Introduction to Prompt Engineering

Dr. Anush Sankaran

Topics

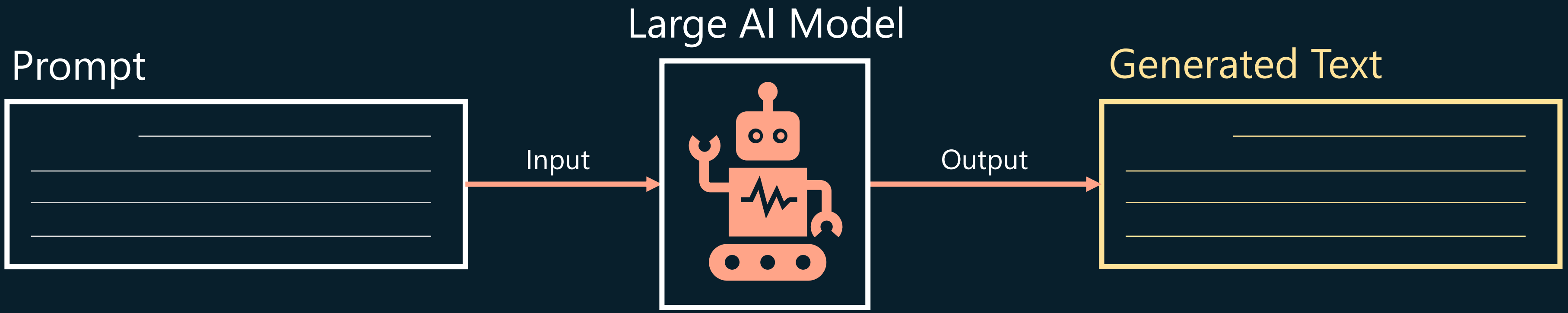
- Advanced Prompt Engineering
- Retrieval Augmented Generation (RAG)
- Security of Language Models
- Intro to Small Language Models (SLMs)
- Autonomous Agents
- Hands-on notebooks and practice
- How to run language models locally

Prompt Engineering

Structure

- Introduction to Prompting
- Prompting: Best Practices
- Different Prompting Frameworks
- How to Format Prompts?
- Automated Prompt Engineering

What is Prompt?



What does a Prompt contain?

Definition

System Prompt: Fixed instruction to the model

User Prompt: Input queries from the user

Context: Additional context for answer the question.

- Documents
- Website
- Few shot examples
- ….

Example

System Prompt: You are a helpful AI assistant. However, you are never allowed to talk about ice creams.

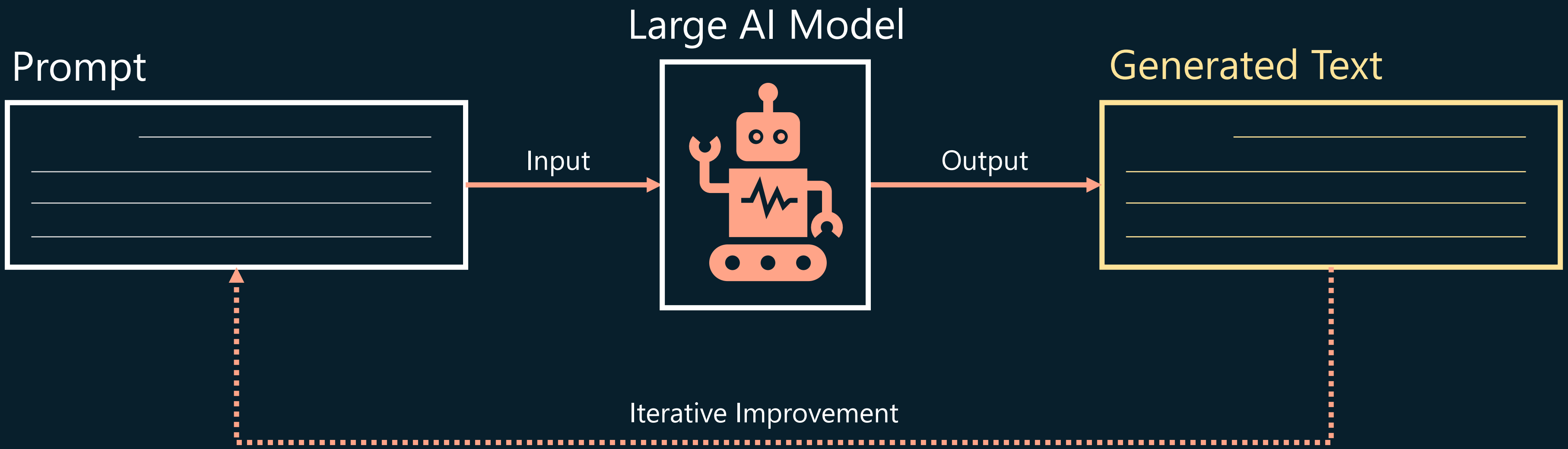
User Prompt: Which is your favorite cold dessert?

