

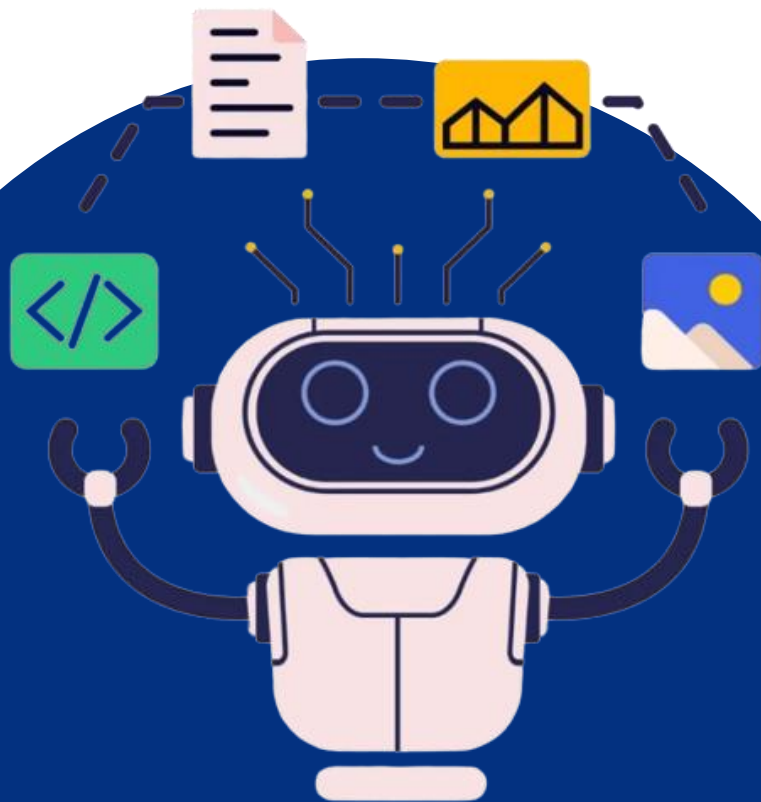
Developers Guide to Effective Prompt Engineering

Mr. Yogender Kumar
Knowledge Associate (AI team)
C-DAC, Bangalore





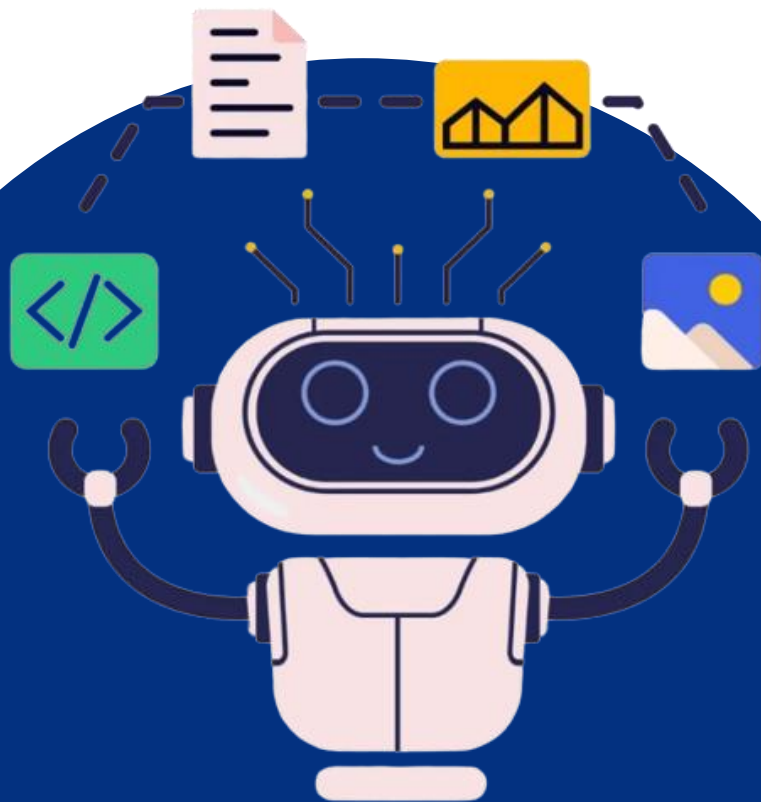
Outline



01. What is Prompt engineering?
02. Why Prompt engineering?
03. Analogies to Understand Prompt Engineering
04. Key Elements of Effective Prompt Engineering



Outline



05. Prompt Types

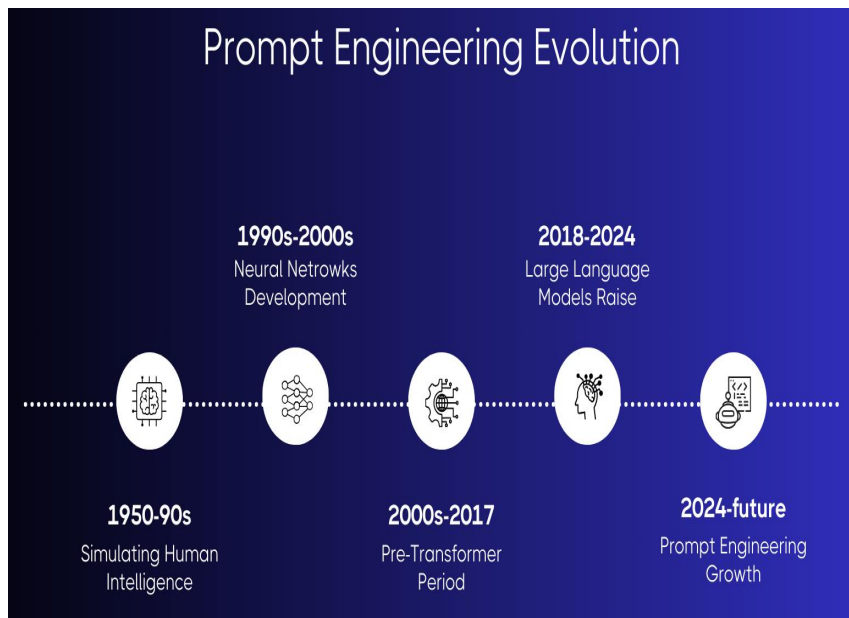
06. Tips for Writing Effective Prompts

07. Real-World Applications of Prompt Engineering

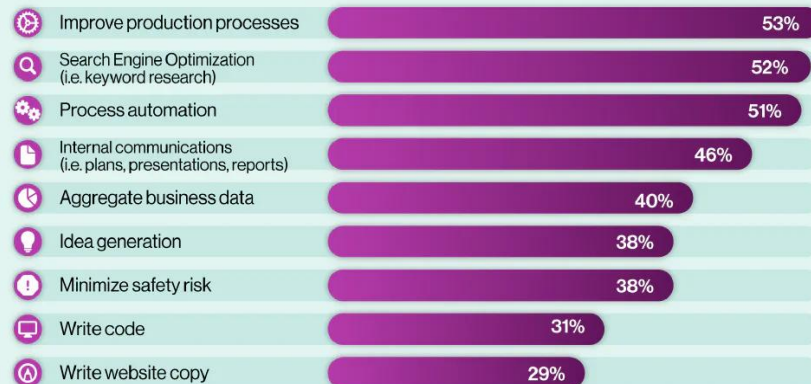
08. Automatic Prompt Optimization

Introduction to Prompt engineering

Prompt engineering is the art of **crafting precise instructions** for AI models to produce the **desired results**. It's a powerful tool for shaping AI's responses and unlocking its full potential.



