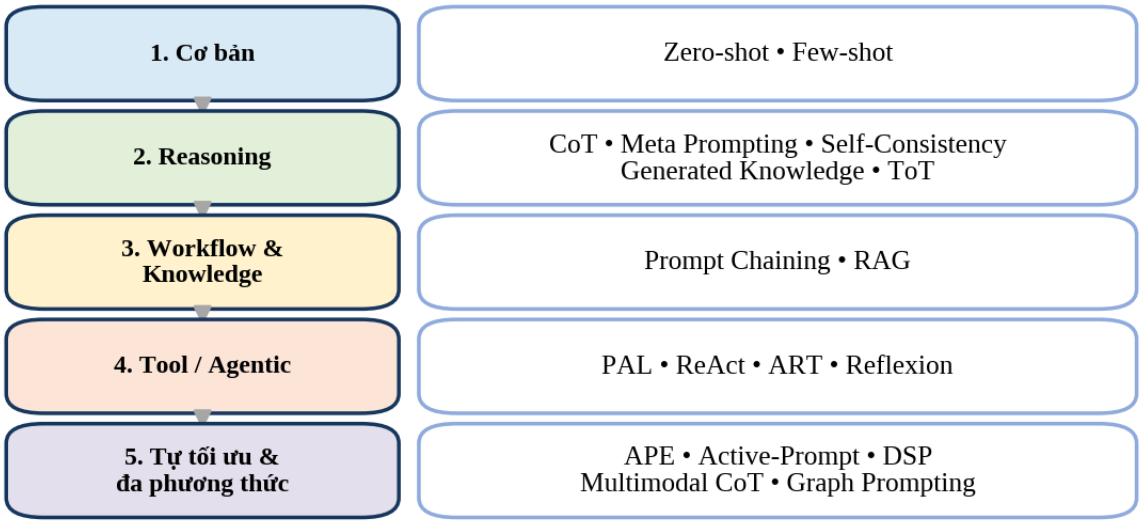
PROMPTING TECHNIQUES

Từ Zero-shot đến RAG, ReAct và Reflexion

Sổ tay tiếng Việt • Định nghĩa rõ ràng • Ví dụ thực tế • Sơ đồ trực quan

Bản đồ các kỹ thuật Prompting



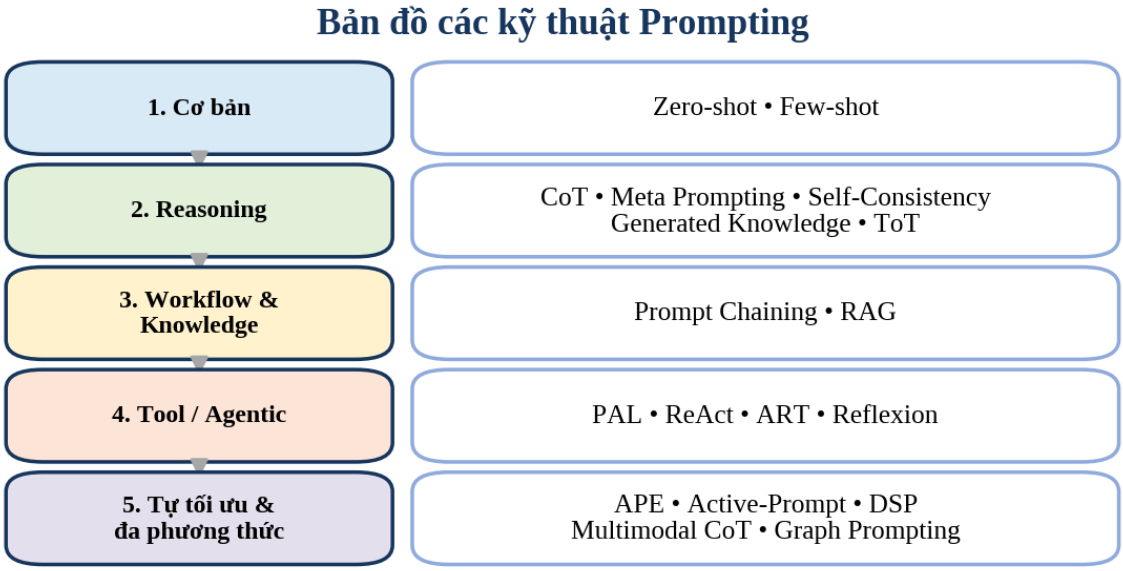
Mục tiêu của cuốn sách

Giúp người đọc hiểu nhanh bản chất của 18 kỹ thuật Prompting, nhận ra điểm khác nhau giữa chúng và biết nên chọn kỹ thuật nào cho từng loại bài toán. Nội dung được biên tập lại từ Prompt Engineering Guide, loại bỏ phần trùng lặp và dùng ví dụ mới để dễ áp dụng.

Biên soạn dựa trên Prompt Engineering Guide (DAIR.AI)

Cách đọc cuốn sách này

Các kỹ thuật Prompting không phải là một danh sách ngang hàng. Một số chỉ là cách viết prompt tốt hơn; một số là cơ chế reasoning; một số lại là cả một workflow có truy xuất dữ liệu, gọi công cụ hoặc học từ feedback. Vì vậy, sách được tổ chức theo 5 tầng độ phức tạp thay vì đi theo thứ tự URL.



Hình 1. Năm tầng kỹ thuật Prompting từ cơ bản đến hệ thống tự tối ưu.

Nguyên tắc sử dụng

- Bắt đầu bằng kỹ thuật đơn giản nhất có thể giải quyết được bài toán. Không nên dùng ToT, ReAct hay Reflexion khi Zero-shot đã đủ.
- Tách rõ ba loại vấn đề: thiếu hướng dẫn, thiếu reasoning, hay thiếu dữ liệu/công cụ. Mỗi loại có kỹ thuật xử lý khác nhau.
- Kỹ thuật nâng cao thường đổi độ chính xác lấy chi phí: nhiều token hơn, nhiều lần gọi model hơn, hoặc phải xây thêm tool/retrieval/memory.
- Luôn đánh giá bằng test set thực tế thay vì chỉ nhìn một vài ví dụ đẹp.

Ghi chú thuật ngữ

