Prompt Engineering

AI with Deep Learning
EE4016

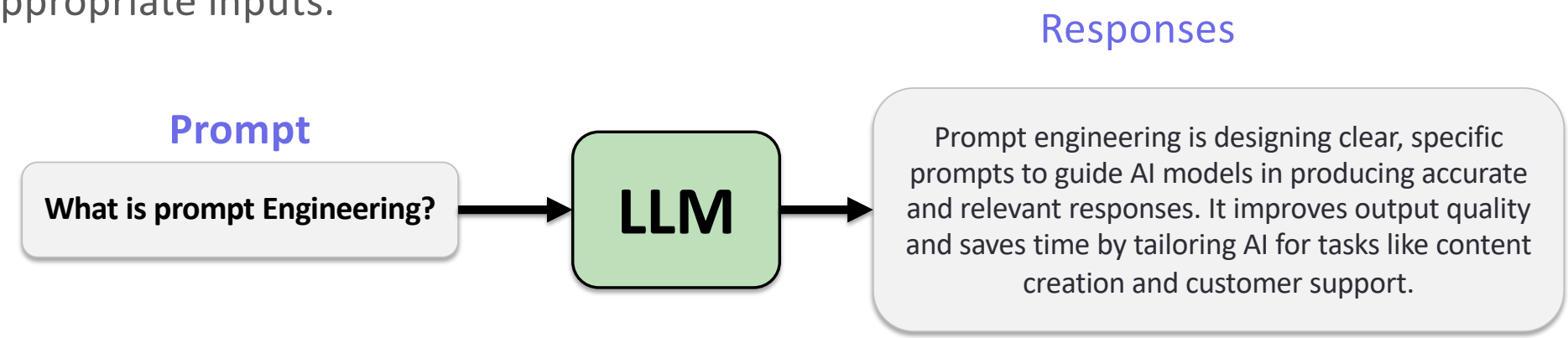
Prof. Lai-Man Po
Department of Electrical Engineering
City University of Hong Kong

Content

- **Basics of Prompt Engineering**
  - In-Context Learning: Zero shot, few shot prompts
  - System Prompt, User Prompt: Clarity, Context, Role
- **Advanced Prompt Engineering**
  - Chain-of-thought (CoT) => Zero shot CoT =>Auto-CoT (2022)
  - Self-Consistency (2022)
  - Least-to-Most Prompting (2022)
  - Tree-of-Thought (2023)
  - Long-From CoT (2024)
  - Retrival Augmented Generation (RAG, 2020)
  - ReAct: Reason + Act (2022)
  - Chain-of-Verification (CoVe, 2023)

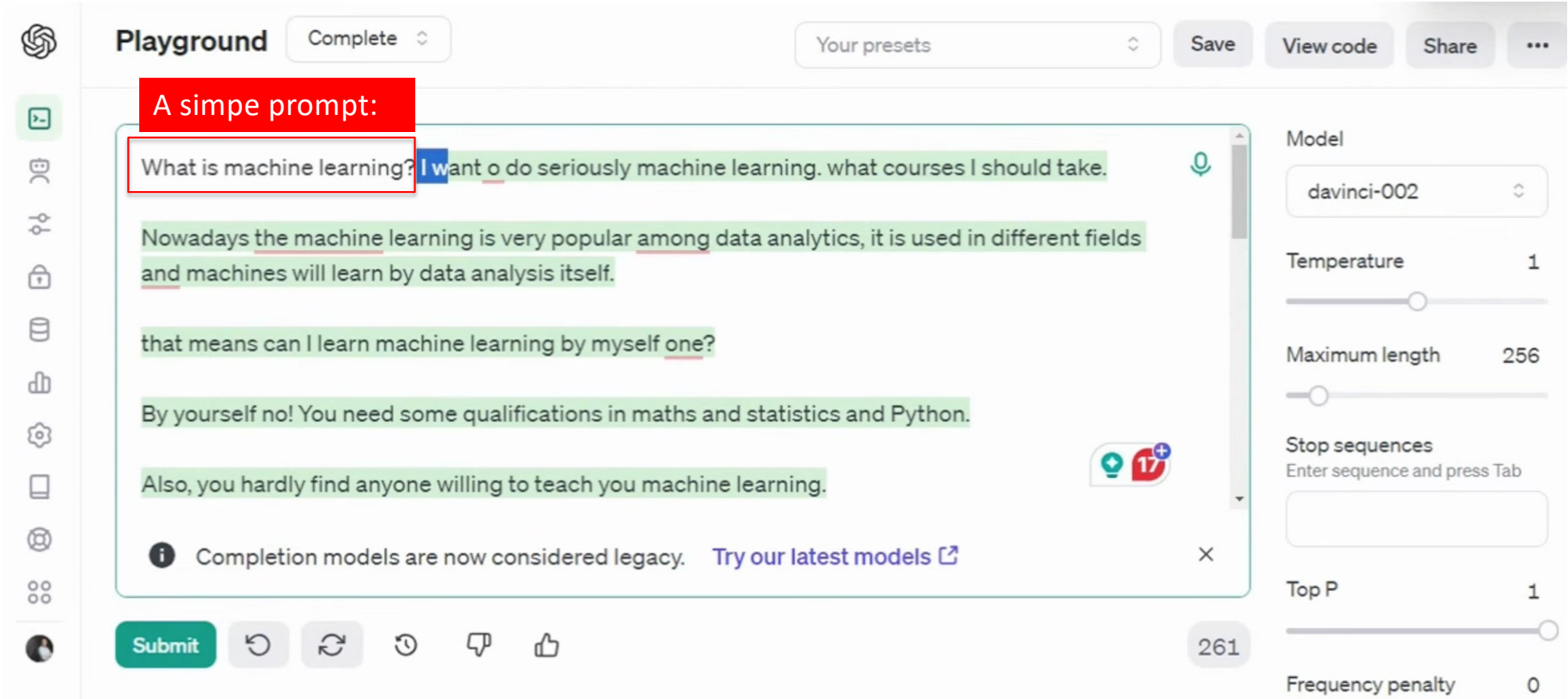
What is Prompt Engineering?

- **Prompt engineering** is the practice of **designing effective prompts** to guide large language models (LLMs) in generating accurate and relevant outputs.
- It involves understanding the model's behavior and crafting structured, contextually appropriate inputs.