Use of AI



Source: Forbes

Percentage of business owners who are using AI in this way

Why Prompt engineering?



Analogies to Understand Prompt Engineering

1

Giving Directions

Imagine telling a new driver, "Take me somewhere nice," vs. giving specific instructions, "Turn left at the next intersection, then take the first right."

2

Ordering at a Restaurant

If you say, "I want something with chicken," you'll get anything. But if you specify, "Grilled chicken with a side of vegetables, no sauce," you'll get exactly what you want.

3

Giving Instructions for a Puzzle

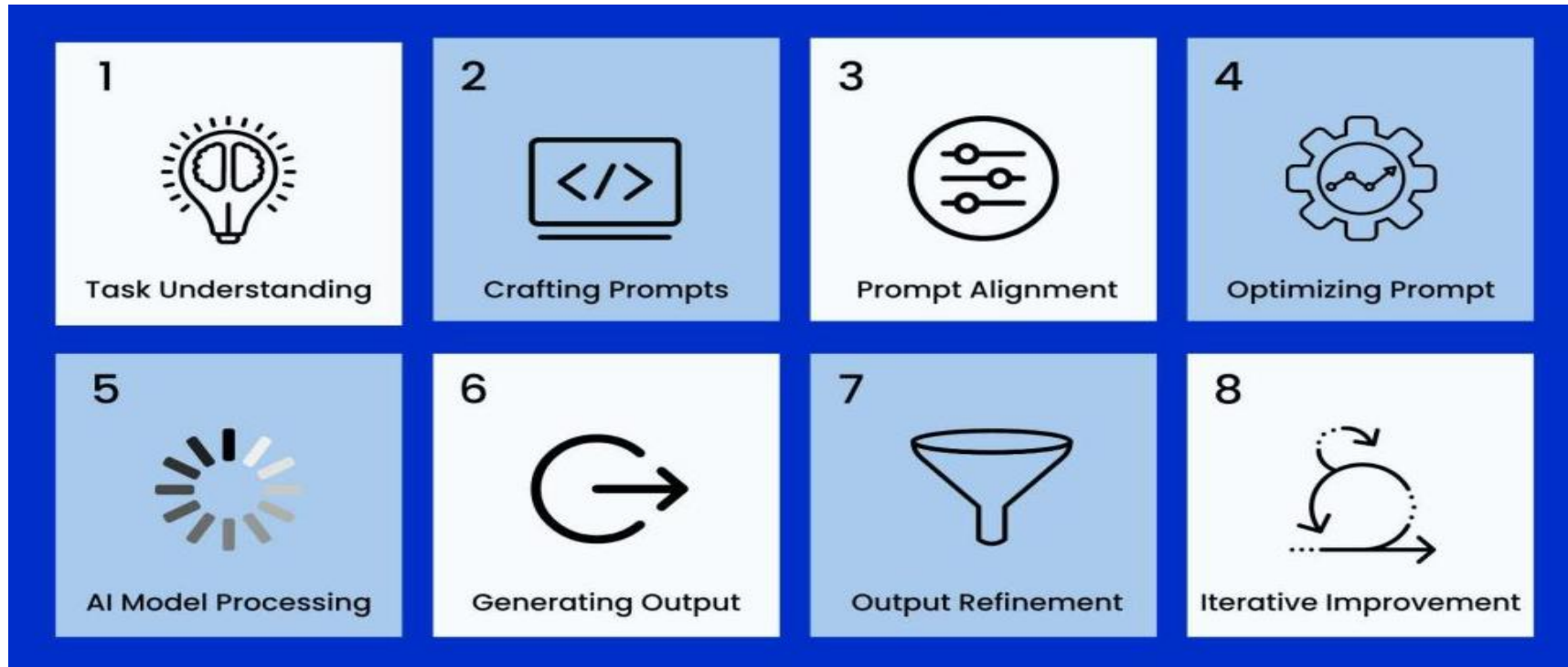
Without instructions, assembling a puzzle can be frustrating. But breaking it down step-by-step, "Start with the corners, then match the edges," helps.

4

Talking to a Librarian

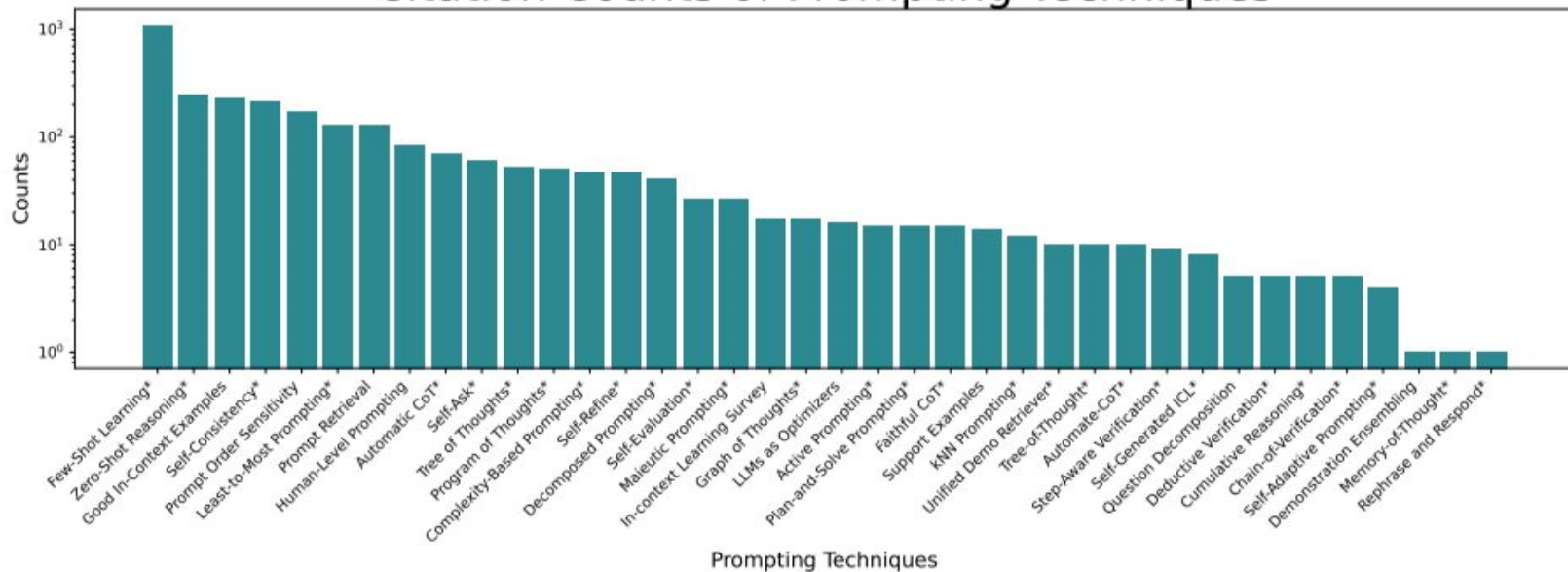
If you ask a librarian for "a good book," the response will be vague. But asking for "books on psychology related to mindfulness" leads to more relevant recommendations.

