HUX AI

Empowering humanity, shaping tomorrow

Agentic LM 101

Dr. Merve Ayyüce KIZRAK
Founder & CEO

Hayriye ANIL
Data Scientist
AI Researcher

Who are we?

HUX AI is a human-centric tech company that offers AI risk management tools, research, training, and consulting to promote ethical and responsible AI.

We aim to align technological innovation with social values for a better future

Why do we do what we do?

To empower organizations and individuals with advanced AI technologies and insights that respect human values and social well-being, fostering deeper understanding and responsible implementation of AI.

Where do we want to go?

To lead AI safety research and consulting, ensuring human-centered AI fosters societal progress and coexists harmoniously with humanity.

Join our community or learn more - www.huxai.tech

Research Internship Program has kicked off!

This summer, we're embarking on a journey filled with real-world, high-impact projects at the intersection of responsible AI, data governance, and policy.

#ResearchInternship

Duration:
6 or 9 weeks
Location:
Fully remote and flexible schedule
Goal:
Mentorship + global community support

Three powerful project tracks:

1 ISO 42001 Starter Guide

Helping organizations take their first steps toward AI management standards with a plain-language guide, self-assessment checklist, and a simple roadmap.

2 AI Literacy Playbook for Project Managers

Making AI governance clear for non-technical leaders with a visual handbook, engaging examples, glossary, and presentation deck.

3 AI Governance Starter Kit for SMEs

Designing tailored governance tools, editable templates, and role/responsibility maps for small and mid-sized enterprises.

This journey isn't just about delivering projects; it's about:

- Clarifying career goals
- Building a global professional network
- Gaining expertise in AI governance
- Developing solutions that shape the future

Stay tuned — we'll share our progress, updates, and final outputs here!

Join our community or learn more - www.huxai.tech

Outline

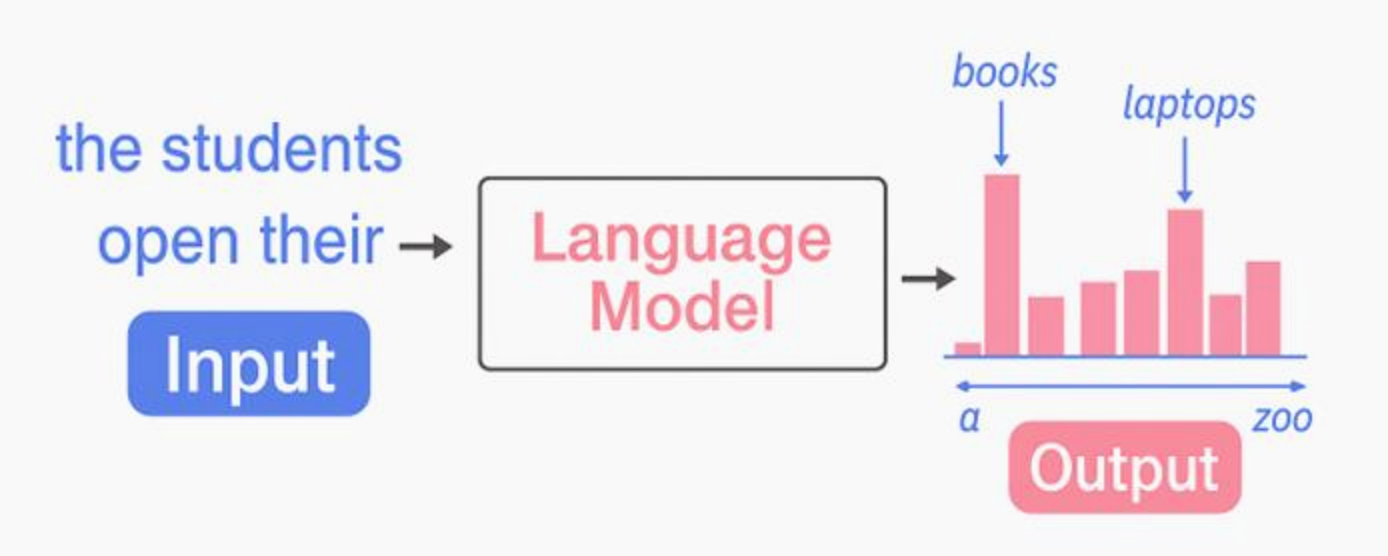
- Introduction
- Language Models: Overview & Training
- Using Language Models in Practice
  - Applications & API usage
  - Prompt Engineering Strategies
- Common Limitations of LMs
- Enhancing LM Capabilities
  - Retrieval-Augmented Generation (RAG)
  - Tool Usage
- From LMs to Agentic LMs
- Agentic LM Design Patterns
  - Planning, Reflection, Tool Usage
  - Multi-agent Collaboration & Example
- Summary & Key Takeaways
- References

Empowering
humanity,
shaping
tomorrow

Language Model

Language Model: Predicts next word given input

- Input: text
- Output: next word prediction



How Language Models are trained

1. Pre-training: Language Model is training with large corpus by next token prediction objective
2. Post-training:
   a) Instruction following training adapts pre-trained LM to follow specific instruction and commands. It is also called as SFT (Supervised Fine-Tuning)
      i. It makes the models easier to use.
      ii. It makes the model to respond in a specific style.

   b) RLHF (Reinforcement Learning with Human Feedback): method that fine-tunes the model using human preference to align generated behaviors with human values and intentions

A Typical Instruction Dataset Template

Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.

Instruction:
(instruction)

Input:
(input)

###Response
(response)

