Analogical Prompting of Large Language Models

OpenAI GPT-5.0: Generated and published : August 17, 2025

It is about analogical prompting of LLMs and it is about how such an analogical prompting procedure works out!

So, the large language model is confronted with a problem statement.
The LLM is presented with a certain question, in other words.

The LLM then generates a row of exemplars/examples: It thus generates comparable problems and their solutions or rather questions and their answers as exemplar/example and then solve original problems, answer the original question according to the self-created exemplars/examples. Right?

Yes, you've captured the essence quite well. Let me lay it out more systematically so it's clearer how **analogical prompting** works in LLMs:

1. Problem **presentation** – The LLM is given a target problem to solve.

2. Exemplar **generation** (self-construction of analogies) – Instead of (or in addition to) relying on provided few-shot examples, the LLM generates its own *analogous problems* along with their solutions.
   - These problems are structurally similar to the original one.
   - The solutions demonstrate a reasoning pattern or schema that could be transferred.

3. Mapping / **transfer** – The LLM then *maps* the reasoning from the generated exemplars onto the target problem.
   - Essentially, the model is using analogy: "If I can solve this similar problem in this way, then the original problem can be solved using the same pattern."

4. Final **solution** – Guided by these analogical exemplars, the model solves the original target problem, ideally in a more systematic and accurate way than if it just jumped straight into solving.

This mimics a human reasoning process: when faced with something unfamiliar, we recall or invent similar problems, see how they're solved, and then apply the same logic.

So yes — the key steps are:

(a) **Generate exemplars** → (b) **Solve exemplars** → (c) **Apply to original problem.**

Would you like me to also show you a **worked-out toy example** (say, in math or logic) where analogical prompting plays out step by step? That might make the process very tangible.

Basically, it is about the self-construction of analogies instead of - or in addition to - relying on provided few-shot examples. It is about generating a row of own analogous problems, while the reasoning for their solution is mapped onto the solution of the initial problem, right?

Exactly — that's the core idea!

Analogical prompting differs from standard few-shot prompting in that:

- **Few-shot prompting** → The model is *given* examples to imitate.
- **Analogical prompting** → The model *creates its own* structurally similar examples (analogous problems), solves them, and then transfers the reasoning pattern to the original target.

So yes:

1. It **self-constructs** analogies (rather than only relying on user-provided ones).
2. It **generates** a row of analogous problems + their solutions**.**
3. It **maps** the reasoning schema from these analogies back to the initial problem to solve it.

This approach gives the model a kind of "scaffold" it builds for itself, which often improves reasoning consistency and reduces errors.

Do you want me to sketch this in a diagram-like flow (Problem → Self-Generated Analogies → Transfer → Solution), so you have a visual summary of the process?

Yes, please.

Here's a clean step-by-step flow you can use as a diagram-like summary of **analogical prompting**:

Problem Statement

1. Generate analogous problems

- LLM creates its own set of problems structurally similar to the target problem.

2. Solve the analogous problems

- LLM works through each analogous problem, producing step-by-step solutions.
- This establishes a *reasoning pattern / schema*.

3. Map reasoning patterns

- The reasoning steps extracted from the analogies are aligned with the structure of the original problem.

4. Solve original problem

- Using the mapped reasoning, the LLM produces the final solution to the target problem.

Visually, you can think of it like this:

```
              [Original Problem]
                      |
                      v
    [Generate Analogous Problems + Solutions]
                      |
                      v
          [Extract Reasoning Pattern]
                      |
                      v
[Apply Pattern to Original Problem → Final Solution]
```

While prompting another LLM on the same issue, this other LLM tells me that *your* statement is not quite right.

<*quote*> In analogical prompting, an LLM doesn't generate its own examples. Instead, it uses retrieval-based analogical prompting (RAP) to find and use similar examples from an external knowledge base to solve a new problem. </*quote*>

and further:

<*quote*> Why Not Generate Examples? The reason LLMs don't typically generate their own examples for this purpose is related to reliability. If an LLM were to generate its own examples, it could be prone to hallucination or creating incorrect examples, which would lead to a flawed solution for the original problem. By relying on a verified external database of solved examples, analogical prompting ensures that the reasoning is grounded in correct information.</*quote*>

Both explanations are partly right — but are talking about **two different variants** that often get lumped together under the same label.

1) "Analogical Prompting" (original paper)

- In the original formulation (Yasunaga et al., ICLR 2024), the model is **explicitly prompted to self-generate** a few analogous problems (with solutions) **before** solving the target. No external database is required. The paper describes this as "self-generating relevant reasoning exemplars" and shows gains over few-shot CoT on several benchmarks. Computer SciencearXiv+1

2) Retrieval-based analogical prompting (a different family)

- Other works build a **library of solved examples** and **retrieve** the most similar cases to form an analogical prompt (sometimes called "retrieval-based analogical prompt construction"). This is closer to RAG/RAP-style methods (retrieve → insert into prompt → solve). arXiv+1
- Retrieval-based analogical solvers also exist in domains like math word problems (e.g., REAL), using nearest neighbors as analogies. ACL AnthologyAmazon Web Services, Inc.

Why the confusion?

- Some people use "RAP" to mean **retrieval-augmented prompting** in general (or even *retrieval-augmented planning*, which is different again), and then equate that with "analogical prompting." That's not what the **original** analogical prompting paper meant. OpenReviewMedium

Reliability trade-offs

- **Self-generated analogies**: no labeled data needed and tailored to the instance; risk is that the model can generate a poor analogy — yet the original paper still reports performance improvements when done with a good template. Computer Science
- **Retrieval-based analogies**: grounded in verified examples and often more **reliable**, but you must build/maintain a high-quality library and your results depend on retrieval quality. arXiv

---

If you want, I can give you two short, ready-to-use prompt templates—one for **self-generated** analogies and one for **retrieval-based** analogies—to try on the same problem and compare.