Key Elements of Effective Prompt Engineering

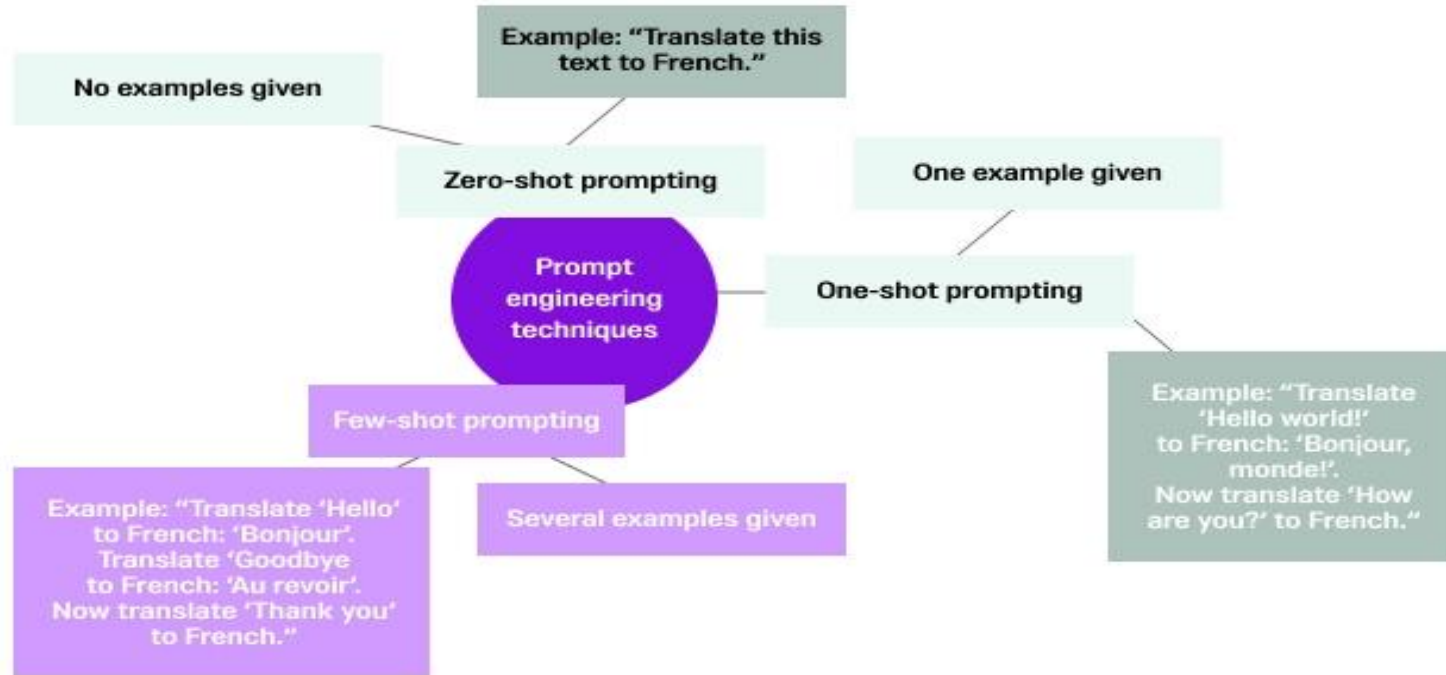


Prompt Types

Citation Counts of Prompting Techniques



Prompt Types



Prompt Types: Chain of thought

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

Chain-of-Thought Prompting

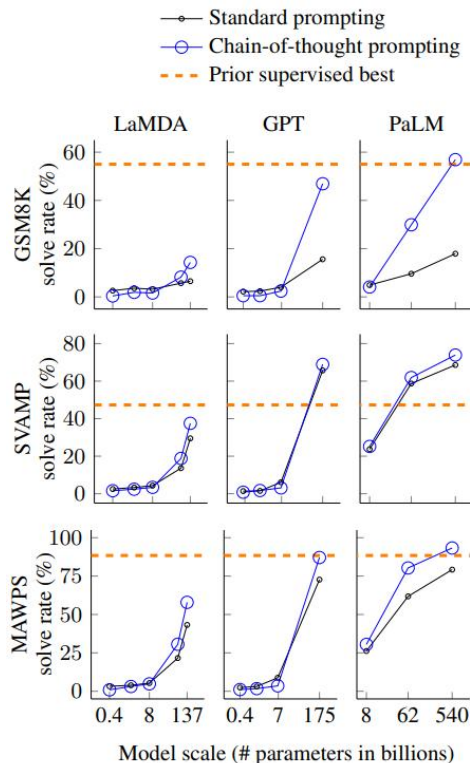
Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

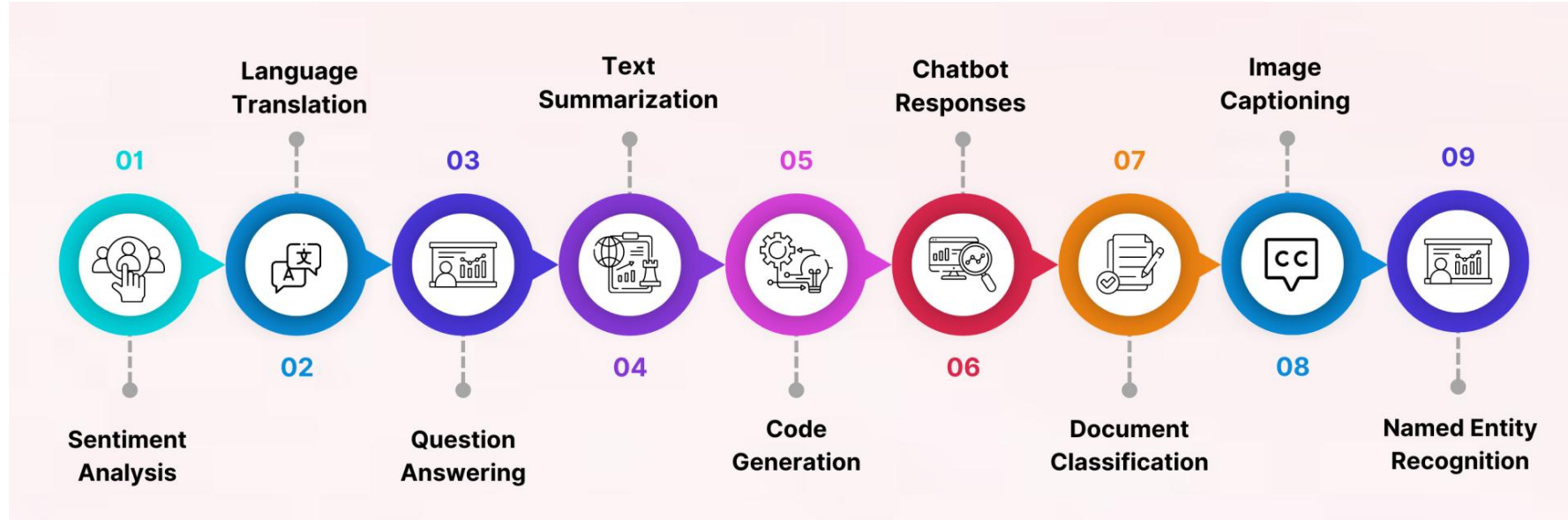
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