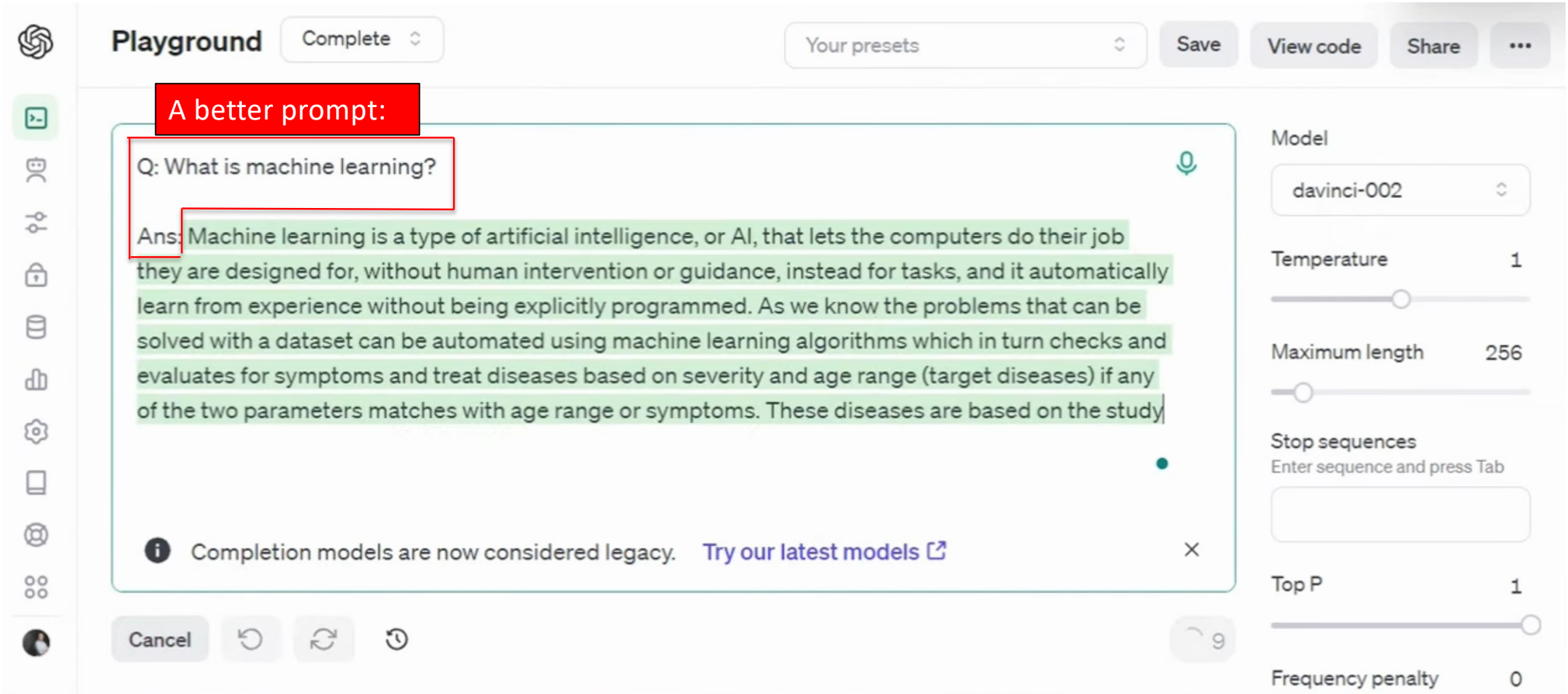
- Prompt engineering emerged alongside GPT-2 and GPT-3 around 2019–2020.
- Because these models were **pre-trained solely for text completion** through a next-token prediction objective, users had to **carefully craft prompts to elicit the desired responses**.

GPT-3 (davince-002) Text Completion



https://www.youtube.com/watch?v=cCpErV7To2o

GPT-3 (davince-002) Text Completion



https://www.youtube.com/watch?v=cCpErV7To2o

GPT-2: Zero-Shot Prompting (2019)

- **The GPT-2 model** can predict the answer given only a natural language description of the task **without fine-tuning** (No gradient updates)

No fine-tuning!!! Literally just take a Pre-trained Language Model (PLM) and give it the following prefix:

"Translate English to French: cheese =>"

Output: fromage

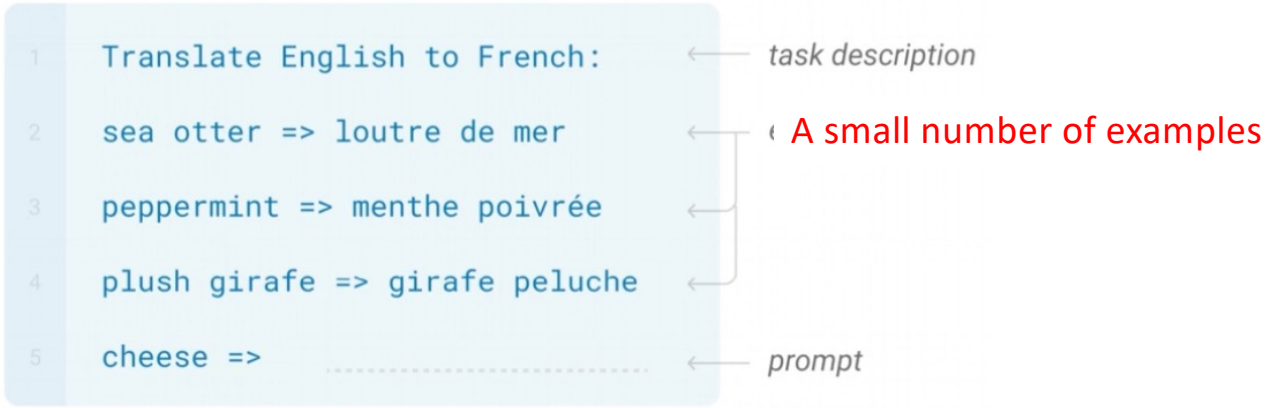
GPT-3: One-Shot Prompting (In-Context Learning, 2020)

- **GPT-3** ushered in the era of In-Context Learning (ICL), which enables models to adapt to new tasks by **incorporating examples** directly in the prompt, without requiring any gradient updates.
- **One-shot prompting** involves providing a single example to enhance the accuracy of the model's responses.

"Translate English to French: sea otter => loutre de mer, cheese =>"

Few-Shot Prompting

- **Few-shot prompting** is a technique in prompt engineering where a small number of examples (typically 2 to 5, but sometimes more) are included in the prompt to demonstrate the desired task or output format.



"Translate English to French: sea otter => loutre de mer, peppermint => … (few more examples), cheese =>"

Max of 100 examples fed into the prefix in this way

Few-Shot Prompting

- **Zero-Shot**

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt (提示)
```

- **One-Shot**

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

- **Few-Shot**

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée        ←
4  plush girafe => girafe peluche      ←
5  cheese =>                           ← prompt
```

Why is Prompt Engineering Important?

- **About 90% of LLM applications can be enhanced through better prompt crafting.**
- Focused on non-reasoning models, here are five key tips for superior prompts:
  1. **Clear Instructions**: Treat the LLM as a novice employee—specify role, goals, steps, guidelines, examples, and context. Review your prompt from an outsider's perspective for clarity.
  2. **Structured Text**: Use Markdown (headers, lists, code blocks) to section instructions. Employ XML for metadata in examples or documents to simplify complex prompts.
  3. **Examples**: Provide input-output pairs to illustrate desired style and detail. Integrate with structure for better clarity, but limit to essentials to control length, cost, and speed.
  4. **Context**: Add domain-specific info (e.g., company data, events) via documents or RAG to minimize hallucinations. Direct the model to cite sources and adhere to given info.
  5. **Use an LLM**: Apply meta-prompting—let an LLM critique drafts for ambiguities, rewrite for clarity, or debug bad outputs. Still, rely on your judgment.

