

- **Unigram (1-gram):** xác suất của một từ không phụ thuộc vào từ phía trước

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

- **Bigram (2-gram):** xác suất một từ xuất hiện sau một từ cho trước

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

- **Trigram (3-gram):** xác suất một từ phụ thuộc vào 2 từ phía trước

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i | w_{i-1}, w_{i-2})$$

- **N-gram:** xác suất 1 từ phụ thuộc vào *N* từ phía trước

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i | w_{i-1}, w_{i-2}, w_{i-N})$$

Ước lượng xác suất

- Sử dụng Maximum Likelihood Estimate

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

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

Ví dụ

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

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

$$P(\texttt{I} \mid \texttt{<s>}) = \frac{2}{3} = .67 \qquad P(\texttt{Sam} \mid \texttt{<s>}) = \frac{1}{3} = .33 \qquad P(\texttt{am} \mid \texttt{I}) = \frac{2}{3} = .67$$

$$P(\texttt{</s>} \mid \texttt{Sam}) = \frac{1}{2} = 0.5 \qquad P(\texttt{Sam} \mid \texttt{am}) = \frac{1}{2} = .5 \qquad P(\texttt{do} \mid \texttt{I}) = \frac{1}{3} = .33$$

Thống kê trên dữ liệu

- Berkeley Restaurant Project (BeRP)
- BeRP chứa các câu tư vấn trong lĩnh vực nhà hàng thành phố Berkeley, California
- Chứa 9222 câu. Ví dụ:
  - 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
  - ...

Đếm đồng xuất hiện

	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

Xác xuất Bigram

- Chuẩn hoá bằng unigrams:

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

- Kết quả

	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

Ví dụ

Cho tập dữ liệu văn bản gồm các câu sau:

<s> *cô ấy dạy môn tin học* </s>

<s> *anh dạy môn toán* </s>

<s> *cô ấy học toán anh ấy dạy* </s>

<s> *môn toán* môn tin đều hay </s>

<s> *anh ấy dạy môn toán hay môn tin* </s>

Xây dựng mô hình ngôn ngữ unigram và bigram?

Đánh giá mô hình

Đánh giá ngoài

- **Extrinsic evaluation**
- Mô hình ngôn ngữ A và B được sử dụng trong một bài toán X khác:
  - Bài toán speech recognition, spelling, machine translation
- So sánh mô hình A với mô hình B tương ứng với so sánh kết quả ứng dụng A với B trong bài toán X.

Đánh giá trong

- **Intrinsic evaluation**
- Sử dụng dữ liệu Test là các câu trong ngôn ngữ
- Sử dụng độ đo Perplexity

Độ đo Perplexity

- Mô hình ngôn ngữ là mô hình dự đoán tốt nhất trên dữ liệu mới
  - Dự đoán xác suất cao nhất cho câu $P(w_1 w_2 \ldots w_n)$
  - Perplexity là xác suất nghịch đảo của tập kiểm tra, được chuẩn hóa bởi số lượng từ

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}}$$

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

- Đối với bigrams:

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

Cực tiểu perplexity tương đương với việc cực đại xác suất

Perplexity

- **Perplexity thấp = mô hình tốt**
- Bộ dữ liệu WSJ
  - Tập huấn luyện 38 triệu từ, kiểm tra 1.5 triệu từ

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

Phương pháp trực quan hóa Shannon

- Chọn ngẫu nhiên một bigram

  (<s>, w) theo xác suất của nó

- Tiếp tục chọn ngẫu nhiên bigram (w, x) theo xác suất của nó

- Chọn liên tục như vậy đến khi gặp từ </s>

- Nối các từ thu được tạo thành choỗi

```
<s> I
    I want
      want to
           to eat
              eat Chinese
                  Chinese food
                          food  </s>
I want to eat Chinese food
```

Giá trị *n*

Giá trị phù hợp của *n là bao nhiêu*?

