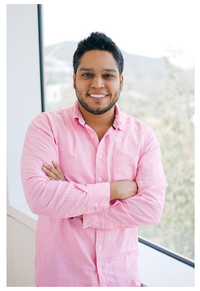
Google Developers

Experts

Developing Agentic Applications Workshop: Intro



Juan Guillermo Gómez
GDE Firebase & GCP & AI/ML
@jggomezt

Juan Guillermo Gómez

- Co-leader and co-founder of GDG Cali.
- Tech Lead at **WordBox** & Founder DevHack.
- Google Developer Expert (GDE) in Firebase & GCP & AI/ML
- BS in System Engineering and a MS in Software Engineering.
- @jggomezt
- devhack.co

SCAN ME

Youtube

Experts

Henry Ruiz

- ➔ Research Scientist at Texas A&M University @ AgriLife Research.
- ➔ Exploring LLM capabilities for remote sensing data (UAVs, Satellite, LiDAR, GPR).
- ➔ Google Developer Expert (GDE) in AI & Google Cloud.
- ➔ Actively contributes to AI community (open-source, public speaking, mentorship).
- ➔ BS in System Engineering and a Ph.D in Computer Science.
- ➔ @devharuiz
- ➔ https://haruiz.github.io/

Experts

Networking

Linkedin

X

Instagram

Onlyfans

Agenda

Módulo 1: Introducción a LLM y Agentes (2 Horas)

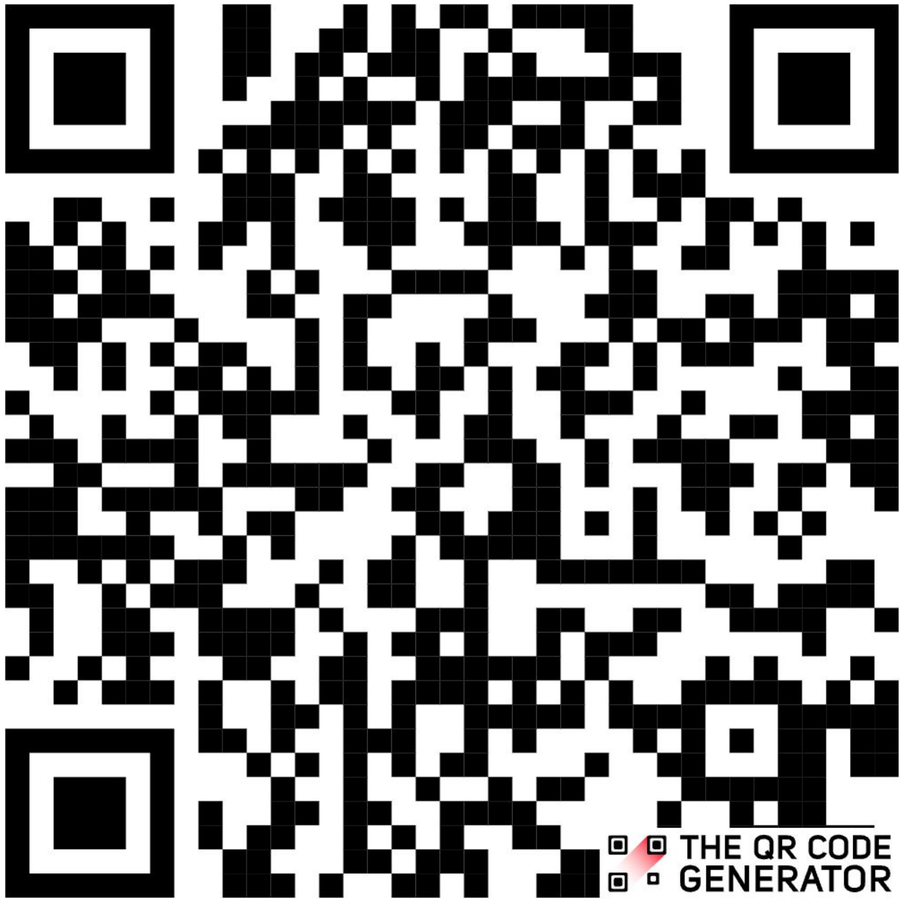
- ¿Qué son los LLM? (Capacidades, limitaciones, conceptos básicos de prompting).
- Definición de Agentes de IA: Más allá de los simples chatbots.
- Componentes Principales de un Agente
- ¿Por qué los Agentes? Casos de Uso y Potencial (Automatización, resolución de problemas complejos, asistentes personales).
- Breve Resumen: Panorama de los frameworks para desarrollo de agentes
- Laboratorio 1: Crear un agente conversacional simple o un agente básico de seguimiento de instrucciones.

Agenda

Módulo 2: Construyendo Tu Primer Aplicación multi agente (2 Horas)

- Diferencias clave entre aplicaciones de agente único (single-agent) y sistemas multiagente.
- Implementación desde cero de una arquitectura básica de sistema multiagente.
- Introducción a técnicas de ajuste posterior (post-training) aplicadas a sistemas basados en agentes.
- Patrones comunes de diseño en aplicaciones orientadas a agentes.
- Laboratorio 2: Introducción al Agent Development Kit (ADK).