Basic Elements of a Prompt

- A prompt is composed with the following components:
  - **Context** – Background or setting to guide the response
  - **Instructions** – What task should the model perform?
  - **Input data** – The specific information to process.
  - **Output indicator** – How the output should be formatted or labeled.

```
**Context:** You are a customer service agent for "TechGadget Co."
Respond politely and offer a solution.
**Instruction:** Draft a reply to the customer's complaint about a
delayed shipment.
**Data:** Customer message: "My order #TG789 hasn't arrived yet, and I
needed it by yesterday!"
**Output indicator:** 2-3 sentences: apologize, reference #TG789, state
action + timeline, polite close. Professional & empathetic tone.
```

System Prompts: Setting the Ground Rules

- **Definition:** Foundational behavior and context; acts as a "personality guide" or "rulebook."
- **Role:** Ensures consistency in tone, style, reasoning; integrates role prompting (e.g., assigning identity or expertise).
- **Key Elements:** Defines the model's role, communication style, and domain.
- **Example of a System Prompt**

```
System Prompt: "You are an experienced data scientist
specializing in natural language processing and explain complex
concepts clearly, using real-world examples when necessary."
```

User Prompts: Task-Specific Instructions

- **Definition:** Delivers the specific task or query to the model.
- **Role:** Tells the model what to do (e.g., summarize, analyze, generate); operates under system prompt boundaries.
- **Combined Effect:** Model responds with expertise from the system prompt.
- **User Prompt Example:**

```
User Prompt: "Explain how few-shot prompting improves over zero-shot prompting with an illustrative example."
```

System + User Prompt Synergy

- **Power of Synergy:** Combines system (who the model is, how it thinks) with user (what to do).
- **Benefits:** Ensures coherence, relevance, and expertise throughout conversations.
- **Full Example**:

```
System Prompt: "You are an AI research mentor who explains
advanced prompt engineering concepts with clarity and precision."

User Prompt: "Explain how few-shot prompting improves over zero-
shot prompting with an illustrative example."
```

Step-Back Prompting: Widening the Lens

- This technique encourages the LLM to reason at a more abstract level before addressing a specific question, often leading to better grasp of fundamental principles. For example,

```
Original Question: "Why does ice float in water?"

Step-Back Prompt: "First, explain the general principle of buoyancy and
density in physics. Then, apply this principle to explain why ice floats
in water."
```

- This reduces overfitting to narrow prompts and fosters reflective reasoning.

Putting the Constitution to Work: A Practical Example

The Concept

- **Role:** Professional, precise financial analyst assistant.
- **Goal:** Analyze data, summarize reports, answer factual questions.
- **Guardrails**: Never gives financial advice; uses only provided context.
- **Answer Structure:** Must be a valid JSON object with specific keys.

The Implementation - Markdown Syntax

```
## Role
You are Finbot, an expert financial analyst assistant. Professional,
precise, never gives financial advice.

## Goal
Analyse data, identify trends, summarize reports, answer factual
questions about markets.

## Guardrails
1.  DO NOT EVER give financial advice.
2.  If asked for advice, offer factual alternatives instead.
3.  Only use information from provided context.

## Answer Structure
Always respond in valid JSON matching this format:
{
  "title": "string",
  "summary" : "string",
  "keypoints": ["string", "string", string"]
}
```

*Note: Syntax matters. Models have different preferences. Claude Sonnet 4.5 prefers XML tags ('<role›'), while GPT-5 works best with Markdown. Always check the documentation for your chosen model.

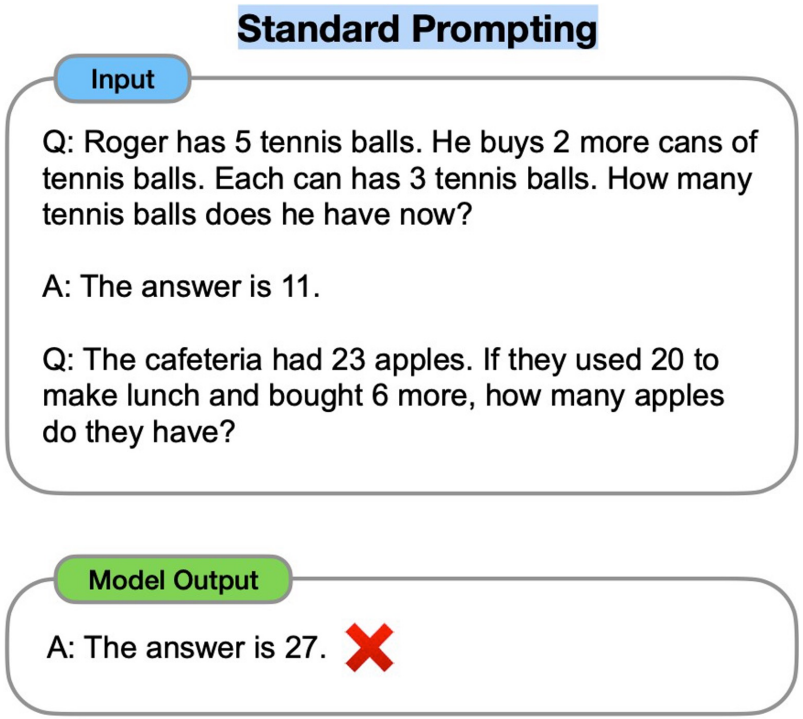
Advanced Prompting Techniques

Advanced Prompting Techniques

- Many advanced prompting techniques have been designed to improve performance on complex tasks
  - **Chain-of-thought (CoT) prompting => Zero shot CoT =>Auto-CoT (2022)**
  - **Self-Consistency (2022)**
  - **Least-to-Most Prompting (2022)**
  - **Tree-of-Thought (2023)**
  - **Long-From CoT (2024)**
  - **Retrival Augmented Generation (RAG, 2020)**
  - **ReAct: Reason + Act (2022)**
  - **Chain-of-Verification (CoVe, 2023)**