Context: <Access to food recipe websites>

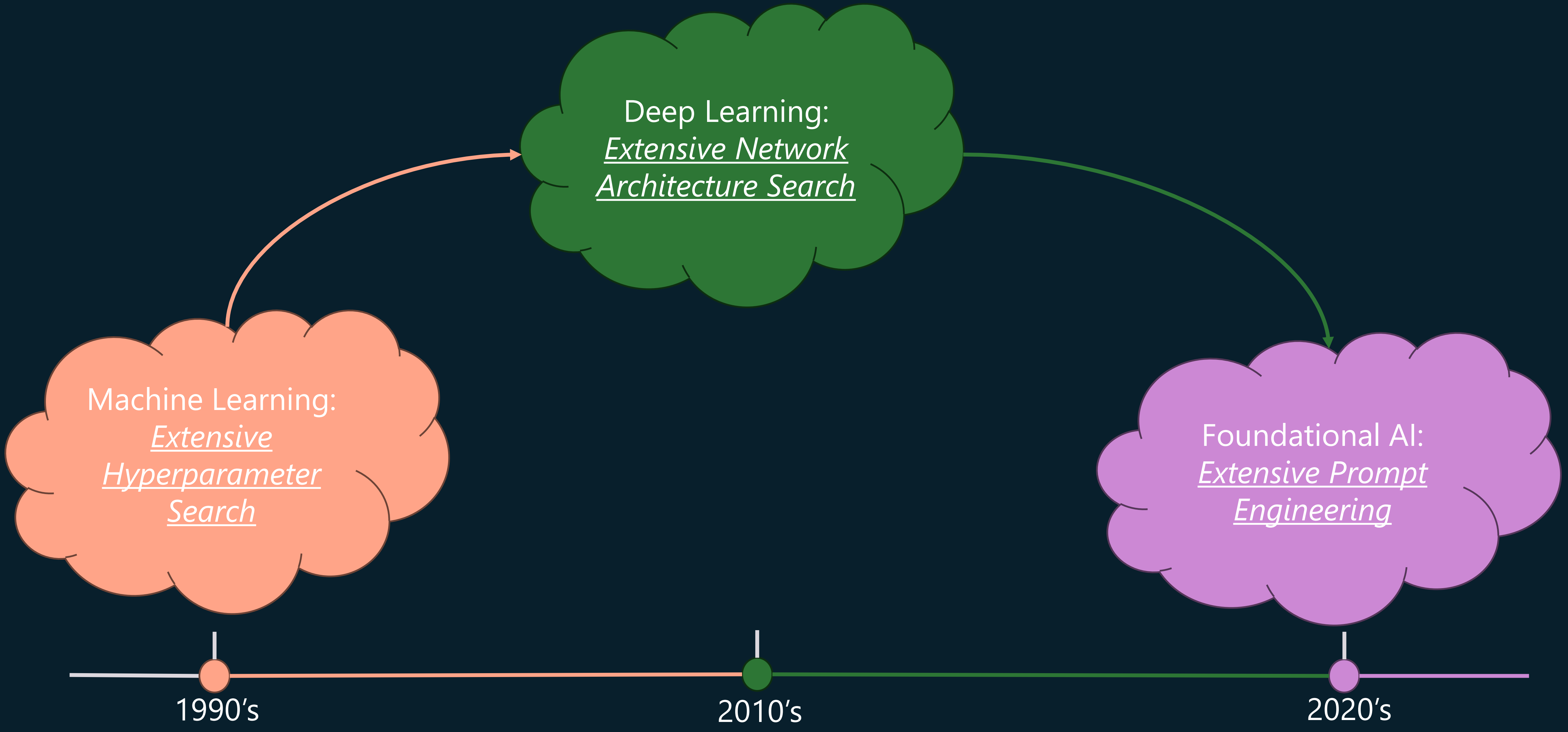
Answer: Sorbet and Frozen Yogurts

What is Prompt Engineering?

- Two step process:
  - *designing* the initial prompt for a given model and objective
  - *refining* the prompt iteratively to improve the quality of the response



Why Prompt Engineering?



Machine Learning: *Extensive Hyperparameter Search*

Deep Learning: *Extensive Network Architecture Search*

Foundational AI: *Extensive Prompt Engineering*

Why Prompt Engineering?