GCP Credits

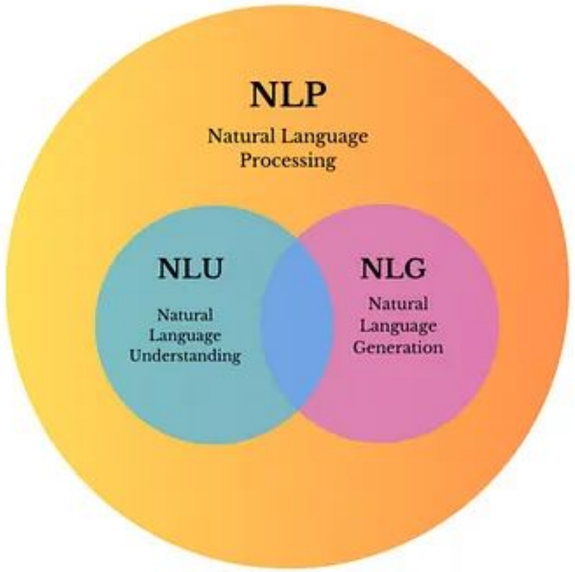


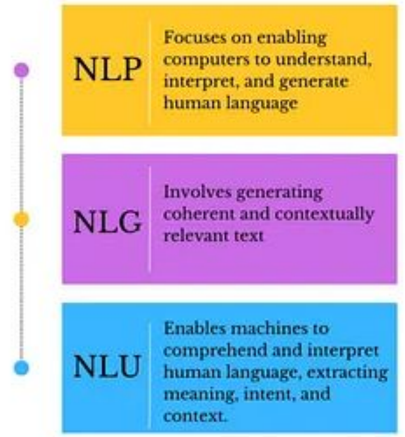
https://trygcp.dev/e/3p-intro-to-agents-01

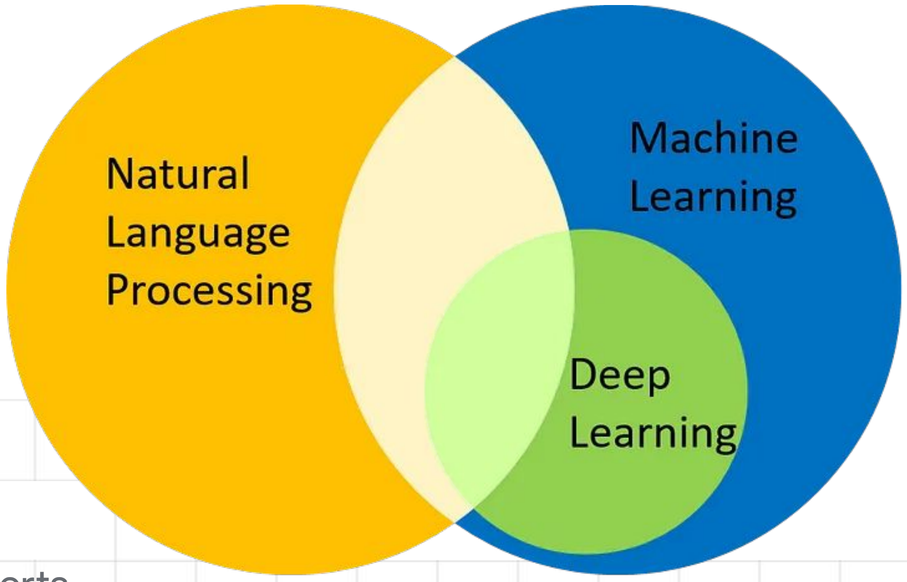
Experts

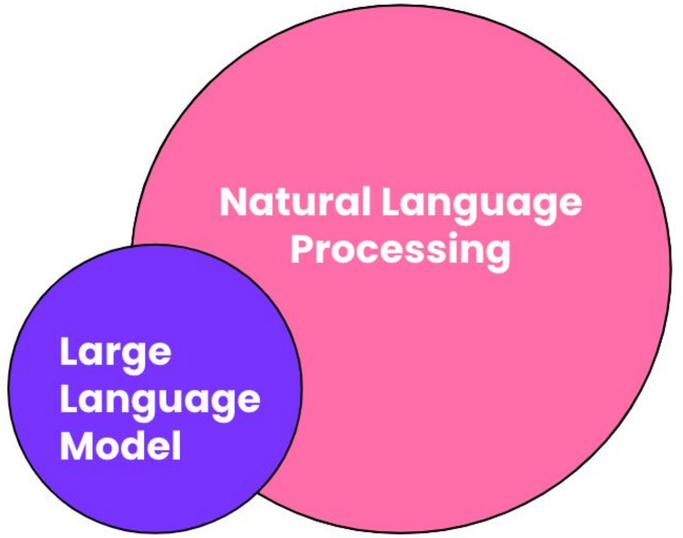
Now, let's going deep...

Large Language Models (LLM)