Các thuật ngữ chuyên ngành được giữ nguyên bằng English và giải thích tiếng Việt trong ngoặc ở lần xuất hiện chính. Một số thuật ngữ quen thuộc như prompt, LLM, output được dùng tự nhiên trong các phần sau để tránh câu văn nặng nề.

Bảng nhìn nhanh: 18 kỹ thuật khác nhau ở đâu?

Kỹ thuật	Nhóm	Ý tưởng cốt lõi
Zero-shot Prompting	Cơ bản	Instruction trực tiếp
Few-shot Prompting	Cơ bản	Học pattern từ vài ví dụ
Chain-of-Thought (CoT) Prompting	Reasoning	Suy luận nhiều bước
Meta Prompting	Reasoning	Khung cấu trúc thay vì nội dung
Self-Consistency	Reasoning	Nhiều đường suy luận + đồng thuận
Generated Knowledge Prompting	Reasoning	Sinh kiến thức trước khi trả lời
Tree of Thoughts (ToT)	Reasoning	Tìm kiếm nhiều nhánh suy luận
Prompt Chaining	Workflow & Knowledge	Chuỗi prompt theo workflow
Retrieval-Augmented Generation (RAG)	Workflow & Knowledge	Truy xuất nguồn ngoài
Program-Aided Language Models (PAL)	Tool / Agentic	Sinh code + runtime thực thi
ReAct	Tool / Agentic	Reasoning ↔ Action ↔ Observation
Automatic Reasoning and Tool-use (ART)	Tool / Agentic	Tự chọn demo + tool-use program
Reflexion	Tool / Agentic	Feedback → reflection → memory
Automatic Prompt Engineer (APE)	Tự tối ưu & đa phương thức	Tự sinh/chọn prompt tối ưu
Active-Prompt	Tự tối ưu & đa phương thức	Chọn câu khó để annotate
Directional Stimulus Prompting (DSP)	Tự tối ưu & đa phương thức	Hint định hướng output
Multimodal CoT Prompting	Tự tối ưu & đa phương thức	Reasoning trên text + image
Graph Prompting / GraphPrompt	Tự tối ưu & đa phương thức	Prompt/adaptation cho graph data

Bảng này chỉ để định vị nhanh; phần sau giải thích từng kỹ thuật bằng ví dụ và sơ đồ.