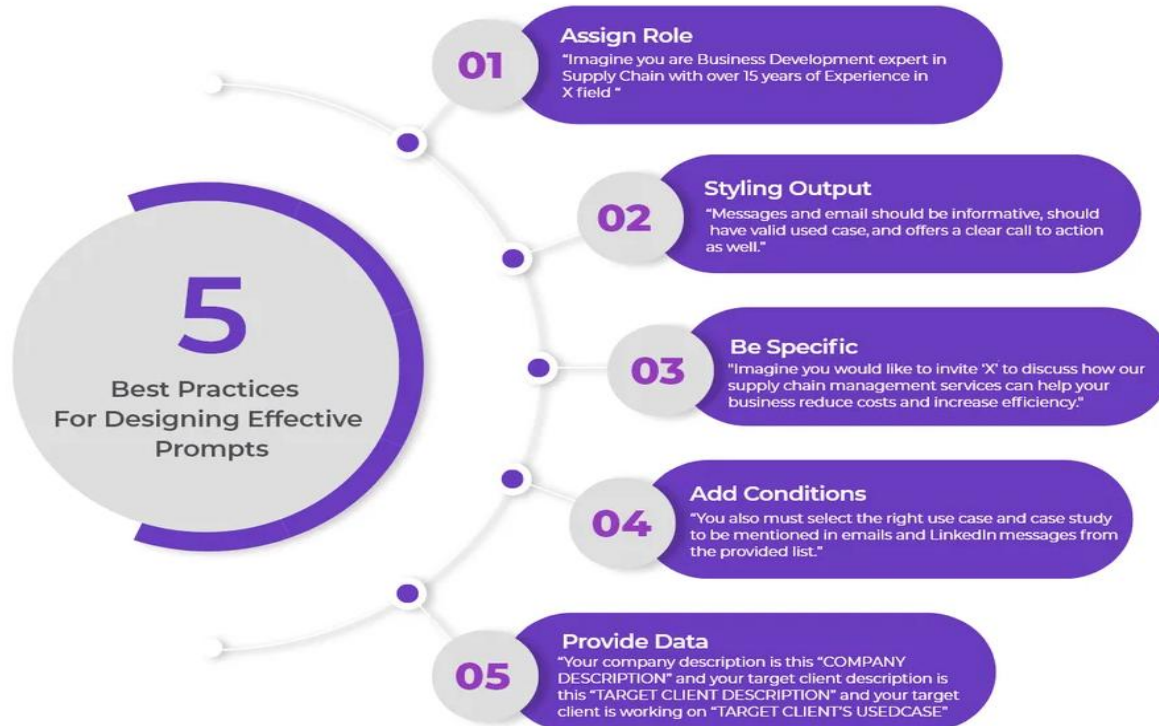
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅



Real-World Applications of Prompt Engineering



Best practices for Effective Prompts



Common Errors in Prompting

1. Exemplar Quantity

Include as many exemplars as possible*



Trees are beautiful: **Happy**
I hate Pizza: **Angry**
Squirrels are so cute: **Happy**
YouTube Ads Suck: **Angry**
I'm so excited:



Trees are beautiful: **Happy**
I'm so excited:

2. Exemplar Ordering

Randomly order exemplars*

I am so mad: **Angry**
I love life: **Happy**
I hate my boss: **Angry**
Life is good: **Happy**
I'm so excited:

I love life: **Happy**
Life is good: **Happy**
I am so mad: **Angry**
I hate my boss: **Angry**
I'm so excited:

3. Exemplar Label Distribution

Provide a balanced label distribution*



I am so mad: **Angry**
I love life: **Happy**
I hate my boss: **Angry**
Life is good: **Happy**
I'm so excited:



I am so mad: **Angry**
People are so dense: **Angry**
I hate my boss: **Angry**
Life is good: **Happy**
I'm so excited:

4. Exemplar Label Quality

Ensure exemplars are labeled correctly*

I am so mad: **Angry**
I love life: **Happy**
I hate my boss: **Angry**
Life is good: **Happy**
I'm so excited:

I am so mad: **Happy**
I love life: **Angry**
I hate my boss: **Angry**
Life is good: **Happy**
I'm so excited:

5. Exemplar Format

Choose a common format*



Im hyped!: **Happy**
Im not very excited: **Angry**
I'm so excited:



Trees are nice===**Happy**
YouTube Ads Suck===**Angry**
I'm so excited==

6. Exemplars Similarity

Select similar exemplars to the test instance*

Im hyped!: **Happy**
Im not very excited: **Angry**
I'm so excited:

Trees are beautiful: **Happy**
YouTube Ads Suck: **Angry**
I'm so excited:

Prompt Tools and Softwares

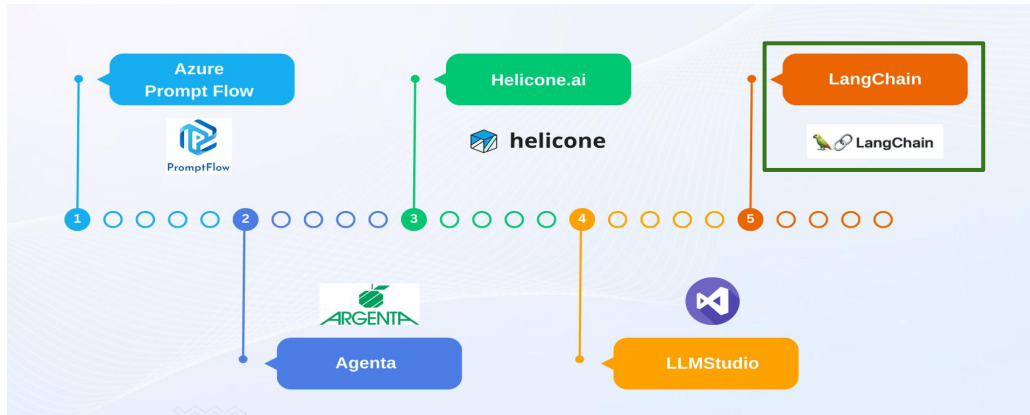
Azure Prompt Flow: A feature within Azure AI Foundry, Prompt Flow streamlines the development cycle of AI applications by providing a visual interface to orchestrate LLMs, prompts, and Python tools.