Input is optional

Note that there are many ways to prompt Language Model

Language Model Usage

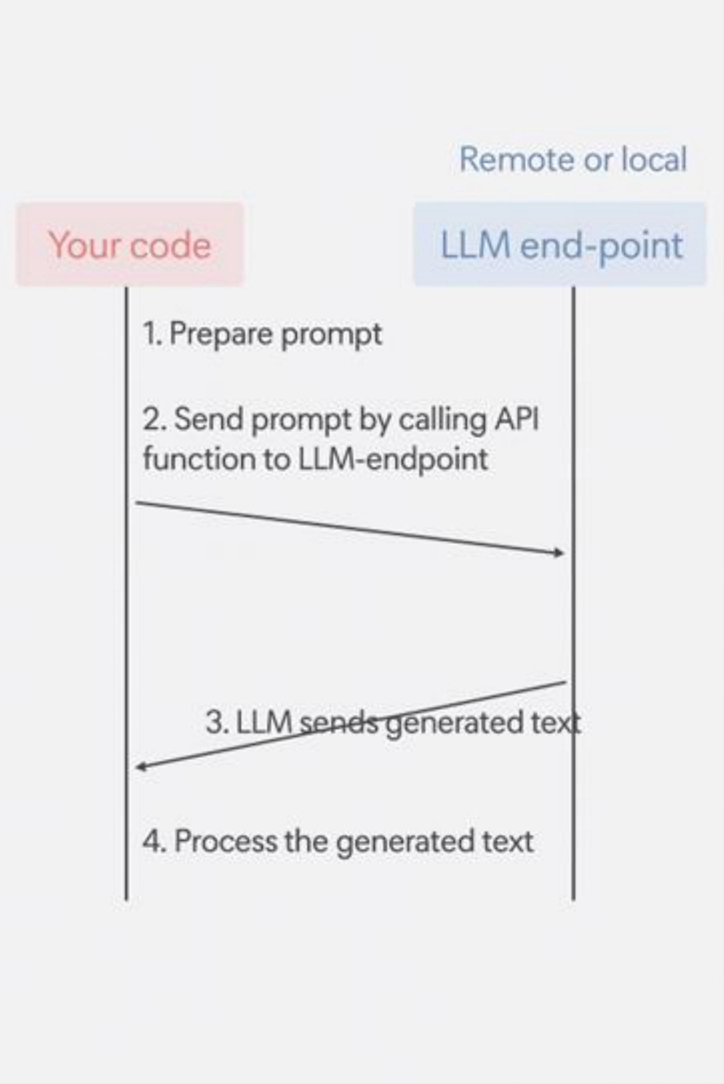
Current Status and Applications

- LMs are rapidly evolving, with new models and products being introduced regularly
- Numerous applications leverage LMs:
  - AI coding assistants
  - Domain-specific AI copilots
  - ChatGPT and other conversational interfaces
- Various deployment options:
  - Cloud API providers
  - Local model deployments

Language Model Usage

Working with APIs

- Text generation via API endpoints
- Integration with service providers
- Implementation steps
  - ➢ Prepare prompts programmatically
  - ➢ Make API calls
  - ➢ Receive and process responses
  - ➢ Iterate or chain with other prompts and tools as needed


Remote or local
Your code
LLM end-point
1. Prepare prompt
2. Send prompt by calling API function to LLM-endpoint
3. LLM sends generated text
4. Process the generated text

Language Model Usage

Working with Prompts

- Natural language interaction through well-crafted prompts
- Key strategies:
  - Write clear, descriptive instructions
  - Include few-shot examples
  - Provide relevent context (static or dynamic)
  - Enable Chain of Thought (CoT) reasoning.Give model time to think
  - Break down complex tasks. Chain complex prompts
  - Systematic trace and evaluation of prompts for performance improvement

Write clear, descriptive instructions

Write clear instruction

LLMs cannot read your mind, so be more specific. The less the model has to guess at what you want, the more likely you'll get it.

Tactic: Include details in your query to get more relevant answers

In order to get a highly relevant response, make sure that requests provide any important details or context. Otherwise you are leaving it up to the model to guess what you mean.

Worse	Better
How do I add numbers in Excel?	How do I add up a row of dollar amounts in Excel? I want to do this automatically for a whole sheet of rows with all the totals ending up on the right in a column called "Total".
Who's president?	Who was the president of Mexico in 2021, and how frequently are elections held?
Write code to calculate the Fibonacci sequence.	Write a TypeScript function to efficiently calculate the Fibonacci sequence. Comment the code liberally to explain what each piece does and why it's written that way.
Summarize the meeting notes.	Summarize the meeting notes in a single paragraph. Then write a markdown list of the speakers and each of their key points. Finally, list the next steps or action items suggested by the speakers, if any.

Include few-shot examples

Tactic: Provide examples

Providing general instructions that apply to all examples is generally more efficient than demonstrating all permutations of a task by example, but in some cases providing examples may be easier. For example, if you intend for the model to copy a particular style of responding to user queries which is difficult to describe explicitly. This is known as “few-shot” prompting.

SYSTEM	Answer in a consistent style.
USER	Teach me about patience.
ASSISTANT	The river that carves the deepest valley flows from a modest spring; the grandest symphony originates from a single note; the most intricate tapestry begins with a solitary thread.
USER	Teach me about the ocean.

Provide relevant context (static or dynamic)

Provide reference text

LLMs can generate fake answers for those they don't know.. Provide reference text (context), like a cheat sheet for a student exam. LLMs will use reference instead of making things up.

Tactic: Instruct the model to answer using a reference text

If we can provide a model with trusted information that is relevant to the current query, then we can instruct the model to use the provided information to compose its answer.

SYSTEM If	Use the provided articles delimited by triple quotes to answer questions. the answer cannot be found in the articles, write "I could not find an answer."
USER	<insert articles, each delimited by triple quotes> Question: <insert question here>

Strategy: Give model time to ”think”

Give the model time to “think”:

LLMs make more reasoning errors when asked answer right away instead of taking time to work it out. Ask for “chain of thought (CoT)” method before answering can help the model to reason toward correct answers.

SYSTEM

Determine if the student’s solution is correct or not.

USER

Problem Statement: I’m building a solar power installation and I need help working out the financials.

- Land costs $100 / square foot
- I can buy solar panels for $250 / square foot
- I negotiated a contract for maintenance that will cost me a flat $100k per year, and an additional $10 / square foot

What is the total cost for the first year of operations as a function of the number of square feet.

Student’s Solution: Let x be the size of the installation in square feet.

1. Land cost: 100x
2. Solar panel cost: 250x
3. Maintenance cost: 100,000 + 10x

Total cost: 100x + 250x + 100,000 + 10x = 450x + 100,000

Model likely answer that it is correct

Strategy: Give model time to ”think”

SYSTEM

First work out your own solution to the problem. Then compare your solution to the student’s solution and evaluate if the student’s solution is correct or not. Don’t decide if the student’s solution is correct until you have done the problem yourself.

USER

Problem Statement: I’m building a solar power installation and I need help working out the financials.

- Land costs $100 / square foot
- I can buy solar panels for $250 / square foot
- I negotiated a contract for maintenance that will cost me a flat $100k per year, and an additional $10 / square foot

What is the total cost for the first year of operations as a function of the number of square feet.

Student’s Solution: Let x be the size of the installation in square feet.