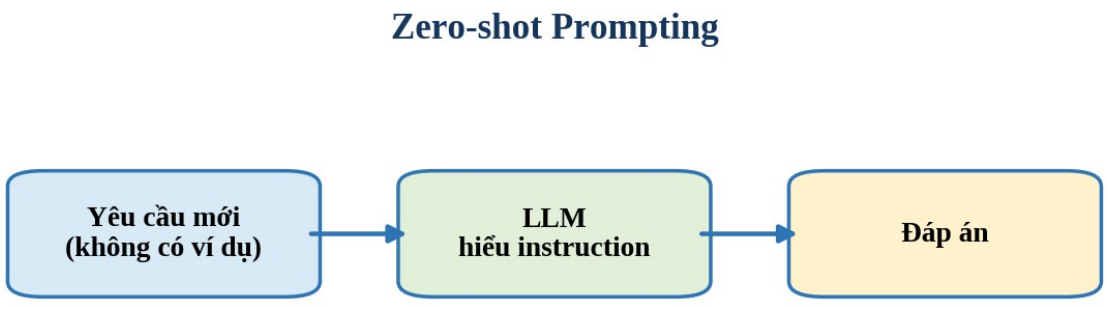
PHẦN I — Kỹ thuật cơ bản

Hai kỹ thuật nền tảng. Nên thử trước khi dùng các phương pháp phức tạp hơn.

Zero-shot Prompting

Zero-shot Prompting — *ra lệnh trực tiếp, không cung cấp ví dụ*

Định nghĩa. Zero-shot Prompting là cách yêu cầu mô hình thực hiện một nhiệm vụ chỉ bằng instruction (chỉ dẫn), không đưa kèm ví dụ mẫu. Mô hình dựa vào kiến thức đã học và khả năng làm theo hướng dẫn để tạo câu trả lời.



Minh hoạ: Zero-shot Prompting

Cách hoạt động

- 1. Viết rõ nhiệm vụ cần làm.
- 2. Cung cấp input cần xử lý.
- 3. Nêu format đầu ra nếu cần.
- 4. Để mô hình tự suy ra cách thực hiện mà không dựa vào demonstration (ví dụ mẫu).

Ví dụ — Phân loại phản hồi khách hàng

Prompt: “Phân loại câu sau thành Positive, Neutral hoặc Negative. Chỉ trả về một nhãn. Câu: ‘Dịch vụ ổn nhưng thời gian chờ hơi lâu.’”
Kết quả mong đợi: Neutral.

Khi nên dùng. Phù hợp với tác vụ quen thuộc như phân loại, tóm tắt, dịch, trích xuất hoặc viết lại khi instruction đã đủ rõ.

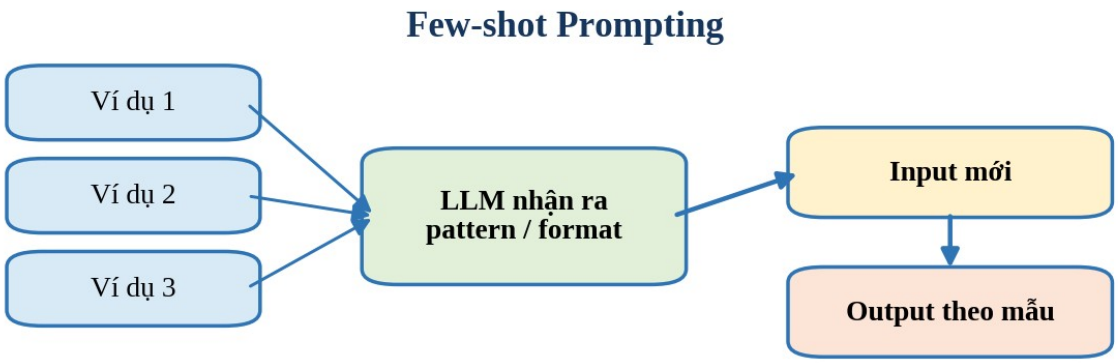
Hạn chế / lưu ý. Nếu yêu cầu có format lạ, tiêu chí ngầm hoặc cách đánh giá đặc thù, Zero-shot dễ hiểu sai. Khi đó nên chuyển sang Few-shot.

