

Outline

- Introduction (5 min)
- Why Go Beyond Pre-trained Models? (5 min)
- Enhancing LLM Performance (7 min)
- AI Agents: The Future (25 min)
- Evaluation, Limitations & Future (3 min)

Introduction to Gen AI

Introduction to Gen AI

Creating the New: What is Generative AI?

Introduction to Gen AI

Creating the New: What is Generative AI?

- AI that generates novel content (text, images, code, audio, etc.) vs. just classifying or predicting.
- Learns patterns and structures from vast amounts of data.
- Examples: Generating marketing copy, writing code, creating artwork, synthesizing speech.

Introduction to Gen AI

Creating the New: What is Generative AI?

- AI that generates novel content (text, images, code, audio, etc.) vs. just classifying or predicting.
- Learns patterns and structures from vast amounts of data.
- Examples: Generating marketing copy, writing code, creating artwork, synthesizing speech.

Discriminative

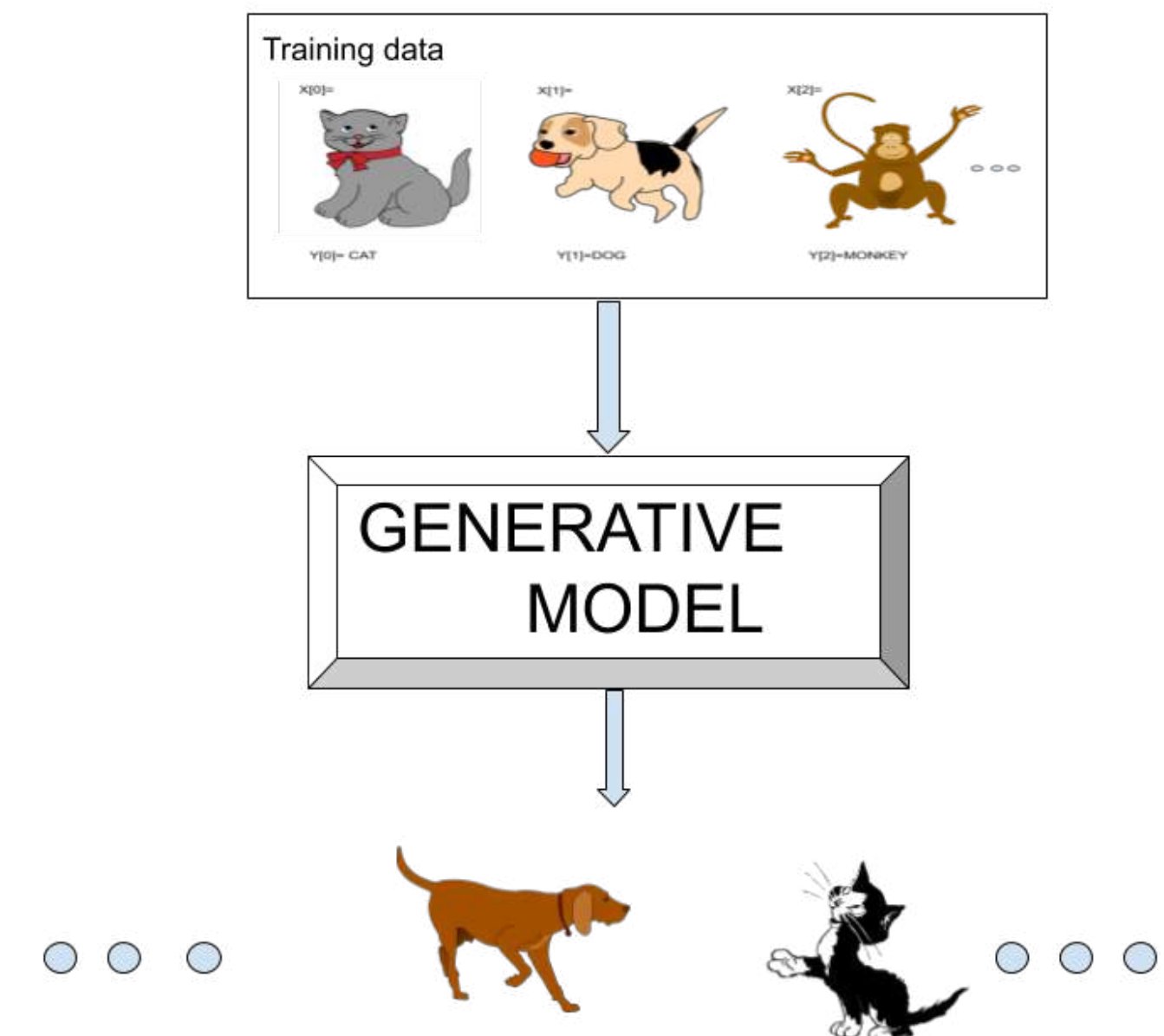


Introduction to Gen AI

Creating the New: What is Generative AI?

- AI that generates novel content (text, images, code, audio, etc.) vs. just classifying or predicting.
- Learns patterns and structures from vast amounts of data.
- Examples: Generating marketing copy, writing code, creating artwork, synthesizing speech.