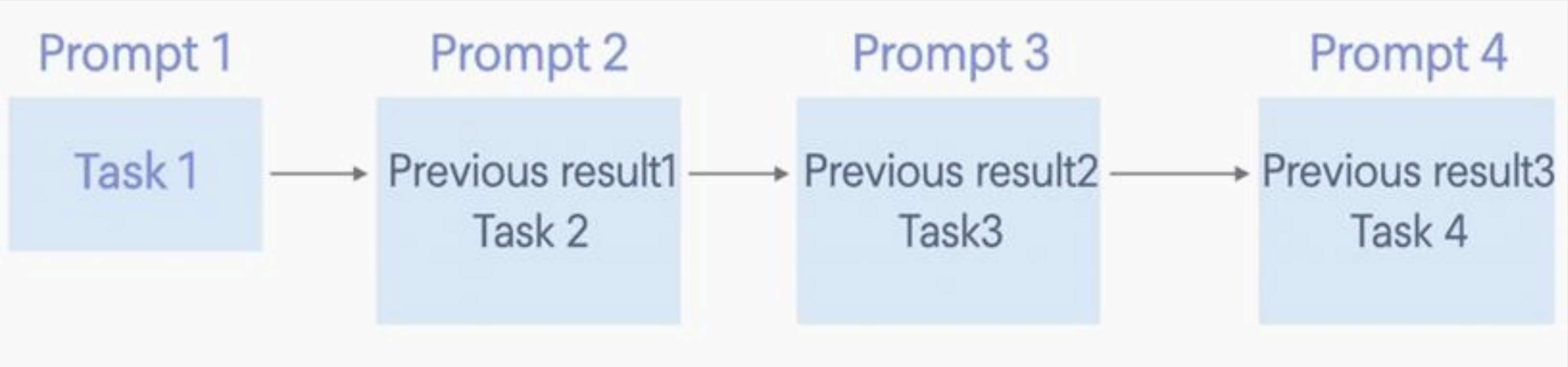
1. Land cost: 100x
2. Solar panel cost: 250x
3. Maintenance cost: 100,000 + 10x

Total cost: 100x + 250x + 100,000 + 10x = 450x + 100,000

Model likely answer it is incorrect

Break down complex task: chain complex prompts

- Split complex tasks into simplier subtasks, and chain them with separate prompts: Complex tasks result in higher errors. Decomposing a complex task into a set of simplier tasks is a good idea



Prompt 1

Task 1

Prompt 2

Previous result1
Task 2

Prompt 3

Previous result2
Task3

Prompt 4

Previous result3
Task 4

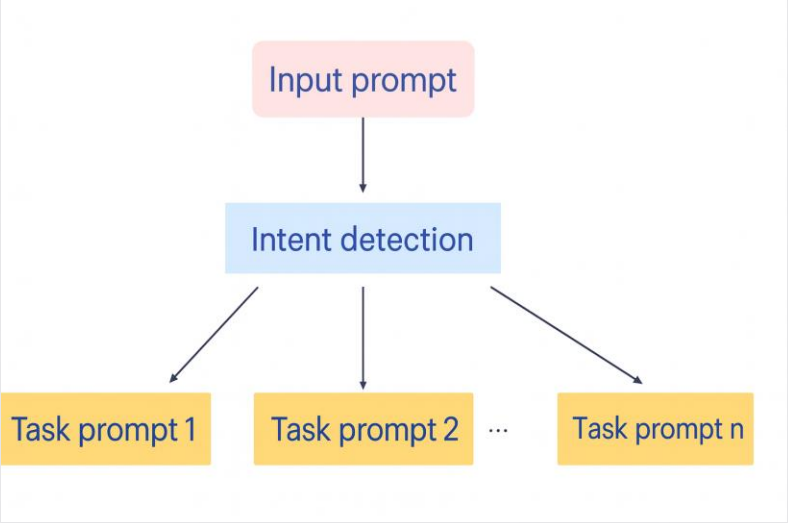
Systematic trace and evaluation of prompts for performance improvement

- Test changes systematically: Improving performance is easier if we can measure it. Prepare ways to evaluate systematically.

1. Have evaluation plan early
2. Prepare evalutaion data
3. LM as a judge together with human evaluation will be a good starting point

Intent Detection, routing

- Handling diverse tasks requiring different prompt strategies
- Benefits:
  - Determines appropriate prompt selection
  - Enables task decomposition
  - Facilitates LM model selection based on task type
- Helps manage complex workflows through systematic task breakdown
- Determine which different LMs to use depending on the detected intent

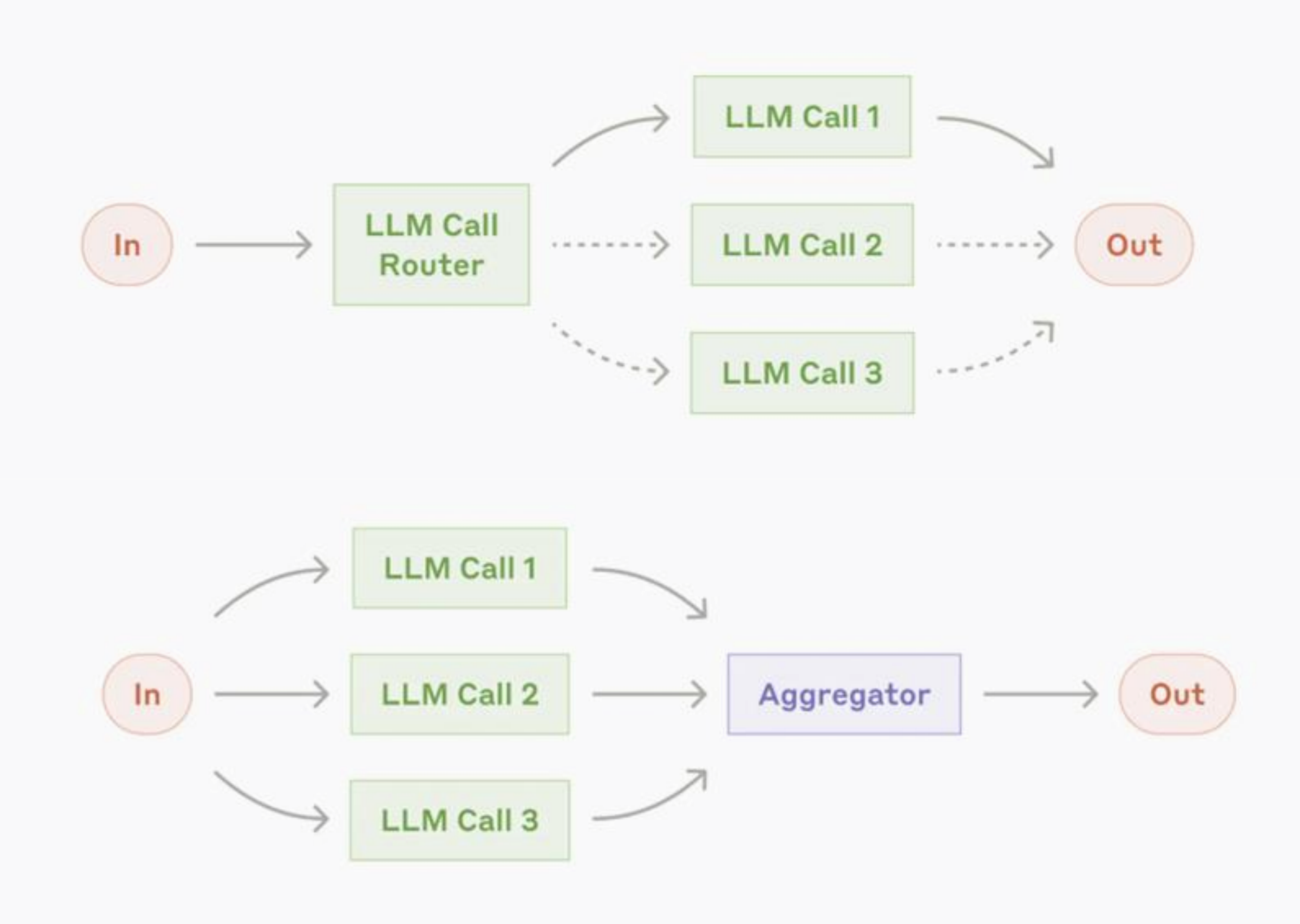


Prompt to identify intent:

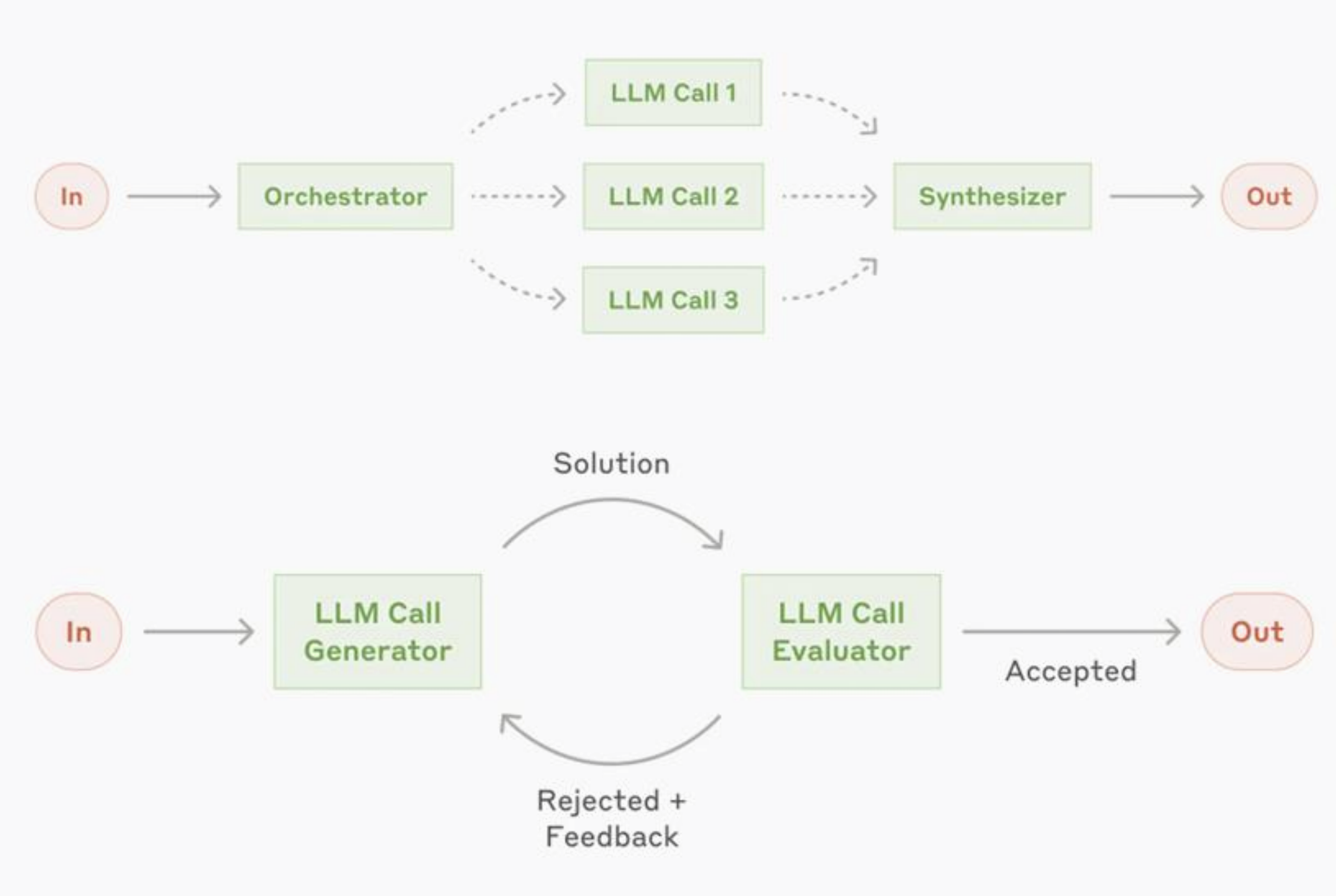
Based on the intent use different prompt to follow up

- It can be iterative calls to LLMs
- It can be different LLMs based on the detected intent

Routing & Parallelization



Orchestrator & Evaluator



Common Limitation of Language Models

- Hallucination: Generation of incorrect information with high confidence
- Knowledge cutoff: Limited to training data timeframe
- Lack of attribution: No direct source citations
- Data privacy: Limited to public training data, no access to proprietary information
- Limited context length: Constraints due to attention mechanism architecture

RAG (Retrieval Augmented Generation)

To augment LMs with knowledge from external sources, Retrieval Augmented Generation (RAG) is commonly used.