Nguồn tham khảo chính: https://www.promptingguide.ai/techniques/zeroshot

Few-shot Prompting

Few-shot Prompting — *dẫn dắt bằng một vài ví dụ*

Định nghĩa. Few-shot Prompting cung cấp một số demonstration (ví dụ mẫu) ngay trong prompt để mô hình nhận ra pattern (mẫu), cách gán nhãn, giọng điệu hoặc cấu trúc đầu ra. Đây là một dạng In-context Learning (học trong ngữ cảnh) mà không cần huấn luyện lại mô hình.



Minh hoạ: Few-shot Prompting

Cách hoạt động

- 1. Chọn một vài ví dụ đại diện và chính xác.
- 2. Giữ format giữa các ví dụ nhất quán.
- 3. Đưa input mới ở cuối prompt.
- 4. Mô hình bắt chước pattern đã quan sát để trả lời.

Ví dụ — Chuẩn hoá tiêu đề ticket hỗ trợ

Ví dụ 1: “Không đăng nhập được” → “[ACCESS] Login failure”
Ví dụ 2: “Không nhận OTP” → “[AUTH] OTP not received”
Input mới: “Thẻ bị từ chối khi thanh toán” → mô hình học format và có thể trả “[PAYMENT] Card declined”.

Khi nên dùng. Hữu ích khi bạn muốn mô hình tuân theo format nội bộ, taxonomy (hệ phân loại), style hoặc quy tắc khó mô tả bằng một câu.

Hạn chế / lưu ý. Ví dụ kém chất lượng sẽ truyền sai pattern. Few-shot cũng không tự giải quyết tốt các bài toán reasoning phức tạp nếu ví dụ chỉ cho đáp án mà không thể hiện cách suy luận.

Nguồn tham khảo chính: https://www.promptingguide.ai/techniques/fewshot

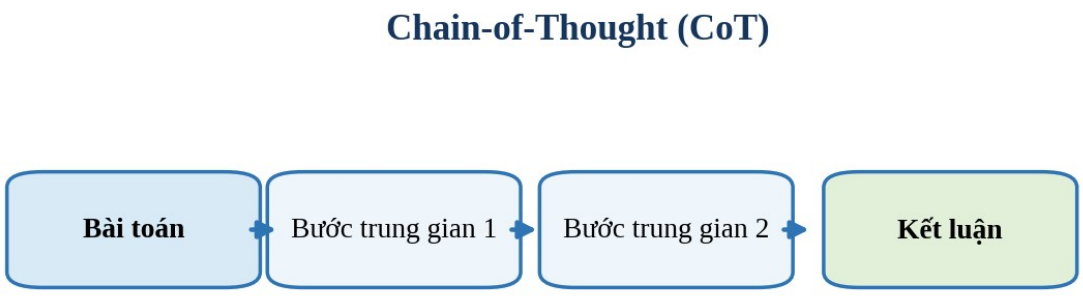
PHẦN II — Kỹ thuật Reasoning (suy luận)

Nhóm này tập trung vào cách mô hình tổ chức quá trình suy luận, từ một chuỗi tuyến tính đến nhiều nhánh tìm kiếm.

Chain-of-Thought (CoT) Prompting

Chain-of-Thought (CoT) Prompting *— suy luận theo chuỗi bước trung gian*

Định nghĩa. Chain-of-Thought (CoT) Prompting khuyến khích mô hình xử lý bài toán qua các intermediate reasoning steps (bước suy luận trung gian) trước khi đưa ra kết luận. Kỹ thuật này đặc biệt hữu ích với toán, logic và câu hỏi nhiều bước.



Minh hoạ: Chain-of-Thought (CoT) Prompting

Cách hoạt động

- 1. Tách bài toán thành các bước hợp lý.
- 2. Có thể dùng Few-shot CoT: ví dụ mẫu vừa có đáp án vừa có cách giải.
- 3. Zero-shot CoT dùng chỉ dẫn ngắn để yêu cầu mô hình suy nghĩ theo từng bước.
- 4. Auto-CoT tự tạo chuỗi suy luận mẫu thay vì viết tay toàn bộ.