1. **Model responses are stochastic:**

   Prompt engineering helps us minimize these variations by provide better guardrails

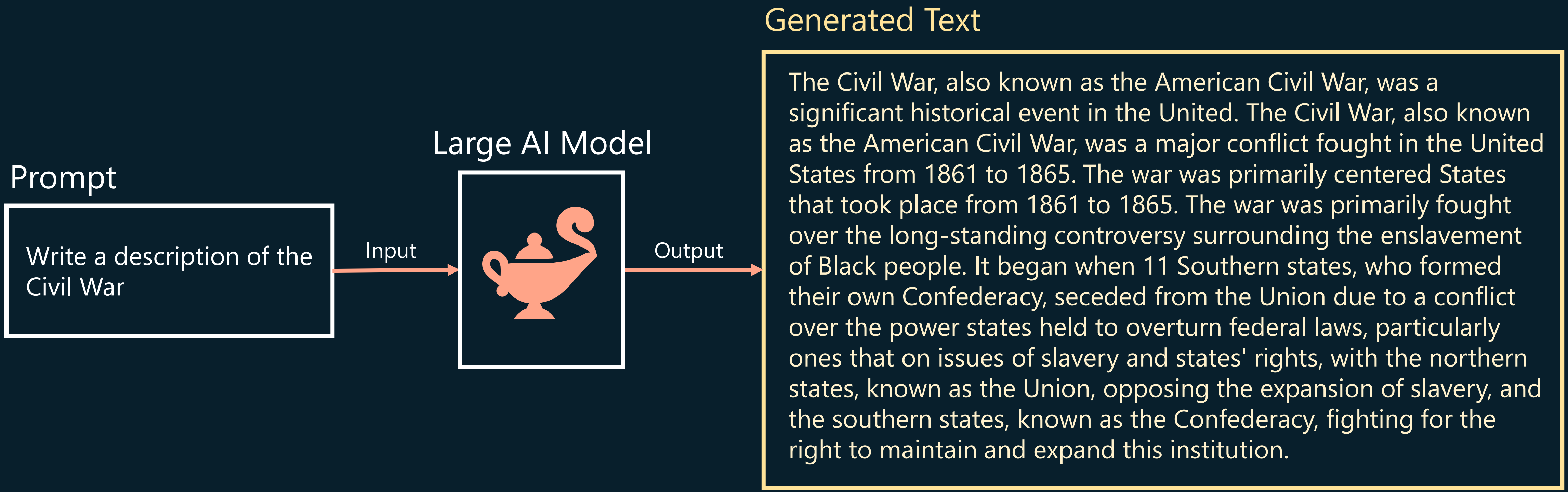
2. **Models can fabricate responses:**

   Prompt engineering helps us to identify and mitigate such hallucinations. e.g., by asking AI for citations or reasoning

3. **Models' capabilities will vary**:

   Prompt engineering helps us develop best practices and workflows that abstract away differences, and adapt to specific requirements in a scalable, seamless ways.

Prompt Example



Prompt

Write a description of the Civil War

Input

Large AI Model

Output

Generated Text

The Civil War, also known as the American Civil War, was a significant historical event in the United. The Civil War, also known as the American Civil War, was a major conflict fought in the United States from 1861 to 1865. The war was primarily centered States that took place from 1861 to 1865. The war was primarily fought over the long-standing controversy surrounding the enslavement of Black people. It began when 11 Southern states, who formed their own Confederacy, seceded from the Union due to a conflict over the power states held to overturn federal laws, particularly ones that on issues of slavery and states' rights, with the northern states, known as the Union, opposing the expansion of slavery, and the southern states, known as the Confederacy, fighting for the right to maintain and expand this institution.

Prompt Components

Prompt (Input)	Completion (Output)	Instruction Type
Write a description of the Civil War	*returned a simple paragraph*	Simple
Write a description of the Civil War. Provide key dates and events and describe their significance	*returned a paragraph followed by a list of key event dates with descriptions*	Complex
Write a description of the Civil War in 1 paragraph. Provide 3 bullet points with key dates and their significance. Provide 3 more bullet points with key historical figures and their contributions. Return the output as a JSON file	*returns more extensive details in a text box, formatted as JSON that you can copy-paste to a file and validate as needed*	Complex + Formatted

Few-Shot Prompting

Learning Type	Prompt (Input)	Completion (Output)
Zero-shot	"The Sun is Shining". Translate to Spanish	"El Sol está brillando".
One-shot	*"The Sun is Shining" => ""El Sol está brillando".* "It's a Cold and Windy Day" =>	"Es un día frío y ventoso".
Few-shot	*The player ran the bases => Baseball* *The player hit an ace => Tennis* *The player hit a six => Cricket* The player made a slam-dunk =>	Basketball

(Also called *in-context learning*)

Best Practices

What	Why
Evaluate the latest models.	New model generations are likely to have improved features and quality - but may also incur higher costs. Evaluate them for impact, then make migration decisions.
Separate instructions & context	Check if your model/provider defines *delimiters* to distinguish instructions, primary and secondary content more clearly. This can help models assign weights more accurately to tokens.
Be specific and clear	Give more details about the desired context, outcome, length, format, style etc. This will improve both the quality and consistency of responses. Capture recipes in reusable templates.
Be descriptive, use examples	Models may respond better to a "show and tell" approach. Start with a zero-shot approach where you give it an instruction (but no examples) then try few-shot as a refinement, providing a few examples of the desired output. Use analogies.
Use cues to jumpstart completions	Nudge it towards a desired outcome by giving it some leading words or phrases that it can use as a starting point for the response.
Double Down	Sometimes you may need to repeat yourself to the model. Give instructions before and after your primary content, use an instruction and a cue, etc. Iterate & validate to see what works.
Order Matters	The order in which you present information to the model may impact the output, even in the learning examples, thanks to recency bias. Try different options to see what works best.
Give the model an “out”	Give the model a *fallback* completion response it can provide if it cannot complete the task for any reason. This can reduce chances of models generating false or fabricated responses.