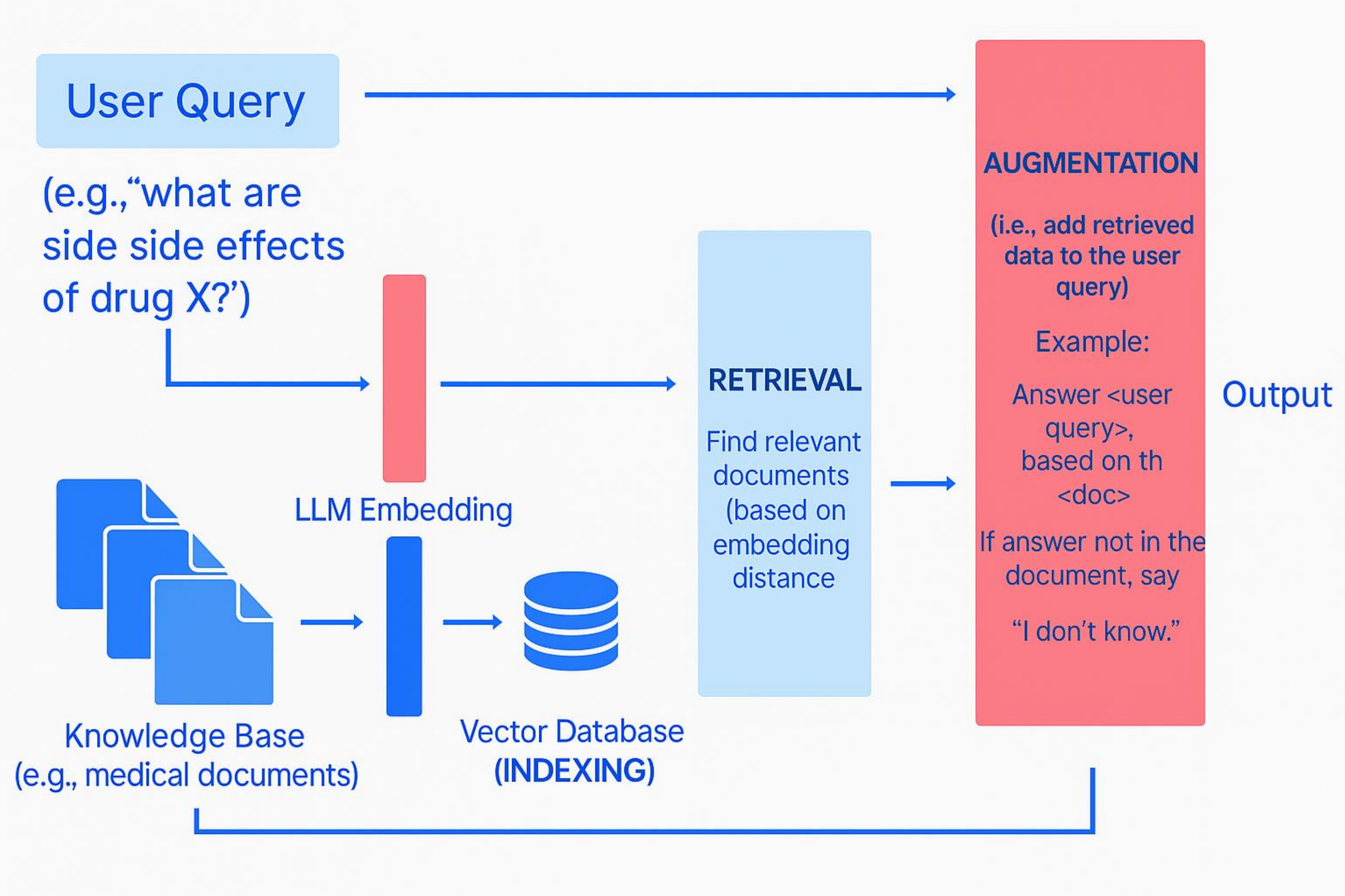
RAG addresses the following challenges

- Hallucination
- Lack of attribution
- Data privacy
- Limited context length

How it works:

1. Text preprocessing:
   a. Chunking large texts appropriately
   b. Converting documents to embeddings
2. Storage:
   a. Maintaining embedding-text pairs
3. Query processing:
   a. Converting queries to embeddings
   b. Performing similarity search (Retrieval)
   c. Generating enhanced prompts with context (Augmentation)
   d. Submitting to LM for final output (Generation)

Question Answering with RAG Example



User Query

(e.g.,“what are side side effects of drug X?’)

LLM Embedding

Knowledge Base
(e.g., medical documents)

Vector Database
(INDEXING)

RETRIEVAL

Find relevant documents (based on embedding distance

AUGMENTATION

(i.e., add retrieved data to the user query)

Example:

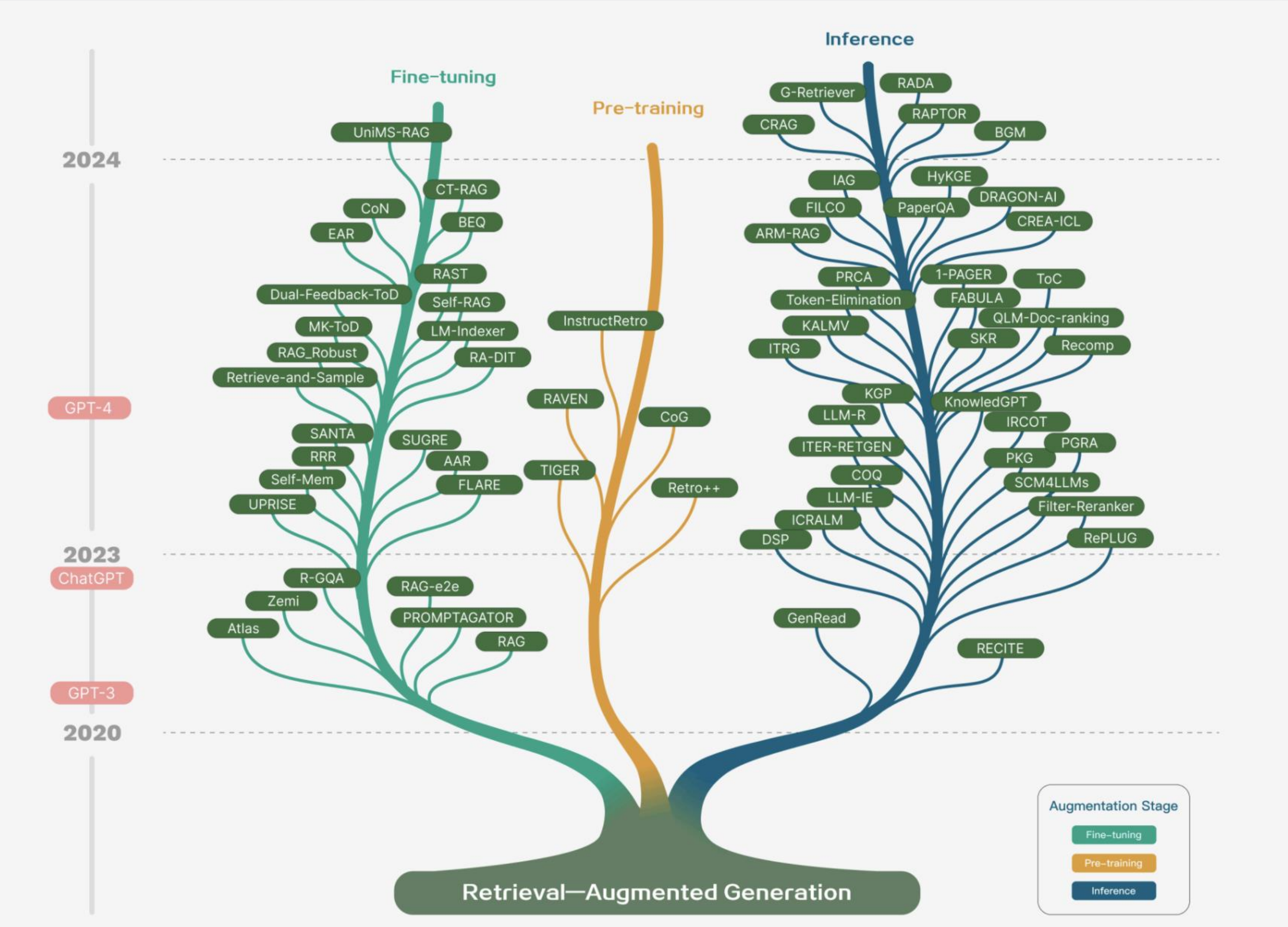
Answer <user query>, based on th <doc>

If answer not in the document, say

“I don’t know.”