Ví dụ — Ước tính chi phí

Bài toán: "Một gói có 12 sản phẩm, giá 35.000 đồng/sản phẩm, giảm 10%. Tổng tiền sau giảm?"
Cách trình bày tốt: xác định giá trước giảm → tính mức giảm → đưa kết quả cuối. Không cần viết lan man ngoài các bước cần thiết.

Khi nên dùng. Dùng khi câu trả lời phụ thuộc vào nhiều bước tính toán hoặc quan hệ logic.

Hạn chế / lưu ý. Chuỗi suy luận có thể nghe hợp lý nhưng vẫn sai. CoT không thay thế dữ liệu đúng hoặc công cụ tính toán chính xác.

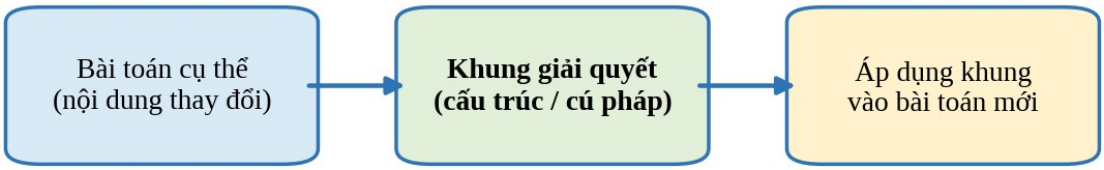
Nguồn tham khảo chính: https://www.promptingguide.ai/techniques/cot

Meta Prompting

Meta Prompting *— dẫn dắt bằng cấu trúc giải quyết vấn đề*

Định nghĩa. Meta Prompting tập trung vào structure (cấu trúc) và syntax (cú pháp) của cách giải, thay vì nhồi nhiều ví dụ nội dung cụ thể. Mục tiêu là đưa cho mô hình một "khung tư duy" có thể tái sử dụng cho nhiều bài toán cùng loại.

Meta Prompting: tập trung vào cấu trúc



Minh hoạ: Meta Prompting

Cách hoạt động

- 1. Xác định cấu trúc chung của bài toán.
- 2. Mô tả vai trò của từng bước hoặc từng trường output.
- 3. Giữ nội dung cụ thể ở mức tối thiểu.
- 4. Áp dụng cùng khung cho các input mới.

Ví dụ — Khung phân tích một quyết định

Prompt khung: “Với mọi phương án, hãy lần lượt đánh giá: Mục tiêu → Giả định → Lợi ích → Rủi ro → Điều kiện để triển khai → Kết luận.” Sau đó chỉ thay nội dung bài toán, không phải viết lại cách phân tích.

Khi nên dùng. Phù hợp với bài toán có cấu trúc lặp lại như phân tích case, toán, coding challenge hoặc đánh giá phương án.

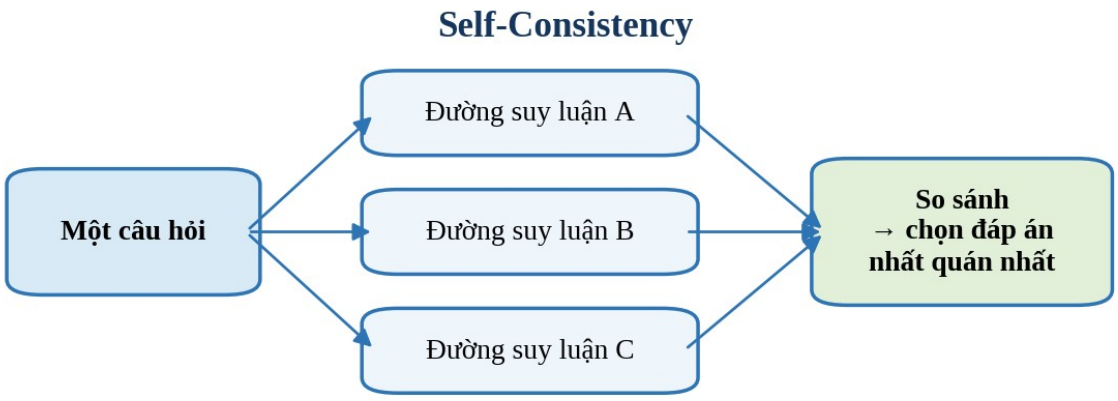
Hạn chế / lưu ý. Meta Prompting giả định mô hình đã có đủ kiến thức nền. Với nhiệm vụ quá mới hoặc quá đặc thù, cấu trúc tốt nhưng nội dung vẫn có thể thiếu.

