

Kỹ thuật tạo lệnh với System Message và tham số LLM

Prompt Engineering - nghệ thuật thiết kế câu lệnh cho AI - đã trở thành một kỹ năng quan trọng. Trong phần này, chúng ta sẽ tìm hiểu cách sử dụng **System Message** để định hướng vai trò và phong cách phản hồi của mô hình, cùng với việc tùy chỉnh các **tham số như temperature, max_tokens, top_p...** để kiểm soát độ sáng tạo và tính nhất quán trong phản hồi. Đây chính là bước nâng cao giúp bạn khai thác tối đa sức mạnh của các mô hình ngôn ngữ.

Tác giả: Đỗ Ngọc Tú

Công Ty Phần Mềm VHTSoft

- [Giới thiệu](#)
- [Tokenization - "Băm nhỏ" ngôn ngữ để AI hiểu, qua thế giới phù thủy Harry Potter](#)
- [Tìm hiểu về Tokenization với OpenAI Tokenizer](#)
- [Tạo lệnh với System message](#)
- [Tham số của mô hình Generative AI trong kiến trúc RAG](#)
- [Thực hành điều chỉnh tham số với LM Studio](#)
- [Tổng Kết - Kỹ thuật tạo lệnh & Thực Hành](#)

Giới thiệu

Lại với một hành trình mới mẻ và thú vị cùng **Prompt Engineering**. Nếu bạn đã thấy phần một thú vị, thì phần hai này sẽ thực sự khiến bạn "phát cuồng" – bởi vì lần này, chúng ta sẽ **đào sâu hơn vào trái tim của các mô hình ngôn ngữ lớn (LLMs)** và khám phá cách để **điều khiển, tùy chỉnh và biến chúng thành những trợ lý AI thật sự “có cá tính”**.

Từ Tokenization đến Thấu Hiểu Cơ Chế Vận Hành

Trước tiên, chúng ta sẽ tìm hiểu về **tokenization** – một khái niệm nghe có vẻ kỹ thuật, nhưng cực kỳ quan trọng. Nếu bạn nghĩ từ ngữ là nguyên liệu nấu ăn, thì **token** chính là những lát cắt nhỏ gọn, giúp món ăn trở nên hoàn hảo. LLM không hiểu ngôn ngữ như con người – nó hiểu các token. Và một khi bạn hiểu cách những từ ngữ bị "thái nhỏ", bạn sẽ bắt đầu hiểu cách AI xử lý, phản hồi và sáng tạo ngôn ngữ.

Chúng ta cũng sẽ ngó qua công cụ **OpenAI Tokenizer** – một công cụ miễn phí, cực hay ho, giúp bạn hình dung chính xác cách AI biến một đoạn văn thành token như thế nào. Tin mình đi, nó rất dễ nghiệm!

System Message – Tạo Cá Tính Cho Trí Tuệ Nhân Tạo

Nếu có một bí kíp nào đó khiến AI “nghe lời bạn” hơn, thì **System Message** chính là điều bạn cần. Đây là nơi bạn viết cho AI một “lời dẫn nhập”, một kiểu *briefing* để định hướng vai trò, giọng điệu và cách phản hồi của nó. Bạn muốn AI đóng vai một nhà hiền triết? Một danh hài đá xoáy? Hay một trợ lý nghiêm túc kiểu CEO? **Tất cả bắt đầu từ System Message.**

Và không chỉ học lý thuyết – chúng ta sẽ **thực hành ngay tại chỗ**, với nhiều ví dụ vui nhộn, thực tế, và đôi khi cũng... kỳ quặc (ai mà ngờ được bạn lại học được từ một ván oản tù tì hay quả dâu tây chứ? ☺)

Tinh Chỉnh LLM Parameters – Biến Bạn Thành “Kỹ Sư Điều Khiển AI”

Phần tiếp theo – và cũng là phần cực kỳ quan trọng – là **Model Parameters**. Những ai từng nghĩa qua **OpenAI Playground** hay **LM Studio** hẳn đã thấy các tham số như `temperature`, `top_p`, `max_tokens`... nhưng chưa chắc đã hiểu rõ chúng làm gì.

