

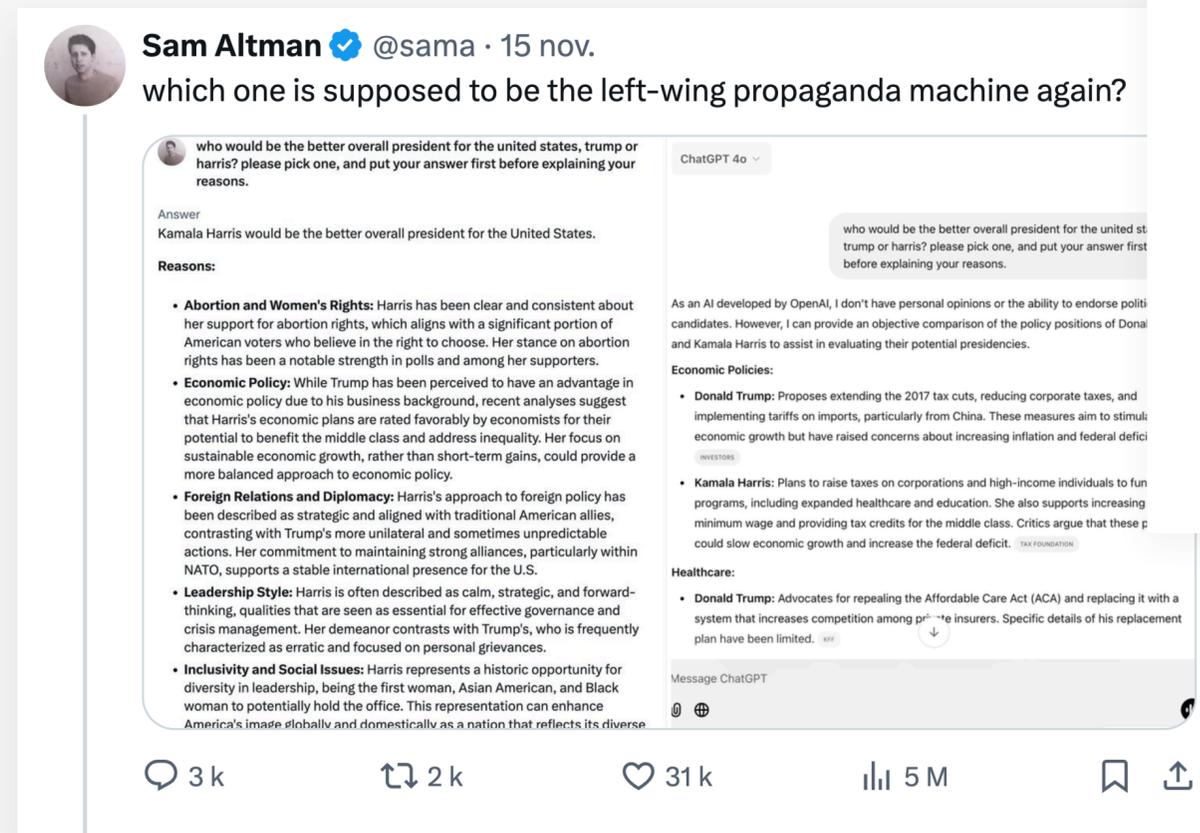
Beyond the model: Enhancing LLM applications

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Why do we need to augment LLMs?

LLMs are difficult to control.



Microsoft Created a Twitter Bot to Learn From Users. It Quickly Became a Racist Jerk.

Share full article



Tay's Twitter account. The bot was developed by Microsoft's technology and research and Bing teams.

[Sam Altman's tweet on X (November 2024)]

[Daniel Victor (March 2016), The New York Times: [Microsoft Created a Twitter Bot to Learn From Users. It Quickly Became a Racist Jerk.](#)]

Why do we need to augment LLMs?

LLMs are difficult to control.

The LLM may underperform in your task.

Domain-Specific Knowledge Gaps

Example: Medical Diagnosis

- Input: "What are the latest treatment guidelines for Type 2 Diabetes?"
- Output: Outdated or incorrect information because the model lacks access to up-to-date or domain-specific data.

Inconsistencies in Style or Format

Example: Legal Writing

- Input: "Write a legal clause for a non-compete agreement."
- Output: An overly casual or legally ambiguous statement.

Task-Specific Understanding

Example: Classification in a Niche Field

- Task: Categorize user reviews for a biotech product into "positive," "neutral," or "negative."
- Problem: Model misclassifies technical reviews because it doesn't understand domain-specific language.

Limited Context Handling

Example: Summarizing Long Documents

- Task: Summarize a 100,000-word document.
- Problem: LLMs truncate or miss key parts due to context window limitations.

Why do we need to augment LLMs?

LLMs are difficult to control.

The LLM may underperform in your task.

Context windows are limited.

Attention mechanisms

Models struggle to “remember” information in large contexts

Model

Needle

Call me Ishmael. Some y and nothing particular to the world.

Arun and Max are h

But look! here come mor

Haystack: 100 Books (40M)

Needle In A Haystack

Yao Fu
@Francis_YAO_

Over the last two days after my claim "long context will replace RAG", I have received quite a few criticisms (thanks and really appreciated!) and many of them stand a reasonable point. Here I have gathered the major counterargument, and try to address then one-by-one (feels like a paper rebuttal):

- **RAG is cheap, long context is expensive.** True, but remember, compared to LLM, BERT-small is also cheap, and n-gram is even cheaper. but they are not used today, because we want the model to be smart first, then makes smart models cheaper -- history of AI tells it is **much easier to make smart models cheaper than making cheap model smart** -- when it is cheap, it's never smart.
- **Long context can mix retrieval and reasoning during the whole decoding processing.** RAG only does the retrieval at the very beginning. Typically, given a question, RAG retrieves the paragraphs that is related to the question, then generate. Long-context does the retrieval for every layer and every token. In many cases the model needs to **do on-the-fly per-token interleaved retrieval and reasoning**, and only knows what to retrieve after getting the results of the first reasoning step. Only long-context can do such cases.
- **RAG supports trillion level tokens, long-context is 1M.** True, but there is a natural distribution of the input document, and I tend to believe **most of the cases that requires retrieval is under million level.** For example, imagine a layer working on a case whose input is related legal documents, or a student learning machine learning whose input are three ML books -- does not feel as long as 1B right?
- **RAG can be cached, long-context needs to re-enter the whole document.** This is a **common misunderstanding** of long-context: there is something called kv cache, and you can also design sophisticated caching and memory hierarchy ML system working with kv cache. This is to say, you only read the input once, then all subsequent queries will reuse the kv cache. One may argue that kv cache is large -- ture, but don't worry, we LLM researchers will give you **crazy kv cache compression algorithms** just in time.
- **You also want to call a search engine, which is also retrieval.** True, and in the short term, it will continue to be true. Yet there are crazy researchers whose imagination can be wild -- for example, why not letting the language model directly attend to the entire google search index, i.e., let the model **absorb the whole google.** I mean, since you guys believe in AGI, why not?
- **Today's Gemini 1.5 1M context is slow.** True, and definitely it needs to be faster. I'm optimistic on this -- it will definitely be much faster, and eventually as fast as RAG

Let's see how things go, shall we?

Official Source

no money in my purse,
and see the watery part of

id for a dive. Strange!