- Về mặt lý thuyết, rất khó xác định
- Tuy nhiên: nhiều nhất có thể (→ tiệm cận với mô hình "hoàn hảo")
- Về mặt thực nghiệm, phổ biến *n* = ***3***
  - Ước lượng tham số? (độ tin cậy, dữ liệu, lưu trữ, không gian, ...)
  - 4 là quá lớn: |V|=60k → 1.296 x 1019 tham số
  - nhưng: 6-7 có thể nếu có đủ dữ liệu: *trong thực tế, chúng ta có thể khôi phục bản gốc từ 7-grams!*

Vấn đề số 0 và các phương pháp làm mịn

Vấn đề bằng 0

- Tập huấn luyện:
  - … denied the allegations
  - … denied the reports
  - … denied the claims
  - … denied the request

- P(“offer” | denied the) = 0

- Tập kiểm tra

… denied the offer

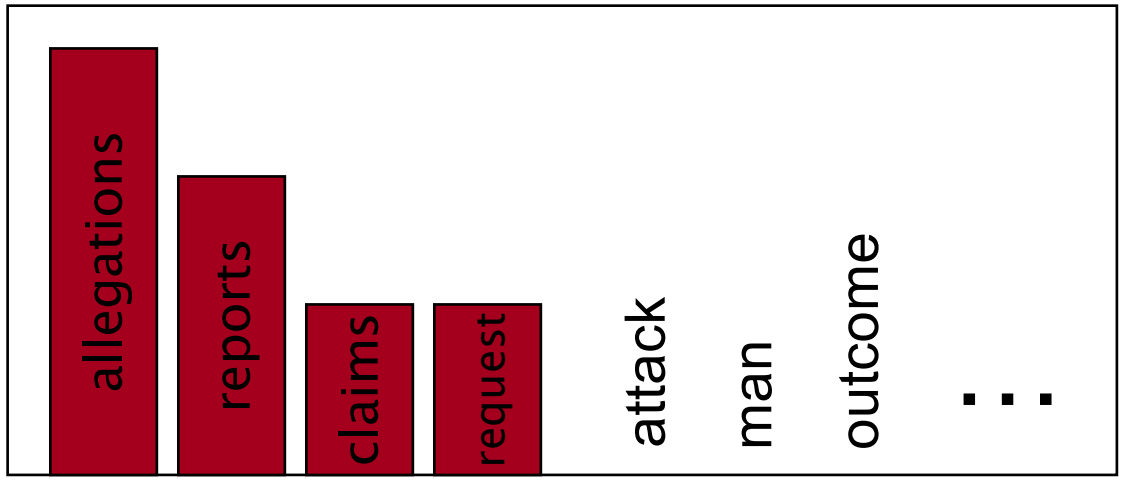
… denied the loan

Điều này có nghĩa rằng chúng ta sẽ gán xác suất bằng 0 cho câu trên

Vấn đề làm mịn

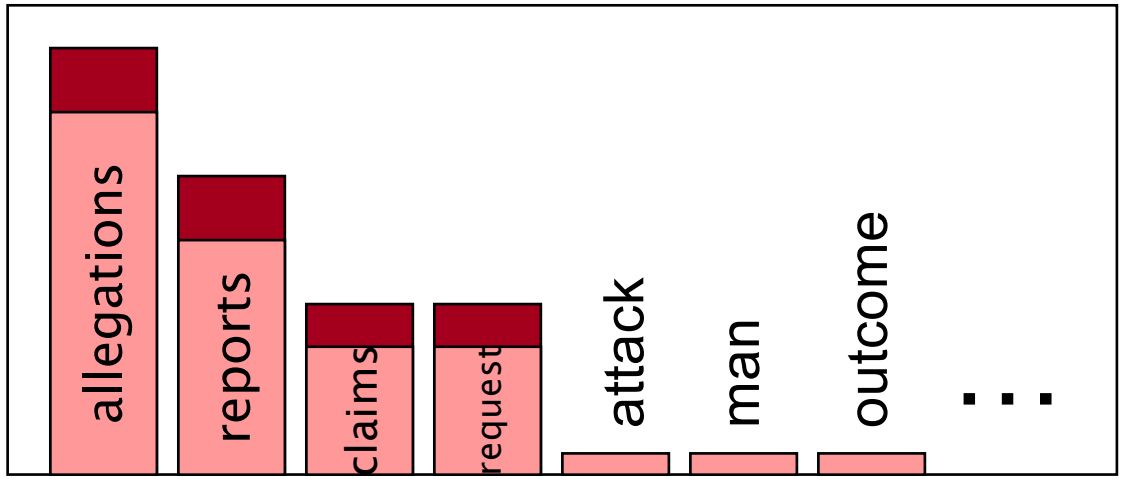
- Khi chúng ta có thống kê thưa:

P(w | denied the)
3 allegations
2 reports
1 claims
1 request



- Điều chỉnh giá trị để tổng quát hóa tốt hơn

P(w | denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other



Ước lượng add-one

- Còn gọi là làm mịn Laplace
- Giả định rằng chúng ta nhìn thấy mỗi từ nhiều hơn 1 lần so với quan sát
- Do đó chúng ta cộng một vào tất cả các lượng đếm được