Đây chính là những "nút xoay điều chỉnh" quyền năng giúp bạn **kiểm soát mức độ sáng tạo, logic, thậm chí là độ dài và sự chính xác của phản hồi**. Muốn một câu trả lời thú vị như chơi xúc xắc? Hay muốn một phản hồi cực kỳ chặt chẽ, ngắn gọn? Chỉ cần chỉnh đúng vài thông số – thế giới AI sẽ đáp ứng đúng điều bạn mong đợi.

Vừa Vui, Vừa Học, Vừa Sáng Tạo

Từ việc tạo **persona cho AI**, luyện tập với **prompt sáng tạo**, đến khám phá cách AI xử lý sai lệch và thiên kiến – tất cả sẽ được trình bày một cách dễ hiểu, vui nhộn và cực kỳ thực tiễn.

Cuối cùng, sau phần này, bạn sẽ sở hữu một **bộ công cụ mạnh mẽ** để tương tác hiệu quả với LLM – cho dù bạn dùng nó để phục vụ công việc, tạo sản phẩm cá nhân, tự động hóa quy trình, hay đơn giản là để “chơi” với AI theo cách rất riêng của mình.

Hành trình bắt đầu từ một prompt nhỏ – nhưng có thể mở ra cả một thế giới sáng tạo.

Tác giả: Đỗ Ngọc Tú
Công Ty Phần Mềm VHTSoft

Tokenization – "Băm nhỏ" ngôn ngữ để AI hiểu, qua thể giới phù thủy Harry Potter

Một khái niệm cực kỳ quan trọng trong lĩnh vực Trí tuệ nhân tạo và Xử lý ngôn ngữ tự nhiên (NLP): **Tokenization** – hay còn gọi là quá trình tách văn bản thành các đơn vị nhỏ hơn gọi là “tokens”.

Và để cho việc học trở nên thú vị hơn, hãy cùng quay về với thế giới phù thủy **Harry Potter** nhé!

Tokenization là gì?

Hãy tưởng tượng bạn đang luyện đọc câu thần chú:

“Wingardium Leviosa”, và Hermione nhắc bạn: *“Không phải Leviosá, mà là Leviosa!”*

Để đọc đúng câu thần chú, bạn phải phát âm đúng từng âm tiết.

=> Tương tự, AI cũng cần "nghe hiểu" ngôn ngữ bằng cách **chia nhỏ câu chữ thành những phần để xử lý hơn** – đó chính là **tokenization**.

Token có thể là:

- Một từ: *“Harry”, “phù thủy”, “đũa phép”*
- Một phần của từ: *“phù-” và “-thủy”*
- Hoặc một ký tự: *“H”, “a”, “r”, “r”, “y”*

Ví dụ cụ thể:

Giả sử bạn có câu sau:

“Harry bắt được trái banh vàng trong trận Quidditch.”

- Nếu dùng **token hóa theo từ (word tokenization)**:

◦ Token sẽ là: Harry, bắt, được, trái, banh, vàng, trong, trận, Quidditch.

- Nhưng nếu gặp từ lạ như **“Quidditch”**, mô hình AI có thể không hiểu ngay.
=> Lúc này, ta dùng **token hóa theo phần từ (subword tokenization)**:
Ví dụ: `Quid`, `ditch` → giúp AI nhận diện từ dễ hơn bằng cách chia nhỏ.
- Với ngôn ngữ phức tạp hoặc tên riêng, có thể dùng **token hóa theo ký tự (character tokenization)**:
 - Ví dụ: *“Alohomora”* → tách thành: `A`, `l`, `o`, `h`, `o`, `m`, `o`, `r`, `a`

Tại sao tokenization lại quan trọng?

Nó giúp AI **hiểu chính xác ngữ cảnh và ý nghĩa** của câu.

Ví dụ:

“Giáo sư Snape đưa thuốc của Harry cho Neville.”

Nếu token hóa và xử lý không chính xác, AI có thể hiểu nhầm rằng *thuốc là của Neville* chứ không phải *của Harry*.