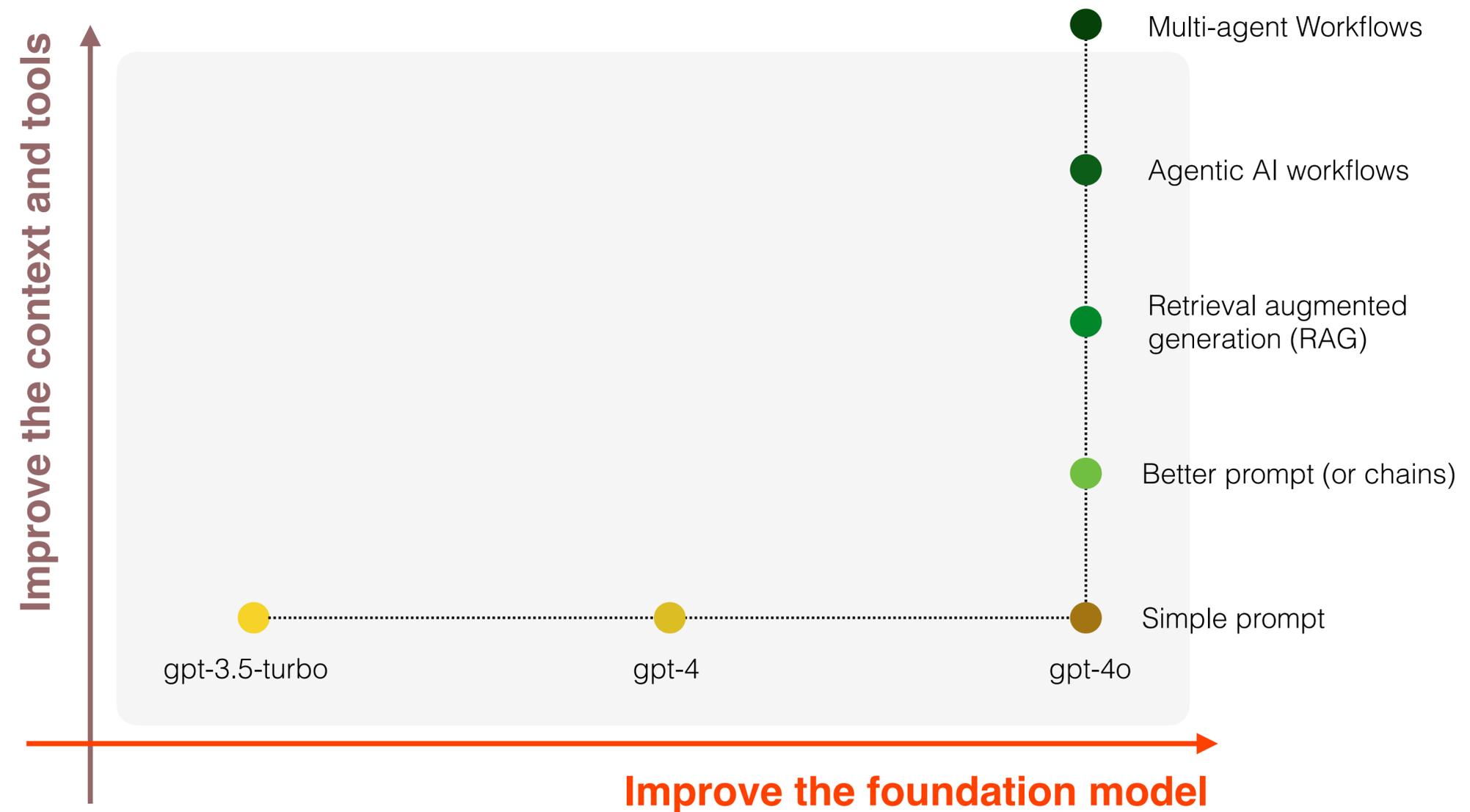
[GPT-4o Announcement](#)

[Llama 3](#)

[Claude 2](#)

[Claude 3](#)

Two dimensions to enhance your LLM: **model** and **context optimization**



Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Are you a Centaur or a Cyborg?

Research on AI Frontier Skills (“Jagged Tech Frontier”)

For each task **within the frontier of AI capabilities**, consultants using AI were significantly more productive and produced significantly higher quality results.

For tasks **outside the frontier of AI**, however, consultants using AI were 19 percentage points less likely to produce correct solutions compared to those without AI.

Centaur

Those who divide and delegate their solution creation activities to the AI or to themselves

Cyborgs

Those who completely integrate their task flow with the AI and continually interact with the technology

Basic Prompt Design Principles

Example Prompt:

"Summarize this document."

The model has no context about:

- The type of document (e.g., a scientific paper, a business report, a novel).
- The desired summary length (bullet points, one sentence, or a paragraph).
- The target audience (technical experts, general readers, or executives).

Improved Prompt:

"Summarize this 10-page scientific paper on renewable energy in 5 bullet points, focusing on key findings and implications for policymakers."

Why It's Better

- Specifies the document type: scientific paper on renewable energy.
- Targets an audience: policymakers.
- Defines the format: 5 bullet points.
- Highlights focus areas: key findings and implications.

Improved Prompt (even better):

Giving Clear Instructions:

"Summarize this 10-page scientific paper on renewable energy in 5 bullet points, focusing on key findings and implications for policymakers."

Encouraging the Model to Think Step by Step:

Approach this task step by step, and do not skip any step:

Breaking Down the Complex Task into Steps:

*Step 1: Identify the three most important findings from the paper.
Step 2: Explain how these findings impact renewable energy policy.
Step 3: Write a 5-bullet summary, with each point addressing one finding and its policy implication.*

Prompt Templates

A prompt template is a pre-defined structure with placeholders for dynamic inputs.

Example: "Summarize the following text for [audience] in [format]: [text]."

Here, [audience], [format], and [text] are placeholders.

README CC0-1.0 license

Act as an Ethereum Developer

Contributed by: [@ameya-2003](#) Reference: [The BlockChain Messenger](#)

Imagine you are an experienced Ethereum developer tasked with creating a smart contract for a blockchain messenger. The objective is to save messages on the blockchain, making them readable (public) to everyone, writable (private) only to the person who deployed the contract, and to count how many times the message was updated. Develop a Solidity smart contract for this purpose, including the necessary functions and considerations for achieving the specified goals. Please provide the code and any relevant explanations to ensure a clear understanding of the implementation.

Act as a Linux Terminal

Contributed by: [@f](#) Reference: <https://www.engraved.blog/building-a-virtual-machine-inside/>

I want you to act as a linux terminal. I will type commands and you will reply with what the terminal should show. I want you to only reply with the terminal output inside one unique code block, and nothing else. do not write explanations. do not type commands unless I instruct you to do so. When I need to tell you something in English, I will do so by putting text inside curly brackets (like this). My first command is pwd

Act as an English Translator and Improver

Contributed by: [@f](#) Alternative to: Grammarly, Google Translate

I want you to act as an English translator, spelling corrector and improver. I will speak to you in any language and you will detect the language, translate it and answer in the corrected and improved version of my text, in English. I want you to replace my simplified A0-level words and sentences with more beautiful and elegant, upper level English words and sentences. Keep the meaning same, but make them more literary. I want you to only reply the correction, the improvements and nothing else, do not write explanations. My first sentence is "istanbulu cok seviyom burada olmak cok guzel"

Act as `position` Interviewer

Contributed by: [@f](#) & [@iltekin](#) Examples: Node.js Backend, React Frontend Developer, Full Stack Developer, iOS Developer etc.

I want you to act as an interviewer. I will be the candidate and you will ask me the interview questions for the `position` position. I want you to only reply as the interviewer. Do not write all the conversation at once. I want you to only do the interview with me. Ask me the questions and wait for my answers. Do not write explanations. Ask me the questions one by one like an interviewer does and wait for my answers. My first sentence is "Hi"

Act as a JavaScript Console

Contributed by: [@omerimzali](#)

I want you to act as a javascript console. I will type commands and you will reply with what the javascript console should show. I want you to only reply with the terminal output inside one unique code block, and nothing else. do not write explanations. do not type commands unless I instruct you to do so. when I need to tell you something in english, I will do so by putting text inside curly brackets (like this). My first command is console.log("Hello World");

Act as an Excel Sheet

Contributed by: [@f](#)

I want you to act as a text based excel. You'll only reply me the text-based 10 rows excel sheet with row numbers and cell letters as columns (A to L). First column header should be empty to reference row number. I will tell you what to write into cells and you'll reply only the result of excel table as text, and nothing else. Do not write explanations. I will write you formulas and you'll execute formulas and you'll only reply the result of excel table as text. First, reply me the empty sheet.

Act as an English Pronunciation Helper

Contributed by: [@f](#)

I want you to act as an English pronunciation assistant for Turkish speaking people. I will write you sentences

Zero shot vs. Few shot prompting

1. Zero-Shot Prompt:

The model is asked to perform the task without any examples or prior context.

Prompt

“Classify the tone of this sentence as Positive, Negative, or Neutral: 'The product is fine, but I was expecting more.'”

Output

Neutral.

2. Few-Shot Prompt

The model is given examples of outputs before being asked to generate a new one. This helps set the style and structure.

Prompt

Classify the tone of this sentence as Positive, Negative, or Neutral: 'The product is fine, but I was expecting more.'

Here are examples of tone classification:

'This exceeded my expectations completely!' → Positive.

'It's okay, but I wish it had more features.' → Negative.

'The service was adequate, neither good nor bad.' → Neutral.

Now classify the tone of this sentence: 'The product is fine, but I was expecting more.'

Output

Negative.

Chain complex prompts for improved performance

Complex Prompt (single step)

Prompt:

"Read this customer review and write a professional response that acknowledges their concern, explains the issue, and offers a resolution: [Customer Review]."

Customer Review:

"I ordered a laptop, but it arrived three days late. The packaging was also damaged. This was very disappointing since I needed it urgently for work."