- Ước lượng MLE:

$$P_{MLE}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

- Ước lượng add-1:

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

Kho dữ liệu Berkeley Restaurant: lượng đếm theo làm mịn bigram Lapıace

	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

Bigrams làm mịn theo Laplace

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

	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

So với xác suất ban đầu

	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

	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

Quay lui và nội suy

- Trong một số trường hợp, việc sử dụng ít ngữ cảnh có thể hiệu quả
  - Điều kiện với ít ngữ cảnh cho các ngữ cảnh mới là chúng ta chưa học nhiều về nó
- **Quay lui:**
  - Sử dụng trigram nếu chúng ta có căn cứ tốt,
  - ngược lại bigram, ngược lại unigram
- **Nội suy:**
  - Kết hợp unigram, bigram, trigram

- Nội suy thường hiệu quả hơn quay lui

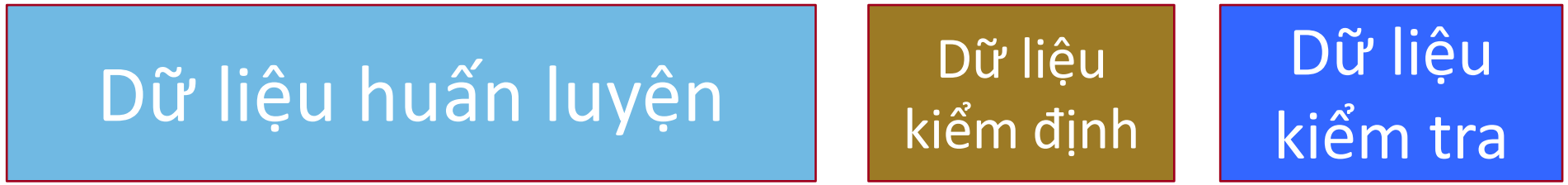
Nội suy tuyến tính

- Nội suy đơn giản $\hat{P}(w_n|w_{n-1}w_{n-2}) = \lambda_1 P(w_n|w_{n-1}w_{n-2}) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n)$ $\sum_i \lambda_i = 1$

- Điều kiện Lambdas với ngữ cảnh:

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1(w_{n-2}^{n-1})P(w_n|w_{n-2}w_{n-1}) + \lambda_2(w_{n-2}^{n-1})P(w_n|w_{n-1}) + \lambda_3(w_{n-2}^{n-1})P(w_n)$$

Chọn Lamda



- Sử dụng bộ dữ liệu kiểm định
- Chọn các giá trị để cực đại xác suất trên bộ kiểm định:
  - Giữ nguyên xác suất N-gram (trên tập dữ liệu huấn luyện)
  - Sau đó tìm các λ sao cho nó có xác suất lớn nhất trên bộ kiểm định:

$$\log P(w_1 \ldots w_n \mid M(\lambda_1 \ldots \lambda_k)) = \sum_i \log P_{M(\lambda_1 \ldots \lambda_k)}(w_i \mid w_{i-1})$$

Từ vựng mở và đóng

- Nếu chúng ta biết trước toàn bộ các từ
  - Bộ từ vựng V là cố định; Đây là bài toán từ vựng đóng
- Thông thường, chúng ta không biết hết các từ
  - **Các từ ngoài từ điển** OOV (**Out Of Vocabulary**)
  - Đây là bài toán từ vựng mở
- Thay vào đó: tạo một từ không biết <UNK>
  - Huấn luyện xác suất của từ không biết <UNK>
    - Tạo một bộ từ vựng cố định L có kích thước V
    - Tại giai đoạn chuẩn hóa văn bản, các từ trong tập huấn luyện không thuộc L được đổi thành <UNK>
    - Tiếp tục chúng ta huấn luyện xác suất của nó như từ thông thường
  - Tại thời điểm giải mã
    - Nếu văn bản đầu vào: sử dụng xác suất UNK cho các từ không thuộc L

Bộ dữ liệu *n*-grams

- Giải quyết vấn đề giá trị lớn, ví dụ, kho Google N-gram
- Cắt tỉ
  - Chỉ lưu N-grams với tuần suất > ngưỡng (threshold).
    - Xóa bỏ các bộ của n-gram bậc cao hơn
- Hiệu quả
  - Sử dụng các cấu trúc dữ liệu hiệu quả như trie
  - Các bộ lọc Bloom: xấp xỉ mô hình ngôn ngữ
  - Lưu trữ các từ như là chỉ số, không phải choỗi
    - Sử dụng mã Huffman để chuyển số lớn của các từ thành 2 byte
  - Lượng hóa xác suất (4-8 bits thay vì kiểu 8 byte float)