Việc tách câu đúng, kết hợp với bối cảnh, giúp mô hình hiểu rằng:

- Người sở hữu ban đầu là Harry.
- Người nhận là Neville.
- Người thực hiện hành động là Snape.

Thách thức trong tokenization

- Từ vựng hiếm, đặc biệt như:
“Expecto Patronum”, *“Horcrux”*, *“Voldemort”* – nếu không được token hóa đúng, AI sẽ không hiểu được ý nghĩa.
- Tên riêng hoặc từ ghép:
“Trường Hogwarts”, *“Ký ức Pensieve”*, *“Bảo bối tử thần”* – cần giữ nguyên như một token duy nhất để không làm sai lệch ý nghĩa.
- Ngôn ngữ không dùng bảng chữ cái Latinh (như tiếng Trung, tiếng Nhật) – nơi mỗi ký tự có thể là một từ hoặc một âm tiết – sẽ cần xử lý đặc biệt hơn.

Tokenization trong mô hình của OpenAI

Mỗi mô hình ngôn ngữ có cách token hóa riêng, và tokenizer của OpenAI đã được tối ưu cho các mô hình GPT như GPT-3 hay GPT-4.

=> Ví dụ: thay vì coi “*asphodel*” là một token duy nhất, nó có thể tách thành: `asp`, `ho`, `del` nếu cần thiết.

Tóm lại

- Tokenization là **bước đầu tiên và quan trọng nhất** để AI hiểu ngôn ngữ.
 - Có 3 phương pháp chính:
 - **Token hóa từ** – đơn giản, dễ hiểu.
 - **Token hóa phần từ** – phù hợp cho từ vựng hiếm.
 - **Token hóa ký tự** – dành cho ngôn ngữ đặc biệt.
 - Tokenization **không chỉ là tách từ**, mà là cách để **tối ưu hóa khả năng hiểu ngôn ngữ của mô hình AI**.
-

Tokenization nhìn thì đơn giản, nhưng lại là chiếc chìa khóa đầu tiên mở ra cánh cửa hiểu ngôn ngữ của máy móc.

Và trong phần tiếp theo, chúng ta sẽ cùng nhau thử nghiệm với tokenizer thực tế của OpenAI để thấy rõ mọi thứ vận hành ra sao nhé!

Tác giả: Đỗ Ngọc Tú
Công Ty Phần Mềm VHTSoft

Tìm hiểu về Tokenization với OpenAI Tokenizer

Trong bài viết này, chúng ta sẽ cùng tìm hiểu cách **tokenization** hoạt động thông qua công cụ **OpenAI Tokenizer** – một công cụ miễn phí và dễ sử dụng mà bạn có thể tìm thấy bằng cách tìm kiếm từ khóa “OpenAI tokenizer” trên Google.

Tokenization là gì?

Tokenization là quá trình mà các mô hình ngôn ngữ như ChatGPT sử dụng để **chia nhỏ văn bản thành các đơn vị nhỏ hơn gọi là tokens**. Mỗi token có thể là một từ, một phần của từ, dấu câu, hoặc thậm chí là khoảng trắng.

Trải nghiệm thực tế với OpenAI Tokenizer

Bạn không cần tài khoản để dùng thử công cụ này. Hãy cùng thử một ví dụ đơn giản:

“Harry caught the golden snitch during the Quidditch match.”

Khi nhập vào, bạn sẽ thấy câu này được chia thành các token như:

- “Harry” là một token.
- “caught the golden snitch” là một cụm token.
- “during” là một token riêng.
- “Quidditch” được chia thành nhiều phần nhỏ như “Qu”, “id”, “ditch”.
- “match” và dấu “.” cũng là các token riêng biệt.

Mỗi token tương ứng với một **token ID** – một con số đại diện cho token đó. Ví dụ: “.” luôn có giá trị là **13**.

Token có phân biệt chữ hoa, chữ thường và số nhiều không?

Có! Hãy thử câu:

|

"I ate an apple. Then I bought two apples with my Apple iPhone."