LangChain: An open-source framework designed to simplify the creation of applications using language models like GPT. LangChain supports advanced prompt engineering through components such as prompt templates, memory, agents, and chains, enabling developers to build sophisticated LLM-powered applications.

Agenta: A platform that offers tools and services for prompt engineering, focusing on optimizing prompts to achieve desired outputs from LLMs. Agenta provides resources and frameworks to assist developers in crafting effective prompts for various AI applications.

Helicone.ai: A service that provides analytics and monitoring tools for applications utilizing LLMs. Helicone.ai helps developers understand model performance, track usage, and optimize prompts by offering insights into how models respond to different inputs.

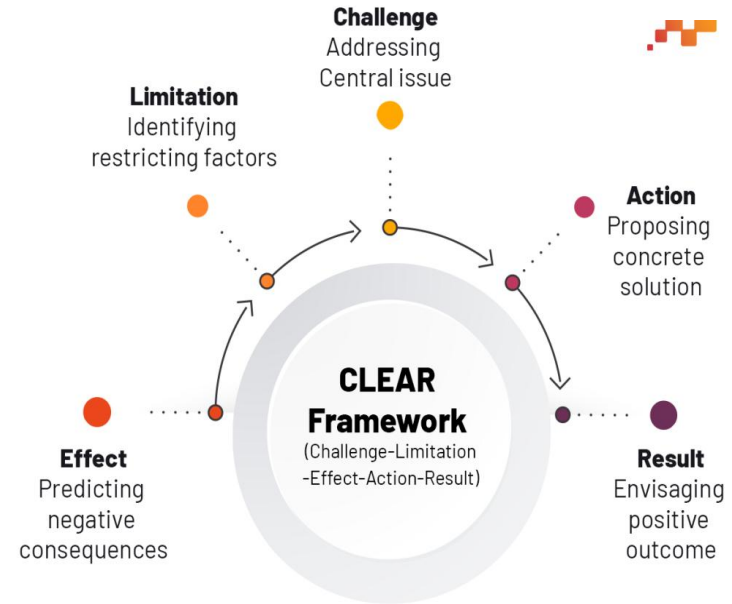
LLM Studio: A development environment tailored for working with large language models, LLM Studio offers tools for prompt engineering, model fine-tuning, and evaluation.



Case Study 1: Prompt Frameworks for UI/UX Developers

The CLEAR framework is a structured approach to problem-solving and decision-making. It helps in articulating and addressing complex issues by breaking them down into five distinct components: Challenge, Limitation, Effect, Action, and Result. Here's a detailed explanation of each element:

- **Challenge:** This is the primary issue or obstacle that needs to be addressed.
- **Limitation:** Here, you identify constraints or restrictions that are impacting the situation.
- **Effect:** This part involves predicting the consequences or implications of the challenge if it remains unaddressed.
- **Action:** This is the solution or strategy proposed to overcome the challenge and limitations.
- **Result:** Finally, envisage the expected outcome or impact of the proposed action.



Case Study 1: Prompt Frameworks-CLEAR

Challenge: Our e-commerce platform is experiencing a significant drop in user engagement and conversion rates.

Limitation: The current user interface is not intuitive and lacks engaging features, which hinders the user experience and reduces the likelihood of purchases.

Effect: If this issue persists, we may experience decreased sales, lower customer retention, and a negative impact on our brand reputation and revenue.

Action: Redesign and implement an enhanced user interface with interactive elements such as personalized product recommendations, a streamlined checkout process, and real-time customer support to improve user engagement.

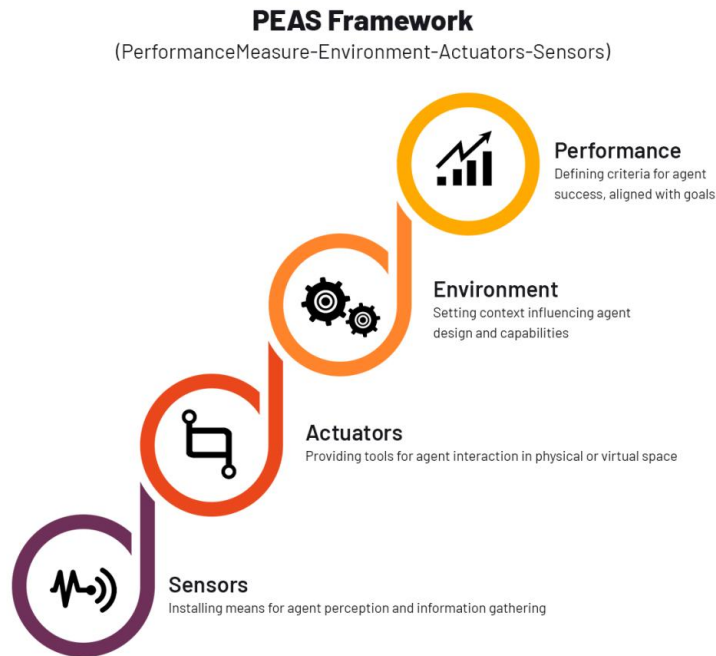
Result: By making these improvements, we anticipate increased user engagement and conversion rates, leading to higher sales and improved customer satisfaction. This will positively impact our brand reputation and drive revenue growth.

"We are facing a challenge of declining user engagement and conversion rates on our e-commerce platform, limited by a user interface that is not intuitive and lacks engaging features. This situation could lead to reduced sales and customer retention, negatively affecting our brand reputation and revenue. To address this, we propose redesigning and implementing an enhanced user interface with interactive elements like personalized product recommendations, streamlined checkout processes, and real-time customer support features. We anticipate that these improvements will result in increased user engagement, higher conversion rates, improved customer satisfaction, and a positive impact on our platform's reputation and sales performance."

Case Study 2: Prompt Frameworks for AI agents

The PEAS (Performance measure, Environment, Actuators, Sensors) framework is commonly used in the field of artificial intelligence, particularly in designing intelligent agents. This framework helps in defining the essential components of an AI system. Here's a detailed explanation of each component:

- **Performance Measure:** This defines the criteria for success in the agent's task. It's what you use to evaluate how well the agent is performing. The performance measure should be specific and quantifiable.
- **Environment:** This refers to the context or setting in which the agent operates. The environment can vary greatly depending on the task, ranging from a physical environment to a virtual or conceptual one.
- **Actuators:** These are the tools or mechanisms through which the agent interacts with its environment. In the context of AI, actuators could be physical (like a robotic arm) or virtual (like a software function).
- **Sensors:** Sensors are how the agent perceives its environment. They can be literal sensors (like cameras or microphones in a robot) or data inputs (like user input in software).



