N-gram Language Models

CS221.Q21.KHTN - Xử lý ngôn ngữ tự nhiên

Bùi Ngọc Thiên Thanh - 23521436
Cáp Kim Hải Anh - 23520036
Hoàng Đức Dũng - 23520328
Nguyễn Thái Sơn - 23521356

Khoa Khoa học Máy tính
Trường Đại học Công nghệ thông tin
VNU-HCM

Tháng 2, 2026

Tables of contents

Table of Contents

Giới thiệu

- Con người có khả năng tự nhiên dự đoán từ tiếp theo trong câu.
- Ví dụ:

  "The water of Walden Pond is so beautifully _____"
- Các từ có khả năng xuất hiện:
  - clear, blue, green
- Các từ ít hợp lý:
  - refrigerator hoặc các từ ngẫu nhiên
- $\Rightarrow$ Ngôn ngữ không hoàn toàn ngẫu nhiên mà có cấu trúc và quy luật.

Language Model (Mô hình ngôn ngữ) là gì?

- **Language Model (LM)** là mô hình dùng để:
  - Dự đoán từ tiếp theo trong câu
  - Đánh giá mức độ tự nhiên của một chuỗi từ
- Mô hình học các mẫu (patterns) trong ngôn ngữ từ dữ liệu văn bản lớn.
- Ví dụ:

Câu tự nhiên

all of a sudden I notice three guys standing on the sidewalk

Câu kém tự nhiên

on guys all I of notice sidewalk three a sudden standing the

Vai trò của việc dự đoán từ

- Nhiều hệ thống AI hiện đại hoạt động dựa trên:

  Next-word prediction (dự đoán từ tiếp theo)

- Ứng dụng:
  - Kiểm tra ngữ pháp (Grammar correction)
  - Nhận dạng giọng nói (Speech recognition)
  - Gợi ý khi nhập văn bản (Text autocomplete)
  - Hệ thống hỗ trợ giao tiếp AAC (Augmentative and Alterna tive Communication).
- Ví dụ:
  - There are > Their are
  - has improved > has improve

Table of Contents

N-gram là gì? (What is an N-gram?)

- **N-gram** là chuỗi gồm n từ xuất hiện liên tiếp trong văn bản.
- Đây là cách đơn giản để biểu diễn ngữ cảnh (context) của từ trong câu.
- Ví dụ:
  - Unigram ($n = 1$): The
  - Bigram ($n = 2$): The water
  - Trigram ($n = 3$): The water of

Ý tưởng chính của N-gram

- Khi con người hiểu ngôn ngữ, các từ gần nhau thường có liên hệ mạnh.
- N-gram mô tả câu bằng các nhóm từ liên tiếp thay vì toàn bộ câu.
- Ví dụ câu:

$$\text{The water of Walden Pond}$$

- Bigram:

$$\text{The water} \rightarrow \text{water of} \rightarrow \text{of Walden} \rightarrow \text{Walden Pond}$$

- Trigram:

$$\text{The water of} \rightarrow \text{water of Walden} \rightarrow \text{of Walden Pond}$$

Vì sao sử dụng N-gram?

- Giúp mô hình hoá ngôn ngữ theo cách đơn giản và trực quan.
- Cho phép máy tính học các mẫu xuất hiện phổ biến của từ.
- Là bước khởi đầu để xây dựng các Language Model phức tạp hơn.

Ý tưởng cốt lõi

Ngữ cảnh gần thường mang nhiều thông tin quan trọng cho việc hiểu ngôn ngữ.

Table of Contents

Bài toán cơ bản (Core Problem)

- Mục tiêu: tính xác suất của một từ w dựa trên lịch sử (history) h:

$$P(w|h)$$

- Ví dụ:

History: *"The water of Walden Pond is so beautifully"*

- Ta muốn tính:

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

Ước lượng bằng tần suất (Relative Frequency Estimation)

- Một cách đơn giản:
  - Dùng corpus (tập văn bản lớn)
  - Đếm số lần xuất hiện
- Ý tưởng:

  Trong tất cả các lần xuất hiện history h, bao nhiêu lần được theo sau bởi từ w?

$$P(w|h) = \frac{C(h, w)}{C(h)}$$

Ví dụ cụ thể

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

- $C(\cdot)$: số lần xuất hiện (Count)
- Ước lượng dựa trên thống kê thực nghiệm (empirical statistics)

Vấn đề của phương pháp đếm trực tiếp

- Ngôn ngữ có tính sáng tạo (Language is creative).
- Rất nhiều câu:
  - chưa từng xuất hiện trong corpus
  - hoặc xuất hiện cực kỳ hiếm
- Ngay cả toàn bộ Internet cũng:
  - Không đủ dữ liệu cho mọi câu hoàn chỉnh.
- $\Rightarrow$ Không thể ước lượng xác suất từ toàn bộ lịch sử dài.