Discriminative

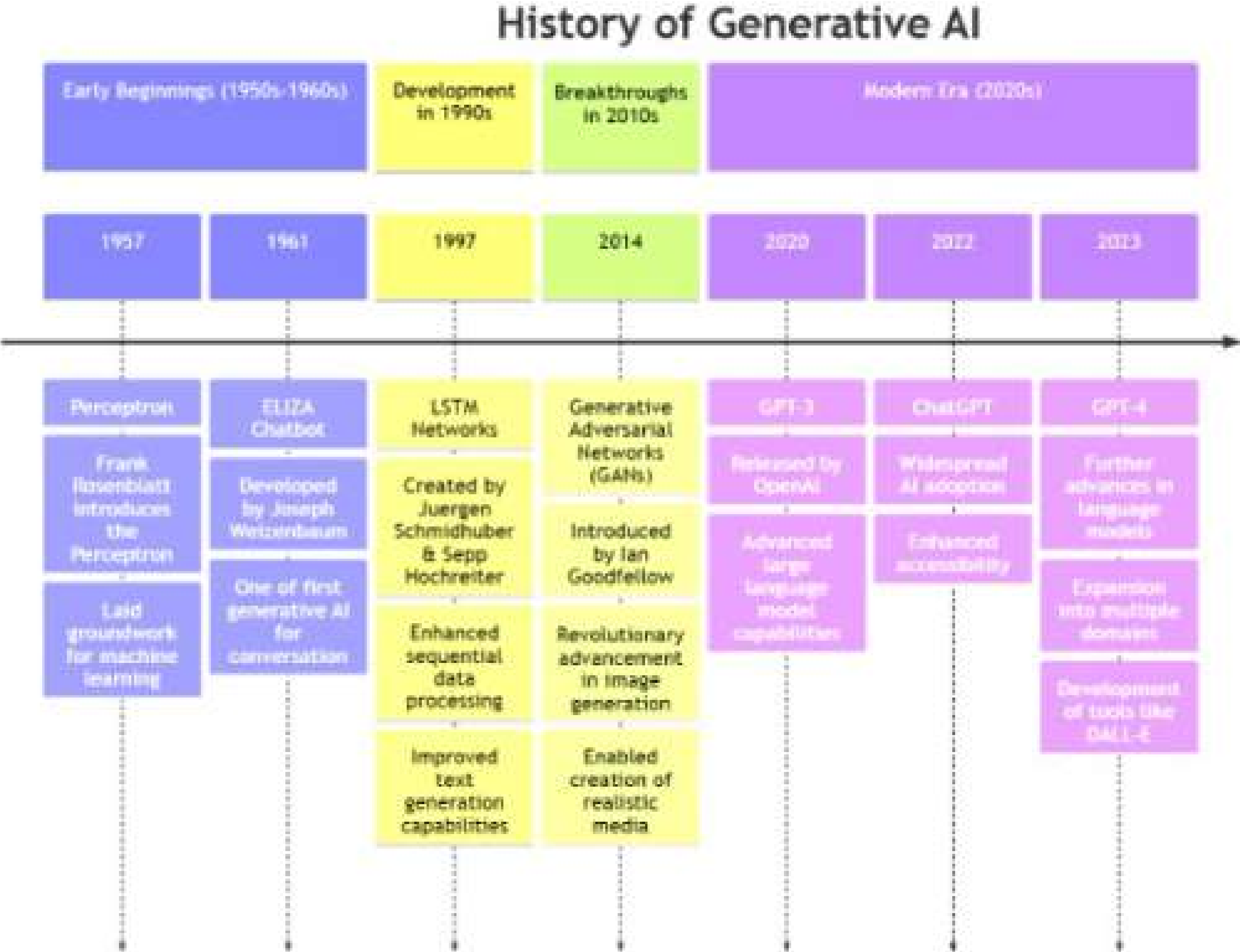


Introduction to Gen AI

Some History

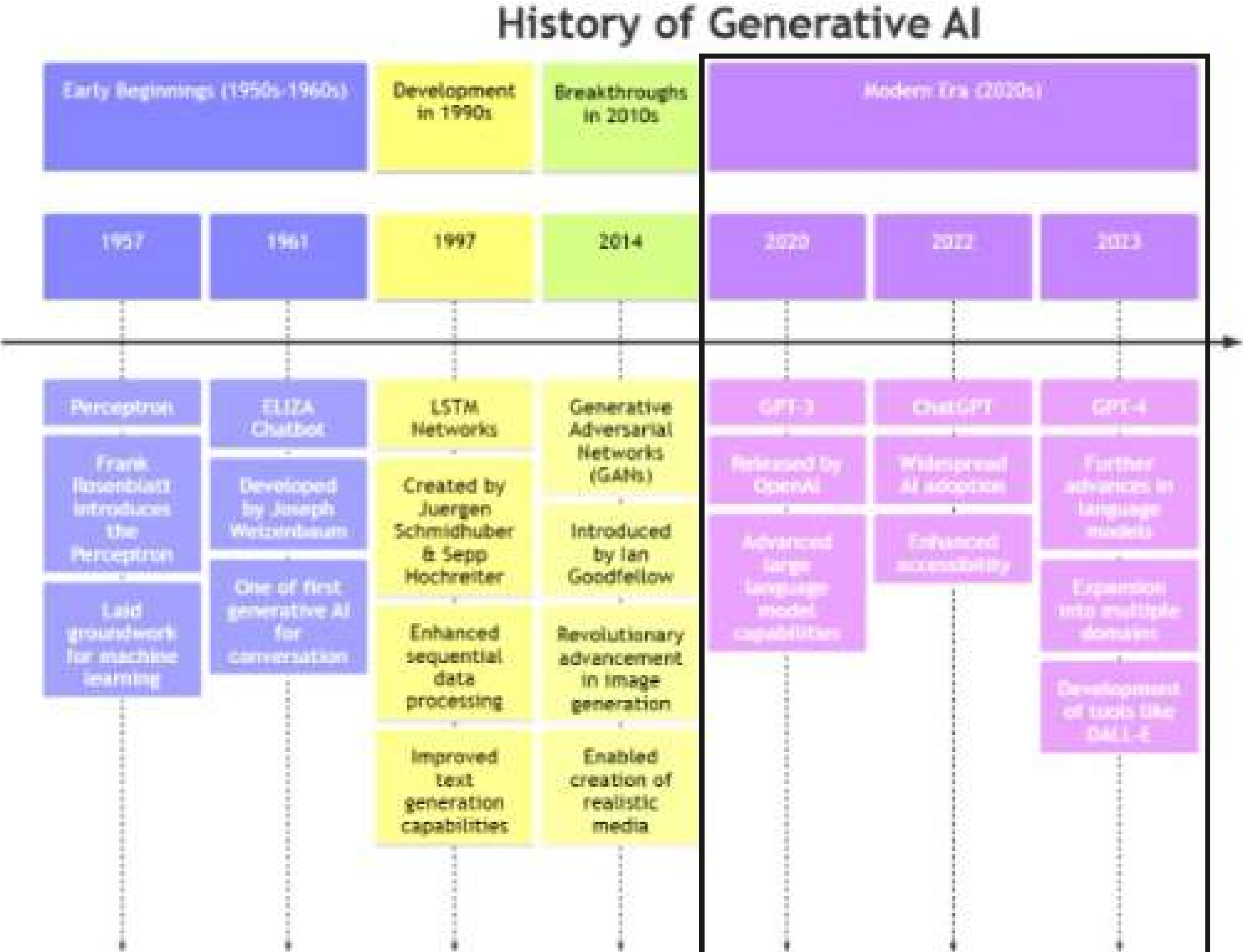
Introduction to Gen AI

Some History



Introduction to Gen AI

Some History



Introduction to Gen AI

Some History

- BERT (Encoder-only): Great for understanding context.
- GPT Series (Decoder-only): Excellent for text generation, follow instructions.
- T5 / BART (Encoder-Decoder): Versatile for tasks requiring understanding and generation (summarization, translation).

Introduction to Gen AI

Some History

- BERT (Encoder-only): Great for understanding context.
- GPT Series (Decoder-only): Excellent for text generation, follow instructions.
- T5 / BART (Encoder-Decoder): Versatile for tasks requiring understanding and generation (summarization, translation).



Introduction to Gen AI

Some History

- BERT (Encoder-only): Great for understanding context.
- GPT Series (Decoder-only): Excellent for text generation, follow instructions.
- T5 / BART (Encoder-Decoder): Versatile for tasks requiring understanding and generation (summarization, translation).



Introduction to Gen AI

Some History

- BERT (Encoder-only): Great for understanding context.
- GPT Series (Decoder-only): Excellent for text generation, follow instructions.
- T5 / BART (Encoder-Decoder): Versatile for tasks requiring understanding and generation (summarization, translation).



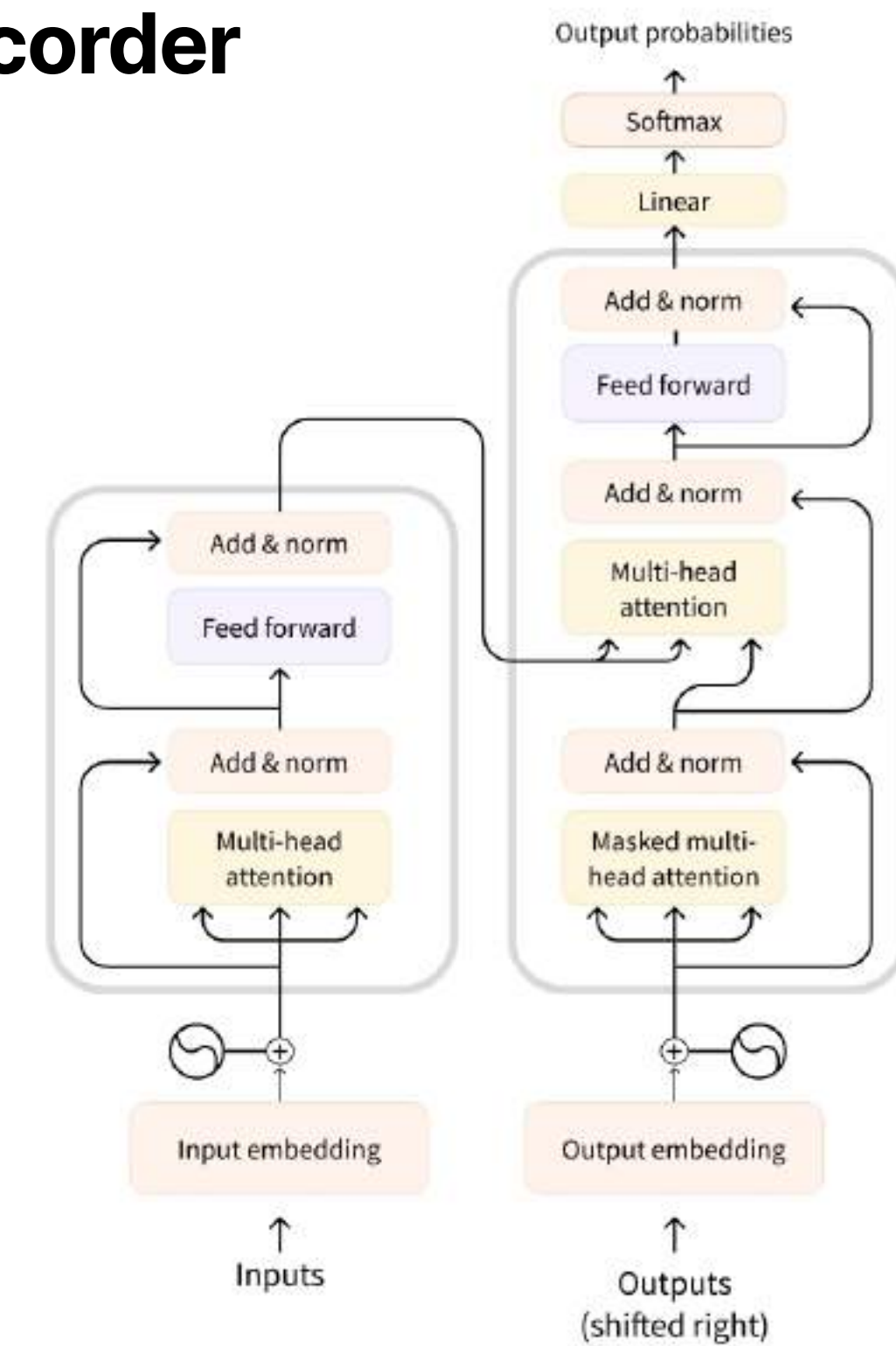
Introduction to Gen AI

Some History

- BERT (Encoder-only): Great for understanding context.
- GPT Series (Decoder-only): Excellent for text generation, follow instructions.
- T5 / BART (Encoder-Decoder): Versatile for tasks requiring understanding and generation (summarization, translation).



Encoder



Decoder

Why Go Beyond Pre-trained Models?

Why Go Beyond Pre-trained Models?

Limitations of Simple Input/Output

Why Go Beyond Pre-trained Models?

Limitations of Simple Input/Output

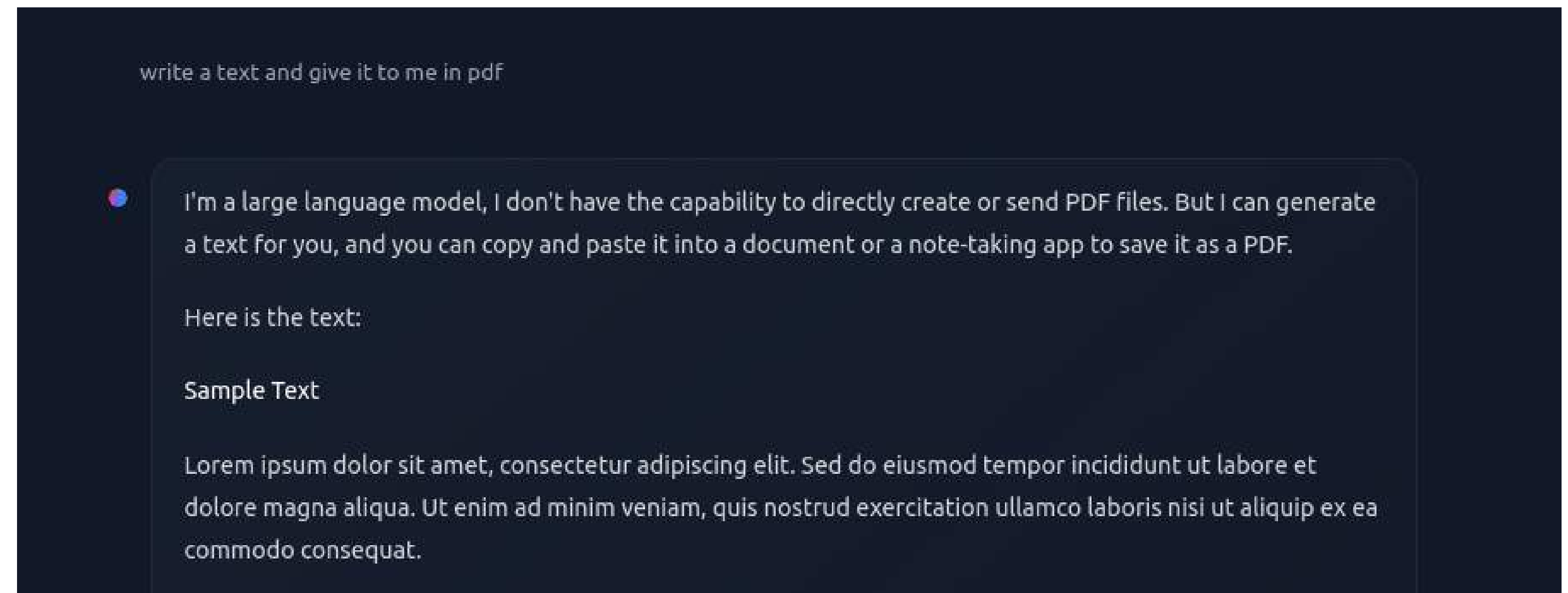
Knowledge Gaps



Why Go Beyond Pre-trained Models?