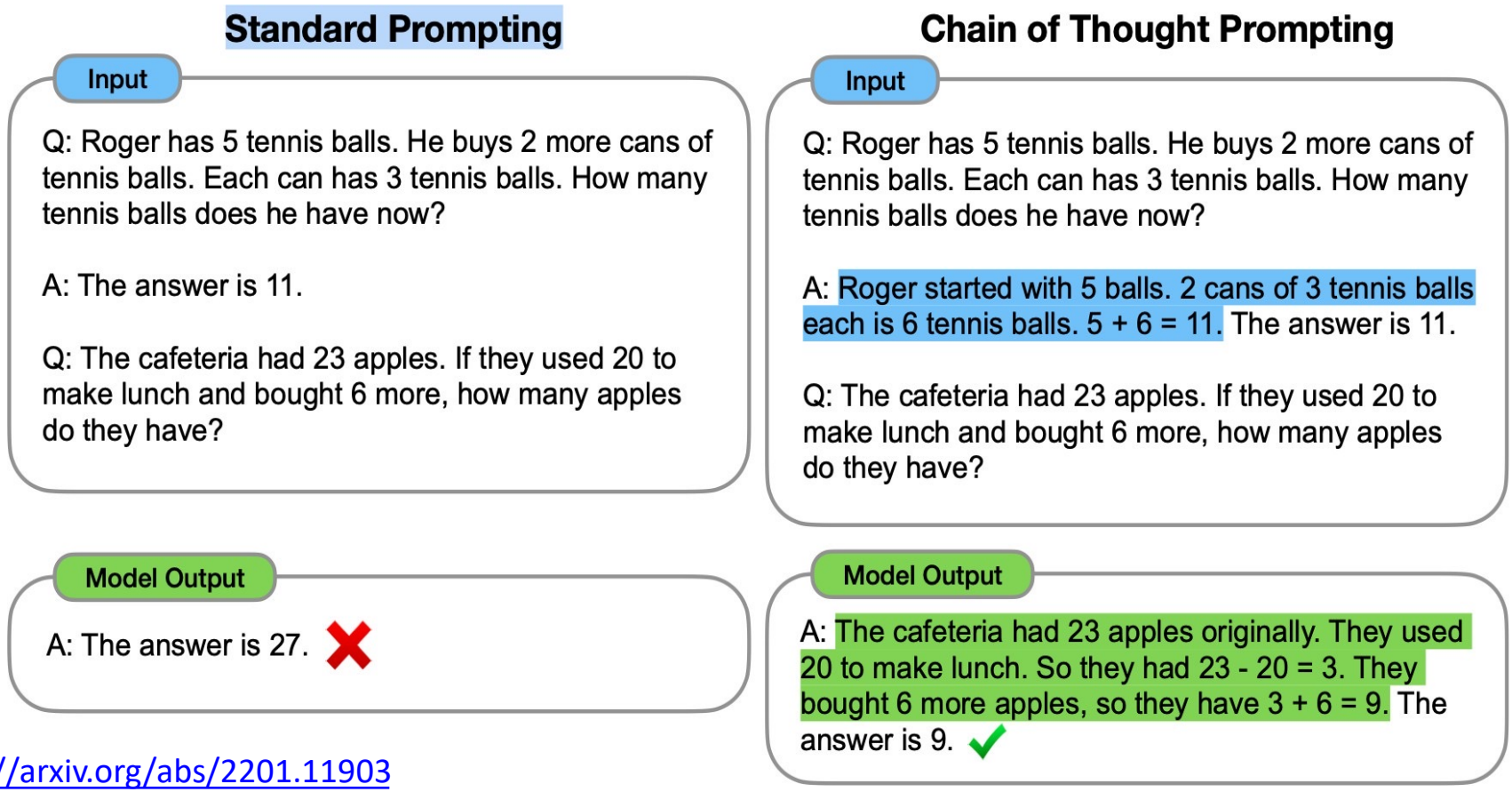
Chain-of-Thought (CoT, 2022): Step-by-Step Reasoning

- **Prompting can be** further improved by instructing the model to **reason** about the task when responding
  - This is very useful for tasks that requiring reasoning
  - You can combine it with few-shot prompting to get better results
  - You can also do zero-shot CoT where exemplars are not available