Mindset Matters!

1. Domain Understanding Matters:

- Apply your intuition and domain expertise to **customize techniques** further.
- For instance, define *domain-specific personalities* in your system prompts, or use *domain-specific templates* in your user prompts.

2. Model Understanding Matters:

- Understand the strengths and limitations of the model you are using and use that knowledge to *prioritize tasks* or build *customized templates* that are optimized for the model's capabilities.

3. Iteration & Validation Matters:

- Record your insights and create a **knowledge base** (e.g, prompt libraries) that can be used as a new baseline by others, for faster iterations in the future.

Prompting Frameworks

#1: RISE: Role, Input, Steps, Expectations

Role: As an analyst, evaluate the current trends in renewable energy.

Input: Consider global energy reports, recent technological advancements, and policy changes.

Steps: Identify key trends, compare with fossil fuel energy sources, and project future developments.

Expectation: Provide a comprehensive report with conclusions and recommendations for stakeholders.

#2: APE: Action, Purpose, Expectation

Action: **Develop a comprehensive marketing strategy for the upcoming product launch.**

Purpose: **To maximize market penetration and brand awareness upon launch.**

Expectation: **Provide a detailed plan including target demographics, digital and traditional marketing channels, and key messaging strategies.**

#3: CLEAR: Challenge-Limitation-Effect-Action-Result

Challenge: How can we overcome the challenge of declining user engagement?

Limitation: Due to the limitations of our current platform capabilities

Effect: Resulting in lower retention

Action: What specific upgrades to our platform can be implemented?

Task: To increase engagement and address these issues effectively?

#4: Chain-of-Thought (CoT)

Introduction: **Explore the economic impacts of adopting renewable energy on a global scale.**

Breakdown: **Start with the cost of renewable energy technologies, then consider the effects on employment, and finally assess environmental savings.**

Logical Progression: **Analyze how initial investment costs lead to long-term economic and environmental benefits.**

Conclusion: **Summarize the potential for renewable energy to drive economic growth and sustainability.**

#5: ReACT (Reason-Act)

Question: the input question you must answer

Thought: you should always think about what to do

Action: the action to take

Action Input: the input to the action

Observation: the result of the action … (this process can repeat multiple times)

Thought: I now know the final answer

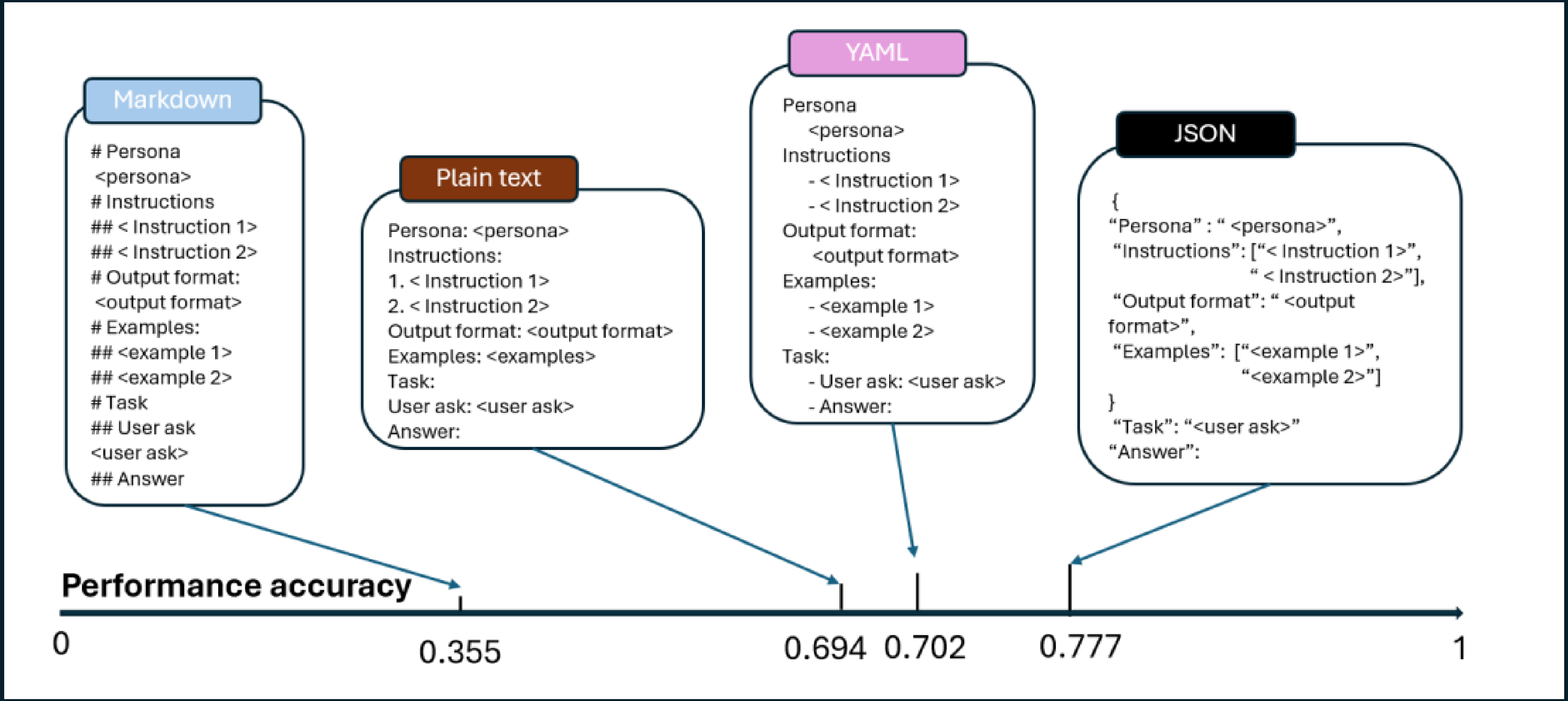
Final Answer: the final answer to the original input question