Limitations of Simple Input/Output

Limited Action



Why Go Beyond Pre-trained Models?

Limitations of Simple Input/Output

Reliability: Prone to hallucination without grounding or specific guidance.



How many 'm's are in the word 'Weather'?



There is one 'm' in the word 'Weather'.



Are you sure?



Apologies for the oversight. You're right; there are no 'm's in the word 'Weather'. Thank you for pointing it out.

Why Go Beyond Pre-trained Models?

Limitations of Simple Input/Output

Garbage in Garbage Out : Prompt engineering is important

IV IVAN SEVERINO

What is a prompt?

CHATGPT

A prompt is a message or instruction given to elicit a response or action. In computing, it often refers to a symbol or set of symbols displayed in a command-line interface, indicating that the system is ready to accept input. In a broader sense, a prompt can also refer to any stimulus that seeks to initiate a reaction or feedback, like a question in a survey or a cue in psychological experiments.

IV IVAN SEVERINO

Please explain what is a prompt, like if I'm 5 years old

CHATGPT

Alright! Imagine you have a toy robot. If you ask it to dance, it dances. The question "Can you dance?" is like giving the robot a little nudge or hint about what to do next. That nudge or hint is called a "prompt." So, a prompt is like a little push or question you give to something (or someone) to get them to do or say something. 😊

Why we care about LLM performance ?

Why we care about LLM performance ?

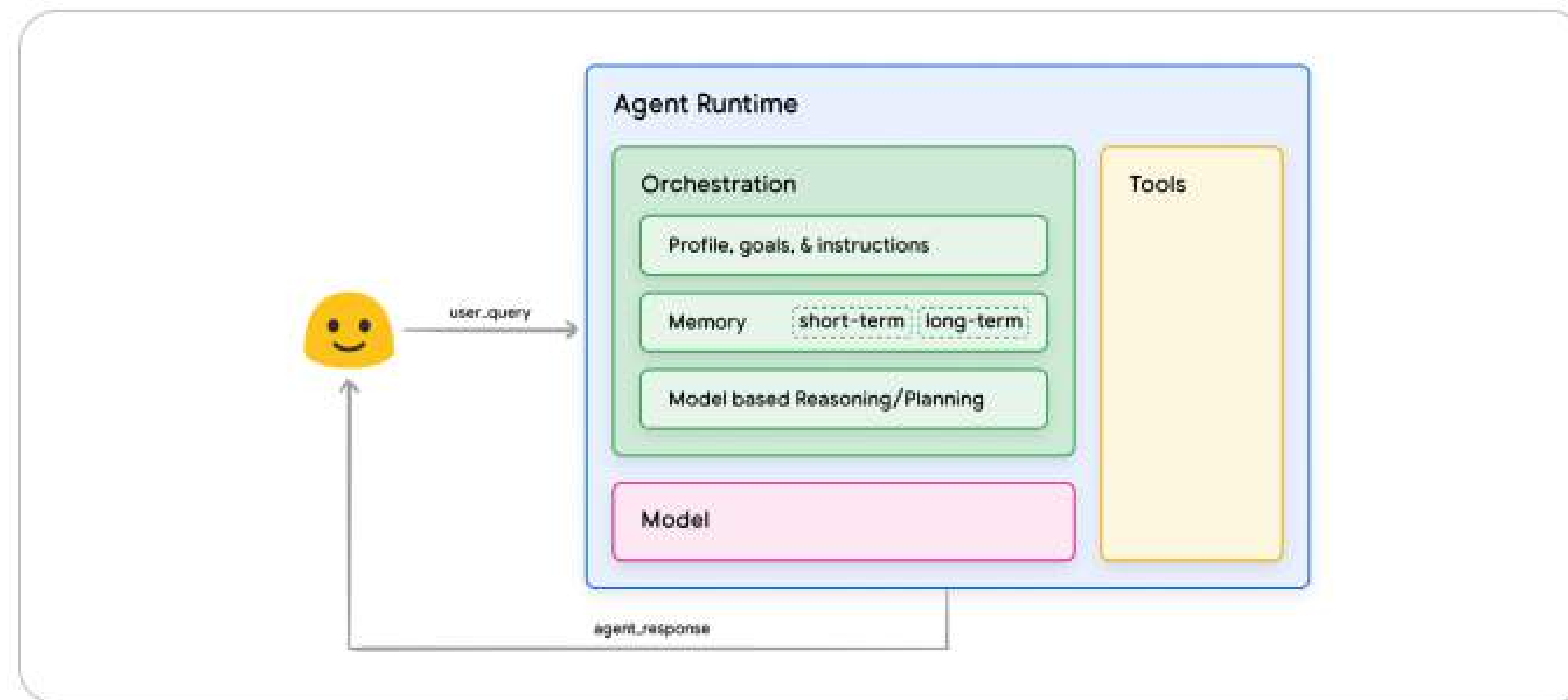


Figure 1. General agent architecture and components

Enhancing LLM Performance

Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

We need to make the underlying **Brain** (LLM)

Smarter :

- Prompt Engineering (PE): Guiding the LLM's reasoning.
- Retrieval-Augmented Generation (RAG): Providing external knowledge.
- Fine-Tuning (FT/PEFT): Adapting the LLM's core skills/style.

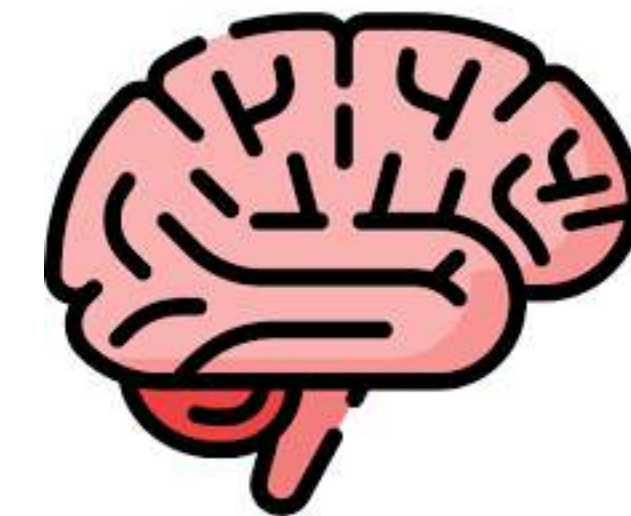
Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

We need to make the underlying **Brain** (LLM)

Smarter :

- Prompt Engineering (PE): Guiding the LLM's reasoning.
- Retrieval-Augmented Generation (RAG): Providing external knowledge.
- Fine-Tuning (FT/PEFT): Adapting the LLM's core skills/style.

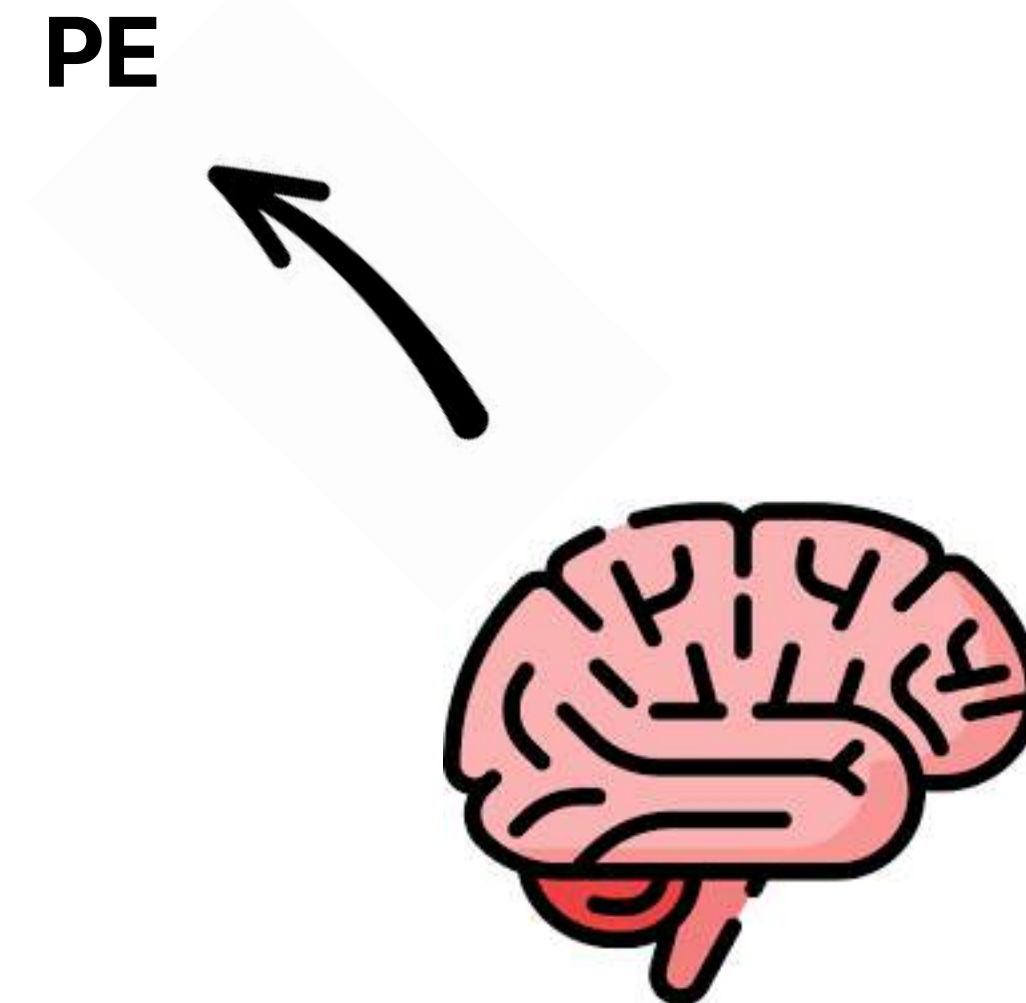


Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

We need to make the underlying **Brain** (LLM)
Smarter :

- Prompt Engineering (PE): Guiding the LLM's reasoning.
- Retrieval-Augmented Generation (RAG): Providing external knowledge.
- Fine-Tuning (FT/PEFT): Adapting the LLM's core skills/style.