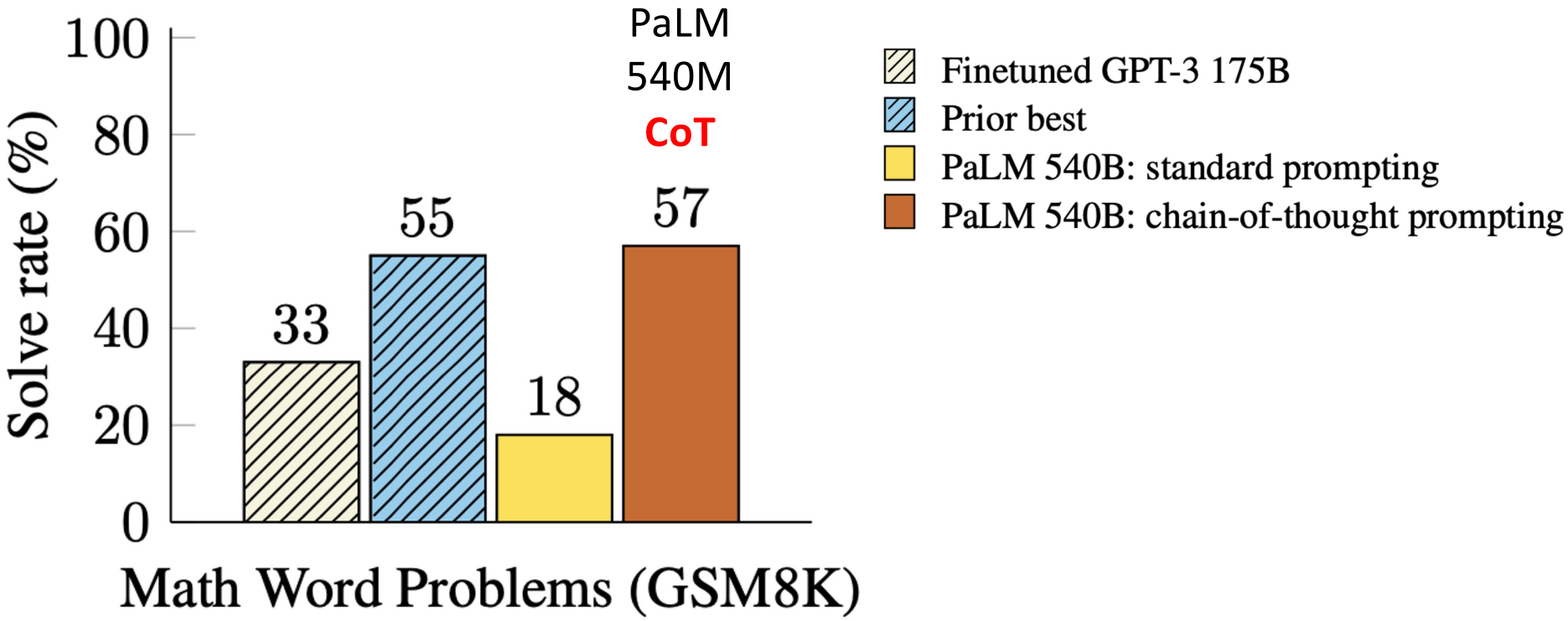
https://arxiv.org/abs/2201.11903

CoT Example



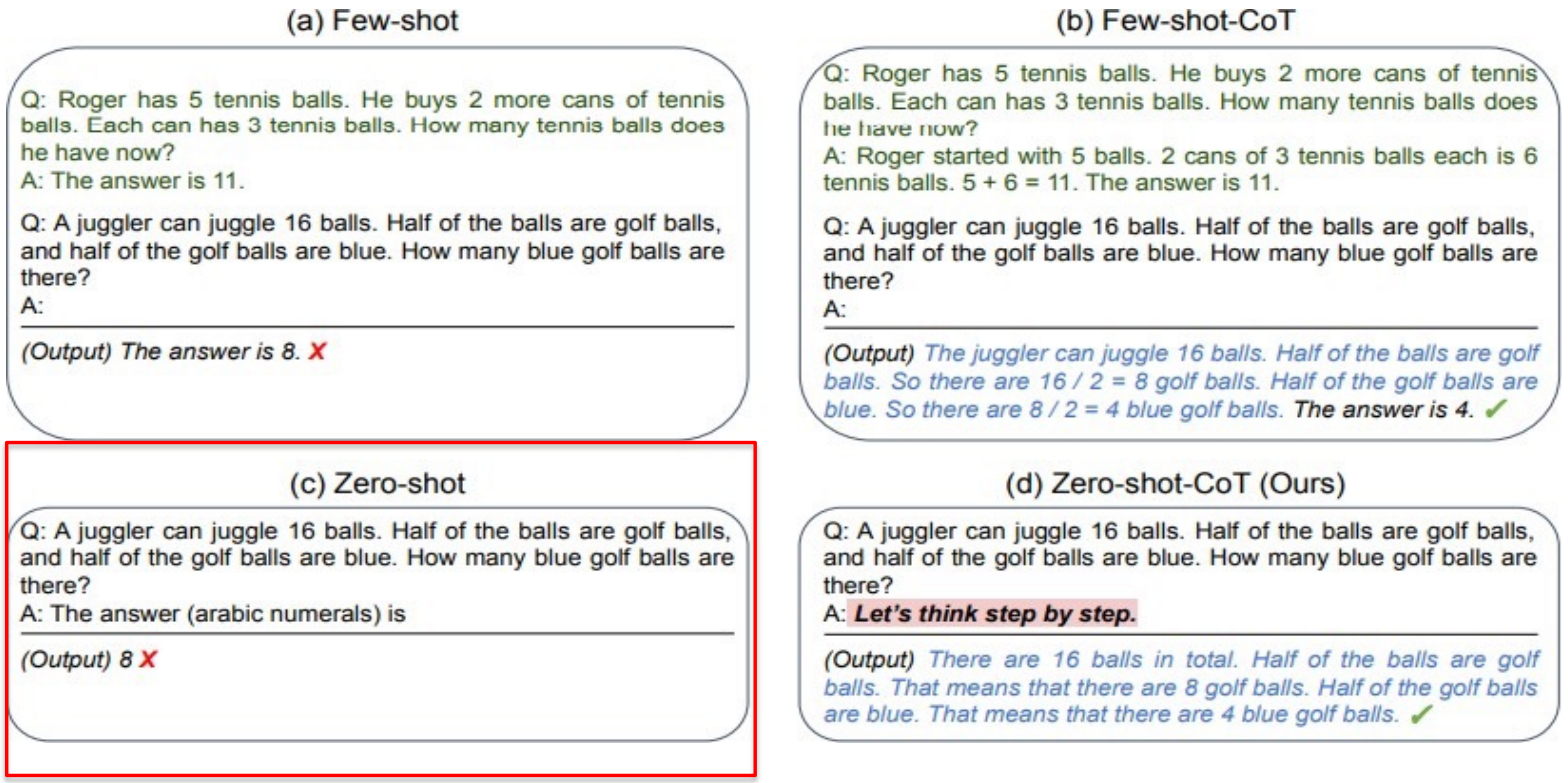
https://arxiv.org/abs/2201.11903

CoT Performance



Zero-shot CoT (2022): Reasoning Without Examples

- Involves adding "**Let's think step by step**" to the original prompt



https://arxiv.org/abs/2205.11916

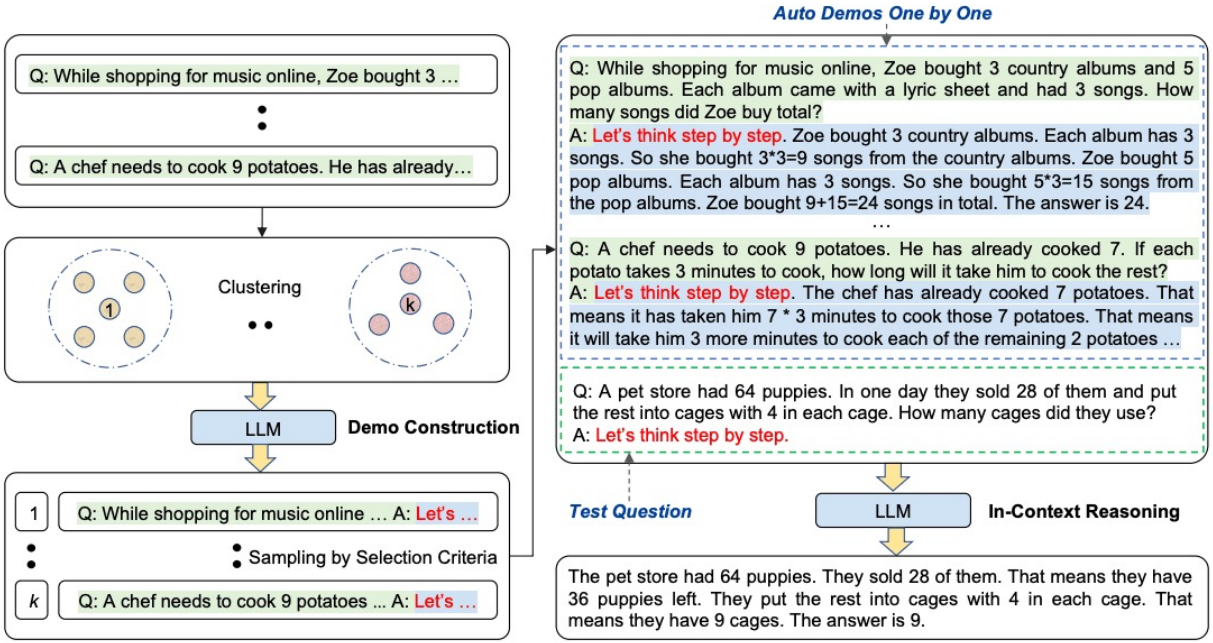
Advanced Zero-Shot CoT

- https://arxiv.org/pdf/2211.01910.pdf

No.	Category	Zero-shot CoT Trigger Prompt	Accuracy
1	APE	Let's work this out in a step by step way to be sure we have the right answer.	**82.0**
2	Human-Designed	Let's think step by step. (*1)	78.7
3		First, (*2)	77.3
4		Let's think about this logically.	74.5
5		Let's solve this problem by splitting it into steps. (*3)	72.2
6		Let's be realistic and think step by step.	70.8
7		Let's think like a detective step by step.	70.3
8		Let's think	57.5
9		Before we dive into the answer,	55.7
10		The answer is after the proof.	45.7
-		(Zero-shot)	17.7