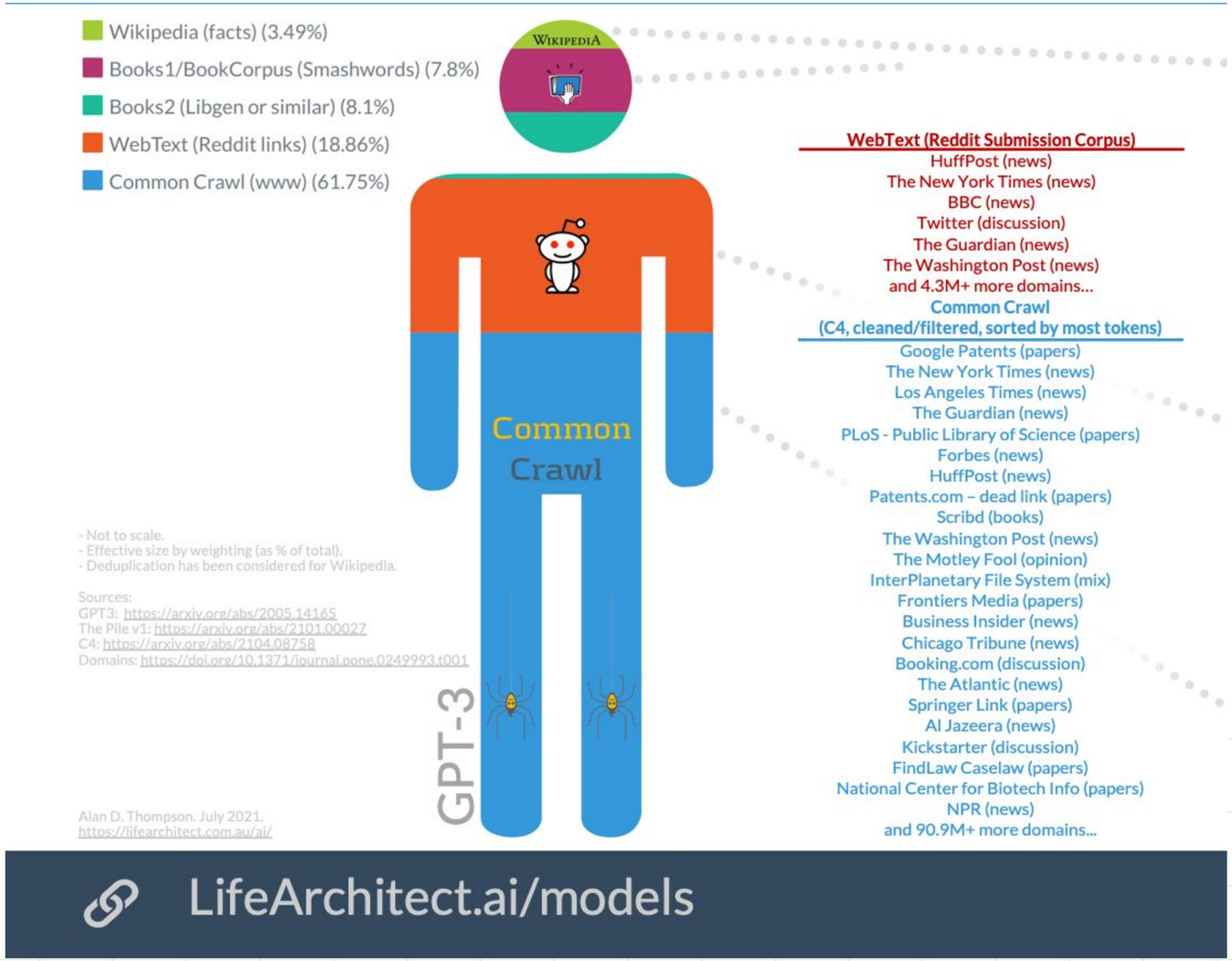
Experts

https://medium.com/@vipra_singh/llm-architectures-explained-nlp-fundamentals-part-1-de5bf75e553a

What are LLMs?

- A large language model (LLM) is a deep learning algorithm (≅ 100 million of parameters) capable to process and understand text, equipped to summarize, translate, predict, and generate text to convey ideas and concepts.

- Large Language Models (LLMs) leverage extensive datasets to identify linguistic patterns and gain a nuanced understanding of written language.



LLMs

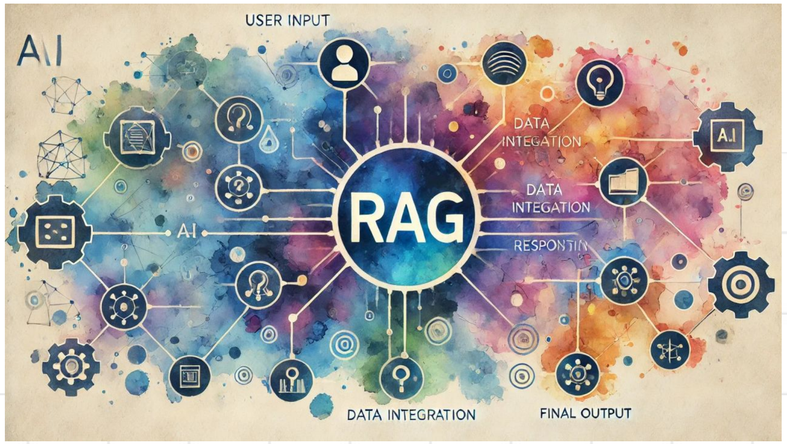

Top 15 LLM Terms You Need to Know
1 Transformers
Self-attention to analyze relationships between words, enabling a deeper understanding of sentences
2 Token
Basic units of text an LLM processes, like words or sub-words.
3 Chunking
Breaking down text into smaller, manageable segments for LLM to analyze.
4 Indexing
catalog for the massive datasets for efficient retrieval
5 Embedding
Represent words in numerical code and such that lets the LLM understand their relationships to each other.
6 Vector Search
Helps LLMs find similar information within their vast datasets using embeddings
7 LLM Agent
In Agent LLM is the central processing unit, orchestrating the sequence of actions required to fulfill a task
8 Vector Database
Stores embeddings allowing for efficient vector search
9 Prompt Engineering
Art of crafting clear and concise instructions for the LLM to achieve the desired outcome
10 Shot Learning
How much instruction an LLM needs to learn a new task.
Zero-Shot, One-Shot, N-Shot
11 Fine Tuning
Training a smaller model on top of a larger one, focusing on a specific task while keeping resource usage in check.
12 Indexing
Machines that can think and learn like humans
13 RAG
RAG teams up large language models with external knowledge bases for more accurate and up-to-date responses.
14 MoE
Allows an LLM to leverage multiple smaller expert models for improved performance on specific tasks
15 LoRA
technique for compressing large LLM models, making them smaller and faster to run on devices
@pvergadia
Follow for more!