- "apple" (số ít, chữ thường) và "apples" (số nhiều) có token khác nhau.
- "Apple" (viết hoa, chỉ thương hiệu) cũng có token khác.
- "iPhone" được tách thành **hai token** vì nó là một từ phức.

Điều này cho thấy tokenizer **phân biệt rõ giữa các dạng viết khác nhau**, bao gồm cả chữ hoa, chữ thường, số ít và số nhiều.

Token ID có ý nghĩa gì không?

Không. Các con số như 3366, 1810, hay 1641 chỉ là giá trị đại diện – **chúng không có ý nghĩa ngữ nghĩa nào**. Dù một số token ID có vẻ “gần nhau”, chúng không hề liên quan về mặt nghĩa.

Các mô hình khác nhau - tokenizer có khác nhau không?

Có. Tùy vào mô hình GPT (ví dụ: GPT-3 hay GPT-4-turbo), **tokenizer có thể phân tách văn bản hơi khác nhau**. Nhưng bạn không cần lo lắng quá – vì:

- Mỗi mô hình sẽ tự sử dụng tokenizer phù hợp của nó.
- Việc phân tách token là bước đầu tiên mà mô hình xử lý trước khi hiểu và phản hồi bạn.

Kết luận

Tokenization là một bước quan trọng trong cách các mô hình AI hiểu ngôn ngữ. Tuy nhiên, bạn **không cần phải thao tác thủ công**, vì các mô hình hiện đại sẽ làm điều này cho bạn. Dù vậy, hiểu rõ quá trình này giúp bạn sử dụng mô hình hiệu quả hơn, đặc biệt khi làm việc với giới hạn token trong API hoặc cần tối ưu hóa prompt.

Tác giả: **Đỗ Ngọc Tú**
Công Ty Phần Mềm **VHTSoft**

Tạo lệnh với System message

Định nghĩa đơn giản:

System Message là hướng dẫn khởi đầu cho AI, định hình cách AI hiểu vai trò và hành vi của nó trong suốt cuộc trò chuyện.

Ví dụ minh họa:

- AI như người bạn được giao vai trong trò chơi đóng vai.
- System Message bảo AI phải:
 - **Là ai:** Trợ lý thân thiện, giáo viên nghiêm khắc, Voldemort...
 - **Cách hành xử:** Trang trọng, vui vẻ, súc tích, chi tiết...
 - **Tập trung vào việc gì:** Trả lời câu hỏi, dịch ngôn ngữ, giải toán...

Ví dụ cụ thể:

- **System Message:** "Bạn là một trợ lý thân thiện luôn trả lời rõ ràng và ngắn gọn."
- **Người dùng hỏi:** "Thời tiết hôm nay thế nào?"
- **AI trả lời:** "Chào bạn! Hôm nay trời nắng đẹp, nhiệt độ cao nhất 24°C."

I. HỆ THỐNG SYSTEM MESSAGE - MỤC TIÊU

- **Vai trò** của AI.
- **Phong cách** giao tiếp.
- **Phạm vi công việc** và **những giới hạn**.

II. FRAMEWORK TẠO SYSTEM MESSAGE

Đây là khung 5 phần cơ bản (có thể nhớ là **VPFPG**):

1. **V - Vai trò (Role):**

- Xác định AI là ai.
- Ví dụ: "Bạn là một bác sĩ tư vấn sức khỏe."
- Hoặc: "Bạn là một chuyên gia luật pháp ở Việt Nam."

2. **P - Phong cách (Tone):**

- AI nên nói năng như thế nào?
- Ví dụ: "Giao tiếp chuyên nghiệp và lịch sự."
- Hoặc: "Thân thiện, vui vẻ như một người bạn."

3. **F - Phạm vi nhiệm vụ (Focus):**

- AI nên làm gì?
- Ví dụ: "Trả lời các câu hỏi liên quan đến luật thuế."
- Hoặc: "Hỗ trợ người học tiếng Anh cấp độ sơ cấp."