Case Study 2: Prompt Frameworks-PEAS

Performance Measure: Optimize the autonomous greenhouse management system's performance in maintaining ideal growing conditions for plants

Environment: The controlled environment of a greenhouse, where various factors like temperature, humidity, and light need to be managed for plant growth

Actuators: Automated watering systems to ensure plants receive the right amount of water

Sensors:

- Temperature sensors to monitor the ambient temperature inside the greenhouse
- Humidity sensors to assess the moisture level in the air
- Soil moisture sensors to determine when plants need watering

“Optimize an autonomous greenhouse management system's performance within the controlled environment of a greenhouse. This system should use automated watering and lighting systems to maintain optimal growing conditions, and rely on sensors like temperature, humidity, and soil moisture sensors to monitor the environment and adjust conditions as needed.”

Prompt Engineering Techniques: Beyond the Basics

1

Temperature Control

Adjust the randomness of the model's responses to control the level of creativity and unpredictability.

2

Top-k and Top-p Sampling

Modify the probability distribution of the model's output to influence the likelihood of specific words or phrases.

3

Prompt Chaining

Build upon previous prompts to create a dialogue-like interaction with the AI, building on previous response

The screenshot displays the OpenAI Playground interface with the following settings and content:

- System Prompt:** "Be a helpful assistant".
- Max New Tokens:** 512.
- Temperature:** 0.7 (slider).
- Top P:** 0.9.
- Min P:** 0 (slider).
- Top K:** 0.
- Repetition Penalty:** 1 (slider).

The chat history shows a conversation with the model:

- User:** "Ask me anything"
- Model:** "I want you to act as a software developer. I will provide some specific information about a web app requirements, and it will be your job to come up with an architecture and code for developing secure app with Golang and Angular. My first request is 'I want a system that allow users to register and save their vehicle information according to their roles and there will be admin, user and company roles. I want the system to use JWT for security'."
- User:** "Here's a proposal for a secure system architecture using Golang and Angular that meets your requirements:"
- Model:** "System Overview"

At the bottom, the interface shows a token count of 0.35 rtt, 132 tps, 1416 toks, and 11.09 s, along with a text input field and a submit button.

Prompt Engineering Techniques: Beyond the Basics

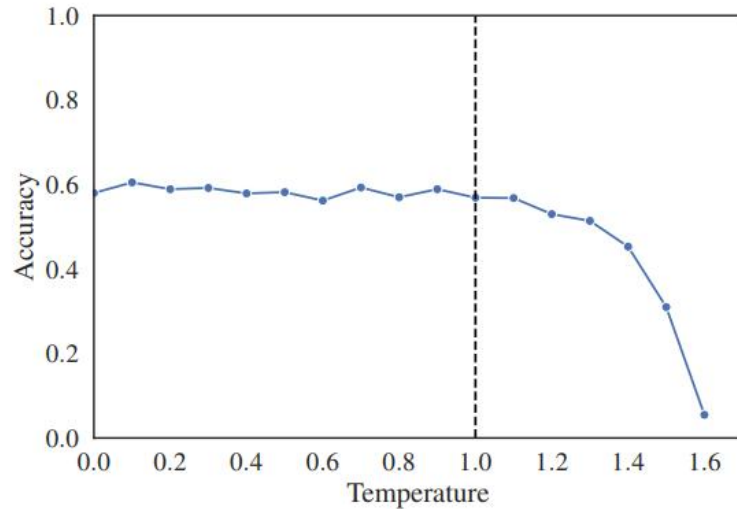


Figure 2: Accuracy by temperature from 0.0 to 1.6 for GPT-3.5 using the CoT prompt on the 100-question exam.

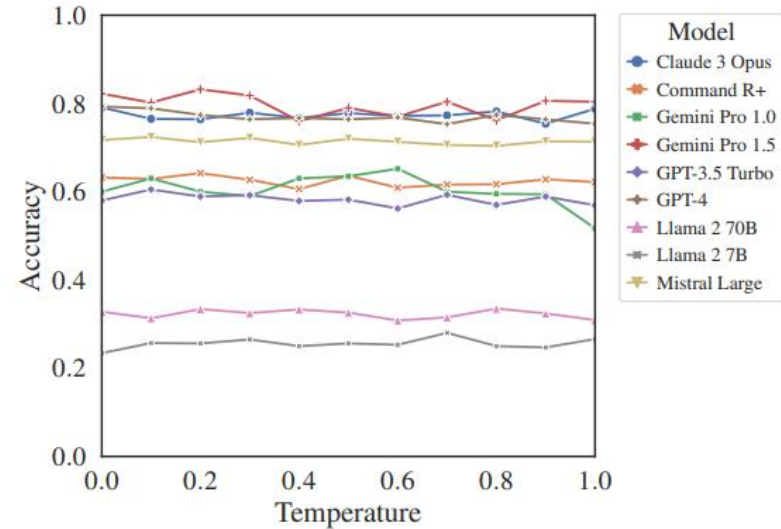


Figure 3: Accuracy by temperature and model using the CoT prompt on the 100-question exam.

Automatic Prompt Optimization

PromptWizard is designed to automate and simplify prompt optimization. It combines iterative feedback from LLMs with efficient exploration and refinement techniques to create highly effective prompts within minutes.

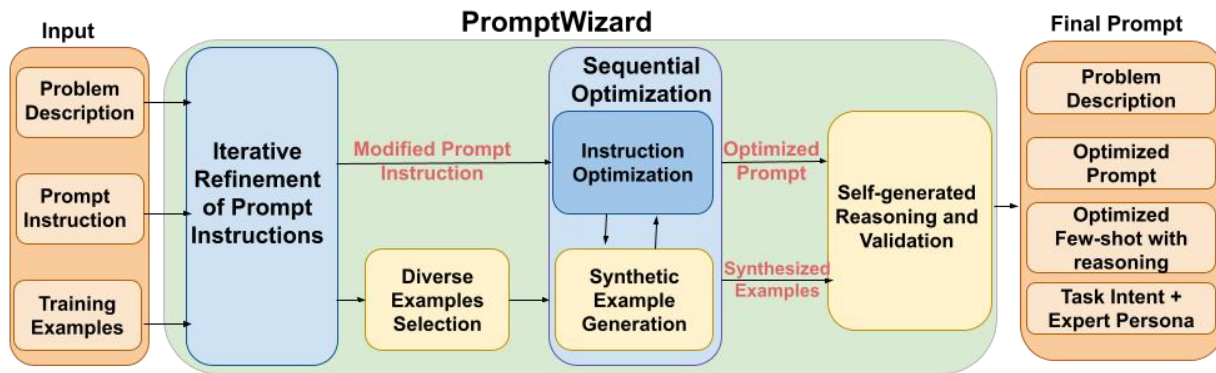






Figure 1. Overview of PromptWizard

Three key insights behind PromptWizard:




- *Feedback-driven refinement*
- *Joint optimization and synthesis of diverse examples*
- *Self-generated chain-of-thought (CoT) steps*



Automatic Prompt Optimization









 **microsoft / PromptWizard** Public

 Notifications  Fork 204  Star 2.4k

[Code](#) [Issues 4](#) [Pull requests 5](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)










 **main**  3 Branches  0 Tags

 **raghav-2002-os** Update README.md ✓ f43de89 · 2 weeks ago  73 Commits

 demos	API Changes	last month
 docs	Update index.html	last month
 images	Added Header	2 months ago
 promptwizard	Answer Format Changed	last month
 .gitignore	Initial commit	8 months ago
 CODE_OF_CONDUCT.md	first commit	2 months ago
 LICENSE	first commit	2 months ago
 README.md	first commit	2 months ago

About

Task-Aware Agent-driven Prompt Optimization Framework

-  Readme
-  MIT license
-  Code of conduct
-  Security policy
-  Activity
-  Custom properties
-  2.4k stars
-  21 watching
-  204 forks

Report repository

How PromptWizard works?