Làm mịn bộ dữ liệu lớn N-grams

- Thuật toán “Stupid backoff” (Brants *et al*. 2007)
- Sử dụng tần số tương quan

$$S(w_i \mid w_{i-k+1}^{i-1}) = \begin{cases} \dfrac{\text{count}(w_{i-k+1}^{i})}{\text{count}(w_{i-k+1}^{i-1})} & \text{if} \quad \text{count}(w_{i-k+1}^{i}) > 0 \\ 0.4 S(w_i \mid w_{i-k+2}^{i-1}) & \text{otherwise} \end{cases}$$

$$S(w_i) = \frac{\text{count}(w_i)}{N}$$

Các thuật toán làm mịn nâng cao

- Một số thuật toán làm mịn cho kết quả tốt
  - Good-Turing
  - Kneser-Ney
  - Witten-Bell
- Đếm những bộ chúng ta chỉ thấy 1 lần
  - Giúp ước lượng để đếm những bộ chưa được nhìn thấy

Một số mô hình ngôn ngữ dùng mạng neural

N lớn bao nhiêu?

- Về mặt lý thuyết không gì là đủ
- Tuy nhiên: lớn nhất có thể (→ gần mô hình “hoàn hảo”)
- Thực nghiệm: ***3***
  - Ước lượng tham số? (tin cậy, dữ liệu, lưu trữ, không gian, ...)
  - 4 là quá lớn: |V|=60k → 1.296 x 1019 tham số
  - nhưng: 6-7 có thể là lý tưởng (có đủ dữ liệu): thực tế, một bản gốc có thể khôi phục từ *7-grams!*

Hạn chế của mô hình ngôn ngữ N-gram

- Khi dữ liệu thưa thì mô hình không chính xác vì các tần suất N-gram không đại diện.
- Mô hình N-gram càng chính xác khi N càng lớn, tuy nhiên khi N lớn thì số lượng N-gram rất lớn và không thực thi được do hạn chế về bộ nhớ và tính toán.
  - Không biểu diễn được phụ thuộc xa, ví dụ:
    - “Hùng sống ở Pháp hồi nhỏ nên anh ấy có thể nói tiếng … khá thạo”
    - “The girl that I met in the train was ...”

Các mô hình ngôn ngữ mạng nơ ron

- Mô hình ngôn ngữ mạng neural - NNLM (Bengio, 2003)
- Mạng hồi quy NNLM (Mikolov, 2010)
- Các mô hình mới: Transformer (2018)