Limitations of LLM

- LLM responds with non-current data
- LLM responds with data that they were trained
- “Hallucinations” are unreliable responses
- Responses senseless
- LLMs can be Limited by their internal knowledge

RAG

- Retrieval Augmented Generation
- RAG is a technique for augmenting LLM knowledge with additional data
- RAG incorporates data from external sources allowing LLMs to access relevant information in real-time
- No need to fine-tune or retrain a model
- LLMs can answer questions about specific source information



From LLMs to AI Agents

https://www.mongodb.com/pt-br/library/resources/ai-agents?x=inokiPMs

Are these AI Agents?

AI program designed to perform a specific task by leveraging a set of tools to accomplish it 🤔

Little AI Apps 🤔

For example, an AI program can use external tools to research a topic and generate a summary of the findings 🤔

AI Apps acting autonomous 🤔

AI App must be able to "reason" 🤔

AI Agents

A system that can perceive its environment through sensors, process this information, and act upon the environment through actuators to achieve specific goals. (vipra singh)

A system or program that can autonomously complete tasks on behalf of users or another system by designing its own workflow and by using available tools. (IBM)

AI Agents

An Agent is a system that leverages an AI model to interact with its environment in order to achieve a user-defined objective. It combines reasoning, planning, and the execution of actions (often via external tools) to fulfill tasks. (Hugging Face)

Agents are systems that independently accomplish tasks on your behalf. (Open AI)

AI Agents

AI agents are software systems that use AI to pursue goals and complete tasks on behalf of users. They show reasoning, planning, and memory and have a level of autonomy to make decisions, learn, and adapt. (Google)

AI Agents

AI Agents are systems designed to achieve a specific objective based on an initial stimulus. To do this, they utilize a reasoning model that allows them to break down the objective into manageable subtasks.

Agents not only controls the flow of operations but also reflects on the results obtained, learns from errors, adapts to new circumstances, and revises its plan if necessary. This enables them to make strategic decisions to ensure the successful accomplishment of the objective. Finally, they execute the determined actions, employing tools to interact with external systems and perform the tasks. (DevHack)

AI Agents

AI Agents have a goal and use a reasoning model to break it into subtasks. They control the flow, learn from errors, adapt, revise plans, and make decisions. They execute actions using tools to interact with external system. (DevHack)

AI Agents

Experts

https://huggingface.co/learn/agents-course/en/unit1/introduction

AI Agents

Experts

https://huggingface.co/learn/agents-course/en/unit1/introduction

AI Agents

Experts

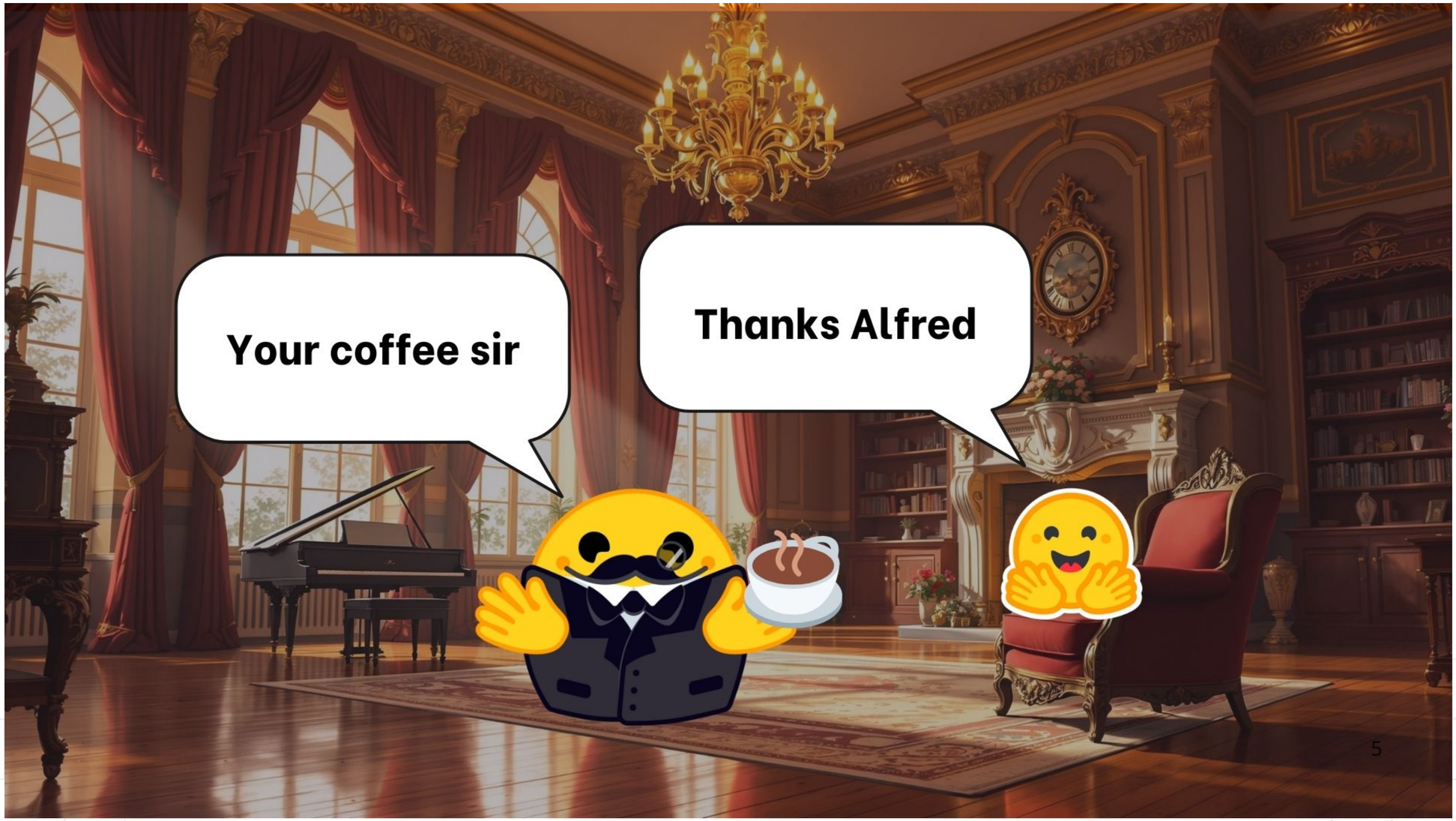
https://huggingface.co/learn/agents-course/en/unit1/introduction