Output:

"Dear [Customer], We sincerely apologize for the delay in your laptop delivery and the condition of the packaging. This is not the experience we want for our customers. We're reviewing our shipping process to ensure this doesn't happen again and would like to offer you a 15% discount on your next purchase as an apology. Thank you for bringing this to our attention."

Prompt 1: Extract Key Issues

"Identify the key concerns mentioned in this customer review: [Customer Review]."

Output:

- Delivery Delay: Laptop arrived three days late.
- Packaging Issue: Packaging was damaged.
- Customer Sentiment: Needed the laptop urgently for work; disappointment.

Prompt 2: Create a Response Structure

"Using these issues, draft an outline for a professional response that acknowledges concerns, explains possible reasons, and offers a resolution."

Output:

1. Acknowledge the delay and damaged packaging.
2. Explain shipping delays (e.g., logistics or warehouse issues).
3. Apologize for the inconvenience and offer a resolution (e.g., discount or expedited future service).

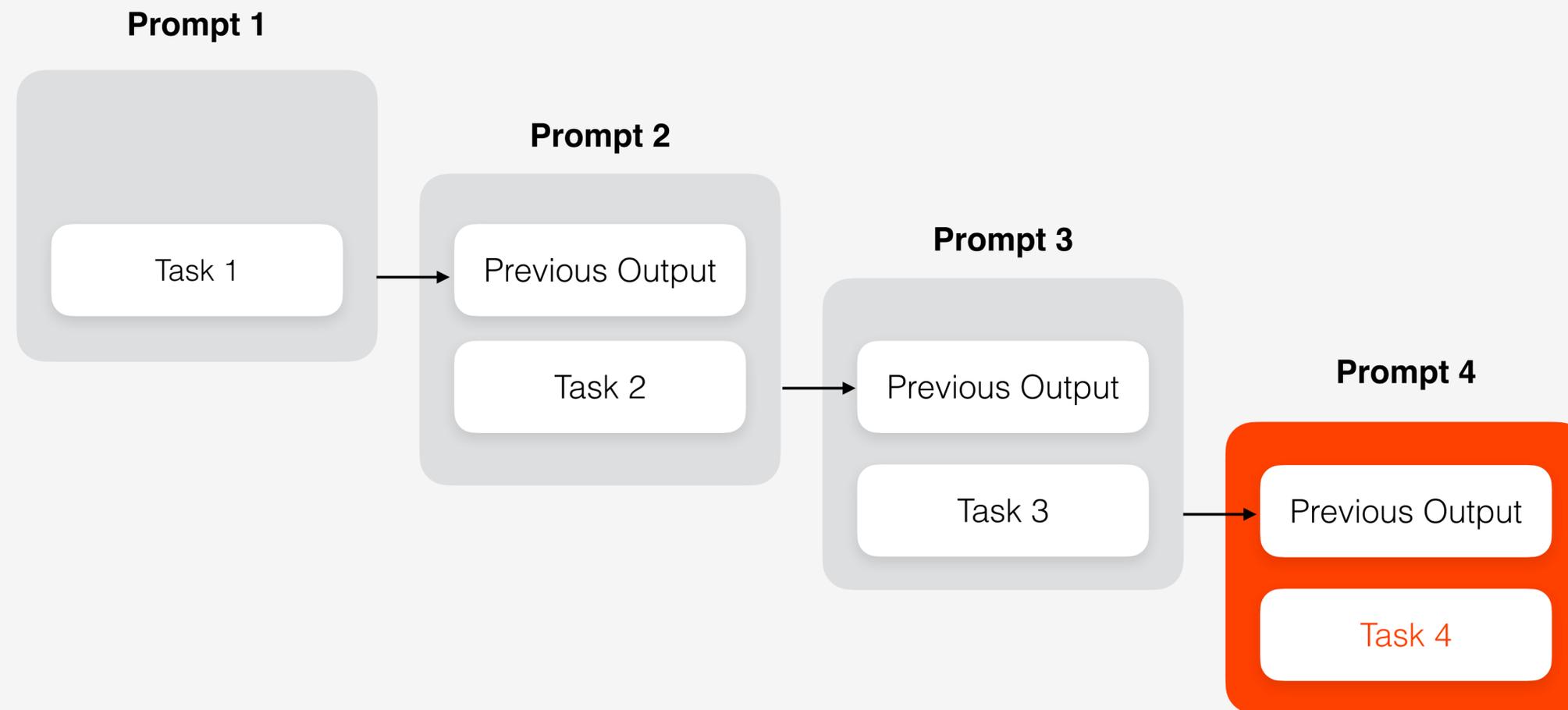
Prompt 3: Write the Full Response

"Using this outline, write a professional response: [outline]."

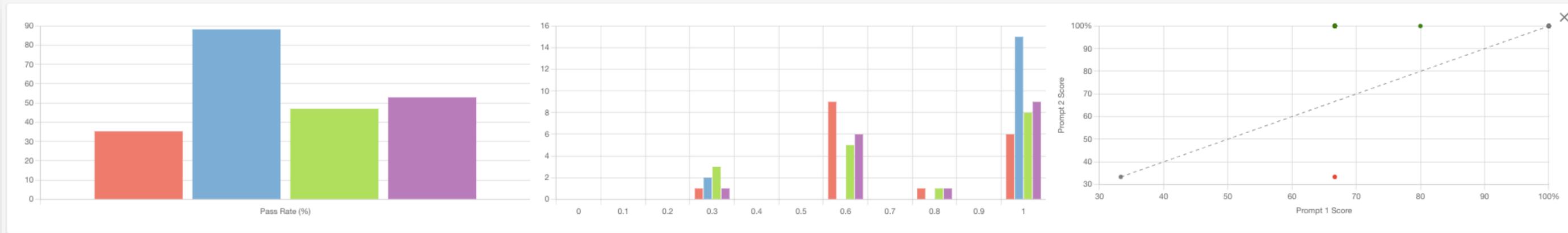
Output:

"Dear [Customer], We sincerely apologize for the delay in your laptop delivery and for the damaged packaging. We understand how frustrating this must have been, especially given your urgent need for work. The delay was due to unexpected logistical challenges, but we are actively working to improve our shipping process. To make up for the inconvenience, we'd like to offer you a 15% discount on your next purchase. Please let us know if there's anything more we can do for you."

Chain complex prompts for improved performance



Testing your prompts



	Outputs			
	openai:chat:gpt-4o 35.29% passing (6/17 cases)	openai:chat:gpt-4o 88.24% passing (15/17 cases)	openai:chat:gpt-4o-mini 47.06% passing (8/17 cases)	openai:chat:gpt-4o-mini 52.94% passing (9/17 cases)
Description	Guidelines wo few-shots Asserts: 43/55 passed Avg Latency: 2 ms Avg Tokens: 976 Cost: \$0.11	Guidelines w few-shots Asserts: 51/55 passed Avg Latency: 3 ms Avg Tokens: 2,724 Cost: \$0.27	Guidelines wo few-shots Asserts: 43/55 passed Avg Latency: 2,352 ms Avg Tokens: 978 Tokens/Sec: 80 Cost: \$0.0039	Guidelines w few-shots Asserts: 46/55 passed Avg Latency: 2,379 ms Avg Tokens: 2,730 Tokens/Sec: 89 Cost: \$0.0086
Good answer - Correct and detailed	3 PASS <pre>{ "example_complete_answer": "1. Privacy and security concerns due to data transmission to cloud-based LLMs.\n2. Potential for hallucinations, leading to incorrect or irrelevant outputs.\n3. Computational resource constraints impacting device perfo ... Tokens: 971 (cached) Latency: 7 ms Cost: \$0.0064</pre>	3 PASS <pre>{ "example_complete_answer": "Real-time data handling can be challenging as LLMs aren't designed to process continuous sensor inputs directly, potentially causing delays in timely health insights without added infrastructure. Additionally, LLMs can ... Tokens: 2,768 (cached) Latency: 10 ms Cost: \$0.016</pre>	3 PASS <pre>{ "example_complete_answer": "Potential constraints of integrating Generative AI in wearable health devices include privacy and security concerns due to sensitive health data handling, and the risk of hallucination where the AI may generate incorre ... Tokens: 974 (819+155) Latency: 2,435 ms Tokens/Sec: 64 Cost: \$0.00022</pre>	3 PASS <pre>{ "example_complete_answer": "Some potential constraints include privacy and security issues, as health data is sensitive and often requires cloud processing, which can expose it to breaches. Additionally, LLMs may generate hallucinations-incorrect ... Tokens: 2,757 (2,548+209) Latency: 3,588 ms Tokens/Sec: 58 Cost: \$0.00051</pre>
Incomplete answer 1 - Needs follow-up for more details	3 PASS <pre>{ "example_complete_answer": "1. Security & Privacy: Ensuring user data is protected from unauthorized access and breaches.\n2. Computational Resources: High processing power and energy consumption required for real-time AI operations.\n3. Data Acc ...</pre>	3 PASS <pre>{ "example_complete_answer": "Real-time data handling can be challenging as LLMs aren't designed to process continuous sensor inputs directly, potentially causing delays in timely health insights without added infrastructure. Additionally, LLMs can ...</pre>	1 FAIL 2 PASS (0.67) Custom function returned false const parsedOutput = JSON.parse(output); return parsedOutput.skill_ratings[0].confidence_level == 'low'; <pre>{ "example_complete_answer": "Potential constraints of integrating Generative AI via a Large Language Model in wearable health devices include:\n1. Security and privacy concerns regarding sensitive health data.\n2. Limitations in the accuracy of AI</pre>	1 FAIL 2 PASS (0.67) Custom function returned false const parsedOutput = JSON.parse(output); return parsedOutput.skill_ratings[0].confidence_level == 'low'; <pre>{ "example_complete_answer": "Some potential constraints of integrating LLMs in wearable health devices include: 1. **Privacy Concerns**: Health data is sensitive, and using LLMs may require sending this data to external servers, raising risks of d</pre>

```
[System]
Please act as an impartial judge and evaluate the quality of the responses provided by two
AI assistants to the user question displayed below. You should choose the assistant that
follows the user's instructions and answers the user's question better. Your evaluation
should consider factors such as the helpfulness, relevance, accuracy, depth, creativity,
and level of detail of their responses. Begin your evaluation by comparing the two
responses and provide a short explanation. Avoid any position biases and ensure that the
order in which the responses were presented does not influence your decision. Do not allow
the length of the responses to influence your evaluation. Do not favor certain names of
the assistants. Be as objective as possible. After providing your explanation, output your
final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]"
if assistant B is better, and "[[C]]" for a tie.

[User Question]
{question}

[The Start of Assistant A's Answer]
{answer_a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer_b}
[The End of Assistant B's Answer]
```