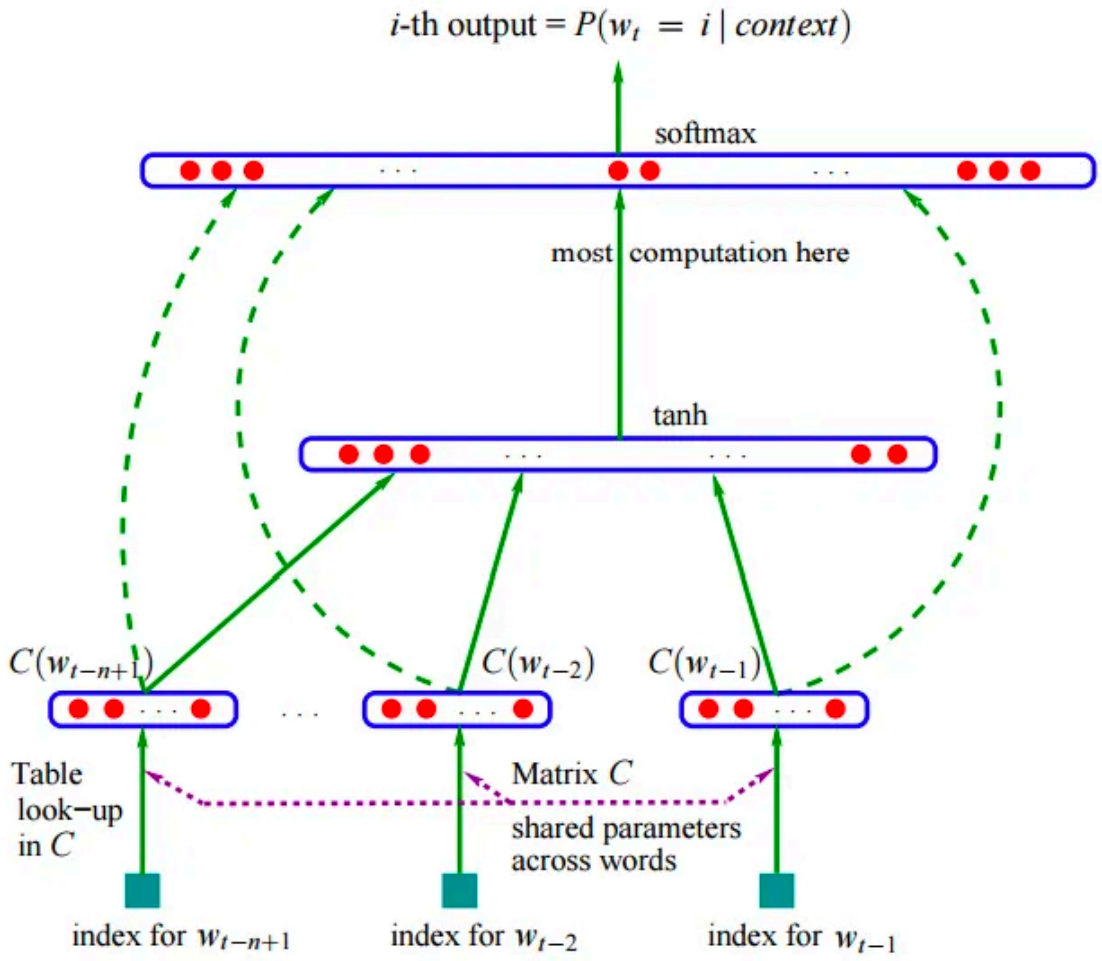
Biểu diễn từ bằng vector

- Các mô hình dựa trên NN sử dụng Word2Vec có tính tổng quát cao.

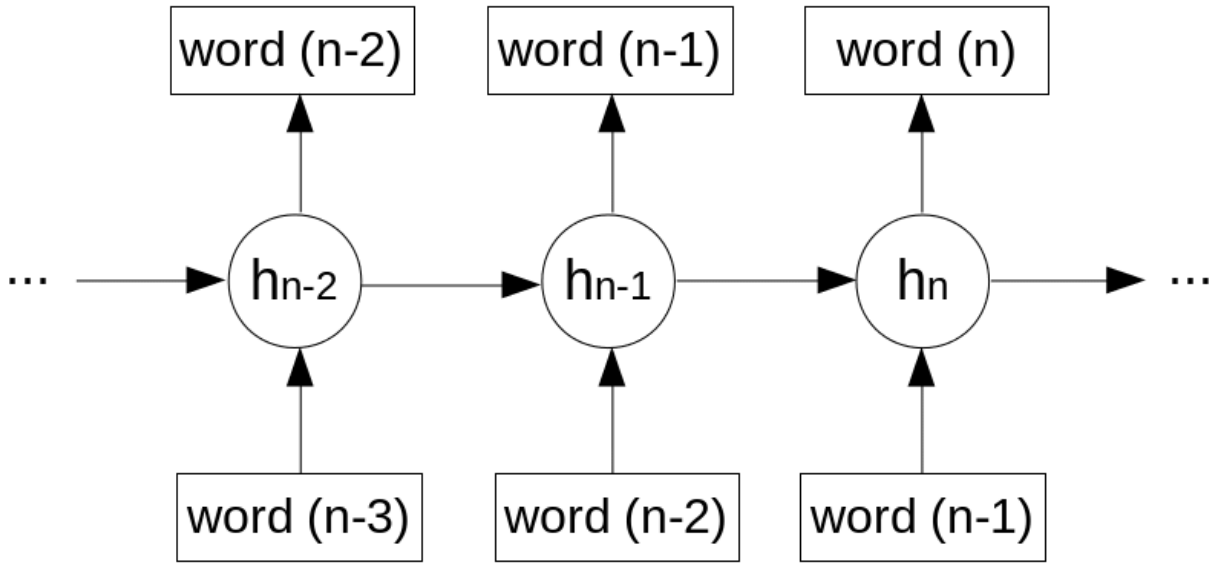
Dimensions

Word vectors					
dog	-0.4	0.37	0.02	-0.34	animal
cat	-0.15	-0.02	-0.23	-0.23	domesticated
lion	0.19	-0.4	0.35	-0.48	pet
tiger	-0.08	0.31	0.56	0.07	fluffy
elephant	-0.04	-0.09	0.11	-0.06	
cheetah	0.27	-0.28	-0.2	-0.43	
monkey	-0.02	-0.67	-0.21	-0.48	
rabbit	-0.04	-0.3	-0.18	-0.47	
mouse	0.09	-0.46	-0.35	-0.24	
rat	0.21	-0.48	-0.56	-0.37	