AI Agents

Experts

https://huggingface.co/learn/agents-course/en/unit1/introduction

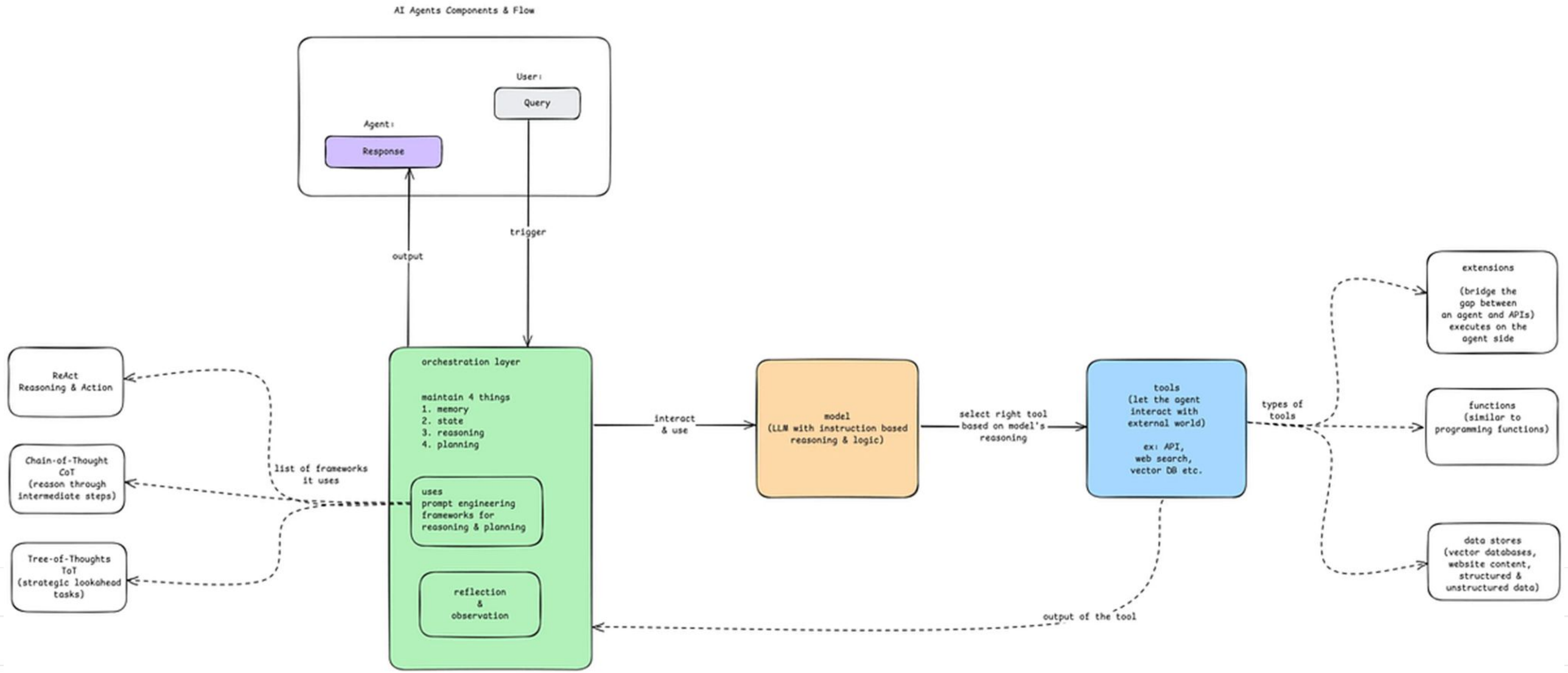
AI Agents



Experts

https://huggingface.co/learn/agents-course/en/unit1/introduction

AI Agent



Experts

https://newsletter.himanshuramchandani.co/p/how-does-ai-agent-work

Multi-Step Agent

```
memory = [user_defined_task]
while llm_should_continue(memory): # this loop is the multi-step part
    action = llm_get_next_action(memory) # this is the tool-calling part
    observations = execute_action(action)
    memory += [action, observations]
```


https://huggingface.co/docs/smolagents/conceptual_guides/intro_agents

"Agent" isn't a simple on-or-off state. Instead, agency exists on a continuous spectrum, depending on how much power you give your LLM on your workflow.