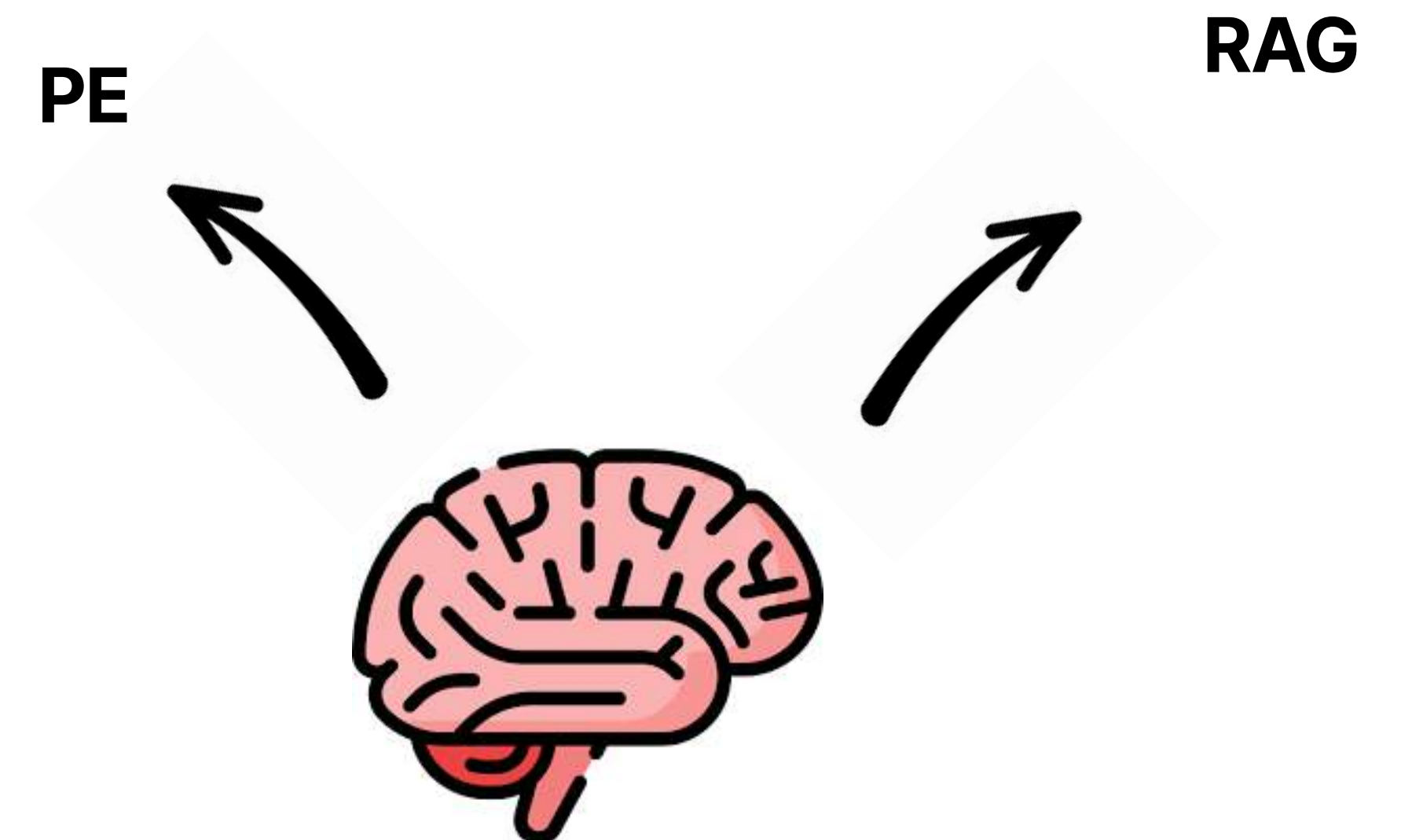


Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

We need to make the underlying **Brain** (LLM)
Smarter :

- Prompt Engineering (PE): Guiding the LLM's reasoning.
- Retrieval-Augmented Generation (RAG): Providing external knowledge.
- Fine-Tuning (FT/PEFT): Adapting the LLM's core skills/style.

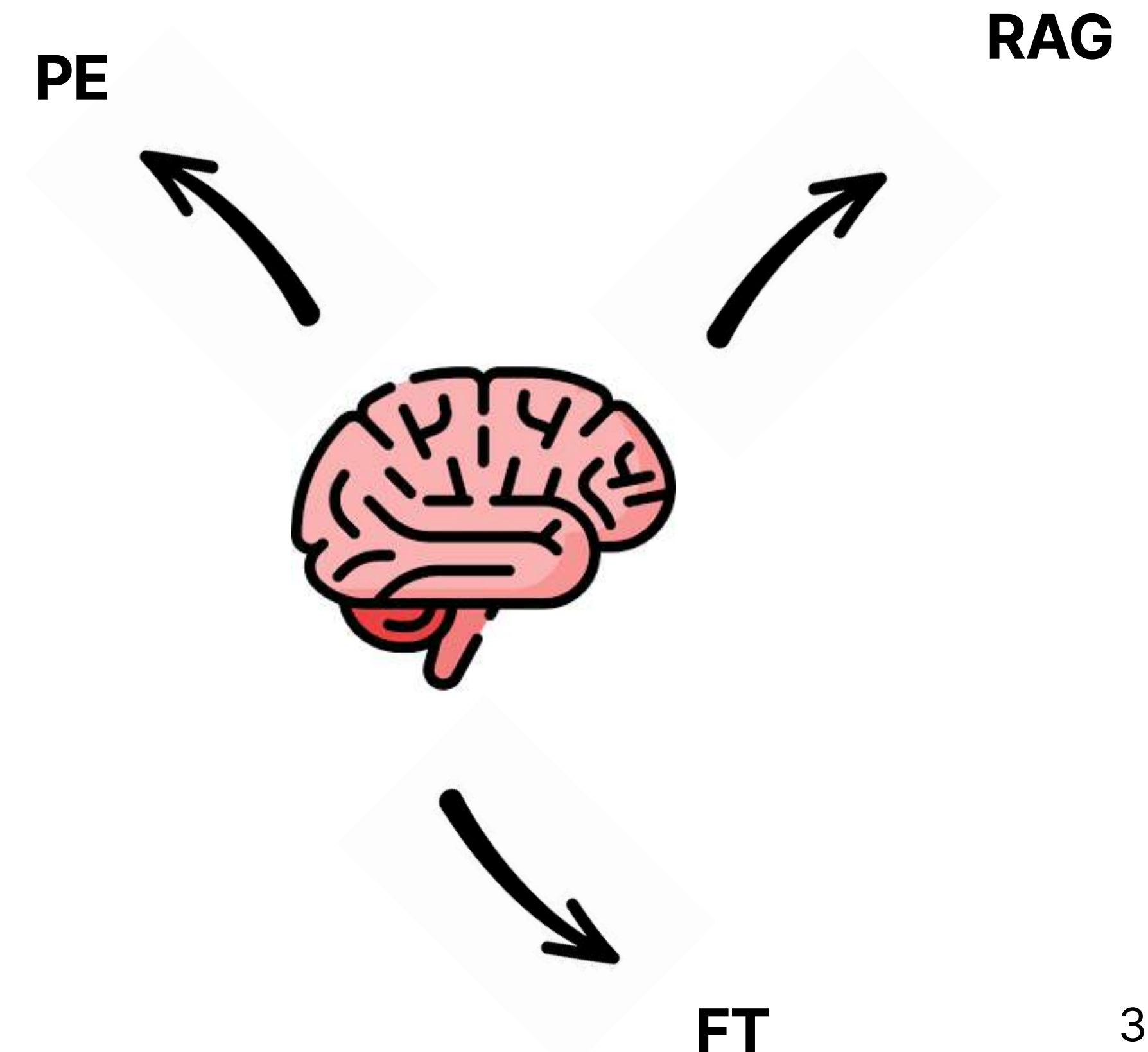


Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

We need to make the underlying **Brain** (LLM)
Smarter :

- Prompt Engineering (PE): Guiding the LLM's reasoning.
- Retrieval-Augmented Generation (RAG): Providing external knowledge.
- Fine-Tuning (FT/PEFT): Adapting the LLM's core skills/style.

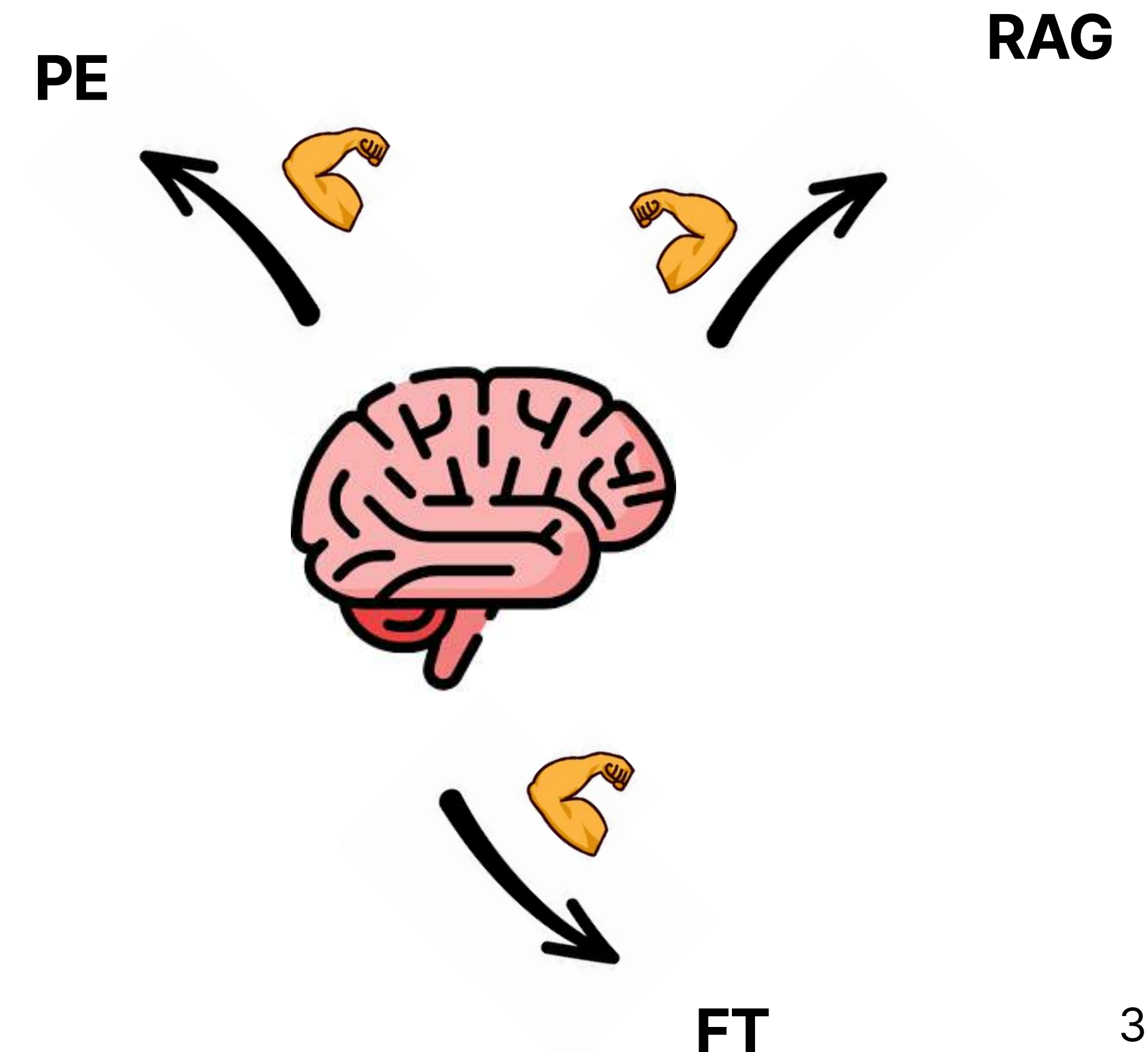


Enhancing LLM Performance

Enhancing the Core LLM: Agent Enablers

We need to make the underlying **Brain** (LLM) Smarter :

- Prompt Engineering (PE): Guiding the LLM's reasoning.
- Retrieval-Augmented Generation (RAG): Providing external knowledge.
- Fine-Tuning (FT/PEFT): Adapting the LLM's core skills/style.