Mô hình ngôn ngữ mạng nơ ron



Mô hình ngôn ngữ mạng nơ ron hồi quy

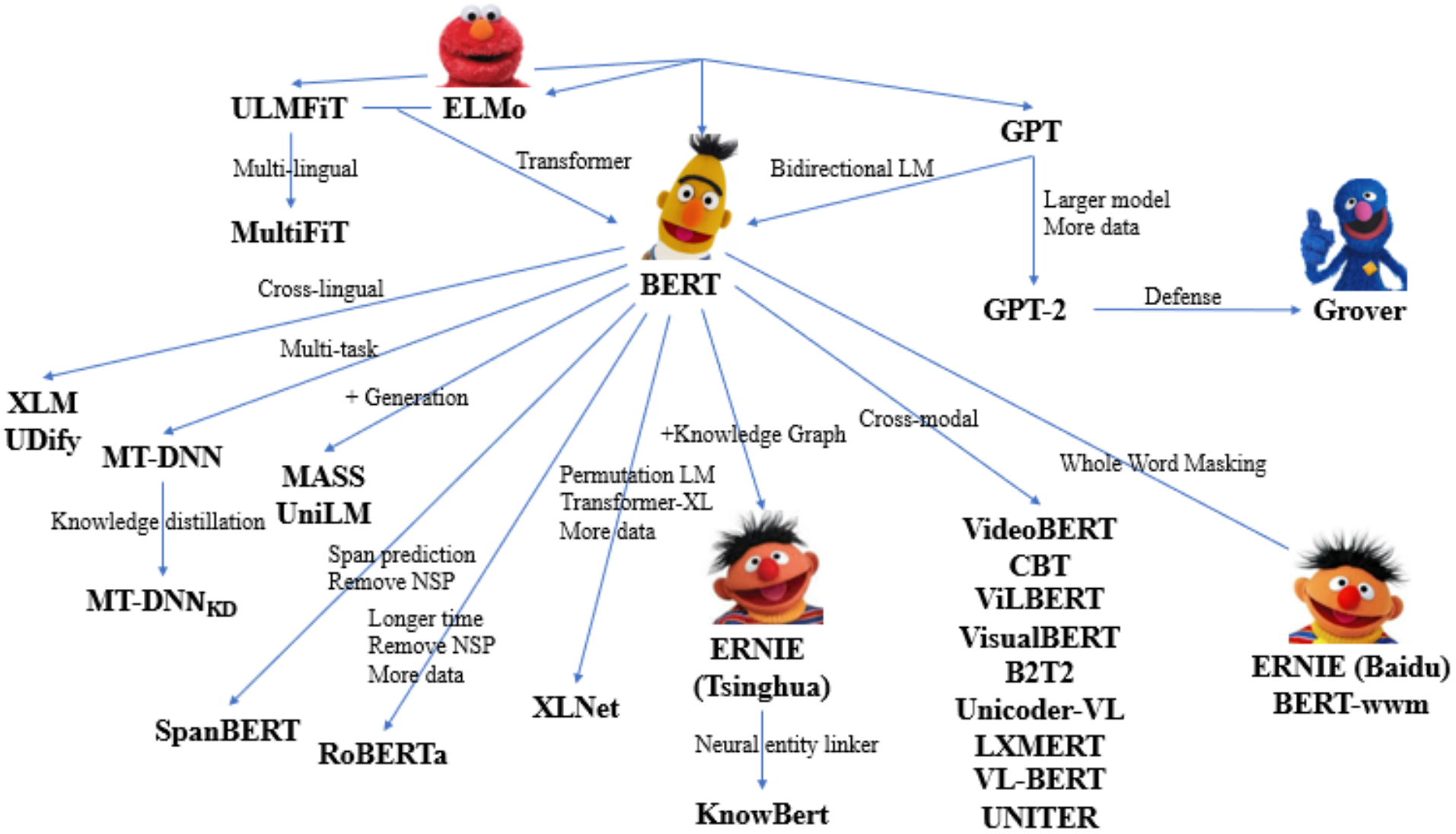


Một số kết quả so sánh

Language Model	$H(H_c)$	PPL	WER
KN5	-	248.0	12.8
RNN	200 (-)	226.2	12.0
RNN-BOW	190 (10)	**218.8**	**11.7**
RNN+KN5	200 (-)	191.6	11.8
RNN-BOW+KN5	190 (10)	**183.0**	**11.3**