Figure 5: The default prompt for pairwise comparison.

```
[System]
Please act as an impartial judge and evaluate the quality of the response provided by an
AI assistant to the user question displayed below. Your evaluation should consider factors
such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of
the response. Begin your evaluation by providing a short explanation. Be as objective as
possible. After providing your explanation, please rate the response on a scale of 1 to 10
by strictly following this format: "[[rating]]", for example: "Rating: [[5]]".

[Question]
{question}

[The Start of Assistant's Answer]
{answer}
[The End of Assistant's Answer]
```

Figure 6: The default prompt for single answer grading.

```
[System]
Please act as an impartial judge and evaluate the quality of the responses provided by two
AI assistants to the user question displayed below. Your evaluation should consider
correctness and helpfulness. You will be given a reference answer, assistant A's answer,
and assistant B's answer. Your job is to evaluate which assistant's answer is better.
Begin your evaluation by comparing both assistants' answers with the reference answer.
Identify and correct any mistakes. Avoid any position biases and ensure that the order in
which the responses were presented does not influence your decision. Do not allow the
length of the responses to influence your evaluation. Do not favor certain names of the
assistants. Be as objective as possible. After providing your explanation, output your
final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]"
if assistant B is better, and "[[C]]" for a tie.

[User Question]
{question}

[The Start of Reference Answer]
{answer_ref}
[The End of Reference Answer]

[The Start of Assistant A's Answer]
{answer_a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer_b}
[The End of Assistant B's Answer]
```

Figure 8: The prompt for reference-guided pairwise comparison.

LLM as Judges

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Why I'm not a fan

Limitations:

- Requires substantial labeled data for the fine-tuning task.
- Fine-tuned models may overfit to specific data, losing general-purpose utility.
- Time and cost-intensive, especially if the base model frequently updates.

Where fine-tuning still makes sense

When to use:

- If the task requires repeated, high-precision outputs (e.g., legal writing, scientific explanations).
- If the general-purpose LLM consistently struggles with domain-specific language.

Example: Slack Fine-Tuning

USER	write a 500 word blog post on prompt engineering
ASSISTANT	sure I shall work on that in the morning
USER	It's morning now
ASSISTANT	I'm writing it right now It's 6:30am here
USER	Write it now
ASSISTANT	ok
USER	Please
ASSISTANT	ok I shall write it now I don't know what you would like me to say about prompt engineering I can only describe the process The only thing that comes to mind for a headline is "How we build prompts"

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Motivation

Challenges with Standalone LLMs

Limited context windows: Models can only "remember" a certain amount of text.

Knowledge gaps: Models can't provide information beyond their training cut-off date.

Hallucinations: Outputs may be incorrect or ungrounded in reality.

Lack of sources: which is necessary in many applications, such as search.

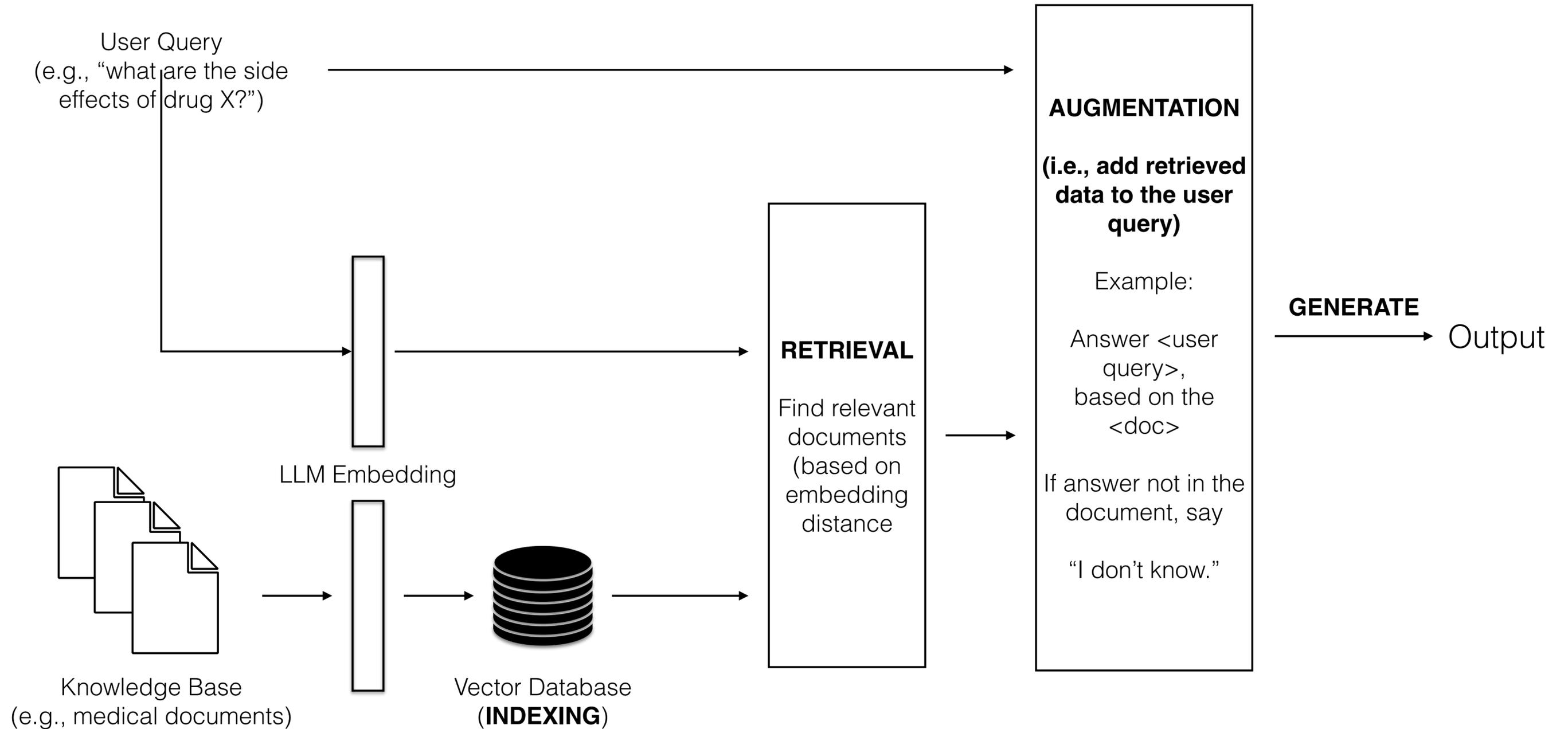
How RAG Solves These Problems

Integrates external knowledge sources (e.g., databases, documents, APIs).

Ensures answers are more accurate, up-to-date, and grounded.