Enhancing LLM Performance

Enhancing LLM Performance

Prompt Engineering (PE): Guiding the Agent's Thoughts

Enhancing LLM Performance

Prompt Engineering (PE): Guiding the Agent's Thoughts

Crafting effective prompts to control LLM behavior:

- **System Prompts:** Define the agent's overall goal, persona, and constraints.
- **Task Decomposition:** Prompts guide the LLM to break down complex goals (e.g., "Think step-by-step...").
- **Output Formatting:** Prompts tell the LLM how to format output to match specific specific tool format.

Enhancing LLM Performance

Prompt Engineering (PE): Guiding the Agent's Thoughts

Crafting effective prompts to control LLM behavior:

- **System Prompts:** Define the agent's overall goal, persona, and constraints.
- **Task Decomposition:** Prompts guide the LLM to break down complex goals (e.g., "Think step-by-step...").
- **Output Formatting:** Prompts tell the LLM how to format output to match specific specific tool format.

Goal	Act as travel guide and provide 3 travel suggestions		
Model	gemini-pro		
Temperature	1	Token Limit	1024
Top-K	40	Top-P	0.8
Prompt	I want you to act as a travel guide. I will write to you about my location and you will suggest 3 places to visit near me. In some cases, I will also give you the type of places I will visit. My suggestion: "I am in Amsterdam and I want to visit only museums." Travel Suggestions:		
Output	1. Rijksmuseum: World-renowned museum, known for Dutch masterpieces like Rembrandt's "The Night Watch." Don't miss the incredible collection of paintings and artifacts showcasing Dutch art and history. 2. Van Gogh Museum: Immerse yourself in the captivating works of Vincent van Gogh. The museum houses the largest collection of his paintings and drawings, including "The Starry Night" and "Sunflowers." 3. Stedelijk Museum Amsterdam: Discover modern and contemporary art from around the world. The museum's collection includes works by Picasso, Kandinsky, and Rietveld, housed in a striking modern building.		

Table 5. An example of role prompting

Enhancing LLM Performance

Prompt Engineering (PE): Guiding the Agent's Thoughts

Crafting effective prompts to control LLM behavior:

- **System Prompts:** Define the agent's overall goal, persona, and constraints.
- **Task Decomposition:** Prompts guide the LLM to break down complex goals (e.g., "Think step-by-step...").
- **Output Formatting:** Prompts tell the LLM how to format output to match specific specific tool format.

Prompt	Q: When my brother was 2 years old, I was double his age. Now I am 40 years old. How old is my brother? Let's think step by step. A: When my brother was 2 years, I was $2 * 2 = 4$ years old. That's an age difference of 2 years and I am older. Now I am 40 years old, so my brother is $40 - 2 = 38$ years old. The answer is 38. Q: When I was 3 years old, my partner was 3 times my age. Now, I am 20 years old. How old is my partner? Let's think step by step. A:
Output	When I was 3 years old, my partner was $3 * 3 = 9$ years old. That's an age difference of 6 years and my partner is older. Now I am 20 years old, so my partner is $20 + 6 = 26$ years old. The answer is 26.

Table 13. An example of chain of thought prompting with a single-shot

Enhancing LLM Performance

Enhancing LLM Performance

RAG: Giving the Agent External Knowledge

Enhancing LLM Performance

RAG: Giving the Agent External Knowledge

Retrieving relevant data from external sources (docs, DBs) and adding it to the prompt at inference time.

- **How Agents Use It:**
 - **As a Tool:** An agent can decide to call a "RAG Tool" to query a knowledge base (e.g., "Search internal docs for Project X details").
 - **Grounding:** Reduces hallucinations by providing factual context for the agent's responses or decisions.

The idea Vector **(in memory)** DBs store data embeddings for fast semantic search.

Enhancing LLM Performance

RAG: Giving the Agent External Knowledge

Retrieving relevant data from external sources (docs, DBs) and adding it to the prompt at inference time.

- **How Agents Use It:**

- **As a Tool:** An agent can decide to call a "RAG Tool" to query a knowledge base (e.g., "Search internal docs for Project X details").
- **Grounding:** Reduces hallucinations by providing factual context for the agent's responses or decisions.

The idea Vector **(in memory)** DBs store data embeddings for fast semantic search.

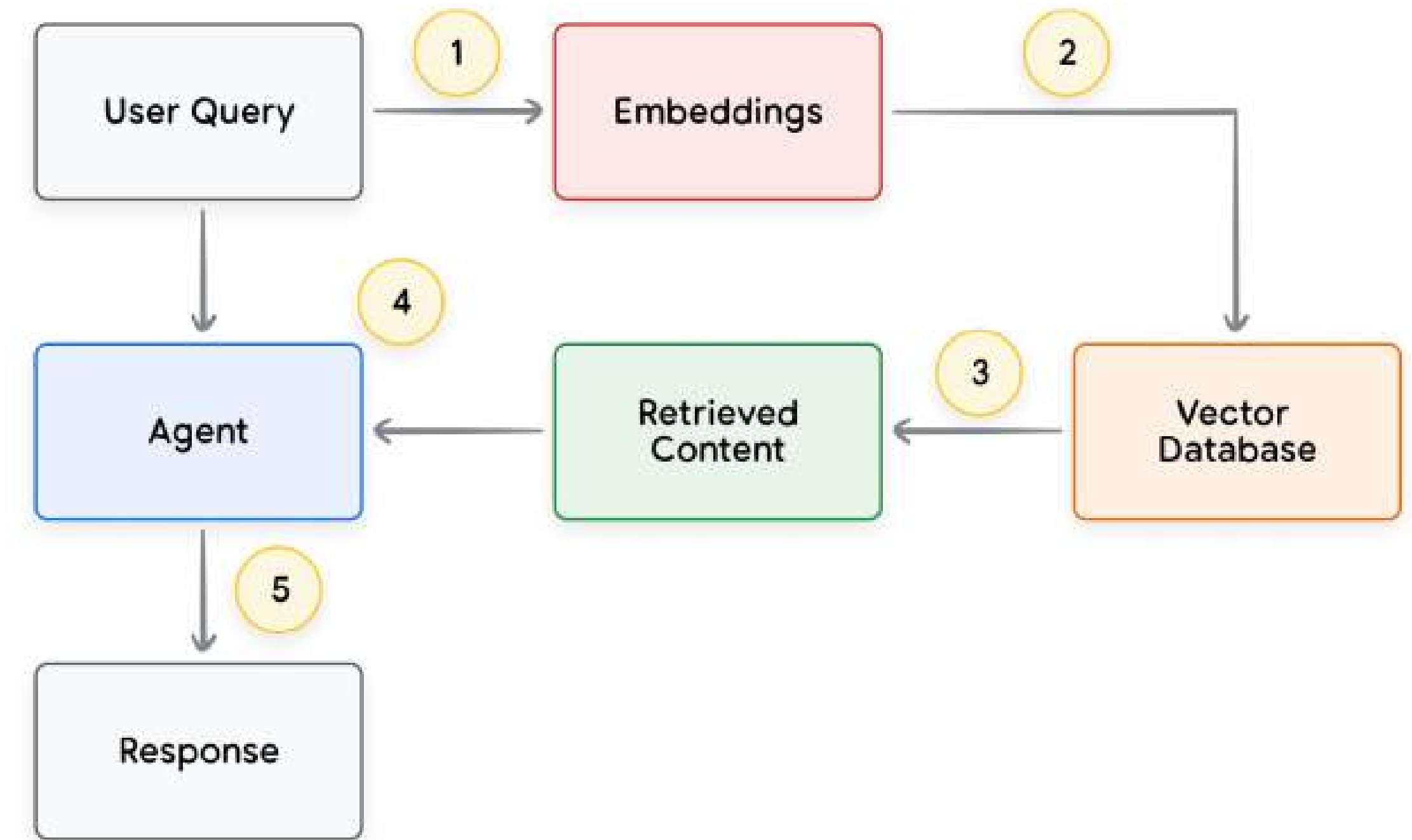


Figure 13. The lifecycle of a user request and agent response in a RAG based application

Enhancing LLM Performance

Enhancing LLM Performance

Fine-Tuning - Parameter-Efficient Fine-Tuning (FT/PEFT): Customizing the Agent's Brain

Enhancing LLM Performance

Fine-Tuning - Parameter-Efficient Fine-Tuning (FT/PEFT): Customizing the Agent's Brain

Adapting a pre-trained LLM on specific data to improve its performance on particular tasks or adopt a certain style/persona.

- **Why for Agents?**
 - Improve reliability on specific, recurring tasks the agent performs.
 - Embed deep domain knowledge or specific formatting rules directly.
 - Make the agent's core reasoning more efficient/accurate for its domain.

PEFT (LoRA/QLoRA): Makes this much more efficient (trains only small adapters).