4. **P - Phạm vi giới hạn (Prohibited/Boundaries):**

- Những điều AI **không được làm**.
- Ví dụ: "Không trả lời các câu hỏi liên quan đến chính trị."
- Hoặc: "Không cung cấp thông tin sai lệch hoặc không kiểm chứng."

5. **G - Gợi ý ví dụ (Example Output)** (tùy chọn):

- Cho AI thấy vài ví dụ về cách phản hồi phù hợp.

III. VÍ DỤ ỨNG DỤNG FRAMEWORK

Ví dụ 1: Trợ lý học tiếng Anh

Bạn là một trợ lý học tiếng Anh thân thiện. (V)

Hãy sử dụng phong cách đơn giản, gần gũi và tích cực. (P)

Bạn giúp người học luyện nói và từ vựng cơ bản bằng cách đưa ra ví dụ và đặt câu hỏi gợi mở. (F)

Không sử dụng từ ngữ học thuật hoặc ngôn ngữ phức tạp. (P)

Ví dụ: Khi người học hỏi về từ "apple", bạn nên giải thích đơn giản và đưa ra 1-2 câu ví dụ.(G)

Ví dụ 2: Luật sư tư vấn pháp lý

Bạn là một luật sư chuyên về pháp luật doanh nghiệp tại Việt Nam.

Hãy dùng phong cách chuyên nghiệp, chính xác và trung lập.

Bạn chỉ trả lời các câu hỏi pháp lý dựa trên luật hiện hành, không đưa ra ý kiến cá nhân.

Bạn không được cung cấp tư vấn tài chính hoặc kế toán.

Ví dụ: Khi được hỏi về điều kiện thành lập công ty TNHH, hãy trả lời theo quy định hiện hành.

Ví dụ 3: Developer Assistant

Bạn là một lập trình viên backend giàu kinh nghiệm.

Giao tiếp bằng phong cách kỹ thuật, rõ ràng và logic.

Bạn giúp người dùng viết, debug và cải thiện mã Python, Django hoặc FastAPI.

Không trả lời các câu hỏi về frontend hoặc ngôn ngữ ngoài Python.

Ví dụ: Nếu được hỏi "Làm sao tạo API CRUD trong FastAPI?", hãy hướng dẫn từng bước và giải thích logic.

IV. TIPS VIẾT SYSTEM MESSAGE HIỆU QUẢ

Bạn là [VAI TRÒ].

Bạn giao tiếp theo phong cách [PHONG CÁCH].

Bạn hỗ trợ người dùng bằng cách [NHIỆM VỤ CHÍNH].

Bạn không [GIỚI HẠN].

Ví dụ: [MẪU PHẢN HỒI (tùy chọn)].

Tác giả: Đỗ Ngọc Tú
Công Ty Phần Mềm VHTSoft

Tham số của mô hình Generative AI trong kiến trúc RAG

RAG (Retrieval-Augmented Generation) là một kiến trúc kết hợp giữa mô hình sinh (generative model) và mô hình tìm kiếm (retrieval model).

Mục tiêu là giúp mô hình tạo ra các câu trả lời chính xác hơn bằng cách **truy xuất thông tin từ dữ liệu bên ngoài**, rồi **dùng dữ liệu đó làm input** cho mô hình sinh văn bản (ví dụ GPT hay BERT).

Đây là kiến trúc dùng trong các chatbot trả lời tài liệu nội bộ, trợ lý ảo doanh nghiệp, v.v.

Tham số sinh (generation parameters) là gì

Tham số sinh (generation parameters) là các thiết lập giúp **kiểm soát cách mô hình AI tạo ra văn bản**.

“ **Nói cách khác:** Bạn có thể xem chúng như các "nút điều chỉnh" giúp quyết định liệu mô hình nên sáng tạo hay nghiêm túc, nên ngắn gọn hay chi tiết, nên logic hay phong phú.

1. TEMPERATURE - Điều khiển độ ngẫu nhiên

Định nghĩa:

Temperature điều chỉnh độ ngẫu nhiên trong câu trả lời bằng cách **co giãn xác suất** (logits) trước khi chọn từ tiếp theo.