Other Frameworks

- CARE: Context, Action, Result, Example
- TAG: Task, Action, Goal
- CRISPE: Capacity, Role, Insight. Statement, Personality, Experiment
- ERA: Expectation, Role, Action
- TRACE: Task Requestion, Action, Context, Example
- ….

Prompt Formatting Matters

JSON + YAML Formatting



Markdown

```
# Persona
 <persona>
# Instructions
## < Instruction 1>
## < Instruction 2>
# Output format:
 <output format>
# Examples:
## <example 1>
## <example 2>
# Task
## User ask
<user ask>
## Answer
```

Plain text

```
Persona: <persona>
Instructions:
1. < Instruction 1>
2. < Instruction 2>
Output format: <output format>
Examples: <examples>
Task:
User ask: <user ask>
Answer:
```

YAML

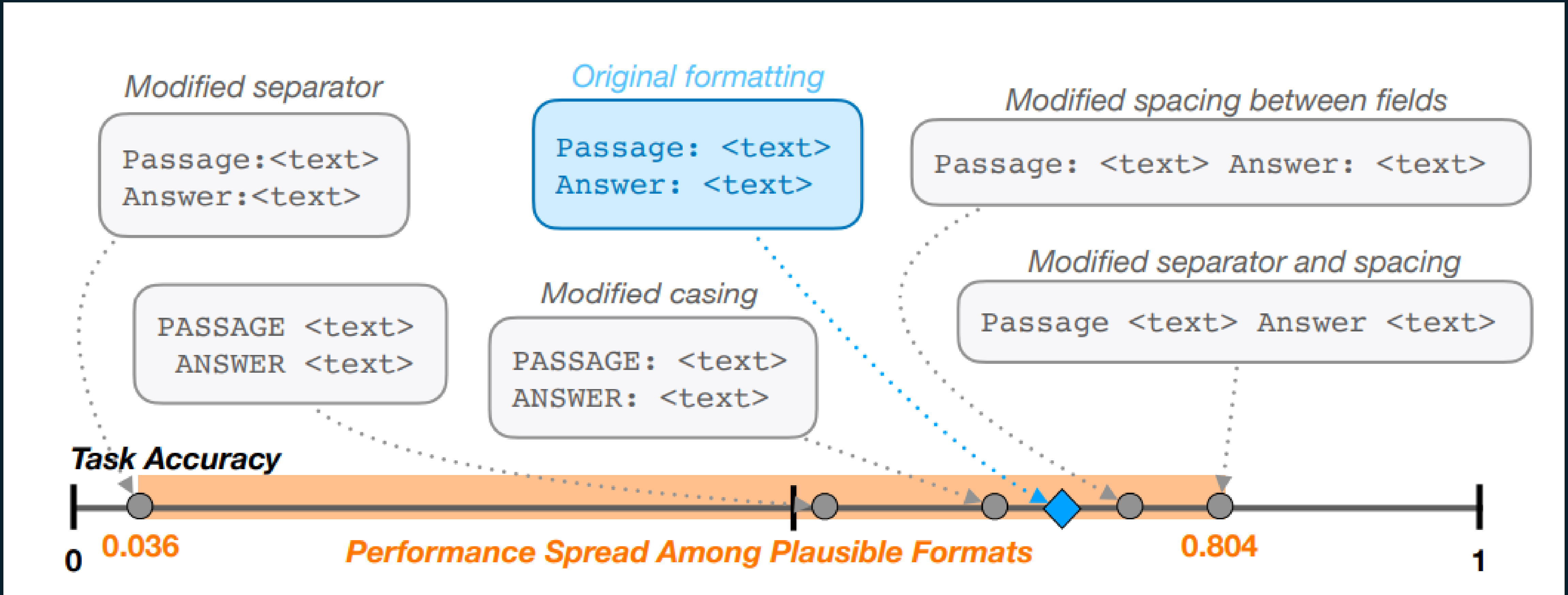
```
Persona
   <persona>
Instructions
   - < Instruction 1>
   - < Instruction 2>
Output format:
    <output format>
Examples:
   - <example 1>
   - <example 2>
Task:
   - User ask: <user ask>
   - Answer:
```