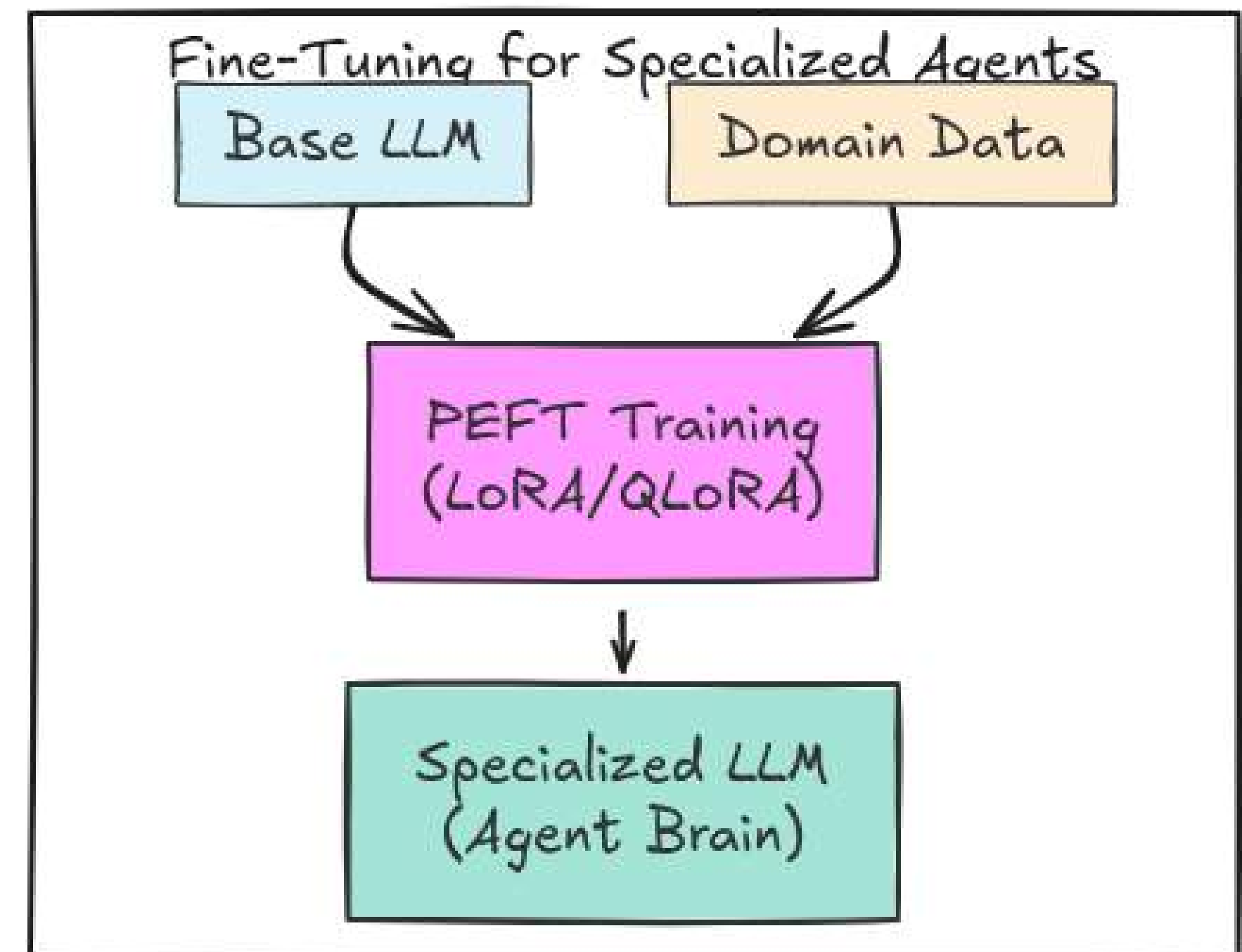
Enhancing LLM Performance

Fine-Tuning - Parameter-Efficient Fine-Tuning (FT/PEFT): Customizing the Agent's Brain

Adapting a pre-trained LLM on specific data to improve its performance on particular tasks or adopt a certain style/persona.

- **Why for Agents?**
 - Improve reliability on specific, recurring tasks the agent performs.
 - Embed deep domain knowledge or specific formatting rules directly.
 - Make the agent's core reasoning more efficient/accurate for its domain.

PEFT (LoRA/QLoRA): Makes this much more efficient (trains only small adapters).



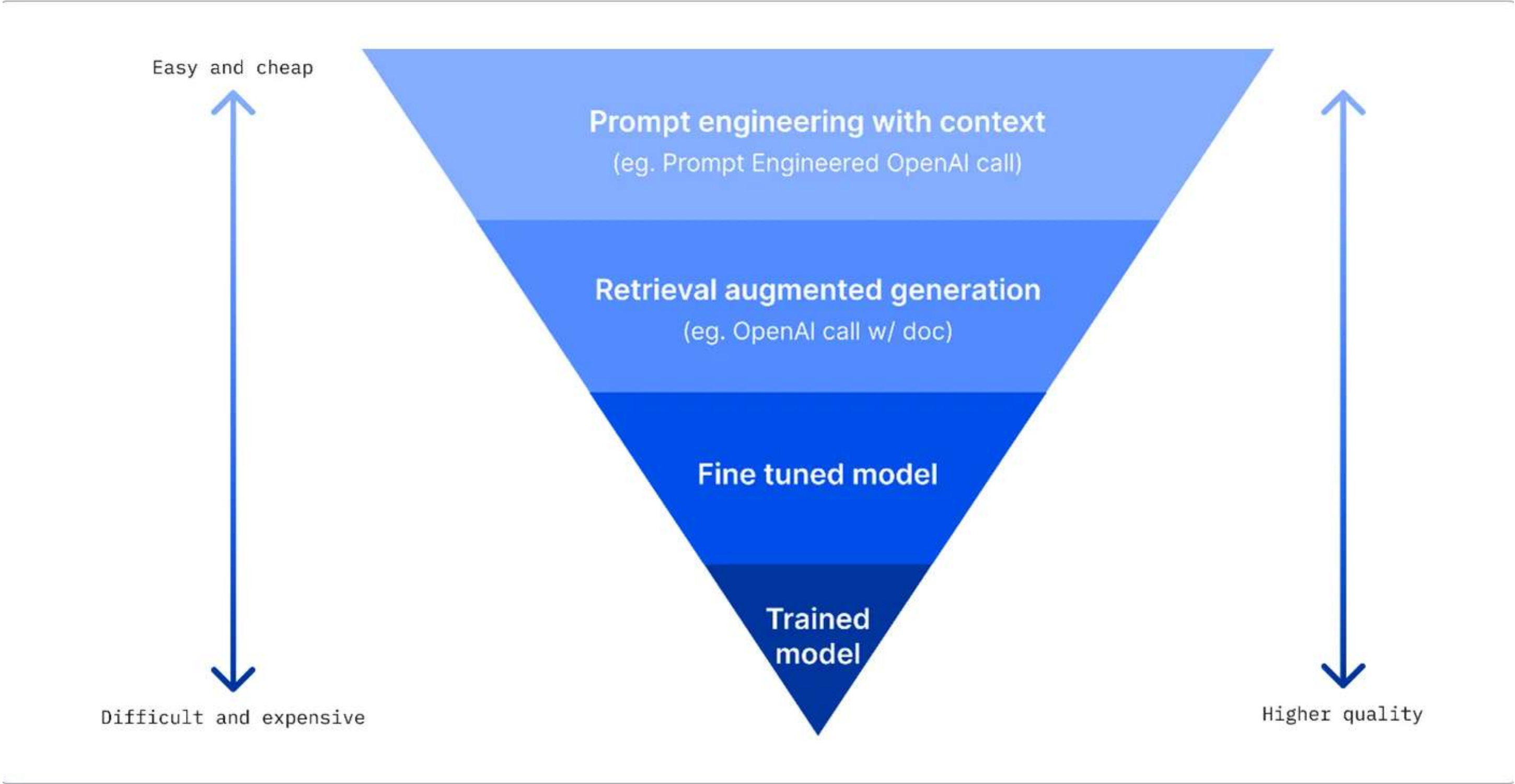
Enhancing LLM Performance

Enhancing LLM Performance

In business :

Enhancing LLM Performance

In business :



AI Agents: The Futrure

AI Agents: The Future

The Agentic Leap: From Response to Action

AI Agents: The Future

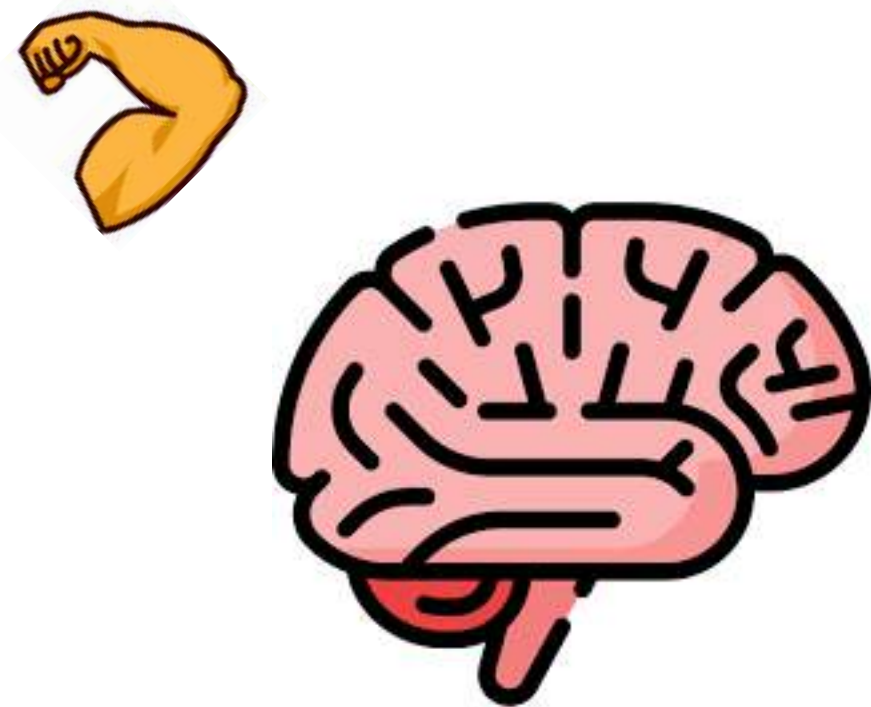
The Agentic Leap: From Response to Action