Ký hiệu sử dụng (Notation)

- Chuỗi n từ:

$$w_1, w_2, ..., w_n \quad \text{hoặc} \quad w_{1:n}$$

- Lịch sử trước từ n:

$$w_{1:n-1} \quad \text{hoặc} \quad w_{<n}$$

- Xác suất đồng thời (Joint Probability):

$$P(w_1, w_2, ..., w_n)$$

- Trong thực tế, mô hình thường làm việc trên tokens (BPE tokens) thay vì words.

Chain Rule of Probability (Quy tắc dây chuyền)

- Ta phân rã xác suất chuỗi bằng Chain Rule:

$$P(X_1 ... X_n) = \prod_{k=1}^{n} P(X_k | X_{1:k-1})$$

- Áp dụng cho ngôn ngữ:

$$P(w_{1:n}) = \prod_{k=1}^{n} P(w_k | w_{1:k-1})$$

Ý nghĩa của Chain Rule

- Xác suất cả câu = tích của các xác suất dự đoán từng từ

$$P(w_1)P(w_2|w_1)P(w_3|w_{1:2})...P(w_n|w_{1:n-1})$$

- Đây chính là nền tảng của:
  - Language Modeling
  - Large Language Models (LLMs)

Vấn đề còn tồn tại

- Trong thực tế:
  - Toàn bộ lịch sử $w_{1:n-1}$ có thể rất dài
  - Số lượng ngữ cảnh tăng theo cấp số nhân
  - Nhiều chuỗi chưa từng xuất hiện trong corpus
- Ngôn ngữ có tính **sáng tạo** (language is creative).
- $\Rightarrow$ Không thể ước lượng chính xác bằng cách đếm trực tiếp.

Chain rule đúng về mặt toán học, nhưng khó áp dụng trong thực tế.

Ý tưởng dẫn đến N-gram

Key Insight

Thay vì sử dụng toàn bộ lịch sử dài, ta chỉ xét một phần ngữ cảnh gần nhất.

- Mục tiêu:
  - Đơn giản hóa việc ước lượng xác suất
  - Giảm số lượng ngữ cảnh cần quan sát
- Ý tưởng này dẫn đến:

Mô hình N-gram

Nội dung

Giả định Markov là gì?

- **Ý tưởng:** Thay vì nhìn lại quá khứ xa xôi, chỉ nhìn vào vài từ gần nhất.
- **Giả định:** Tương lai chỉ phụ thuộc vào hiện tại (ngữ cảnh cục bộ).

"Đừng để quá khứ ảnh hưởng quá nhiều đến tương lai"

Công thức Xấp xỉ (Approximation)

Thay vì tính toán trên toàn bộ chuỗi $w_{1:n-1}$:

$$P(w_n|w_1, w_2, ..., w_{n-1})$$

Ta sử dụng phép xấp xỉ Markov:

$$\approx P(w_n|w_{n-k}, ..., w_{n-1})$$

- Chỉ quan tâm đến k từ đứng ngay trước đó.

Nội dung

Mô hình Bigram (2-gram)

Trường hợp đơn giản nhất: Chỉ nhìn lại **1 từ** $(N-1)$.

Công thức Bigram

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-1})$$

Ví dụ: *"I want to eat pizza"*

- **Thực tế:** $P(\text{pizza}|\text{I want to eat})$
- **Bigram:** $P(\text{pizza}|\text{eat})$
- $\rightarrow$ Chỉ cần biết từ "eat"để đoán "pizza".

Tổng quát hóa N-gram

- **Bigram ($N = 2$):** Nhìn lại 1 từ.
- **Trigram ($N = 3$):** Nhìn lại 2 từ.
- **4-gram ($N = 4$):** Nhìn lại 3 từ.