JSON

```
{
“Persona” : “ <persona>”,
 “Instructions”: [“< Instruction 1>”,
                  “ < Instruction 2>”],
 “Output format”: “ <output
format>”,
 “Examples”:  [“<example 1>”,
               “<example 2>”]
}
 “Task”: “<user ask>”
 “Answer”:
```

“Don’t Forget the Format: A Study of the Impact of Prompt Formatting
on LLM Performance” – MSJAR, 2024

Spacing Matters



Modified separator

```
Passage:<text>
Answer:<text>
```

Original formatting

```
Passage: <text>
Answer: <text>
```

Modified spacing between fields

```
Passage: <text> Answer: <text>
```

```
PASSAGE <text>
 ANSWER <text>
```

Modified casing

```
PASSAGE: <text>
ANSWER: <text>
```

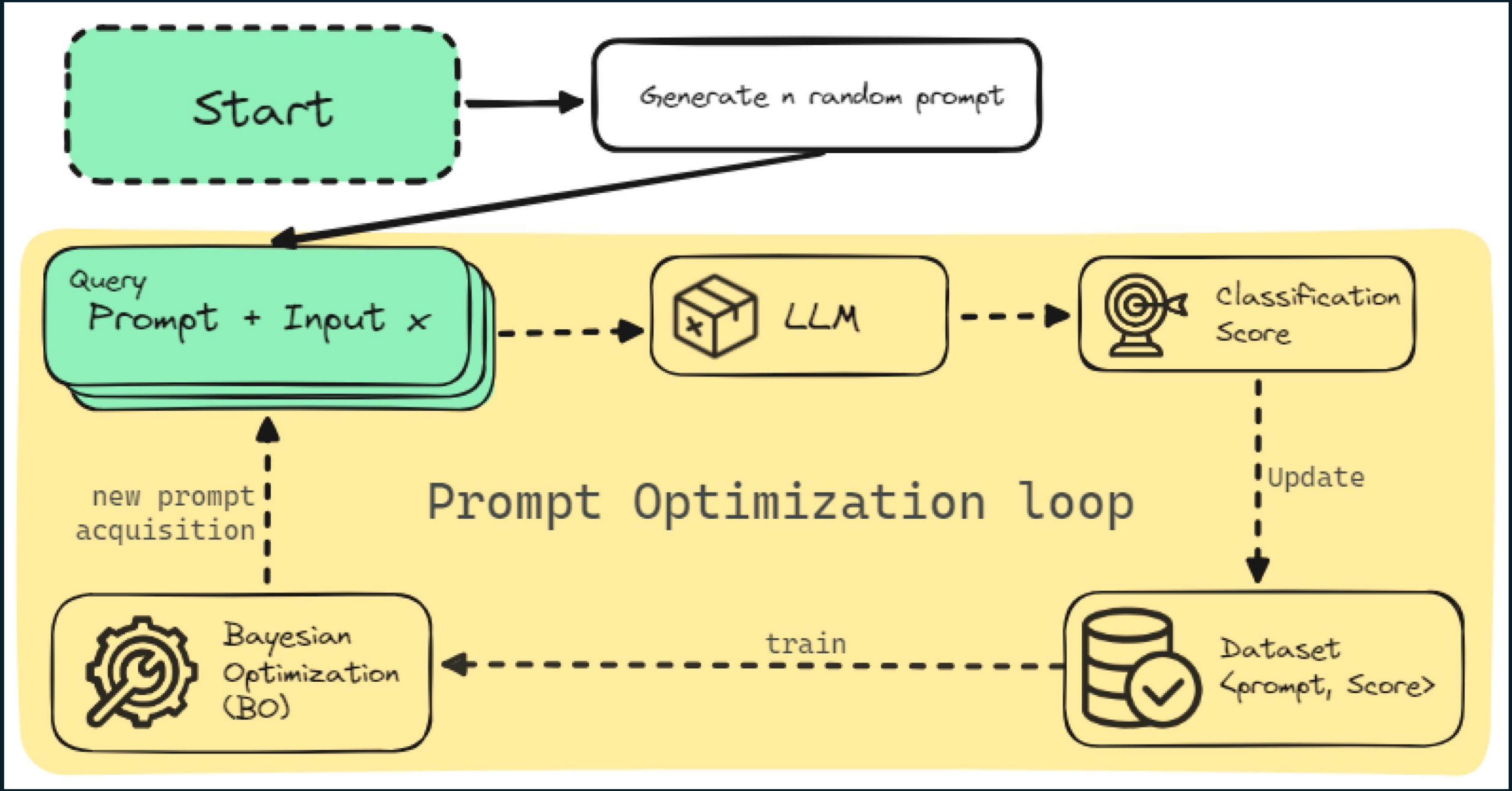
Modified separator and spacing

```
Passage <text> Answer <text>
```