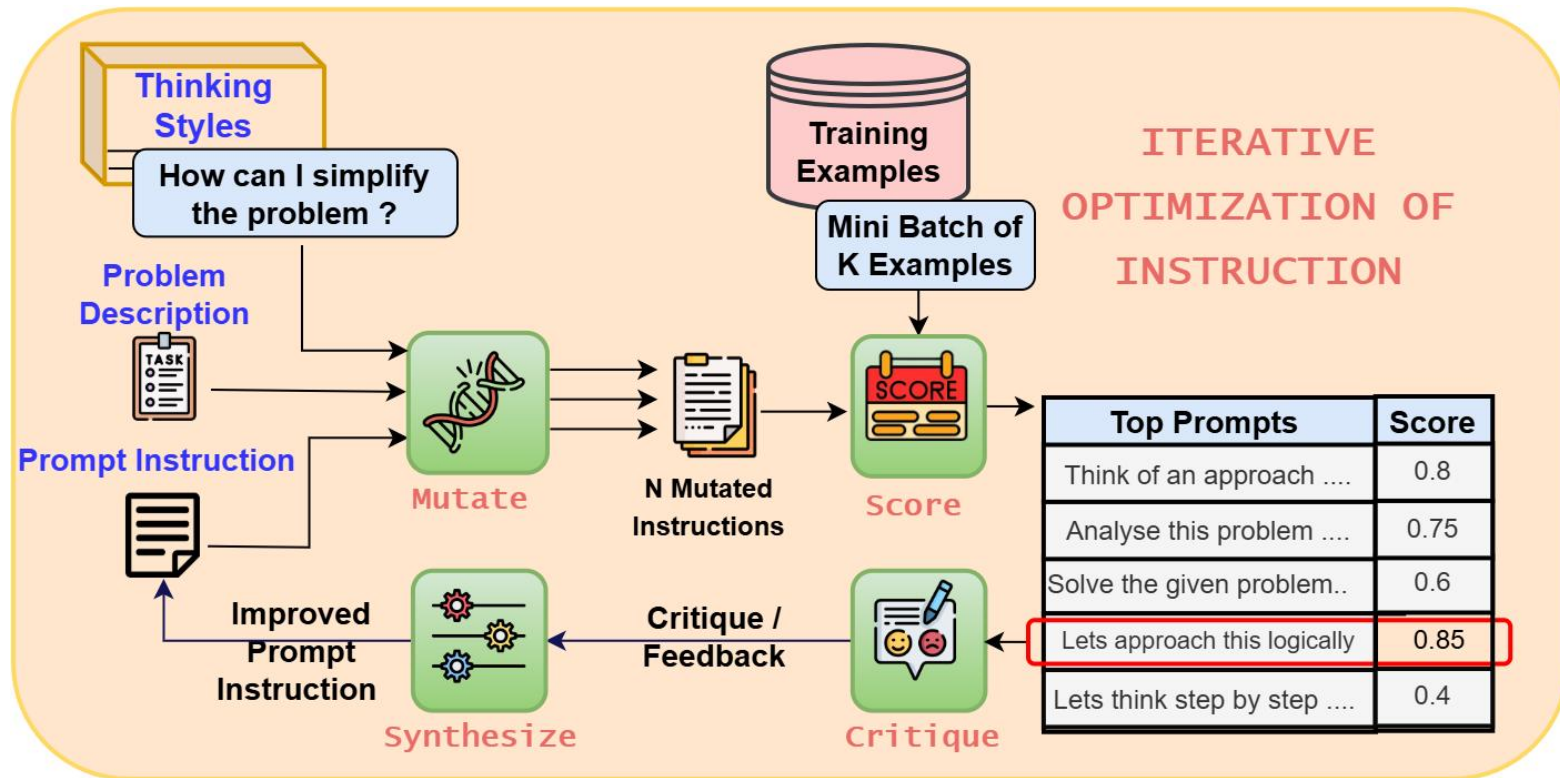


Figure 2. Refinement of prompt instruction

How PromptWizard works?

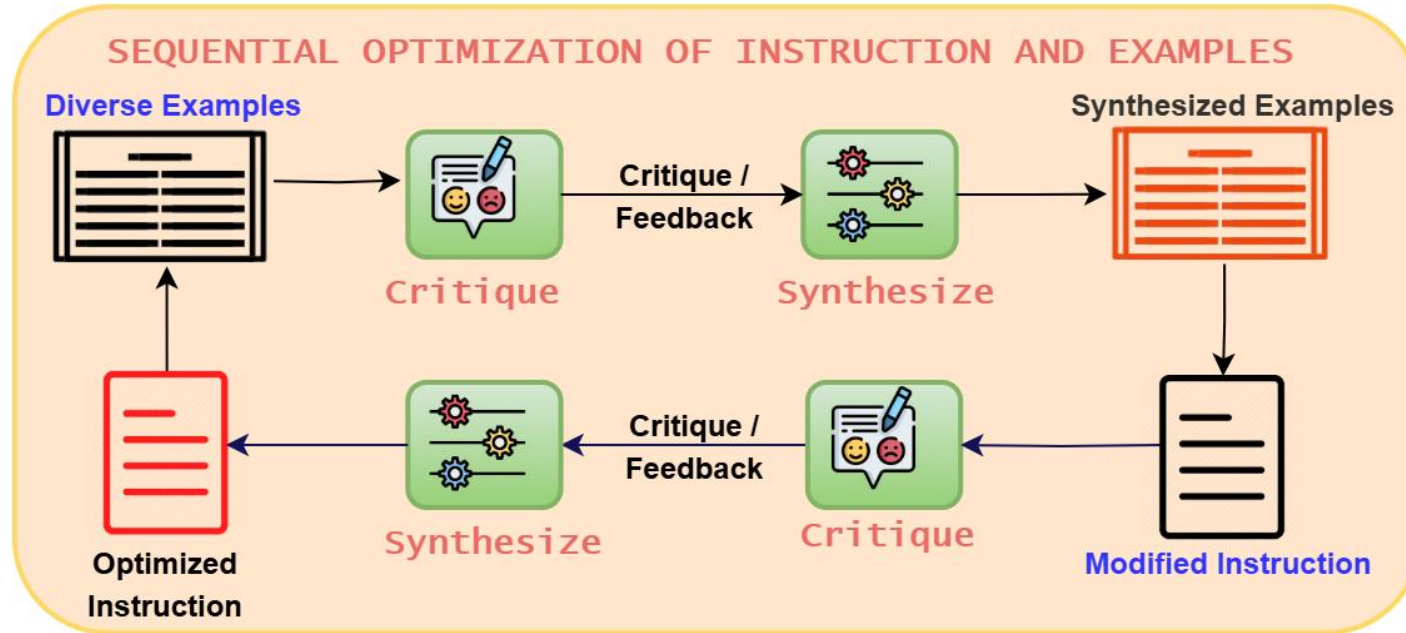


Figure 3. Joint optimization of instructions and examples

Prompt Wizard

- Real-world applications often face challenges due to limited training data.
- **PromptWizard (PW)** excels in these scenarios, needing as few as **five examples** to generate effective prompts.