Output

RAG (Retrieval Augmented Generation)



Tool Usage

LMs can generate function signatures to interact with external tools, enabling capabilities such as making API calls or executing code.

Tool usage address the following challenges

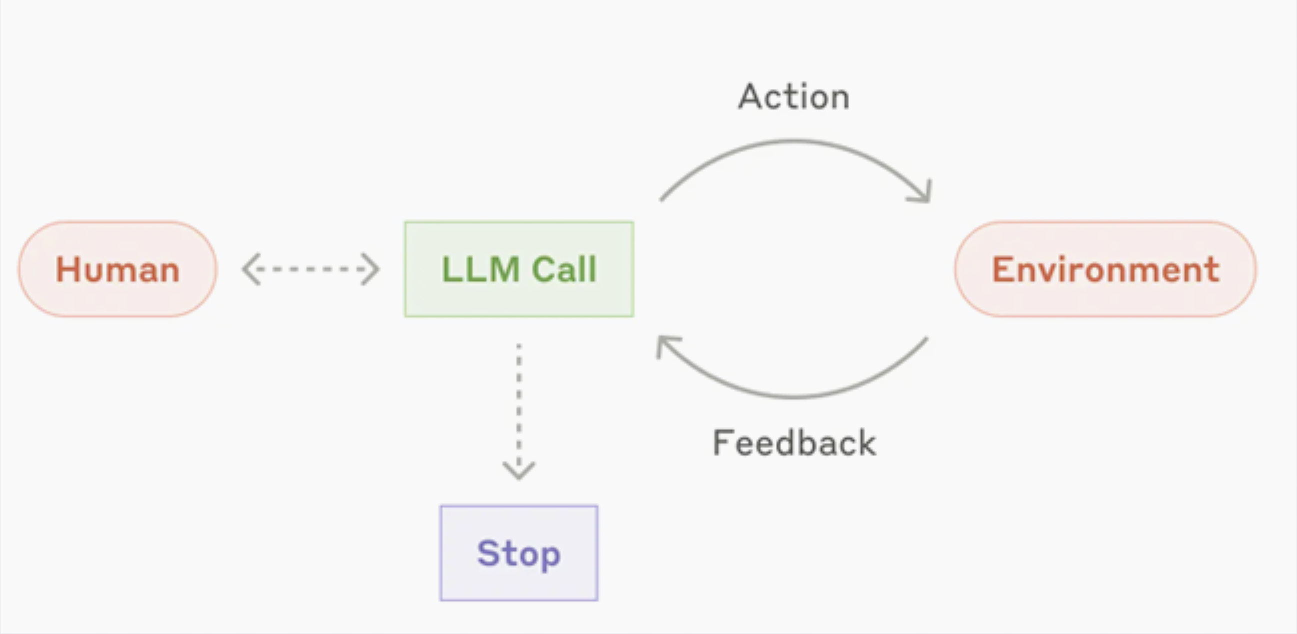
- Real-time information
- Computations

- Example: “What is the best cafe based on user reviews?”
  - LLM will generate {tool: web-search, query: "cafe reviews"}
  - External tool searches and provides result to LM
- Example: “What is the weather in San Francisco?”
  - LLM will generate {tool: get-weather, query: "SF"}
  - External tool is called and provides result to LM
- Example: “If I invest $100 at 7% compound interest for 12 years, what do I have at the end?”
  - LLM generates Python code with this: {tool: python-interpreter, code: "100 * (1+0.07)**12"}
  - Generates code, runs it, and produces the output

Agentic Language Models

Interact with environments

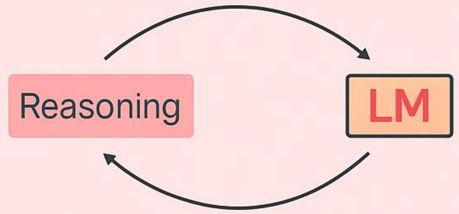
- Simpe LM use cases involve text in and text out
- Inreract with environment



Agentic Language Models

ReAct (Reason and Action)

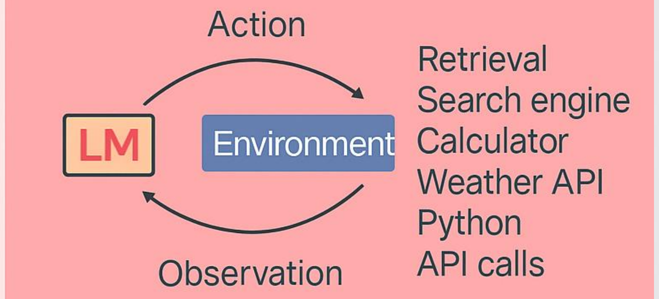
CoT (chain of thought)



Flexible and general to augment inference-time generation

Lacking external information

RAG/Retrieval/Tool use



Lacking reasoning

Flexible and general to augment knowledge, computation, feedback

Agentic Language Models

Overview

A progression of LM usage

- Ability to Reason and Act:
  - Reason about tasks: break down complex tasks and plan actions
  - Execute actions: use external tools, code, and retrieve information
  - Interact with environment: using tools
  - Learn from feedback

Example task:

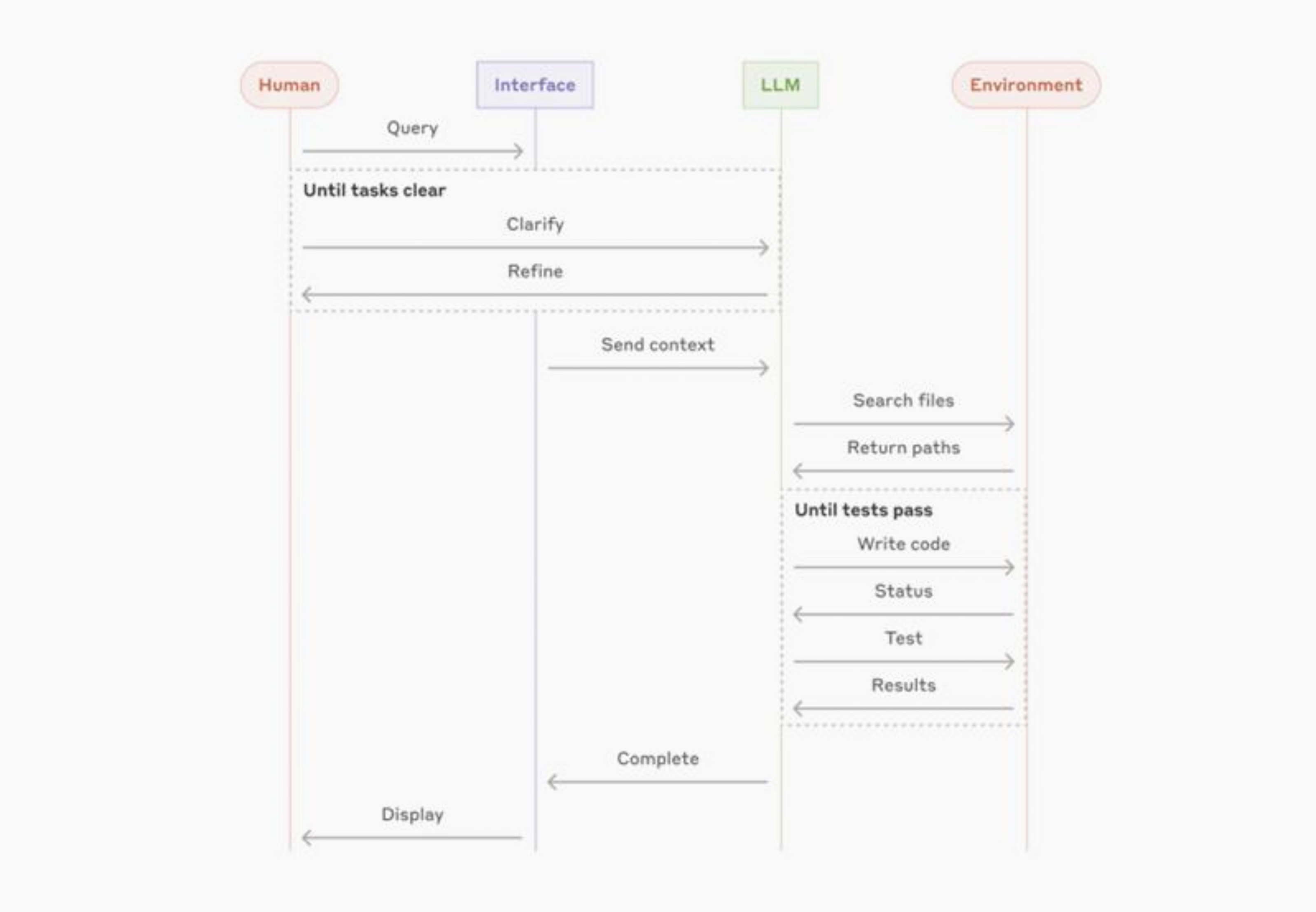
Customer support case: “Can I get a refund for product Foo?”

- Check the refund policy
- Check the customer order information
- Check the product information
- Generate a refund plan and response

Agentic Language Models

- Agent workflow: LMs iterate over documents or tasks, using external tools or code to:
  - Research topics using web data, summarize findings, and present output
  - Prepare plans for software fixes and iteratively propose code solutions
- Utilize LMs with different roles (such as generator and critics) iteratively
- Generate plans for using external tools
- Agentic LM usage can achieve more complex tasks than non-iterative and LM-only patterns

Agentic Language Models



Human

Interface

LLM

Environment

Query

Until tasks clear

Clarify

Refine

Send context

Search files

Return paths

Until tests pass

Write code

Status

Test

Results

Complete

Display

Agentic Language Models

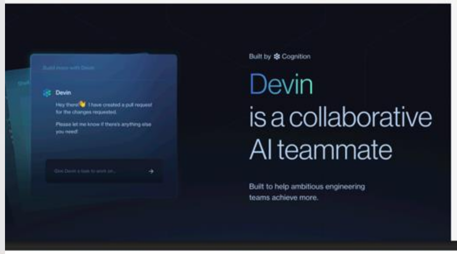
Real-World Applications

- Software Development
  - Code generation and review
  - Bug detection and fixing

- Research and Analysis
  - Data gathering and synthesis
  - Report generation

- Task Automation
  - Workflow optimization
  - Process automation

AI software engineer



Devin

Hey there👋 I have created a pull request for the changes requested.

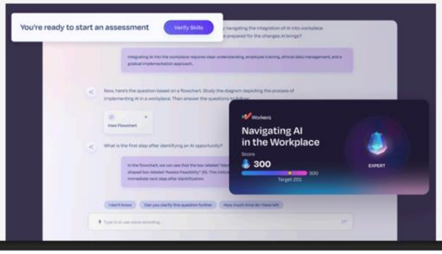
Please let me know if there's anything else you need!

Built by Cognition

Devin is a collaborative AI teammate

Built to help ambitious engineering teams achieve more.

AI psychometrician



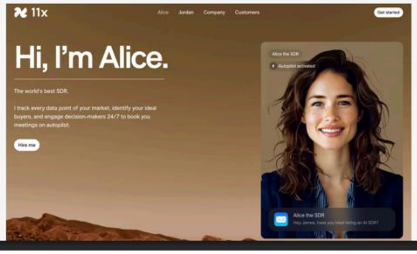
You're ready to start an assessment

Integrating AI into the workplace requires clear understanding, employee training, ethical data management, and a gradual implementation approach.

Now, here's the question based on a flowchart. Study the diagram depicting the process of implementing AI in a workplace. Then answer the questions to follow.

What is the first step after identifying an AI opportunity?

AI SDR

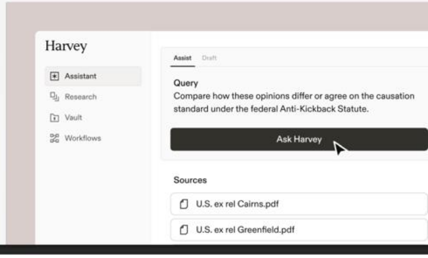


Hi, I'm Alice.

The world's best SDR.

I track every data point of your market, identify your ideal buyers, and engage decision-makers 24/7 to book you meetings on autopilot.

AI lawyer



Query

Compare how these opinions differ or agree on the causation standard under the federal Anti-Kickback Statute.