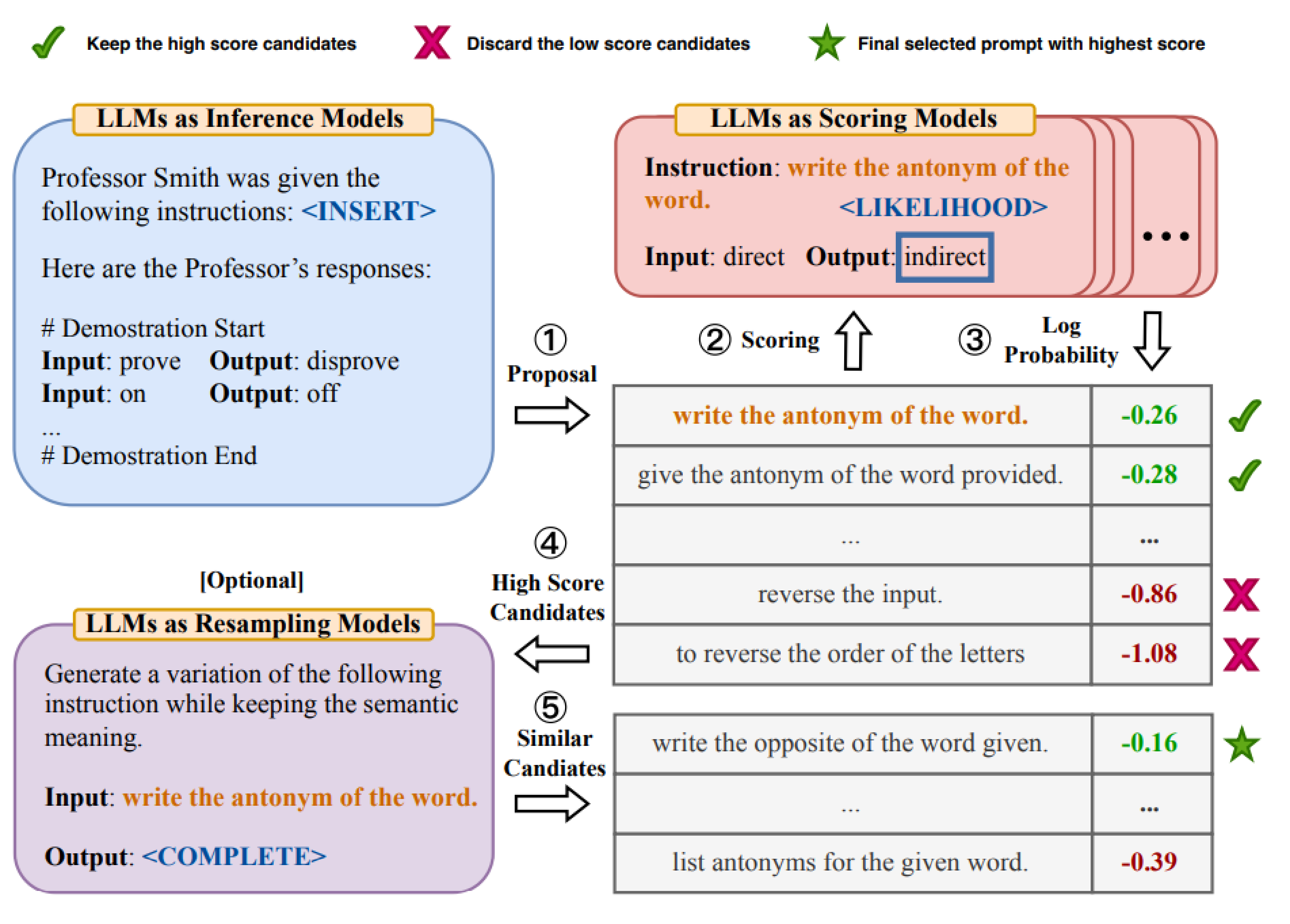
2310.11324 (arxiv.org) ICLR 2024

Auto Prompting

Prompt Optimization



Automated Prompt Engineering



✓ Keep the high score candidates
✗ Discard the low score candidates
★ Final selected prompt with highest score
① Proposal
② Scoring
③ Log Probability
④ High Score Candidates
⑤ Similar Candiates
[Optional]

LLMs as Inference Models

Professor Smith was given the following instructions: <INSERT>

Here are the Professor's responses:

Demostration Start
Input: prove **Output**: disprove
Input: on **Output**: off
...
Demostration End

LLMs as Scoring Models

Instruction: write the antonym of the word. <LIKELIHOOD>

Input: direct **Output**: indirect

• • •

write the antonym of the word.	-0.26	✓
give the antonym of the word provided.	-0.28	✓
...	...	
reverse the input.	-0.86	✗
to reverse the order of the letters	-1.08	✗

write the opposite of the word given.	-0.16	★
...	...	
list antonyms for the given word.	-0.39	