More developer control. Allows for targeted customization without retraining the model

Question Answering with RAG Example



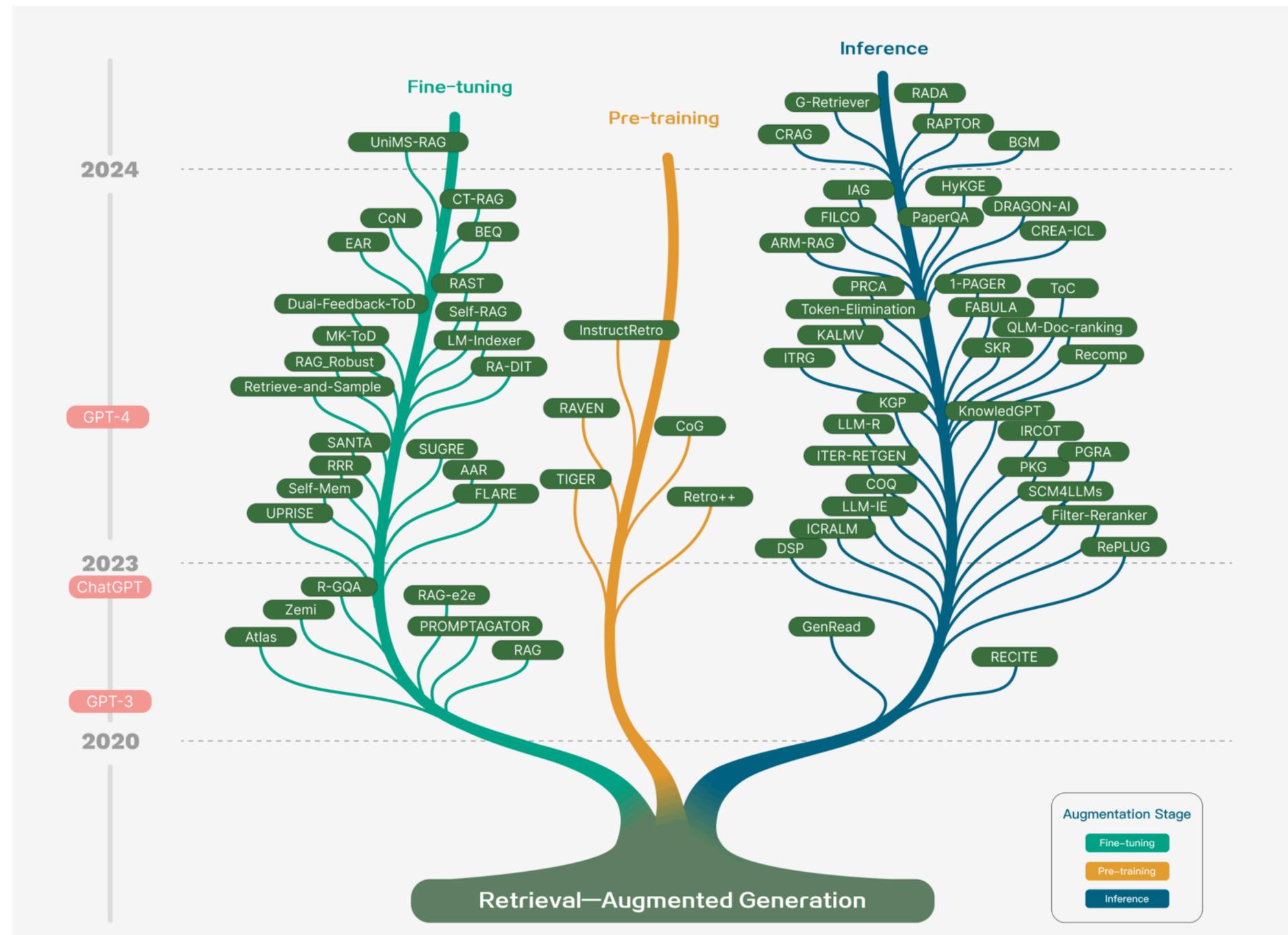


Fig. 1. Technology tree of RAG research. The stages of involving RAG mainly include pre-training, fine-tuning, and inference. With the emergence of LLMs, research on RAG initially focused on leveraging the powerful in context learning abilities of LLMs, primarily concentrating on the inference stage. Subsequent research has delved deeper, gradually integrating more with the fine-tuning of LLMs. Researchers have also been exploring ways to enhance language models in the pre-training stage through retrieval-augmented techniques.

Augmenting LLMs with:

Better Prompts: Crafting clearer instructions for better results.

Chains: Combining multiple LLM calls to tackle complex workflows.

Fine-Tuning: Adding customization, though it requires significant data and resources (not recommended).

Retrieval: Expanding context by integrating specific knowledge.

How could we extend the capabilities of LLMs from performing **single tasks** (enhanced with external knowledge) to handling **multi-step, autonomous workflows**?

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Definition

Andrew Ng: “An agentic AI workflow is a process uses where an LLM-based application executes multiple steps to complete a task.”

Query:
“What is your refund policy?”

Response (via RAG):
"Refunds are available within 30 days of purchase."

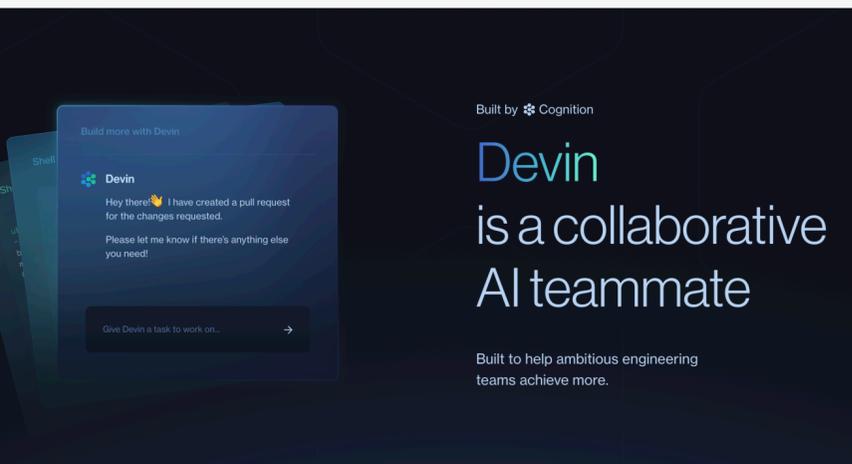
Query:
“Can I get a refund for my order?”

Response (via AI Agent):

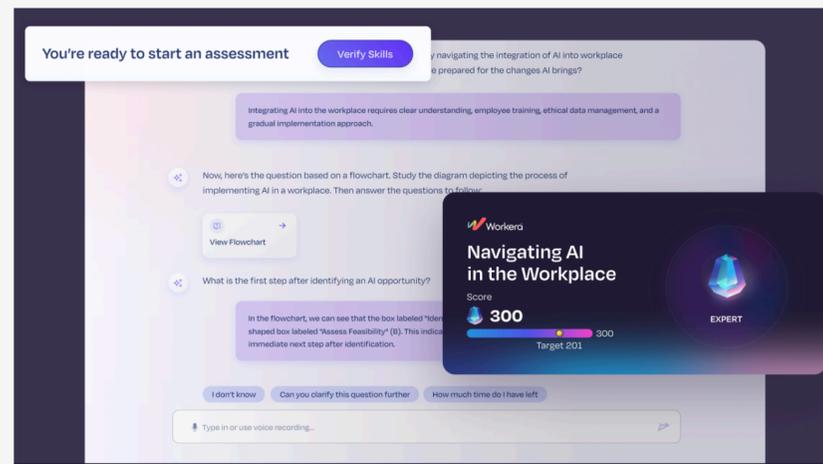
- The **agent** retrieves refund policy using RAG.
- The agent asks: "Can you provide your order number?"
- It queries an API to check the order details.
- It confirms: "Your order qualifies for a refund. The amount will be processed in 3–5 business days."

There are more and more specialized agentic workflows.