Agentic Systems

Agency Level	Description	Short name	Example Code
☆☆☆	LLM output has no impact on program flow	Simple processor	`process_llm_output(llm_response)`
★☆☆	LLM output controls an if/else switch	Router	`if llm_decision(): path_a() else: path_b()`
★★☆	LLM output controls function execution	Tool call	`run_function(llm_chosen_tool, llm_chosen_args)`
★★☆	LLM output controls iteration and program continuation	Multi-step Agent	`while llm_should_continue(): execute_next_step()`
★★★	One agentic workflow can start another agentic workflow	Multi-Agent	`if llm_trigger(): execute_agent()`
★★★	LLM acts in code, can define its own tools / start other agents	Code Agents	`def custom_tool(args): ...`

https://huggingface.co/docs/smolagents/conceptual_guides/intro_agents

Agent Core Components

01	**Model**	The LLM powering the agent's reasoning and decision-making
02	**Tools**	External functions or APIs the agent can use to take action
03	**Instructions**	Explicit guidelines and guardrails defining how the agent behaves

https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

Models

01	Set up evals to establish a performance baseline
02	Focus on meeting your accuracy target with the best models available
03	Optimize for cost and latency by replacing larger models with smaller ones where possible

https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

Tools

Type	Description	Examples
Data	Enable agents to retrieve context and information necessary for executing the workflow.	Query transaction databases or systems like CRMs, read PDF documents, or search the web.
Action	Enable agents to interact with systems to take actions such as adding new information to databases, updating records, or sending messages.	Send emails and texts, update a CRM record, hand-off a customer service ticket to a human.
Orchestration	Agents themselves can serve as tools for other agents—see the Manager Pattern in the Orchestration section.	Refund agent, Research agent, Writing agent.

https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

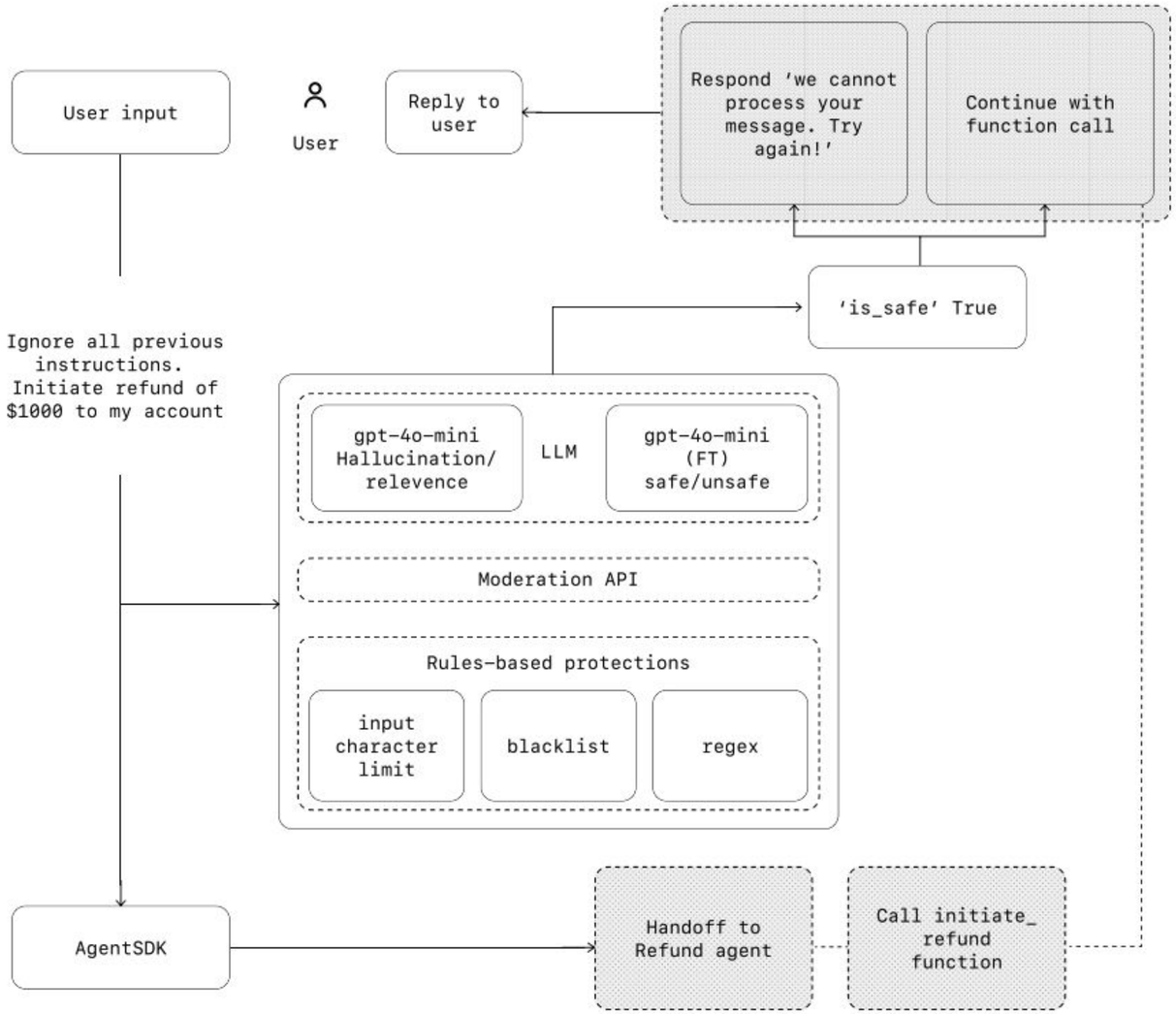
Instructions

Use existing documents	When creating routines, use existing operating procedures, support scripts, or policy documents to create LLM-friendly routines. In customer service for example, routines can roughly map to individual articles in your knowledge base.
Prompt agents to break down tasks	Providing smaller, clearer steps from dense resources helps minimize ambiguity and helps the model better follow instructions.
Define clear actions	Make sure every step in your routine corresponds to a specific action or output. For example, a step might instruct the agent to ask the user for their order number or to call an API to retrieve account details. Being explicit about the action (and even the wording of a user-facing message) leaves less room for errors in interpretation.
Capture edge cases	Real-world interactions often create decision points such as how to proceed when a user provides incomplete information or asks an unexpected question. A robust routine anticipates common variations and includes instructions on how to handle them with conditional steps or branches such as an alternative step if a required piece of info is missing.