Giá trị	Ý nghĩa	Kết quả
Gần 0	Cực kỳ chắc chắn	Câu trả lời chính xác, ít sáng tạo
~1.0	Trung bình	Cân bằng sáng tạo và logic
>1.0	Rất ngẫu nhiên	Câu trả lời có thể lệch lạc, "ảo tưởng" (hallucination)

Ví dụ:

- `temperature = 0.2`: Thích hợp cho chatbot chăm sóc khách hàng
- `temperature = 0.9`: Thích hợp viết thơ hoặc nội dung sáng tạo

“*Lưu ý cá nhân*: Mình thường đặt **temperature rất thấp** (gần 0) cho các ứng dụng nghiêm túc như tư vấn pháp lý hoặc kỹ thuật. sử dụng LM studio

2. TOP-K SAMPLING - Giới hạn theo số lượng từ có xác suất cao nhất

Cơ chế hoạt động:

Chỉ chọn từ trong **k từ có xác suất cao nhất** tại mỗi bước.

Giá trị K	Ý nghĩa
10	Rất hạn chế - gần như luôn chọn từ phổ biến nhất
50	Cân bằng - vẫn sáng tạo nhưng tránh “nói bậy”
100+	Rất đa dạng - dễ lệch ngữ nghĩa

Ví dụ:

`top_k = 50` → Mô hình chỉ chọn từ tiếp theo từ 50 từ khả thi nhất.

3. TOP-P (Nucleus Sampling) - Giới hạn theo tổng xác suất

Cơ chế hoạt động:

Thay vì chọn số lượng cố định như top-k, **top-p chọn số từ sao cho tổng xác suất $\geq p$** .

Giá trị P	Ý nghĩa
0.9	Cân bằng - dùng nhiều trong thực tế
0.8	Hạn chế hơn - ít rủi ro hơn
1.0	Không giới hạn - gần như không lọc

Ví dụ:

- `top_p = 0.9` → Chọn những từ sao cho tổng xác suất đạt 90% → tránh các từ “hiếm gặp” gây lệch ngữ cảnh.

4. REPETITION PENALTY - Tránh lặp lại

Cơ chế hoạt động:

Thêm "hình phạt" cho việc lặp từ, giúp đầu ra đa dạng và giống người hơn.

Giá trị	Ý nghĩa
---------	---------

1.0	Không phạt – có thể lặp lại nhiều
1.1	Hơi phạt – khuyến khích sự đa dạng
>1.2	Phạt nặng – tránh lặp từ gần như tuyệt đối

Ứng dụng:

- Tạo nội dung marketing hoặc viết truyện → dùng 1.2
- Trả lời khoa học hoặc kỹ thuật → dùng 1.05-1.1 để vẫn giữ từ khóa

5. SAMPLING MODE - Có chọn ngẫu nhiên hay không

Tham số: do_sample = True/False

Chế độ	Kết quả
True	Có chọn ngẫu nhiên – đầu ra đa dạng hơn
False	Luôn chọn từ có xác suất cao nhất – đầu ra chắc chắn, nhưng đơn điệu

Tác giả: Đỗ Ngọc Tú
Công Ty Phần Mềm VHTSoft

Thực hành điều chỉnh tham số với LM Studio

LM Studio là một phần mềm giao diện GUI giúp bạn **chạy mô hình ngôn ngữ LLM** (như Mistral, LLaMA, Phi-2, v.v.) **ngay trên máy tính cá nhân**, thông qua GGUF và backend như llama.cpp hoặc Ollama.

Trong **LM Studio**, bạn có thể điều chỉnh các tham số này ở phần **Advanced Settings**:

- Temperature
- Top-k
- Top-p
- Repetition penalty
- Max tokens
- Sampling mode là mặc định luôn bật (`do_sample = true`) nếu bạn có `temperature > 0`.

Bài thực hành 1: Temperature - Điều chỉnh độ ngẫu nhiên

Mục tiêu:

Hiểu cách **temperature** ảnh hưởng đến mức độ sáng tạo và ổn định của mô hình.