RNN-BOW LM to combine short term (RNN) and long term (BOW) information (Haidar & Kurimo, 2016)

Các mô hình LM huấn luyện trước



Mô hình đã huấn luyện cho tiếng Việt (PhoBERT)

- PhoBERT
  - Dựa trên RoBERTa huấn luyện theo thủ tục tương tự như BERT.
  - Có 2 phiên bản “base” & “large”

Model	#params	Arch.	Pre-training data
vinai/phobert-base	135M	base	20GB of texts
vinai/phobert-large	370M	large	20GB of texts

  - PhoBERT vượt trội các Phương pháp cho nhiều bài toán NLP: POS tagging, NER, NLI

Kết quả cho 1 số bài toán NLP

NER (word-level)	
Model	F_1
BiLSTM-CNN-CRF [♦]	88.3
VnCoreNLP-NER (Vu et al., 2018) [♦]	88.6
VNER (Nguyen et al., 2019b)	89.6
BiLSTM-CNN-CRF + ETNLP [♠]	91.1
VnCoreNLP-NER + ETNLP [♠]	91.3
XLM-R_{base} (our result)	92.0
XLM-R_{large} (our result)	92.8
$PhoBERT_{base}$	93.6
$PhoBERT_{large}$	**94.7**

NLI (syllable- or word-level)	
Model	Acc.
–	–
BiLSTM-max (Conneau et al., 2018)	66.4
mBiLSTM (Artetxe and Schwenk, 2019)	72.0
multilingual BERT (Devlin et al., 2019) [■]	69.5
$XLM_{MLM+TLM}$ (Conneau and Lample, 2019)	76.6
XLM-R_{base} (Conneau et al., 2020)	75.4
XLM-R_{large} (Conneau et al., 2020)	79.7
$PhoBERT_{base}$	78.5
$PhoBERT_{large}$	**80.0**

Một số ví dụ và kết quả

Model	*Ground Truth*	*User Input*	*Model Prediction*
Transformer [13]	could you try ringing her is that ok thanks i will yes i am playing is not can i call you no material impact	couks you tru ringing her is that ok thanka i will yew i am playing if not can i call you no material impact	coucks your ringhing ing is that on thank i will yew i amplaying if not an cally no material micat
LM+SpellCheck	could you try ringing her is that ok thanks i will yes i am playing is not can i call you no material impact	couks you tru ringing her is that ok thanka i will yew i am playing if not can i call you no material impact	coke you ' ringing her is that ok hanka i will yes i am playing if not can i call you no material impact
CCEAD (ours)	could you try ringing her is that ok thanks i will yes i am playing is not can i call you no material impact	couks you tru ringing her is that ok thanka i will yew i am playing if not can i call you no material impact	could you try ringing her is that ok thanks i will yew i am playing if not can i call you no material impact

Ứng dụng mô hình ngôn ngữ cho soát lỗi chính tả

Ứng dụng mô hình ngôn ngữ cho soát lỗi chính tả

- Sử dụng n-gram
- Dùng mô hình Seq2Seq để sửa lỗi

Tổng kết

- Mô hình ngôn ngữ quan trọng, có nhiều ứng dung.
- Mô hình ngôn ngữ dựa vào *N*-gram
- Đánh giá mô hình
- Xử lý vấn đề zero và các phương pháp smoothing
- Mô hình hiện đại dựa vào mạng neural có hiệu quả hơn

Ký hiệu: N_c = Tần suất của tần suất c

- N_c = đếm những bộ chúng ta thấy c lần
- Ví dụ: Sam I am I am Sam I do not eat

```
I    3
sam  2
am   2
do   1
not  1
eat  1
```

$N_1 = 3$

$N_2 = 2$

$N_3 = 1$

Làm mịn Good-Turing

- Bạn đang câu cá và bắt được
  - 10 cá chép, 3 cá rô, 2 cá trắng, 1 cá trout, 1 cá hồi, 1 lươn = 18 cá
- Khả năng con tiếp theo là cá trout?
  - 1/18
- Khả năng tiếp theo là loài mới (i.e. cá trê hoặc cá vược)
  - Sử dụng ước lượng những thứ chúng ta thấy 1 lần để ước lượng cái mới.
  - 3/18 (vì N_1=3)
- Giả sử như vậy, xác suất của loài tiếp theo là trout?
  - Phải nhỏ hơn 1/18
  - Làm sao để ước lượng?