Sources

U.S. ex rel Cairns.pdf

U.S. ex rel Greenfield.pdf

Agentic Language Model Design Patterns

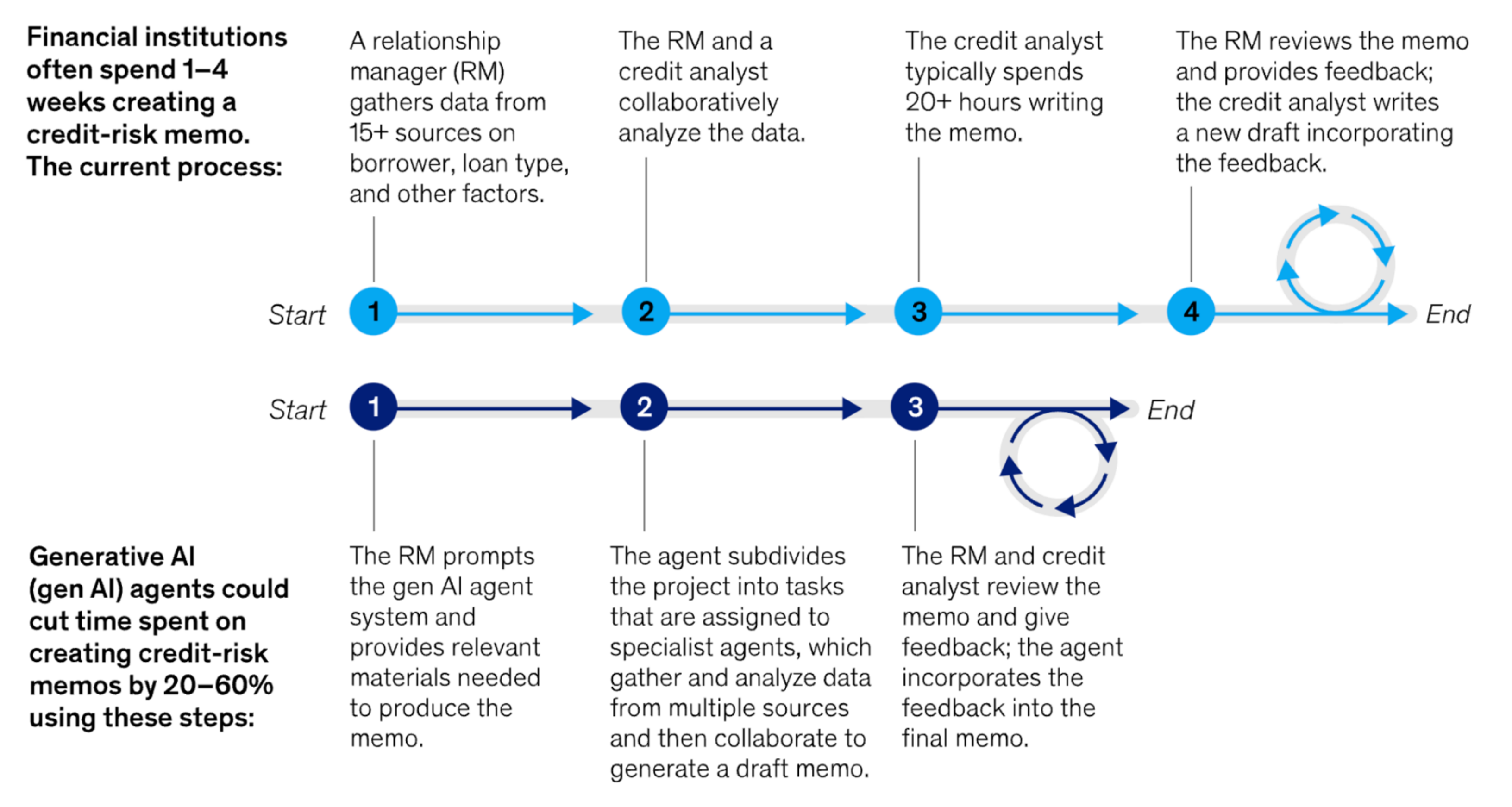
Here are some key agent design patterns:

- Planning: Multi-step planning to achieve goals
- Reflection: Examines its own work and improves
- Tool usage: Utilizes web search, code executions
- Multi-agent collaboration: Splits work and facilitates discussion

Paradigm Shift

Aspect	Traditional Software	AI Agent 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 change to rely more on Agentic workflows



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

1. A relationship manager (RM) gathers data from 15+ sources on borrower, loan type, and other factors.
2. The RM and a credit analyst collaboratively analyze the data.
3. The credit analyst typically spends 20+ hours writing the memo.
4. The RM reviews the memo and provides feedback; the credit analyst writes a new draft incorporating the feedback.

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

1. The RM prompts the gen AI agent system and provides relevant materials needed to produce the memo.
2. The agent subdivides the project into tasks that are assigned to specialist agents, which gather and analyze data from multiple sources and then collaborate to generate a draft memo.
3. The RM and credit analyst review the memo and give feedback; the agent incorporates the feedback into the final memo.

Multi-agent

Design a multi-agent system for smart home automation

- Climate control agent: Manages temperature and humidity based on setting and weather conditions
- Lighting control agent: Manages lighting based on schedule and time of day
- Security agent: Monitors cameras and alerts when there is a security breach
- Energy management agent: Monitors energy usage and suggests savings opportunities
- Entertainment agent: Manages TVs, audio, and streaming services based on user preferences
- Orchestration agent: Coordinates all agents to ensure smooth operation

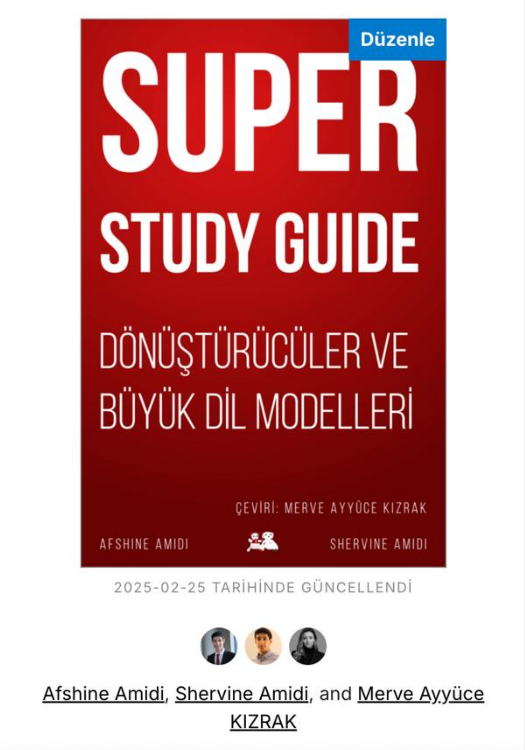
Summary

- Agentic LMs represent the next evolution in AI assistance
- Key advantages:
  - Autonomous reasoning and action
  - Tool integration capabilities
  - Iterative improvement through feedback
  - Complex task decomposition
- Growing applications across software development, research, and automation
- Continuous evolution of patterns and capabilities

References

The following are references for the presentation. We learned from these sources and reused content and images from them:

- Agentic Design Patterns Part 1
- Large Language Model Agents, MOOC Fall 2024
- Natural Language Processing with Deep Learning, 2024
- Building Effective Agents
- RAG and AI Agents from Deep Learning
- Tool Use and LLM Agent Basics from Advanced NLP
- What are AI Agents?
- Frontiers-of-AI-Agents-Tutorial
- Agentic AI: A Progression of Language Model Usage



SUPER STUDY GUIDE

DÖNÜŞTÜRÜCÜLER VE BÜYÜK DİL MODELLERİ

ÇEVİRİ: MERVE AYYÜCE KIZRAK

AFSHINE AMIDI SHERVINE AMIDI

2025-02-25 TARİHİNDE GÜNCELLENDİ

Afshine Amidi, Shervine Amidi, and Merve Ayyüce KIZRAK

SCAN ME

Thank you!

huxai.tech

Empowering
humanity,
shaping
tomorrow