Big Brain LLM

AI Agents: The Future

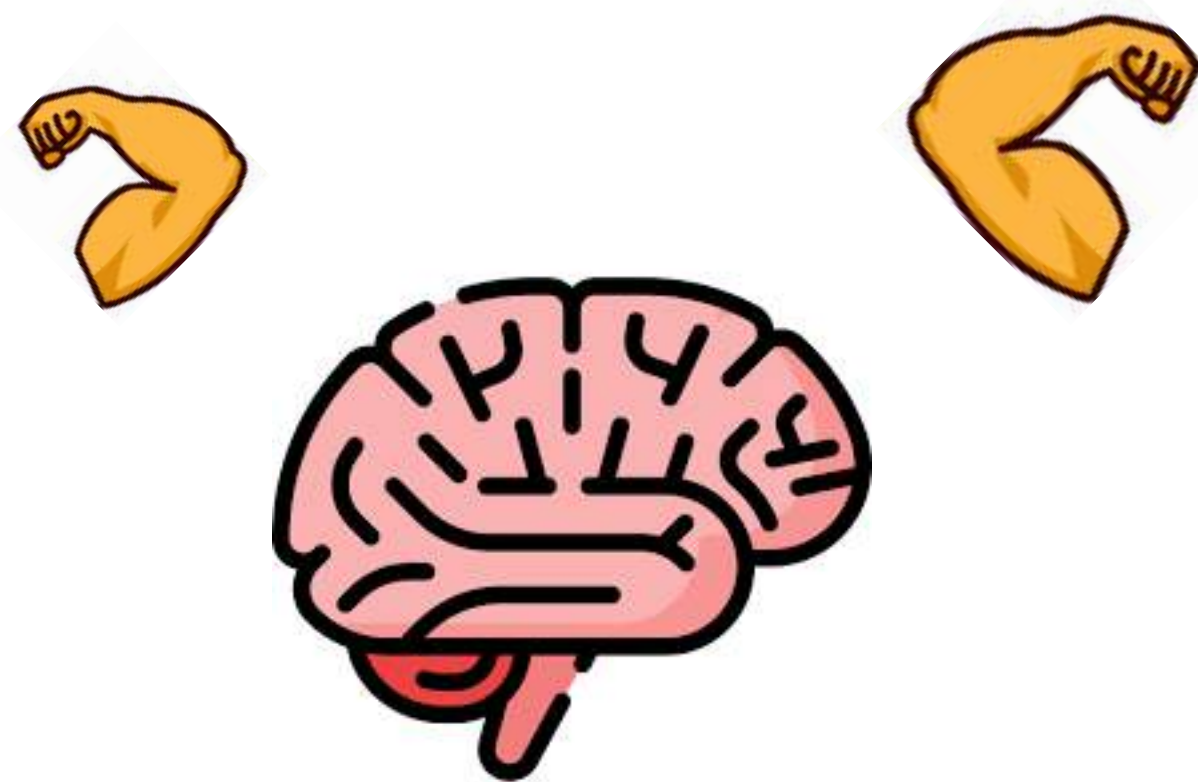
The Agentic Leap: From Response to Action



Big Brain LLM

AI Agents: The Future

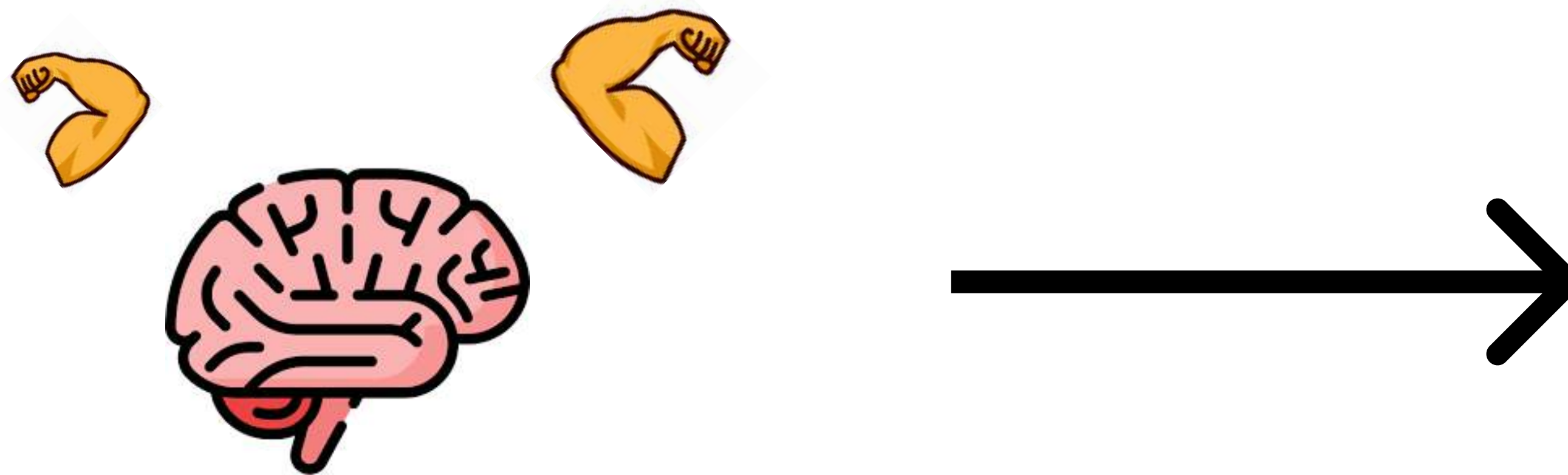
The Agentic Leap: From Response to Action



Big Brain LLM

AI Agents: The Future

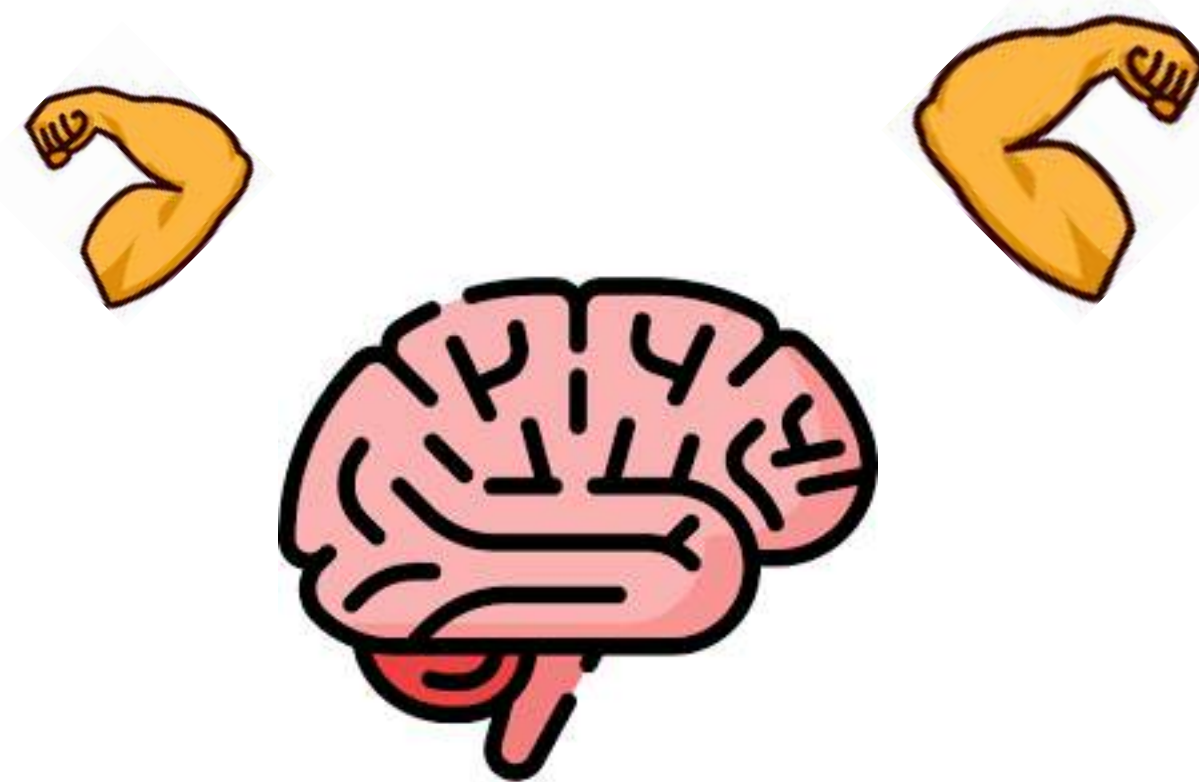
The Agentic Leap: From Response to Action



Big Brain LLM

AI Agents: The Future

The Agentic Leap: From Response to Action



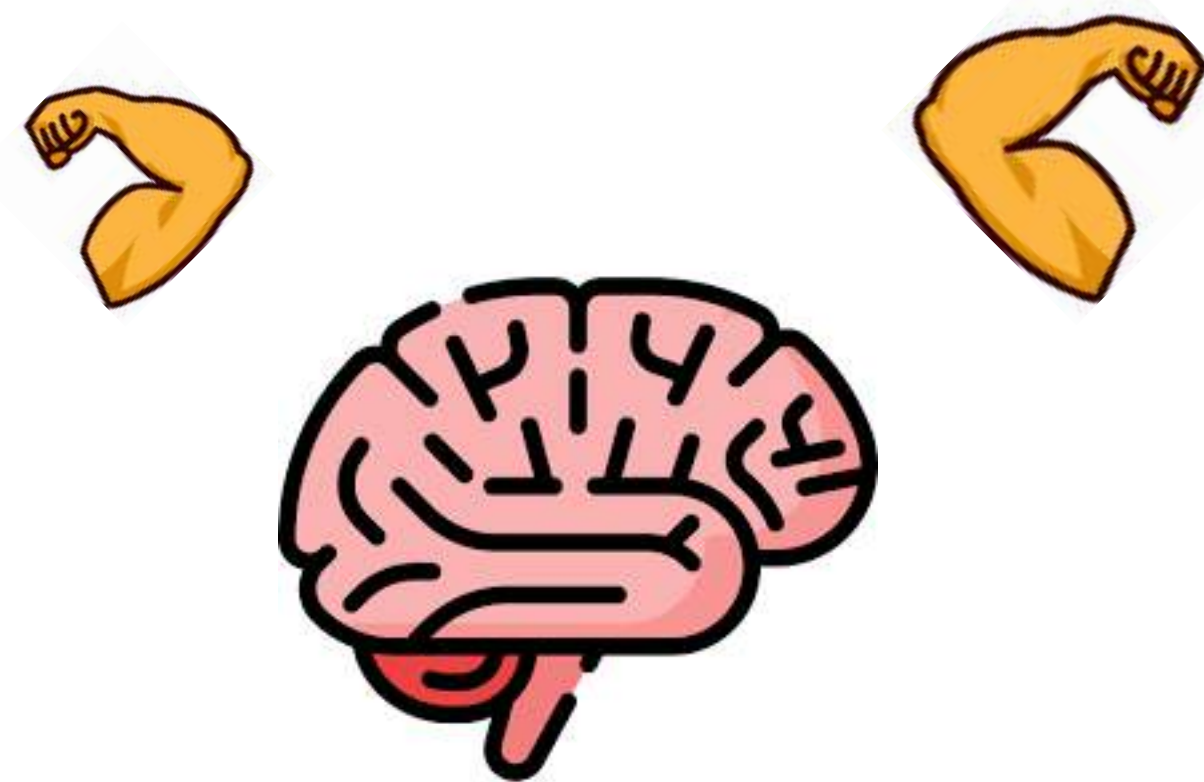
Big Brain LLM

**We need to use it
effectively !!!**



AI Agents: The Futrure

The Agentic Leap: From Response to Action



Big Brain LLM

We need to use it
effectively !!!