Làm mịn Good Turing

$$P^*_{GT}(\text{things with zero frequency}) = \frac{N_1}{N}$$

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

- Chưa thấy (các vược hoặc cá trê)
  - c = 0:
  - MLE p = 0/18 = 0
  - P^*_{GT} (chưa thấy) = N_1/N = 3/18
- Thấy 1 lần (cá trout)
  - c = 1
  - MLE p = 1/18
  - C*(trout) = 2 * N2/N1
    = 2 * 1/3
    = 2/3
  - P^*_{GT}(trout) = 2/3 / 18 = 1/27

Kết quả tính toán Good-Turing

- Các số từ Church and Gale (1991)
- 22 triệu từ của AP Newswire

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

Count c	Good Turing c*
0	.0000270
1	0.446
2	1.26
3	2.24
4	3.24
5	4.22
6	5.19
7	6.21
8	7.24
9	8.25

Nội suy trừ hao tuyệt đối

- Trừ đi một lượng nào đó (ví dụ 0.75)

Bigram được trừ hao

Trọng số nội suy

$$P_{\text{AbsoluteDiscounting}}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) - d}{c(w_{i-1})} + \lambda(w_{i-1})P(w)$$

unigram

  - Có thể giữ giá trị của d cho các tần suất 1 và 2
- Nhưng chỉ ử dụng cho unigram P(w)?

Làm mịn Kneser-Ney I

- Ước lượng tốt hơn cho xác suất của bậc thấp unigrams!
  - Trò chơi Shannon: *I can't see without my reading*_____? *glasses*
  - "Francisco" là phổ biến hơn "glasses"
  - ... nhưng "Francisco" luôn luôn theo sau "San" *Francisco*
- unigram rất hữu ích nếu chúng ta chưa quan sát được bigram này!
- Thay vì P(w): "Khả năng là w"
- $P_{continuation}$(w): "Khả năng của w xuất hiện tiếp theo của câu truyện?
  - Với mỗi từ, đếm số lượng kiểu bigram đầy đủ
  - Mọi kiểu bigram tiếp tục của câu truyện mà ta đã thấy lần đầu

$$P_{CONTINUATION}(w) \propto \left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|$$

Làm mịn Kneser-Ney II

- Bao nhiêu lần w xuất hiện tiếp theo của câu truyện:

$$P_{CONTINUATION}(w) \propto \left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|$$

- Chuẩn hóa bởi tổng số kiểu bigram

$$\left|\{(w_{j-1}, w_j) : c(w_{j-1}, w_j) > 0\}\right|$$

$$P_{CONTINUATION}(w) = \frac{\left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|}{\left|\{(w_{j-1}, w_j) : c(w_{j-1}, w_j) > 0\}\right|}$$

Làm mịn Kneser-Ney III

- Số lượng các kiểu từ nhìn thấy trước w

$$|\{w_{i-1} : c(w_{i-1}, w) > 0\}|$$

- Chuẩn hóa bởi số từ trước tất cả các từ:

$$P_{CONTINUATION}(w) = \frac{\left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|}{\sum_{w'} \left|\{w'_{i-1} : c(w'_{i-1}, w') > 0\}\right|}$$

- Một từ thường xuyên (Francisco) xuất hiện trong 1 ngữ cảnh (San) luôn có xác suất tiếp theo thấp

Làm mịn Kneser-Ney IV

$$P_{KN}(w_i \mid w_{i-1}) = \frac{\max(c(w_{i-1}, w_i) - d, 0)}{c(w_{i-1})} + \lambda(w_{i-1}) P_{CONTINUATION}(w_i)$$

λ là một hằng số chuẩn hóa; lượng xác suất mà chúng ta trừ hao

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} \left| \{w : c(w_{i-1}, w) > 0\} \right|$$

Trừ hao chuẩn hóa

Số lượng các kiểu từ mà có xuất hiện thể sau w_{i-1}
= # các kiểu từ được trừ hao
= # số lần chúng ta áp dụng chuẩn hóa trừ hao

Làm mịn Kneser-Ney: Công thức đệ quy

$$P_{KN}(w_i \mid w_{i-n+1}^{i-1}) = \frac{\max(c_{KN}(w_{i-n+1}^{i}) - d, 0)}{c_{KN}(w_{i-n+1}^{i-1})} + \lambda(w_{i-n+1}^{i-1}) P_{KN}(w_i \mid w_{i-n+2}^{i-1})$$

$$c_{KN}(\bullet) = \begin{cases} count(\bullet) & \text{for the highest order} \\ continuationcount(\bullet) & \text{for lower order} \end{cases}$$

Continuation count = Số ngữ cảnh từ đơn duy nhất cho $\bullet$