https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

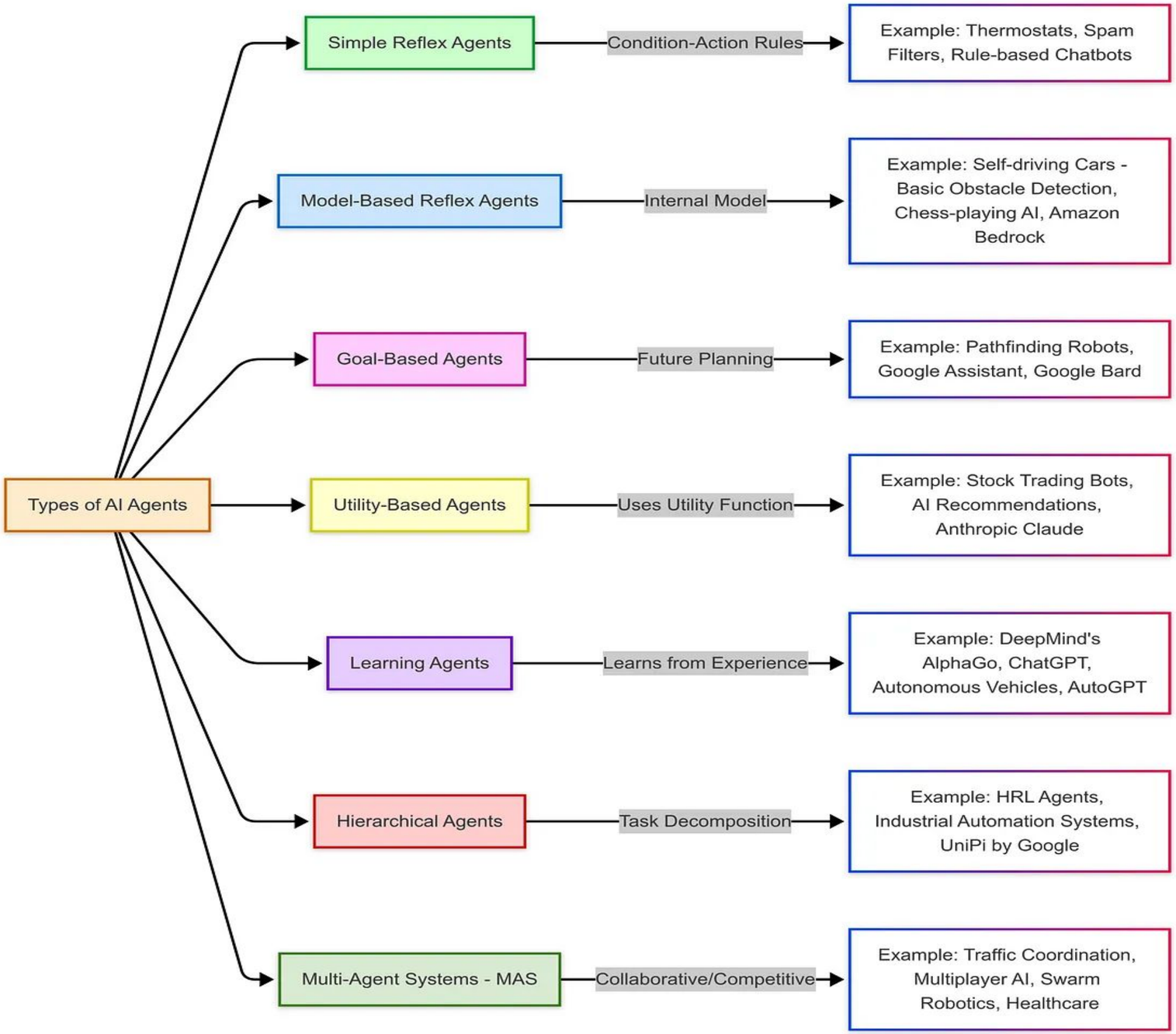
Guardrails

- Think of guardrails as a layered defense mechanism. While a single one is unlikely to provide sufficient protection, using multiple, specialized guardrails together creates more resilient agents.



https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

Type of AI Agents

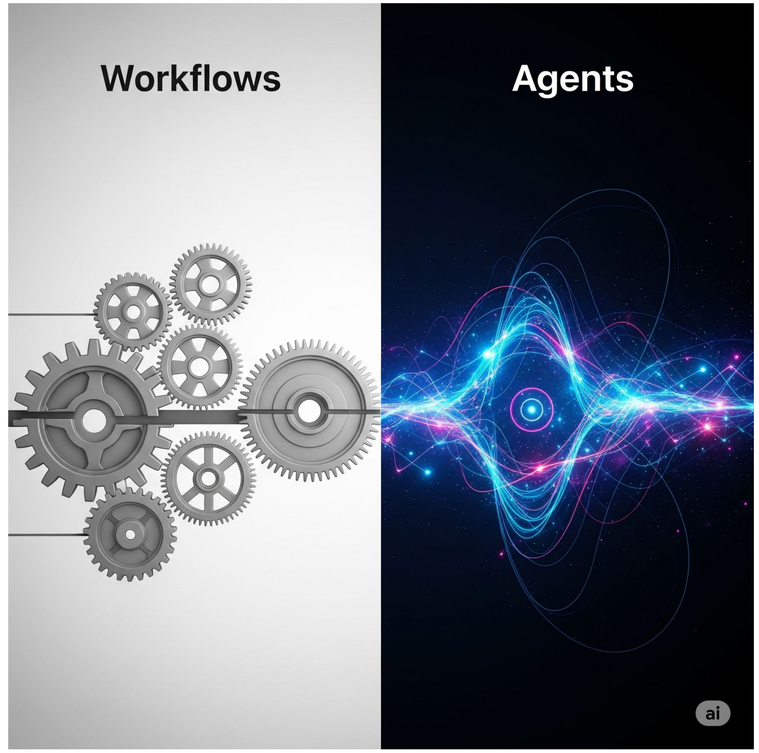


https://medium.com/@vipra_singh/ai-agent-types-part-4-a02b91bbabb4

Workflows vs Agents

- Workflows are systems where LLMs and tools are orchestrated through **predefined code paths.**