Ví dụ: *"Vui lòng không hút thuốc"* (Dự đoán từ "thuốc")

Mô hình	**Ngữ cảnh sử dụng**	**Xấp xỉ**
Bigram	hút	$P(\text{thuốc}\|\text{hút})$
Trigram	không hút	$P(\text{thuốc}\|\text{không hút})$

Vấn đề khi N-gram có quy mô lớn

Trong thực tế, mô hình N-gram có thể rất lớn, gây ra:

- Xác suất cực nhỏ $\rightarrow$ tràn số dưới (underflow)
- Số lượng n-gram tăng theo cấp số mũ
- Bộ nhớ lưu trữ rất lớn
- Tính toán chậm

Vấn đề khi N-gram có quy mô lớn

Trong thực tế, mô hình N-gram có thể rất lớn, gây ra:

- Xác suất cực nhỏ $\rightarrow$ tràn số dưới (underflow)
- Số lượng n-gram tăng theo cấp số mũ
- Bộ nhớ lưu trữ rất lớn
- Tính toán chậm

Bài toán không còn chỉ là NLP mà là bài toán hệ thống.

Vấn đề 1: Tràn số dưới (Numerical Underflow)

Xác suất câu được tính bằng:

$$P = p_1 \times p_2 \times p_3 \times \ldots$$

Vì $p_i \leq 1$, khi nhân nhiều xác suất nhỏ:

$$0.01 \times 0.02 \times 0.005 \rightarrow 10^{-9}$$

Nếu nhân nhiều hơn nữa:

$$\rightarrow 10^{-40} \approx 0$$

Máy tính làm tròn về 0 $\Rightarrow$ mất thông tin.

Giải pháp: Tính trong không gian log

Thay vì nhân xác suất:

$$p_1 \times p_2 \times p_3$$

Ta chuyển sang:

$$\log p_1 + \log p_2 + \log p_3$$

Giải pháp: Tính trong không gian log

Thay vì nhân xác suất:

$$p_1 \times p_2 \times p_3$$

Ta chuyển sang:

$$\log p_1 + \log p_2 + \log p_3$$

Khôi phục lại xác suất khi cần:

$$P = \exp(\log P)$$

- Tránh underflow
- Ổn định số học
- Tính toán hiệu quả hơn

Ngữ cảnh dài hơn (Higher-order N-grams)

Khi có nhiều dữ liệu hơn, ta dùng:

- Bigram (n=2)
- Trigram (n=3)
- 4-gram, 5-gram

Ví dụ:

$$P(w_n \mid w_{n-2}, w_{n-1})$$

Ngữ cảnh dài hơn (Higher-order N-grams)

Khi có nhiều dữ liệu hơn, ta dùng:

- Bigram (n=2)
- Trigram (n=3)
- 4-gram, 5-gram

Ví dụ:

$$P(w_n \mid w_{n-2}, w_{n-1})$$

Nếu kích thước từ vựng là V:

Số khả năng n-gram $= V^n$

Tăng theo cấp số mũ.

Dữ liệu N-gram quy mô rất lớn

Một số tập dữ liệu lớn:

- Google Web 5-gram (1 nghìn tỷ từ)
- Google Books N-grams (800 tỷ token)
- COCA (1 tỷ từ)

Dữ liệu N-gram quy mô rất lớn

Một số tập dữ liệu lớn:

- Google Web 5-gram (1 nghìn tỷ từ)
- Google Books N-grams (800 tỷ token)
- COCA (1 tỷ từ)

Lưu trữ toàn bộ bảng n-gram:

$\Rightarrow$ Tốn bộ nhớ khổng lồ

Các kỹ thuật tối ưu hệ thống

1. Lượng tử hóa (Quantization)

- Dùng 4–8 bit thay vì số thực 8 byte

2. Hash 64-bit

- Không lưu chuỗi từ đầy đủ trong RAM

3. Reverse Trie

- Tra cứu n-gram hiệu quả

4. Pruning

- Loại bỏ n-gram hiếm

Infini-gram (∞-gram)

Ý tưởng:

- Không tiền tính toán toàn bộ bảng n-gram
- Tính xác suất tại thời điểm suy luận
- Sử dụng cấu trúc dữ liệu **suffix array**

Infini-gram (∞-gram)

Ý tưởng:

- Không tiền tính toán toàn bộ bảng n-gram
- Tính xác suất tại thời điểm suy luận
- Sử dụng cấu trúc dữ liệu **suffix array**

Cho phép sử dụng ngữ cảnh dài tùy ý, ngay cả với corpus hàng nghìn tỷ token.

Tổng kết

Vấn đề	**Giải pháp**
Underflow	Log-space
Bộ nhớ lớn	Quantization
Quá nhiều n-gram	Pruning
Ngữ cảnh dài	Infini-gram

Tổng kết

Vấn đề	**Giải pháp**
Underflow	Log-space
Bộ nhớ lớn	Quantization
Quá nhiều n-gram	Pruning
Ngữ cảnh dài	Infini-gram

Mở rộng N-gram là bài toán kỹ thuật và tối ưu hệ thống.