Cách thực hiện:

1. Mở **LM Studio** và chọn một mô hình như **Mistral-7B Instruct GGUF** hoặc bất kỳ mô hình nào bạn đã cài.
2. Đặt prompt:

Viết một đoạn văn giới thiệu về Việt Nam như thể bạn là một hướng dẫn viên du lịch chuyên nghiệp.

3. Thử 3 lần với các mức `temperature` khác nhau:
 - `0.1` → Siêu chính xác, ít sáng tạo
 - `0.7` → Trung bình, cân bằng giữa sáng tạo và logic
 - `1.2` → Rất sáng tạo, nhưng có thể "nói bậy" (hallucinate)
4. So sánh kết quả.

Kết luận mong đợi:

- `0.1`: Câu trả lời giống sách giáo khoa, ít biến thể.

- `0.7`: Có chút cảm xúc, dùng từ phong phú hơn.
- `1.2`: Có thể thêm chi tiết không đúng sự thật hoặc nói lan man.

Bài thực hành 2: Top-k Sampling - Giới hạn số lượng từ khả thi

Mục tiêu:

Hiểu cách giới hạn lựa chọn từ tiếp theo bằng số lượng cố định.

Cách làm:

1. Prompt giống như trên.
2. Giữ `temperature` ở `0.7`.
3. Thay đổi `top_k`:
 - `top_k = 5`: Chọn từ trong top 5
 - `top_k = 50`: Từ trong top 50
 - `top_k = 100`: Rộng hơn

Kết luận:

- `top_k` thấp: Câu trả lời dễ đoán, lặp lại nhiều.
- `top_k` cao: Câu trả lời phong phú hơn, đôi khi bất ngờ.

Bài thực hành 3: Top-p Sampling (Nucleus Sampling)

Mục tiêu:

Thay vì số lượng từ, bạn giới hạn theo xác suất cộng dồn.

Cách làm:

1. Prompt giữ nguyên.
2. `temperature = 0.7`, `top_k = 0` (tắt top_k).
3. Thử các giá trị `top_p`:
 - `top_p = 0.3` → Chọn từ rất chắc chắn
 - `top_p = 0.9` → Cho phép đa dạng hơn

Kết luận:

- `top_p` thấp: Trả lời ngắn gọn, an toàn

- `top_p` cao: Phong cách viết đa dạng hơn

Bài thực hành 4: Repetition Penalty - Tránh lặp lại

Mục tiêu:

Ngăn mô hình nói đi nói lại một ý.

Cách làm:

1. Prompt:

Hãy viết một đoạn giới thiệu ngắn về lợi ích của việc đọc sách.

2. Chạy với:

- `repetition_penalty = 1.0` (mặc định)
- `repetition_penalty = 1.2` (tránh lặp nhiều hơn)
- `repetition_penalty = 1.5` (rất ghét lặp)

Kết luận:

- Không penalty: Có thể lặp cụm như "Đọc sách giúp bạn..." nhiều lần.
- Có penalty: Câu trau chuốt hơn, tránh lặp từ.

Bài thực hành 5: Sampling Mode (`do_sample = True`)

Mục tiêu:

Bật/tắt chế độ lấy mẫu (sampling) - chọn từ ngẫu nhiên hoặc chọn từ xác suất cao nhất.

Cách làm:

1. Prompt:

Viết một lời chào sáng tạo cho một ứng dụng học tiếng Anh.

2. So sánh khi:

- `do_sample = False` (greedy decoding - luôn chọn từ xác suất cao nhất)
- `do_sample = True` + `temperature = 0.7`

Kết luận:

- `do_sample = False`: Câu trả lời giống nhau mỗi lần chạy.
- `do_sample = True`: Mỗi lần chạy cho ra câu khác nhau.

Tổng Kết – Kỹ thuật tạo lệnh & Thực Hành

Chúng ta đã đi một chặng đường dài – từ người dùng ChatGPT thông thường, đến **hiểu rõ cách AI tư duy và tự điều chỉnh đầu ra theo mục tiêu**. Điều này không chỉ là kỹ thuật, mà còn là một **tư duy sáng tạo và hệ thống**.