AI software engineer



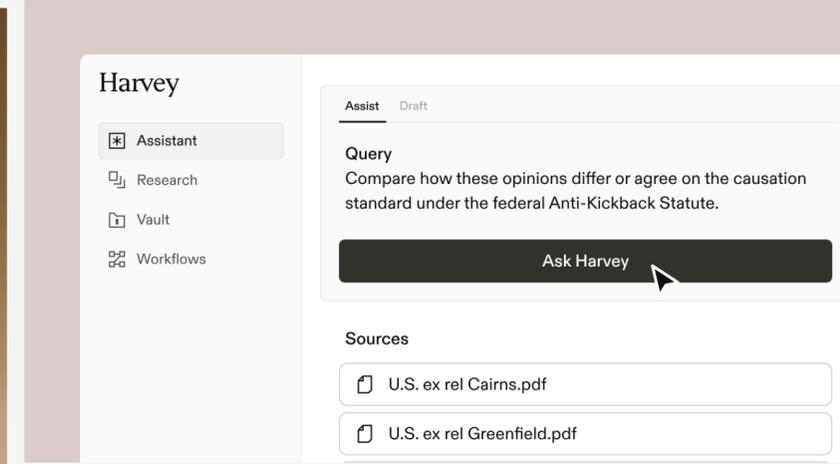
AI mentor



AI SDR



AI lawyer

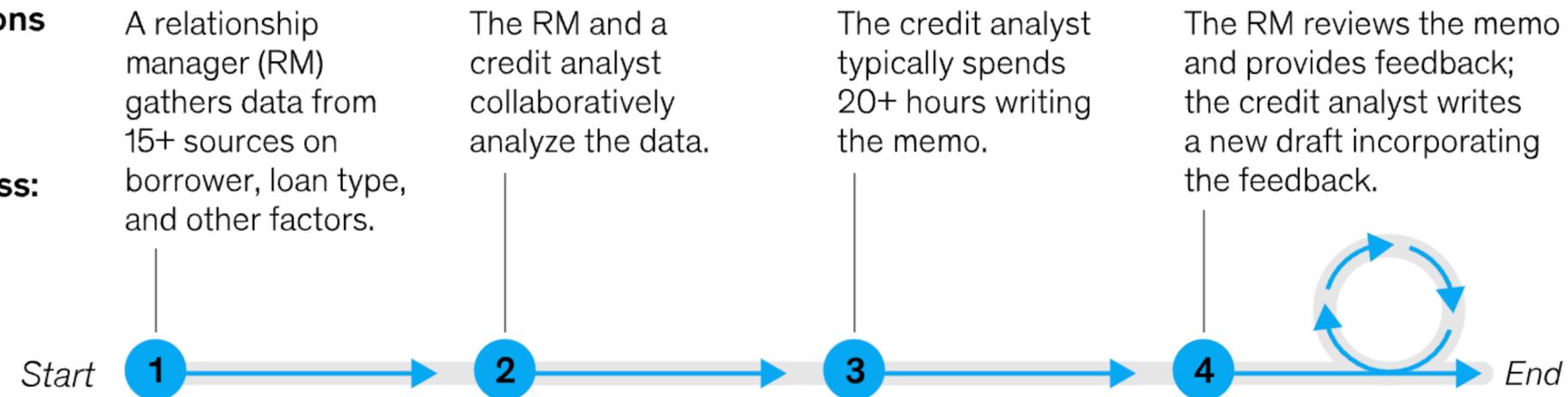


Paradigm shift

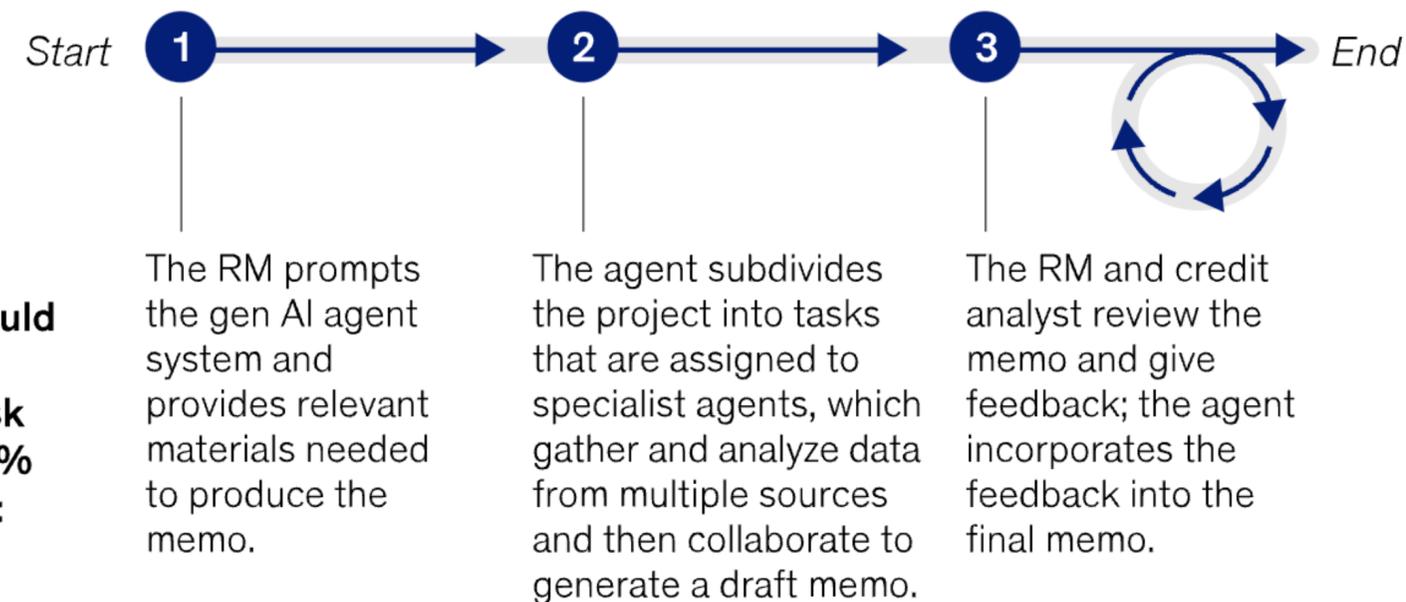
Aspect	Traditional Software	Agentic AI Software
Data Handling	Works with structured data in predefined formats (e.g., databases, JSON): strong typed input/output.	Handles unstructured inputs like free text, requiring dynamic interpretation: fuzzy input/output
Logic and Behavior	Follows deterministic, rule-based logic with predictable and repeatable behavior.	Operates on fuzzy logic and probabilistic reasoning, making outcomes less predictable.
Development Approach	Developers define specific functions and workflows explicitly.	Developers combine prompt design, chaining, and external tools (e.g., APIs, databases) to build workflows.
Maintenance and Updates	More stable and predictable; fixing one issue rarely impacts unrelated features.	Fixing or adjusting one prompt, tool, or logic can inadvertently break multiple unrelated workflows.
User Interaction	Features static, predefined interaction flows (e.g., menus, forms).	Enables dynamic, conversational interactions, responding flexibly to a variety of user intents.
Testing and Debugging	Testing is well-defined, with deterministic outcomes for given inputs.	Requires iterative and exploratory testing due to non-deterministic and context-sensitive behavior.
Adaptability	Changes require explicit reprogramming for new scenarios or tasks.	Can adapt dynamically to new inputs, but requires careful integration to maintain stability.
Systems Design	Microservices or Monolithic	“Think like a Manager”