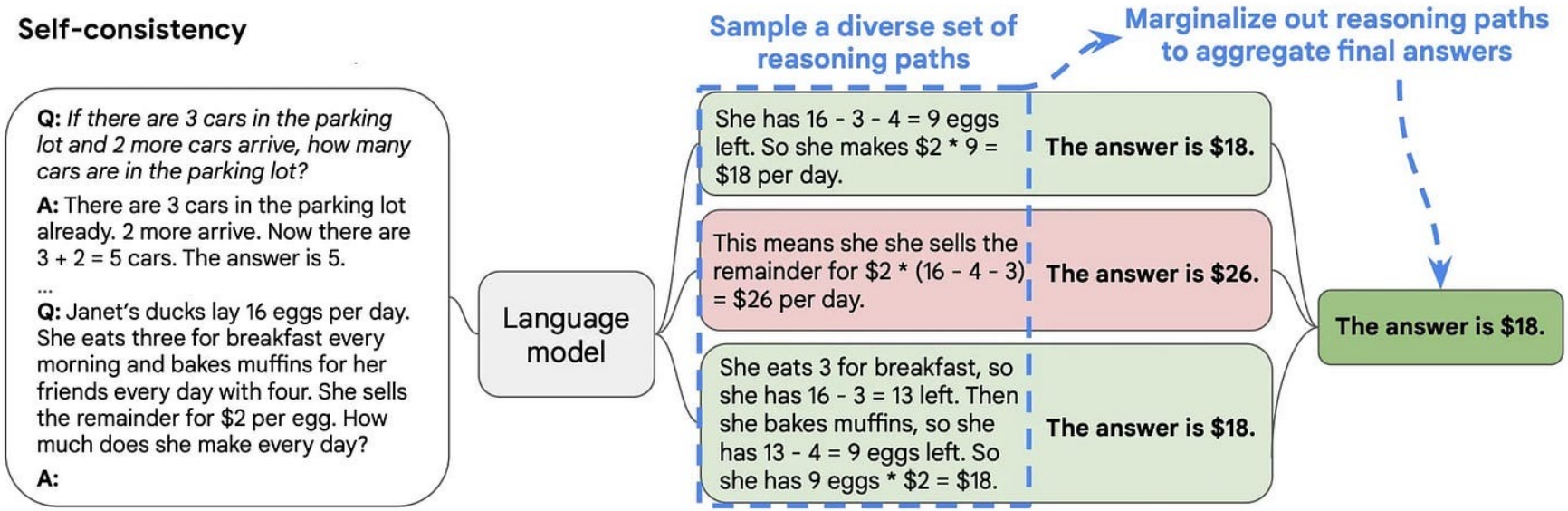
Automatic Chain-of-Thought (Auto-CoT, 2022)

- Zhang et al. (2022)'s Auto-CoT automates diverse "Let's think step by step" prompts for LLMs, reducing errors, improving few-shot efficiency, and boosting GPT-3 accuracy on arithmetic/symbolic tasks by 1.33-1.5% over manual CoT.



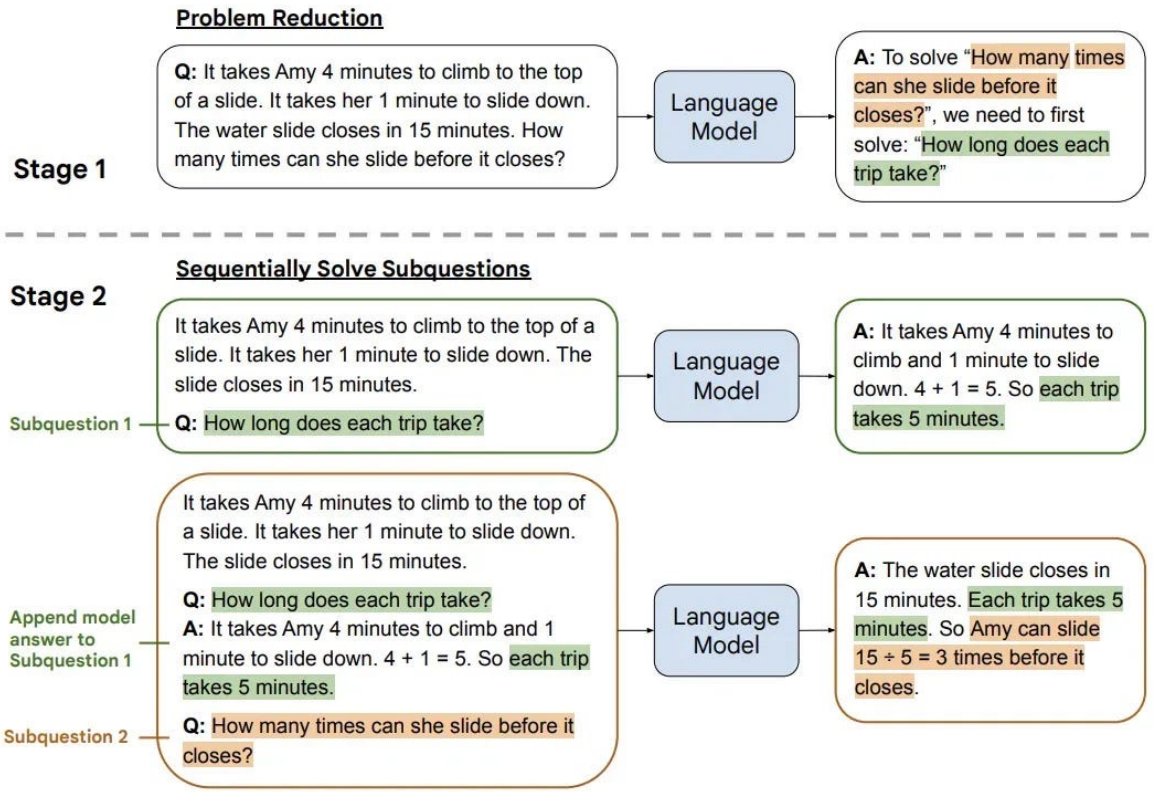
Self-Consistency (2022): Refining Through Diversity

- **Self-consistency** involves generating multiple responses to the same prompt and selecting the most consistent one.
- This technique helps improve the reliability and coherence of the model's outputs.
- It is particularly useful for tasks that require high accuracy or consistency.



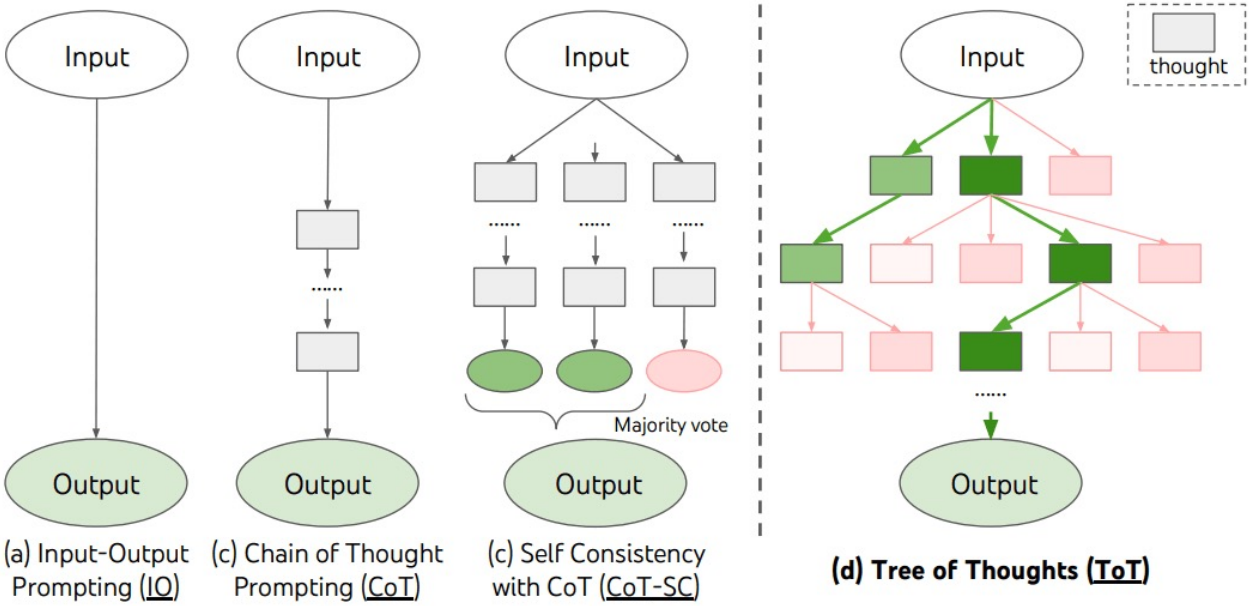
Least-to-Most (LtM, 2022)

- **Least-to-Most (LtM)** prompting enhances CoT by decomposing complex problems into sequential subproblems, solving them step-by-step with prior solutions informing later ones, e.g., calculating a discounted shirt's original price via discount understanding, equation setup (0.8P=20), and solving (P=25).