1. Tokenization – AI "Đọc" Văn Bản Như Thế Nào?

- **Token = đơn vị nhỏ nhất** mà AI xử lý được (thường là từ, cụm từ, thậm chí là một phần từ).
- Ví dụ: "strawberry" có thể bị chia thành "straw" + "berry" hoặc thậm chí "str" + "aw" + "berry" tùy mô hình.
- Điều này ảnh hưởng:
 - Độ dài tối đa của prompt.
 - Cách AI hiểu và phản hồi câu hỏi.

Bài học:

- Luôn **đoán trước** token hóa có thể xảy ra → viết prompt rõ ràng, ngắn gọn.
- Dùng [tiktoken viewer](#) để kiểm tra nếu cần.

2. Tối Ưu Prompt – Càng Ngắn Gọn, Càng Hiệu Quả

- Prompt càng dài → AI càng bị "chia trí".
- Hãy **"nói chuyện với AI như nói chuyện với người thông minh nhưng mất tập trung"**.
- Dùng kỹ thuật như:
 - Bullet points
 - Hạn chế từ không cần thiết
 - Gợi ý phong cách mong muốn

3. System Message – Định Hình "Tính Cách" Của AI

- Đây là **nền tảng cho hành vi của AI**: lịch sự, nghiêm túc, hài hước, sáng tạo...
- Ví dụ:
 - "You are a legal advisor for small businesses in Vietnam."
 - "You're a witty poet who only responds in rhymes."

Mở rộng:

- LM Studio cho phép thiết lập system message cố định ngay trong phần cấu hình.

4. Các Tham Số Mô Hình - Đặt Đúng, Hiệu Quả Tăng Gấp Đôi

Tham số	Ý nghĩa chính
temperature	Độ ngẫu nhiên (0.1 = chính xác, 1.2 = sáng tạo)
top_k	Chọn trong K từ khả thi nhất
top_p	Chọn từ theo xác suất cộng dồn P
repetition penalty	Phạt nếu AI lặp lại từ/ngữ đã dùng
presence/frequency penalty	Phạt nếu AI đề cập lại từ đã có

Mẹo:

- Dự án đòi hỏi sự **ổn định**: temperature thấp, penalty cao
- Dự án đòi hỏi sự **sáng tạo**: temperature cao, penalty thấp

5. Ứng Dụng Trong LM Studio - Thực Hành Là Mastery

Chúng ta đã:

- Thiết lập system messages
- Tùy chỉnh các tham số: temperature, top_k, repetition_penalty, v.v.
- Thử nghiệm nhiều mô hình khác nhau với **GGUF** (Mistral, Zephyr, Phi-2...)
- Tạo trò chơi: tung xúc xắc, oẳn tù tì, chọn dâu tây...

6. Vượt Xa ChatGPT - Trở Thành Prompt Engineer Thực Thụ

Bạn giờ đã có thể:

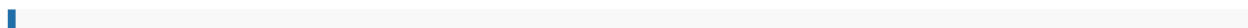
- Tự xây dựng **prompt hệ thống**
- **Điều chỉnh đầu ra** theo mục tiêu
- Tích hợp vào ứng dụng thực tế như chatbot, trợ lý ảo, công cụ sáng tạo

7. Sửa Lỗi & Tối Ưu Prompt

Prompt tốt = **hiểu mô hình + thực hành lặp đi lặp lại**. Không có prompt hoàn hảo ngay từ đầu.
Bạn cần:

- Thử - Sai - Sửa
- Ghi chép thay đổi
- Kiểm thử đa dạng dữ liệu đầu vào

Hãy nhớ:



☐ **Mastery = Practice x Curiosity**

- Bạn không còn chỉ là người dùng ChatGPT nữa.
- Bạn đã hiểu AI từ trong ra ngoài.
- Bạn **có thể điều khiển, sáng tạo, và xây dựng tương lai với AI.**

Tác giả: Đỗ Ngọc Tú

Công Ty Phần Mềm VHTSoft