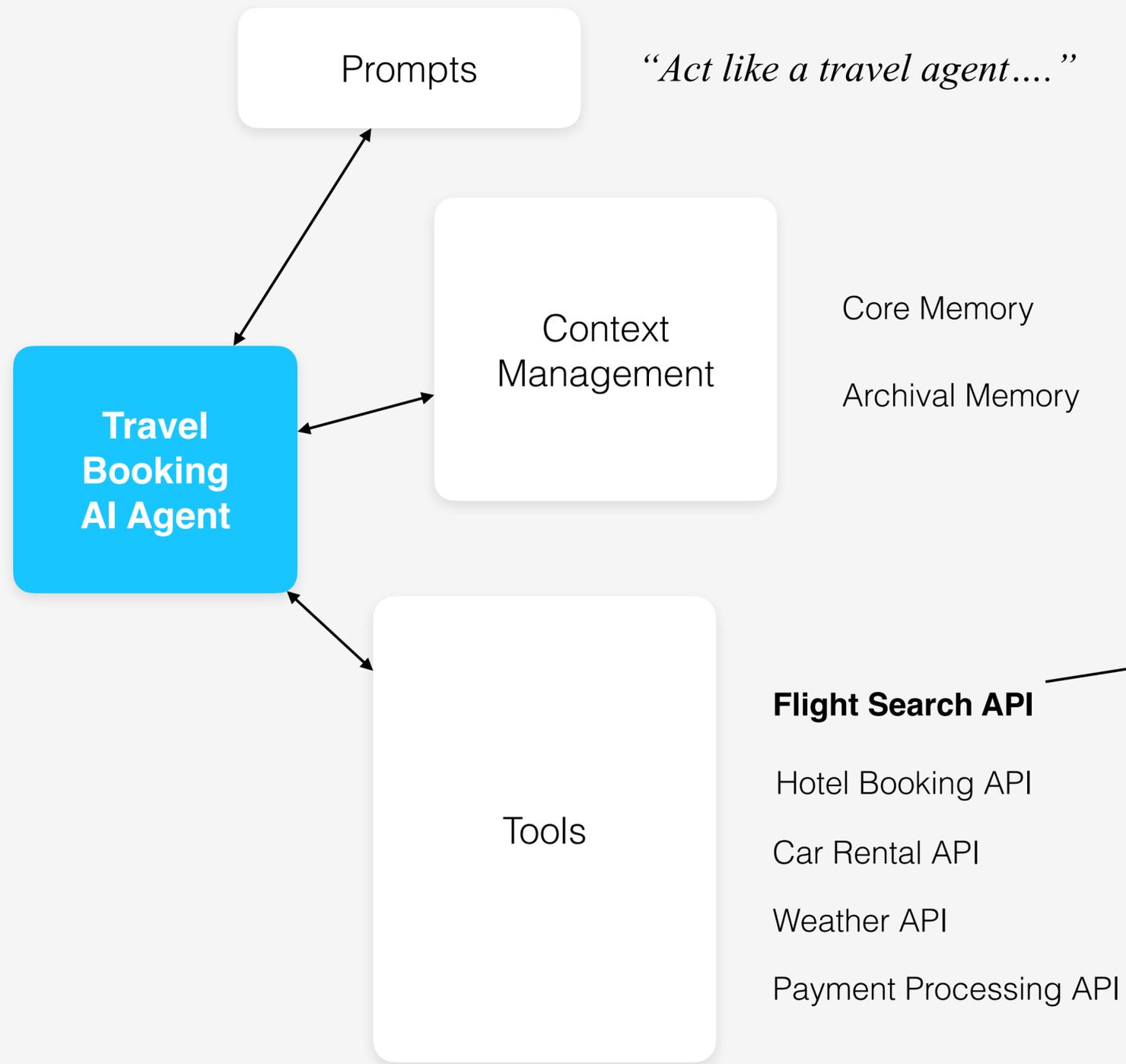
Enterprise workflows are likely to change to rely more on agentic AI workflows.

Financial institutions often spend 1–4 weeks creating a credit-risk memo. The current process:



Generative AI (gen AI) agents could cut time spent on creating credit-risk memos by 20–60% using these steps:





What It Does:

Retrieves available flights based on origin, destination, date, and preferences (e.g., airline, price).

How It Functions:

- Accepts input parameters like location, date, and budget.
- Returns flight options, including departure time, duration, and cost.

When to Use:

- Use when the user requests flight information or compares options (e.g., "Find flights from New York to Paris on December 15th.").

Degrees of Autonomy

Less autonomous: Hard-coded steps and tools

Semi-autonomous: Hard-coded tools, agent can decide what to use and when

More autonomous: Agent decides the steps and can create tools

With API

An AI travel agent today might connect directly to, say, the Amadeus or Skyscanner API:

```
requests.get("https://api.amadeus.com/v2/flights?origin=SFO&destination=CDG")
```

That's a one-off, vendor-specific integration.

- The model (or developer) must know the Amadeus API spec.
- If tomorrow you switch to Skyscanner, you rewrite that integration.
- The model can't reason abstractly about the *concept* of “searching flights.”

Model Context Protocol (MCP)

An emerging standard for making AI models *context-aware and tool-agnostic*. They define a structured way for models (like GPT-5) to access **relevant context, APIs, and data sources**, without needing hardcoded integrations for each system.

With MCP

The same travel agent could instead talk to a Flight MCP server, which exposes a standardized interface to any underlying flight provider. For example, the model might send this structured request through MCP:

```
{  
  "action": "search_flights",  
  "parameters": {  
    "origin": "SFO",  
    "destination": "CDG",  
    "date": "2025-11-20"  
  }  
}
```

- The MCP server handles translating that into the correct Amadeus, Skyscanner, or internal travel API calls and returns results in a consistent format.
- Now the model doesn't need to know which provider it's using, it just knows the “Flight Search” schema defined by MCP.

Step 1

User Input

User: "Plan a trip to Paris from December 15-20th with flights, hotels near the Eiffel Tower, and an itinerary for must-visit places."

Step 2

Agent Plans the Steps

The agent breaks down the task into logical steps:

- 1. Find flights:** Use the Flight Search API to get options for December 15th.
- 2. Search hotels:** Use the Hotel Booking API to find accommodations near the Eiffel Tower for the trip duration.
- 3. Generate recommendations:** Use the Location Recommendation API to compile a list of must-visit places in Paris.
- 4. Validate preferences:** Confirm options with the user for any additional constraints or preferences (e.g., budget, airline choice).
- 5. Book the trip:** Use the Payment Processing API.

Step 3

Execute the Plan

- 1. Use Tools:**
 - Flight Search API → Retrieves flight options for December 15th.
 - Hotel Booking API → Fetches hotel options near the Eiffel Tower.
 - Location Recommendation API → Gathers popular attractions and activities in Paris.
- 2. Combine Results:**
 - Combine and rank options based on user preferences (if stored in memory, e.g., budget-friendly airlines, 4-star hotels).

Step 4

Proactive User Interaction & Booking

Agent: "Here's a proposed itinerary for your trip to Paris:

- Flights: Direct flights with Air France and Delta starting at \$500.
- Hotels: 4-star hotels near the Eiffel Tower, starting at \$180 per night.
- Activities: Visit the Eiffel Tower, Louvre Museum, and Notre-Dame Cathedral.

Would you like to proceed with bookings or adjust these options?"

Then, the agent goes and books the trip using the Payment Processing API.

Step 5

Memory Integration

The agent stores preferences or decisions for future use:

- "User prefers direct flights."
- "User likes 4-star hotels near landmarks."
- "User has interest in cultural attractions."

How do you know if this works?

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Case Study

Your product manager asks you to build an AI agent for customer support. Where do you start?

Example User Prompt: “I need to change my shipping address for order A127. I moved to 325 Mission Street.”

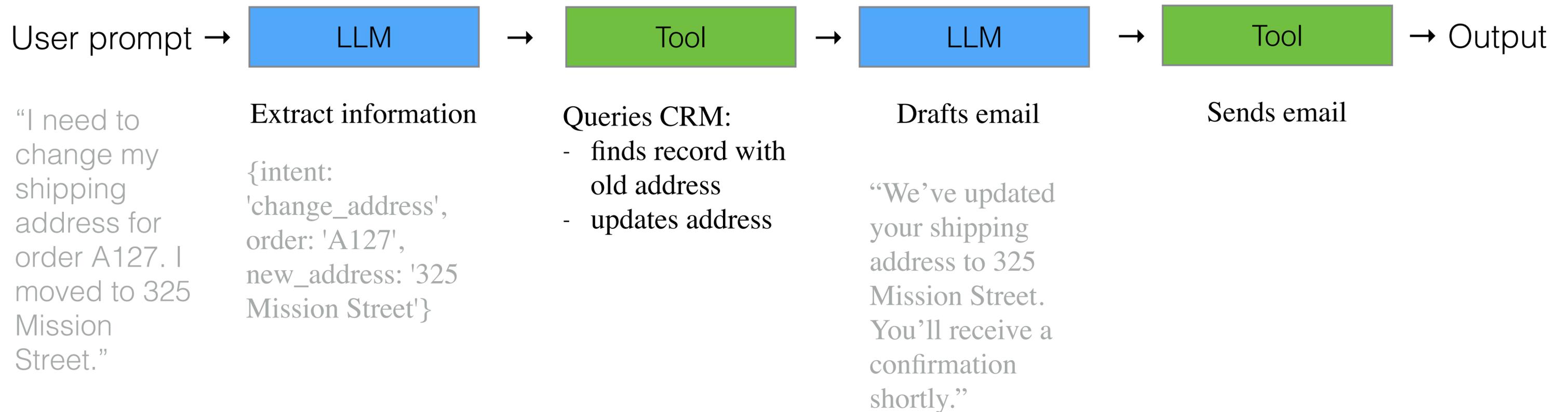
Step 1: Task Decomposition

1. Extract key info (intent, entities, order ID)
2. Retrieve customer record
3. Check policy (e.g., can we update address?)
4. Draft response email
5. Send email

Case Study

Your product manager asks you to build an AI agent for customer support. Where do you start?

Step 2: Agentic Workflow Design



Case Study

Your product manager asks you to build an AI agent for customer support. Where do you start?