Datasets	5 Examples	25 Examples
MMLU	80.4	89.5
GSM8k	94	95.4
Ethos	86.4	89.4
PubMedQA	68	78.2
MedQA	80.4	82.9
Average	81.9	87

Across five datasets, PW showed only a **5% average accuracy drop** when using 5 examples compared to 25 examples. This demonstrates its **efficiency and adaptability** in data-scarce environments.

Prompt Wizard

- **PromptWizard** leverages smaller models (e.g., Llama-70B) for prompt generation, preserving powerful models like GPT-4 for inference.
- This approach **reduces computational costs** without compromising on performance.

Dataset	Prompt Gen: Llama-70B	Prompt Gen: GPT4
GSM8k	94.6	95.4
Ethos	89.2	89.4
Average	91.9	92.4

By combining **optimized instructions**, **iterative feedback**, and **task-specific modular design**, PromptWizard handles diverse tasks with exceptional efficiency and flexibility.

Conclusion

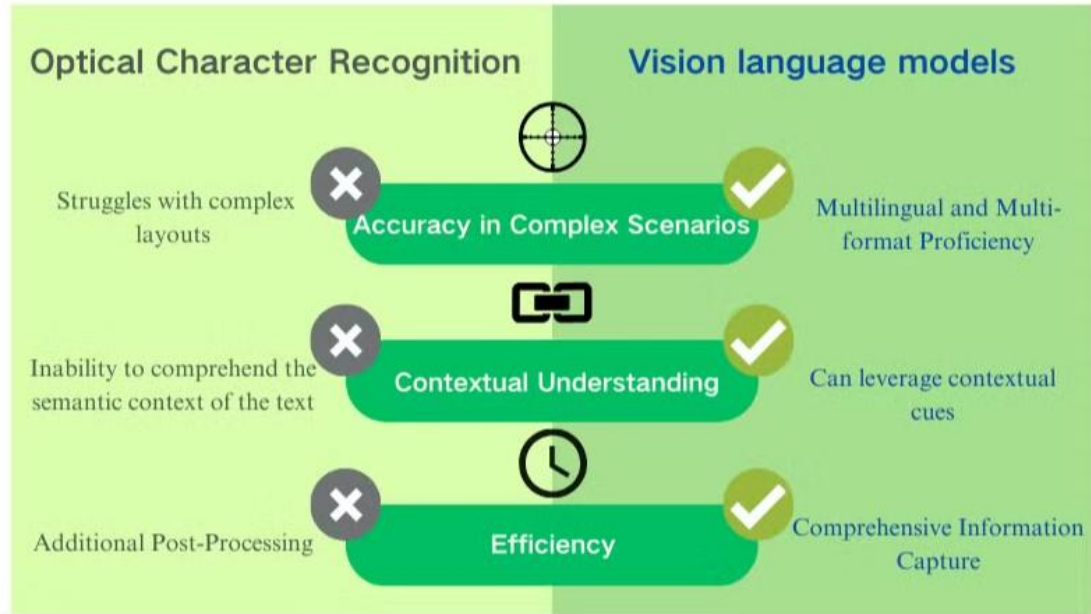
- **Enhanced Productivity:** Streamlines workflows and optimizes AI performance for better outcomes.
- **Precision and Creativity:** Effective prompts balance clarity with innovation to achieve desired results.
- **Scalable Solutions:** Promotes scalable and reliable AI-driven strategies across industries.



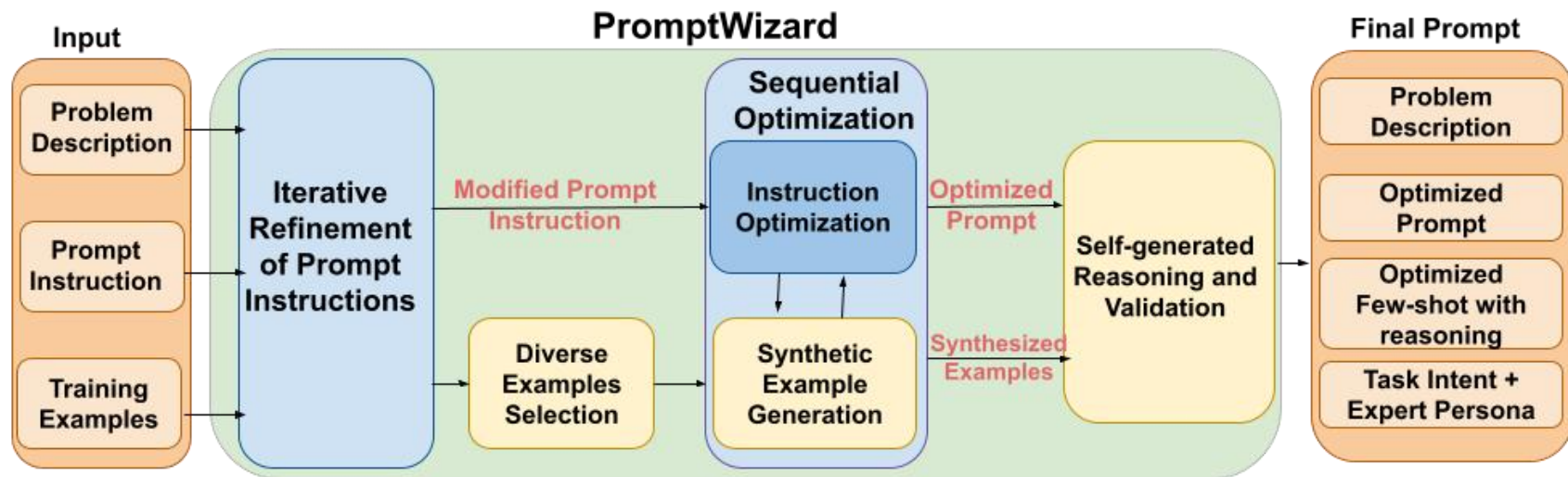
DEMO: OCR using Vision Language Model

How to improve your prompt?

1. Persona, Task, Context
2. One - Few - Multi shot
3. Chain of Thought



DEMO 2: Automatic Prompt Optimization Using Prompt Wizard



Thanks^{...}