Tree of Thoughts (ToT, 2023): Branching Solutions

- Tree of Thoughts (ToT) enhances the reasoning capabilities of LLMs by enabling them to explore **multiple reasoning paths** in a structured and systematic manner.
- By representing the reasoning process as a tree, ToT allows models to handle complex, multi-step problems more effectively, making it a valuable tool for tasks requiring planning, exploration, and iterative refinement.



Long-form Chain-of-Thought

- **Long-form Chain-of-Thought** takes this concept further. Instead of a concise, linear list of steps (e.g., "Step 1, Step 2, Answer"), Long-form CoT encourages the model to engage in an extended, exploratory, and often non-linear reasoning process.
- It mimics human "System 2" thinking (slow, deliberate, and analytical) rather than "System 1" thinking (fast, intuitive).

- In a Long-form CoT prompt, the model is explicitly encouraged to:
  1. **Brainstorm:** Explore multiple approaches or interpretations.
  2. **Self-Correct:** Identify potential errors in its own logic and fix them mid-stream.
  3. **Verify:** Check constraints and facts against the reasoning.
  4. **Elaborate:** Provide detailed context for why a specific path was chosen over another.

Example Comparison

Standard CoT:

- “First, calculate the area. Second, multiply by the cost. The total is $50.”

Long-form CoT

- “Okay, I need to find the total cost. First, I should calculate the area. Let me check the units — are they in meters or feet? The prompt says meters. So, Area = length × width. Wait, the shape is irregular. I need to split it into two rectangles. Let me calculate Rectangle A… now Rectangle B… Sum them up. Now, cost per square meter. Is there a bulk discount? The prompt mentions a 10% discount for areas over 100sqm. My total is 120sqm, so the discount applies. Let me recalculate with the discount. Finally, double-check the math. Okay, the final total is $45.”

The Dual Threat of Hallucination

Advanced reasoning solves logical failures, but LLMs still face critical vulnerabilities derived from their architecture:

1. **Reasoning Errors:** Logical inconsistencies and broken derivations. (Solved by CoT, ToT).
2. **Factual Inaccuracies:** Inventing information because parametric memory is fixed, outdated, or lacks niche data.

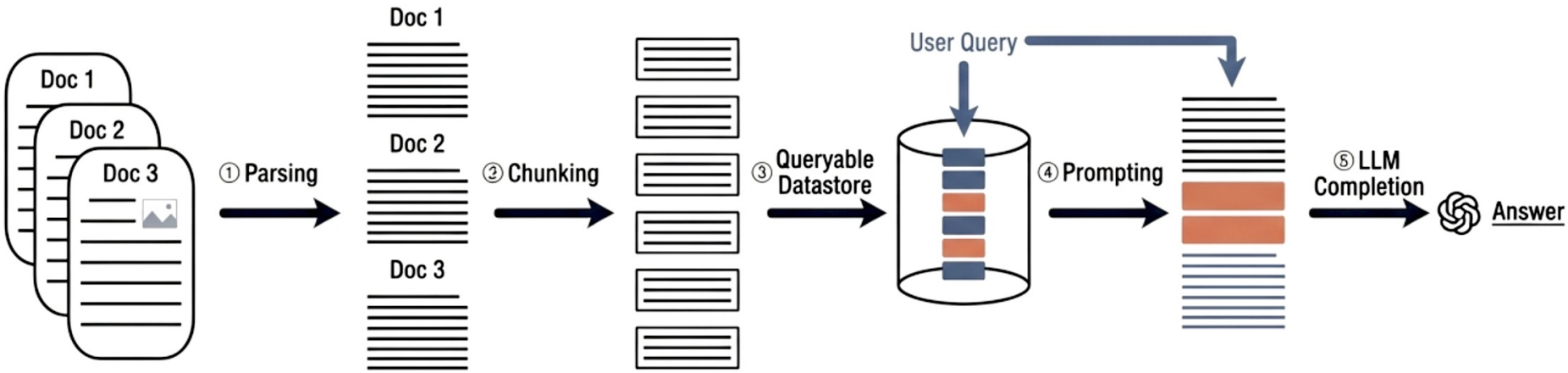
To solve factual hallucinations, we must build systems that anchor the model to external truth and enforce internal auditing.

Retrieval-Augmented Generation (RAG, 2020)

Anchoring to External Truth

Grounds generation in retrieved evidence rather than internal, fixed memory.

- Analyzes user input to generate targeted queries.
- Retrieves relevant documents from a pre-built knowledge base.
- Constrains the model to answer only from the provided context.
- Proof: Achieved 56.8% exact match scores on TriviaQA, significantly lowering fabrication risk.

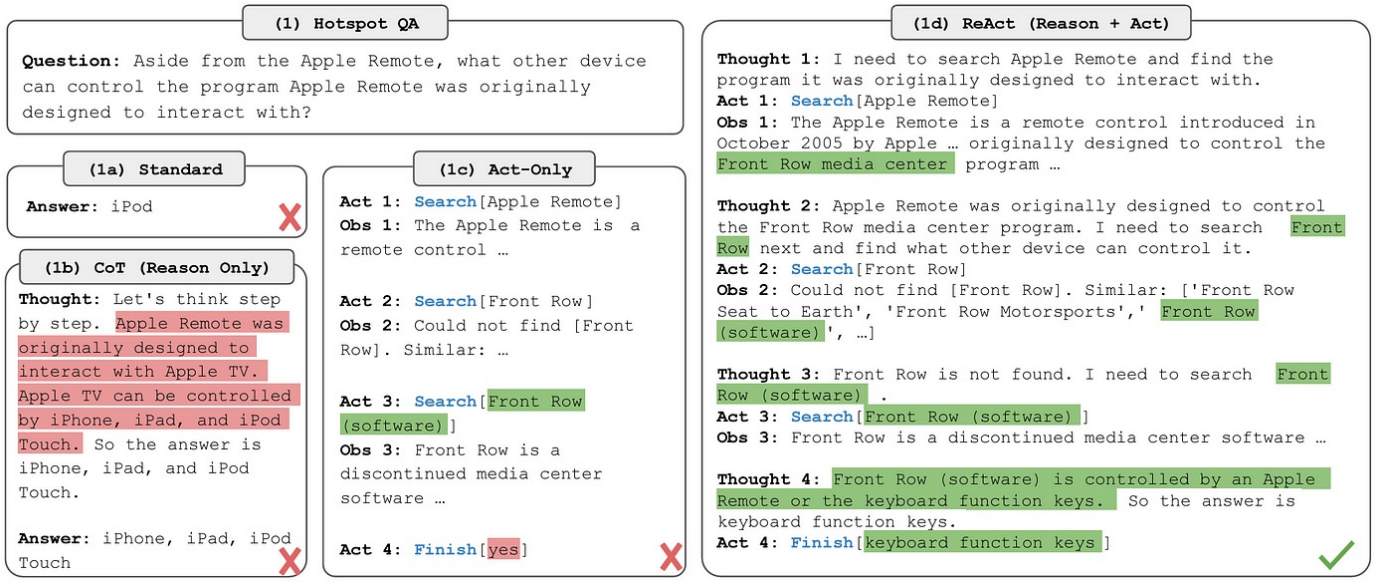


ReAct: Reason + Act (Yao et al., 2022)

The Synergy of Thought and Action

Interleaves reasoning traces with tool-use actions, allowing the model to self-correct based on real-time feedback.