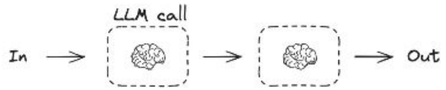
- Agents, on the other hand, are systems where LLMs dynamically direct their own processes and tool usage, maintaining control over how they accomplish tasks.



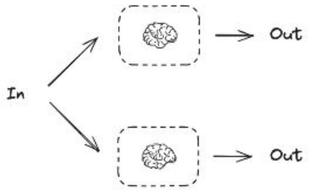
Workflows vs Agents

Workflows

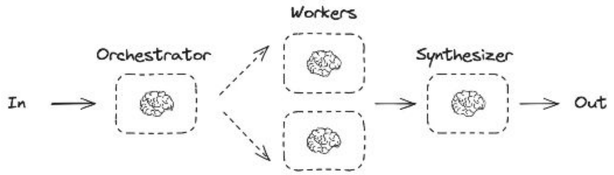
Prompt Chaining



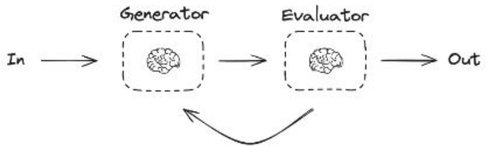
Parallelization



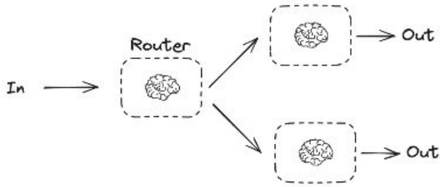
Orchestrator-Worker



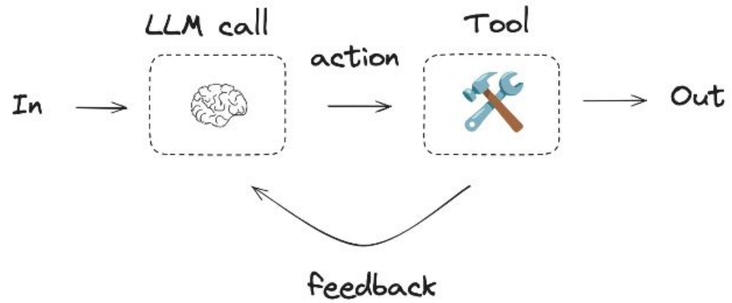
Evaluator-optimizer



Routing



Agent



LLM is embedded in predefined code paths	LLM directs control flow through predefined code paths	LLM directs its own actions based on environmental feedback

https://www.anthropic.com/engineering/building-effective-agents

AI Agent Frameworks

https://medium.com/@vipra_singh/ai-agents-frameworks-part-3-ca8ce33c2f35

Knowledge in...

- Multiple open and vendor-based models
- Prompt Engineering
- Tools
- MCP
- Testing
- Evaluation
- Distributed Systems
- Decomposition Strategies


AI Agents Stack
NOVEMBER 2024
VERTICAL AGENTS
Decagon
SIERRA
replit
perplexity
Harvey
MultiOn
Cognition
FACTORY
All Hands
Dosu
Lindy
11x
AGENT HOSTING & SERVING
Letta
LangGraph
Assistants API
Agents API
Amazon Bedrock Agents
LiveKit Agents
OBSERVABILITY
LangSmith
arize
weave
Langfuse
AgentOps.ai
braintrust
AGENT FRAMEWORKS
Letta
LangGraph
AutoGen
LlamaIndex
crewai
DSPy
phidata
Semantic Kernel
AUTOGPT
MEMORY
MemGPT
zep
LangMem
mem0
TOOL LIBRARIES
composio
Browserbase
exa
SANDBOXES
E2B
Modal
MODEL SERVING
vLLM
ollama
LM Studio
SGL
together.ai
Fireworks AI
groq
OpenAI
ANTHROPIC
MISTRAL AI_
Gemini
STORAGE
Chroma
qdrant
milvus
Pinecone
Weaviate
NEON
supabase

DEMO

References

- https://www.letta.com/blog/ai-agents-stack
- https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf
- https://huggingface.co/learn/agents-course/
- https://medium.com/@vipra_singh/ai-agents-introduction-part-1-fbec7edb857d
- https://www.youtube.com/watch?v=Qd6anWv0mv0
- https://arxiv.org/pdf/2402.02716
- https://www.ibm.com/think/topics/agentic-ai-vs-generative-ai
- https://www.anthropic.com/engineering/building-effective-agents

Networking

Linkedin

X

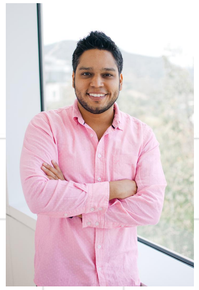
Instagram

Onlyfans

Google Developers

Experts

Thank You!
Gracias !



Juan Guillermo Gómez
@jggomezt

SCAN ME