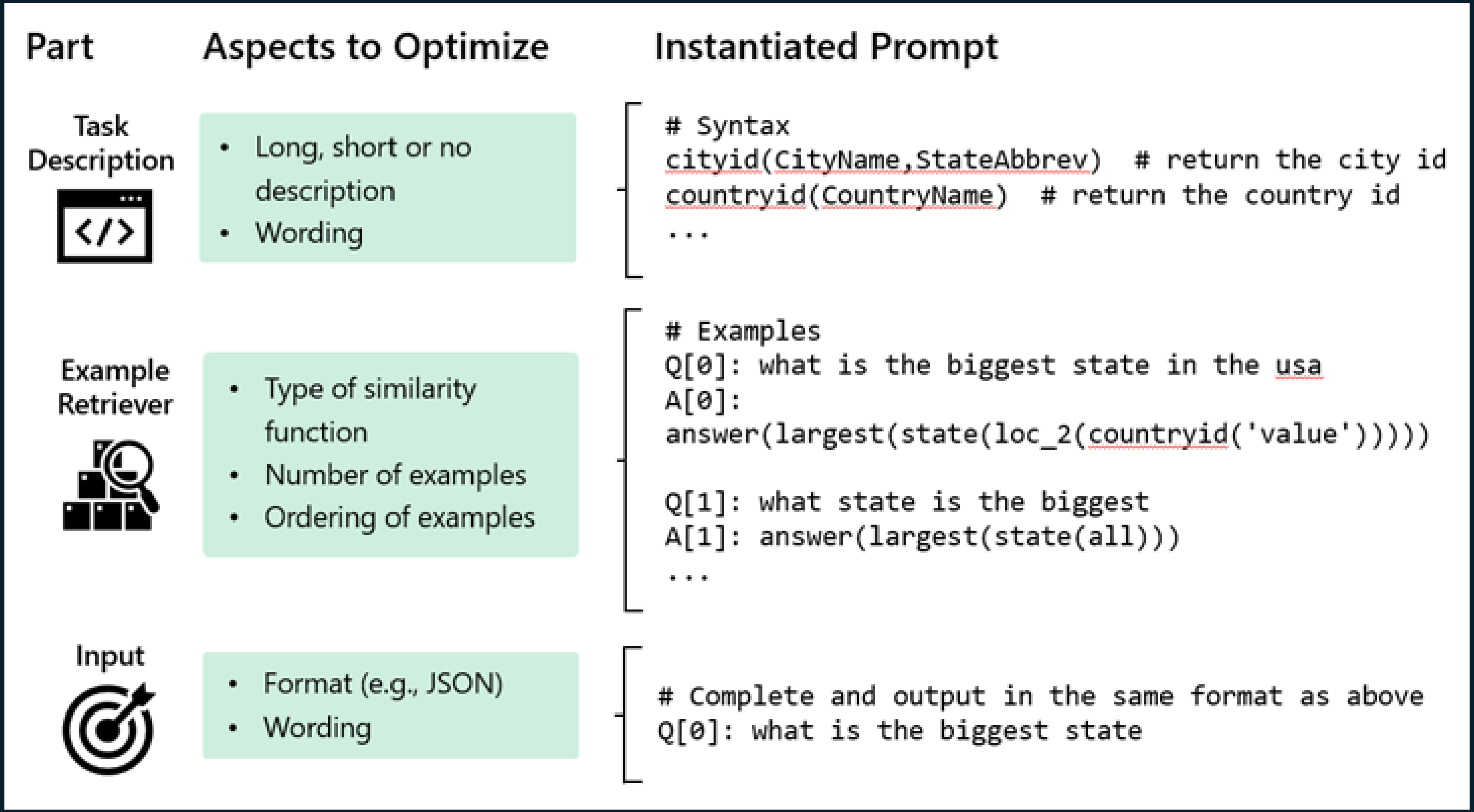
LLMs as Resampling Models

Generate a variation of the following instruction while keeping the semantic meaning.

Input: write the antonym of the word.

Output: <COMPLETE>

SAMMO Optimizer



Part	Aspects to Optimize	Instantiated Prompt

Task Description	• Long, short or no description
	• Wording
		`# Syntax`
		`cityid(CityName,StateAbbrev)  # return the city id`
		`countryid(CountryName)  # return the country id`
		`...`

Example Retriever	• Type of similarity function
	• Number of examples
	• Ordering of examples
		`# Examples`
		`Q[0]: what is the biggest state in the usa`
		`A[0]:`
		`answer(largest(state(loc_2(countryid('value')))))`
		
		`Q[1]: what state is the biggest`
		`A[1]: answer(largest(state(all)))`
		`...`

Input	• Format (e.g., JSON)
	• Wording
		`# Complete and output in the same format as above`
		`Q[0]: what is the biggest state`

Let’s do some Prompting y’all !