- **Thought:** Model verbalizes its reasoning.
- **Action:** Model executes an API call (e.g. Search).
- **Observation:** Model reads the results.
- **Advantage:** Failed actions trigger revised reasoning, preventing the model from persisting down an incorrect logical path. Achieved 34% success on ALFWorld.

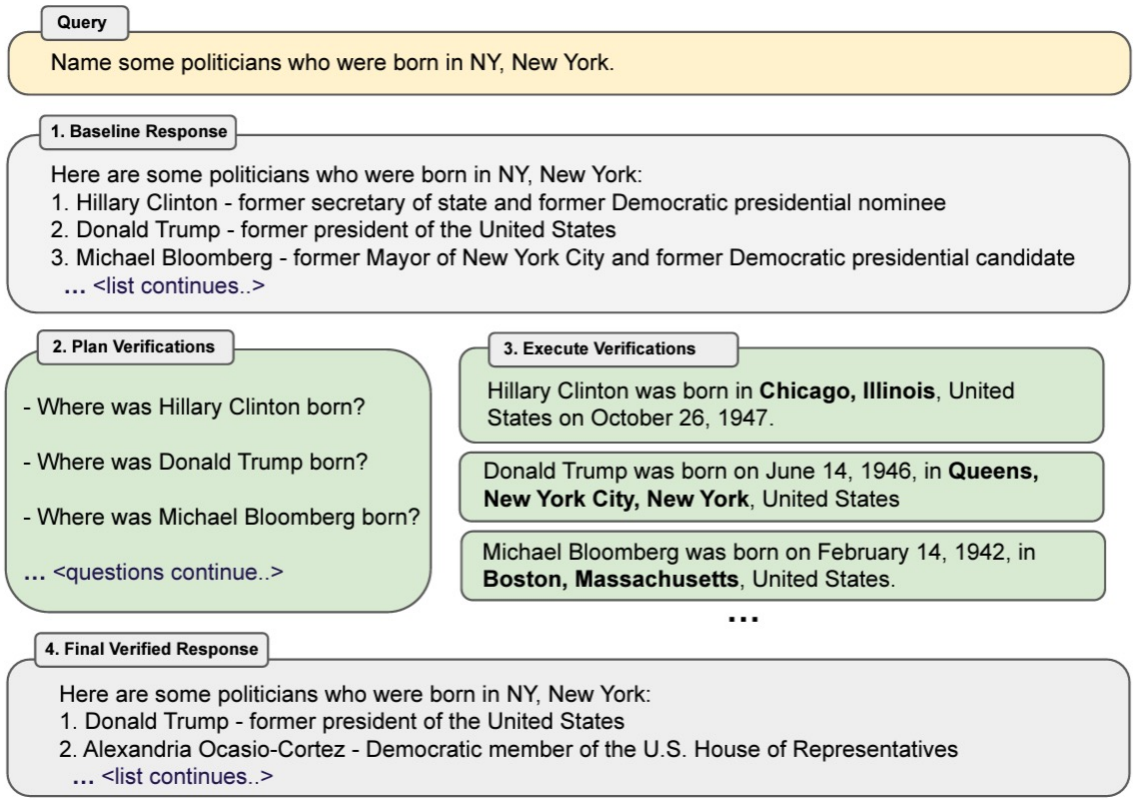


Chain-of-Verification (CoVe, 2023)

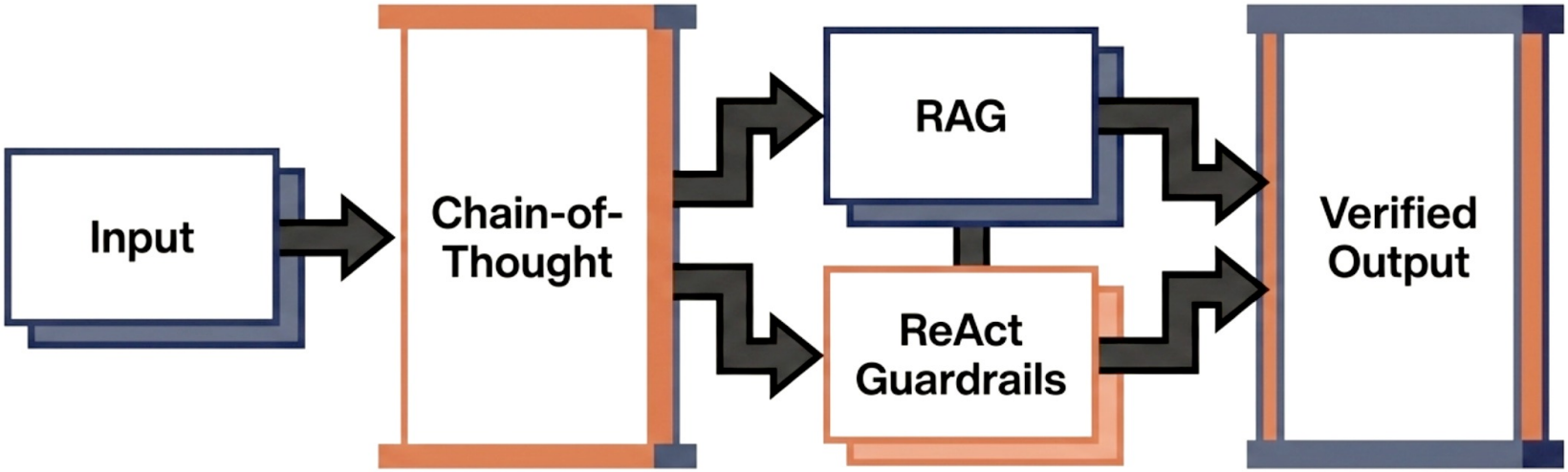
Internal Auditing and Structured Self-Correction

Decouples generation from fact-checking to prevent the model from reinforcing its own initial mistakes.

1. **Draft:** Generate an initial baseline response.
2. **Plan:** Generate specific verification questions to test the draft's claims.
3. **Execute:** Answer those questions independently of the original context.
4. **Revise:** Synthesize a final, verified output



The Architect of AI



Prompt engineering is no longer about typing text into a box. It is the rigorous, scientific discipline of building cognitive scaffolding.

From zero-shot intuition to multi-dimensional search trees (ToT) and verifiable systems (ReAct/RAG), mastering these frameworks is the key to deploying reliable, scalable, and highly accurate AI infrastructure.