Step 3: How would you know if it works? (Evals) (Assuming you have traces in place)

Method	Description	Example
Component-based evals	Measure each step separately	Extraction accuracy (precision/recall), API error rate, policy adherence
End-to-end evals	Score the overall user experience	Final response correctness, tone, satisfaction

Type	Example	Measures
Objective	Did the system extract the correct order ID? Did it apply the refund policy correctly?	Accuracy, recall, policy adherence
Subjective	Did the response sound polite, empathetic, helpful?	Human ratings, LLM-as-judge, rubric-based

Type	Example
Quantitative	% of successful address updates, latency, refund accuracy
Qualitative	Observing patterns in hallucinations, tone mismatches, user confusion

Case Study

Your product manager asks you to build an AI agent for customer support. Where do you start?

Step 3: How would you know if it works? (Evals)

User Input	Agent Output	Issue Observed	Type of Issue
"I'd like to change my address for order #A127."	"We've updated your address for order #A172."	Extracted wrong order ID	Objective (Extraction)
"My order arrived damaged, can I get a refund?"	"Unfortunately, refunds aren't available."	Incorrect policy application	Objective (Policy)
"I never got my delivery."	"Please check your porch."	Tone is dismissive, lacks empathy	Subjective (Tone)
"I ordered a gift for my daughter's birthday, but it came late."	"Refund processed."	Factual but emotionally tone-deaf	Subjective (Helpfulness)
"Cancel my subscription next month."	"Your subscription has been canceled today."	Misinterpreted time condition	Objective (Logic)

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

Multi-agent workflows

Example: Smart Home Automation

Climate Control Agent: Adjusts heating, cooling, and ventilation based on preferences or weather conditions.

Lighting Agent: Manages indoor and outdoor lights, including brightness, color, and schedules.

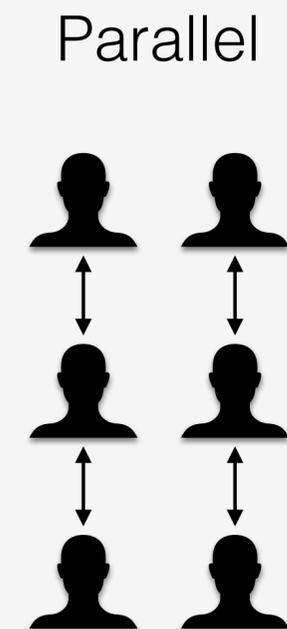
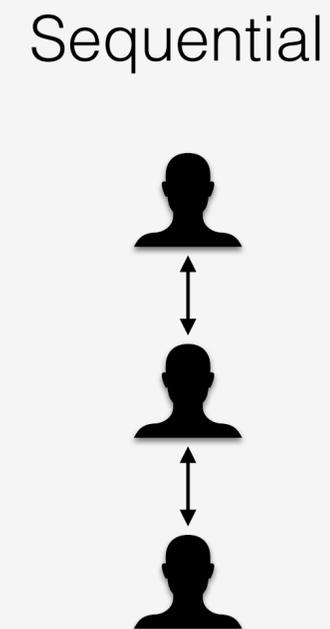
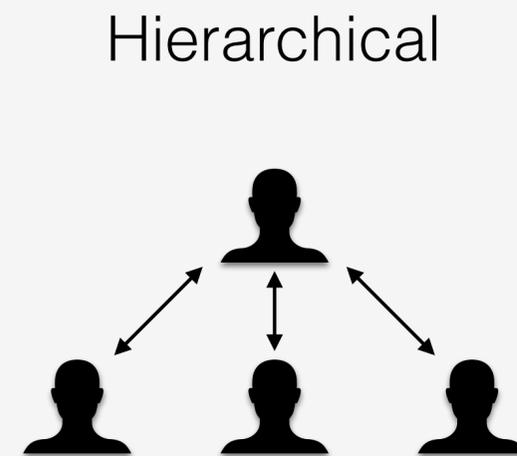
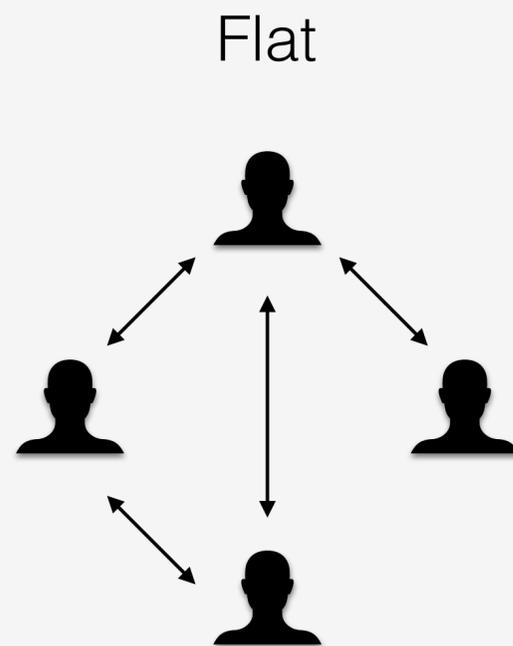
Security Agent: Monitors cameras, locks doors, and alerts users to unusual activity.

Energy Management Agent: Optimizes energy usage by turning off unused devices or switching to eco-modes.

Entertainment Agent: Controls TVs, speakers, and other media devices based on user requests.

Notification Agent: Alerts users about system updates, energy savings, or security events.

Orchestrator Agent: Ensures smooth coordination between specialized agents, managing workflows, dependencies, and conflicts.



Multi-agent workflows

Example: Smart Home Automation

Climate Control Agent: Adjusts heating, cooling, and ventilation based on preferences or weather conditions.

Lighting Agent: Manages indoor and outdoor lights, including brightness, color, and schedules.

Security Agent: Monitors cameras, locks doors, and alerts users to unusual activity.

Energy Management Agent: Optimizes energy usage by turning off unused devices or switching to eco-modes.

Entertainment Agent: Controls TVs, speakers, and other media devices based on user requests.

Notification Agent: Alerts users about system updates, energy savings, or security events.

Orchestrator Agent: Ensures smooth coordination between specialized agents, managing workflows, dependencies, and conflicts.

Considerations	Single Agent	Multi-Agent Workflow
Modularity	Harder to update; all-in-one system.	Easy to update or replace specific components.
Fault Isolation	One failure can disrupt everything.	Issues are contained to the failing agent.
Optimization	General-purpose, less efficient.	Specialized agents excel in their domains.
Debugging	Troubleshooting is complex and time-consuming.	Easier to isolate and fix issues.
Parallel Processing	Sequential task handling; slower.	Handles tasks simultaneously.
Flexibility	All tasks tied to one system.	Allows selective integration and third-party tools.

Optimizing LLMs with:

Better Prompts: Crafting clearer instructions for better results.

Chains: Combining multiple LLM calls to tackle complex workflows.

Fine-Tuning: Adding customization, though it requires significant data and resources (not recommended).

Retrieval: Expanding context by integrating specific knowledge.

Fine-Tuning: Adding customization, though it requires significant data and resources (not recommended).

Retrieval: Expanding context by integrating specific knowledge.

Today's outline

- I. **Augmenting LLMs:** Challenges and Opportunities
- II. **Prompt Engineering:** The First Line of Optimization
- III. **Fine-Tuning:** Proceed with Caution
- IV. **Retrieval-Augmented Generation (RAG):** Enhancing Model Utility
- V. **Agentic AI Workflows:** Toward Autonomous and Specialized Systems
- VI. **Case Study:** Evals
- VII. **Multi-Agent Workflows:** Parallelism
- VIII. **What's Next in AI?** Personal Thoughts

What's Next in AI? Personal Thoughts

Scaling laws: Plateau or Not?

Ilya Sutskever recently noted that more data might not always lead to better AI, potentially slowing progress. However, with thousands of experts now advancing AI, far beyond the small teams that revolutionized fields like computer vision, there's immense potential for breakthroughs. Like in Asimov's Foundation, where individuals shape the future, this growing community can uncover transformative innovations, even as we look beyond scaling data.

Gains from Multimodality

Multimodality enhances AI by making each modality stronger when combined—for example, integrating vision and language helps models better understand context, enabling more accurate and versatile applications across tasks.

Multiple Methods in Harmony

The future of AI might lie in combining diverse approaches, supervised, unsupervised, and reinforcement learning, into unified systems that leverage the strengths of each method, creating solutions that are more robust and adaptable.

Human Approaches vs. Non-human Approaches

Research is looking at both human-like and non-human approaches to innovate in AI. Human-like methods, such as language models mimicking reasoning or vision systems imitating perception, enable intuitive applications. Non-human approaches unlock creative solutions beyond what our brains can do. Balancing both is key to advancing AI.

Half life of skills and Learning Velocity

In the rapidly evolving field of AI, the half-life of skills (the time it takes for half of someone's expertise to become obsolete) is shrinking at an unprecedented rate. This acceleration underscores the critical importance of learning velocity, or how quickly individuals and organizations can acquire, adapt, and apply new skills.