Nguồn tham khảo chính: https://www.promptingguide.ai/techniques/meta-prompting

Self-Consistency

Self-Consistency — *lấy nhiều đường suy luận và chọn đáp án nhất quán*

Định nghĩa. Self-Consistency tạo nhiều reasoning paths (đường suy luận) khác nhau cho cùng một câu hỏi, rồi chọn đáp án xuất hiện nhất quán nhất. Nó là cách nâng độ tin cậy của CoT thay vì chỉ dùng một lần sinh câu trả lời.



Minh hoạ: Self-Consistency

Cách hoạt động

- 1. Sinh nhiều cách giải độc lập.
- 2. Mỗi cách có thể đi theo một đường suy luận khác.
- 3. So sánh kết quả cuối.
- 4. Chọn đáp án được nhiều đường suy luận đồng thuận nhất.

Ví dụ — Bài toán logic có nhiều cách tiếp cận

Thay vì hỏi một lần "Có bao nhiêu lịch trình hợp lệ?", hệ thống chạy 5–10 lần với biến thiên hợp lý. Nếu phần lớn đường suy luận cùng cho kết quả 6, đáp án 6 được ưu tiên hơn một kết quả đơn lẻ khác biệt.

Khi nên dùng. Hữu ích cho arithmetic reasoning (suy luận số học), commonsense reasoning (suy luận thường thức) và bài toán có thể giải bằng nhiều đường.

Hạn chế / lưu ý. Tốn nhiều token và thời gian hơn. Nếu tất cả đường suy luận cùng dựa trên một giả định sai, "đồng thuận" vẫn có thể sai.

Nguồn tham khảo chính: https://www.promptingguide.ai/techniques/consistency

Generated Knowledge Prompting

Generated Knowledge Prompting — *tạo kiến thức hỗ trợ trước khi trả lời*

Định nghĩa. Generated Knowledge Prompting yêu cầu mô hình tạo ra knowledge (kiến thức) liên quan trước, rồi dùng phần kiến thức đó làm context để đưa ra dự đoán hoặc câu trả lời cuối.

Generated Knowledge Prompting



Minh hoạ: Generated Knowledge Prompting

Cách hoạt động

- 1. Nhận câu hỏi.
- 2. Sinh các facts/knowledge có liên quan.
- 3. Ghép kiến thức hữu ích vào prompt.
- 4. Dùng chúng để giải quyết câu hỏi chính.

Ví dụ — Câu hỏi thường thức

Câu hỏi: “Trong golf, người có điểm cao hơn có thắng không?”
Bước kiến thức: “Golf thường tính số gậy; người chơi cố hoàn thành với số gậy thấp hơn.”
Sau khi có kiến thức này, mô hình dễ trả lời đúng hơn: Không.

Khi nên dùng. Phù hợp với câu hỏi cần kiến thức thường thức hoặc kiến thức nền nhưng chưa cần hệ thống truy xuất dữ liệu ngoài.

Hạn chế / lưu ý. Kiến thức do chính mô hình sinh ra có thể sai. Nếu tính đúng-sai quan trọng, RAG với nguồn đáng tin cậy thường phù hợp hơn.

Nguồn tham khảo chính: https://www.promptingguide.ai/techniques/knowledge

Tree of Thoughts (ToT)

Tree of Thoughts (ToT) — *khám phá nhiều nhánh suy luận có đánh giá và quay lui*

Định nghĩa. Tree of Thoughts (ToT) mở rộng CoT từ một chuỗi tuyến tính thành một tree (cây) gồm nhiều hướng suy nghĩ. Hệ thống có thể sinh các candidate thoughts (nhánh ứng viên), đánh giá chúng, giữ nhánh tốt, loại nhánh kém và backtrack (quay lui) khi cần.