Orchestration Layer

AI Agents: The Future

The Agentic Leap: From Response to Action

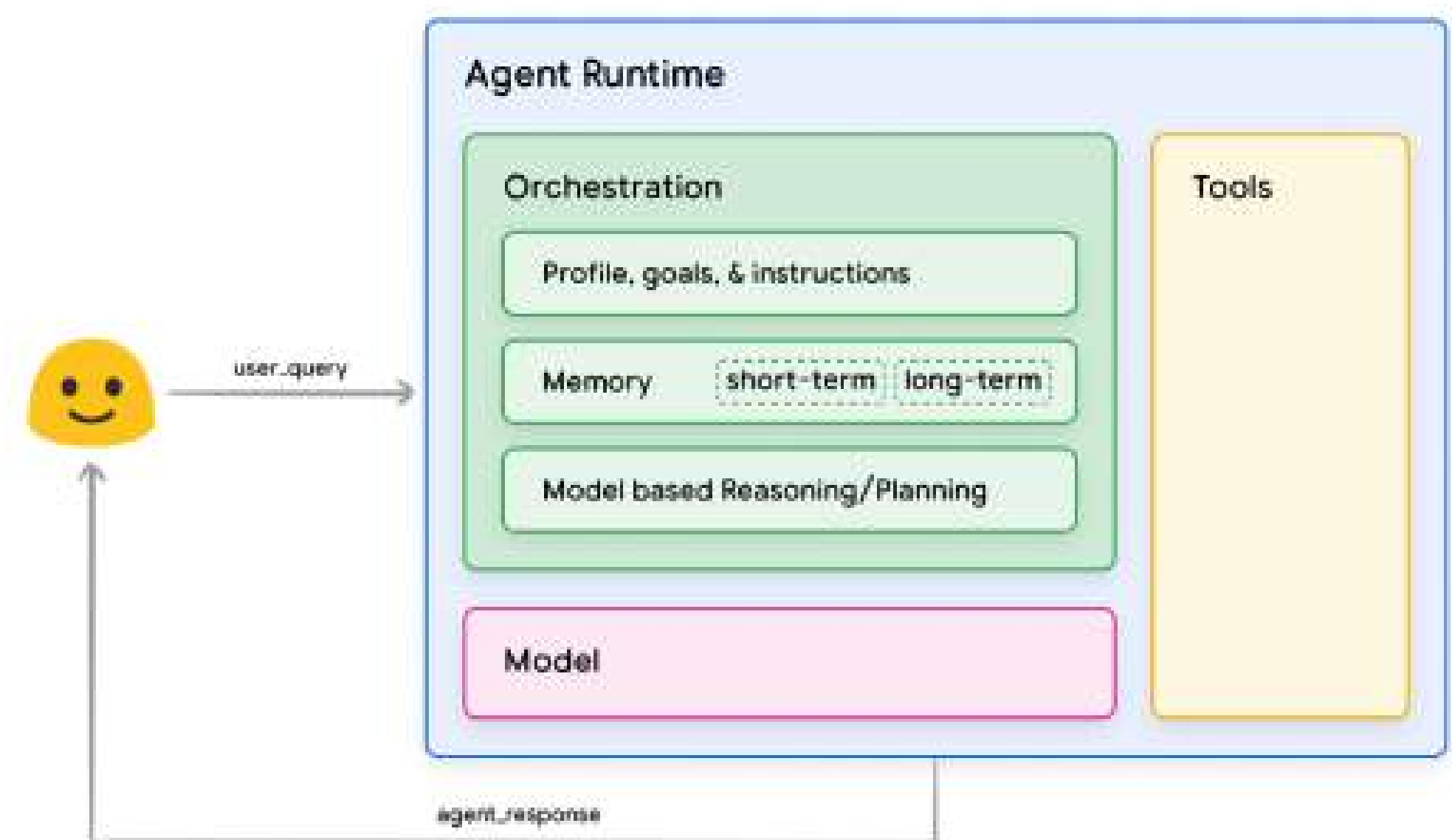
An Agent is a system that uses an AI model (**Big Smart Brain**) to interact with its environment in order to achieve a **user-defined objective**. It combines reasoning, planning, and the execution of actions (often via external tools) to fulfill tasks.



Orchestration Layer

AI Agents: The Future

The Agentic Leap: From Response to Action



Orchestration Layer

AI Agents: The Future

The Agentic Leap: From Response to Action

- The orchestration Layer is what makes the agent able to manage the interaction between LLM, Tools, and Memory.
- Executes the planning/action cycle.



Orchestration Layer

AI Agents: The Futrure

AI Agents: The Future

The Agentic Leap: From Response to Action

AI Agents: The Future

The Agentic Leap: From Response to Action

Imagine an agent is a human :

AI Agents: The Future

The Agentic Leap: From Response to Action

Imagine an agent is a human :



LLM is the **brain**

AI Agents: The Future

The Agentic Leap: From Response to Action

Imagine an agent is a human :



LLM is the **brain**



Tools Are the **Hands**

AI Agents: The Futrure

The Agentic Leap: From Response to Action

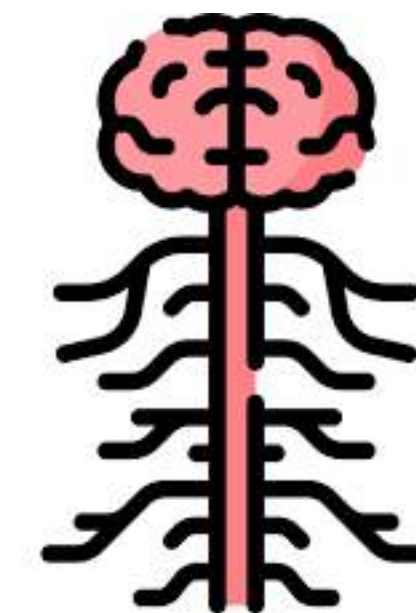
Imagine an agent is a human :



LLM is the **brain**

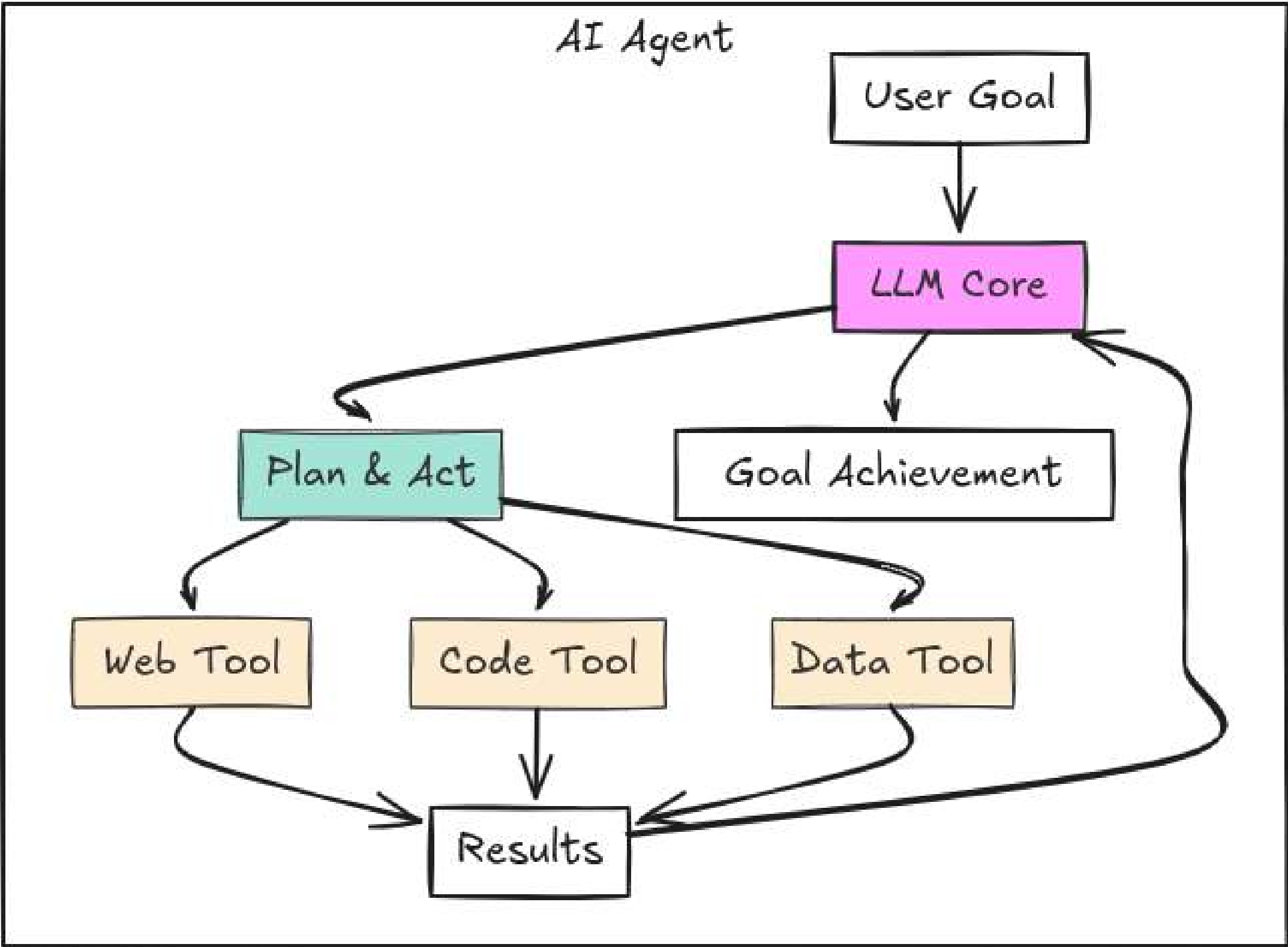


Tools Are the **Hands**

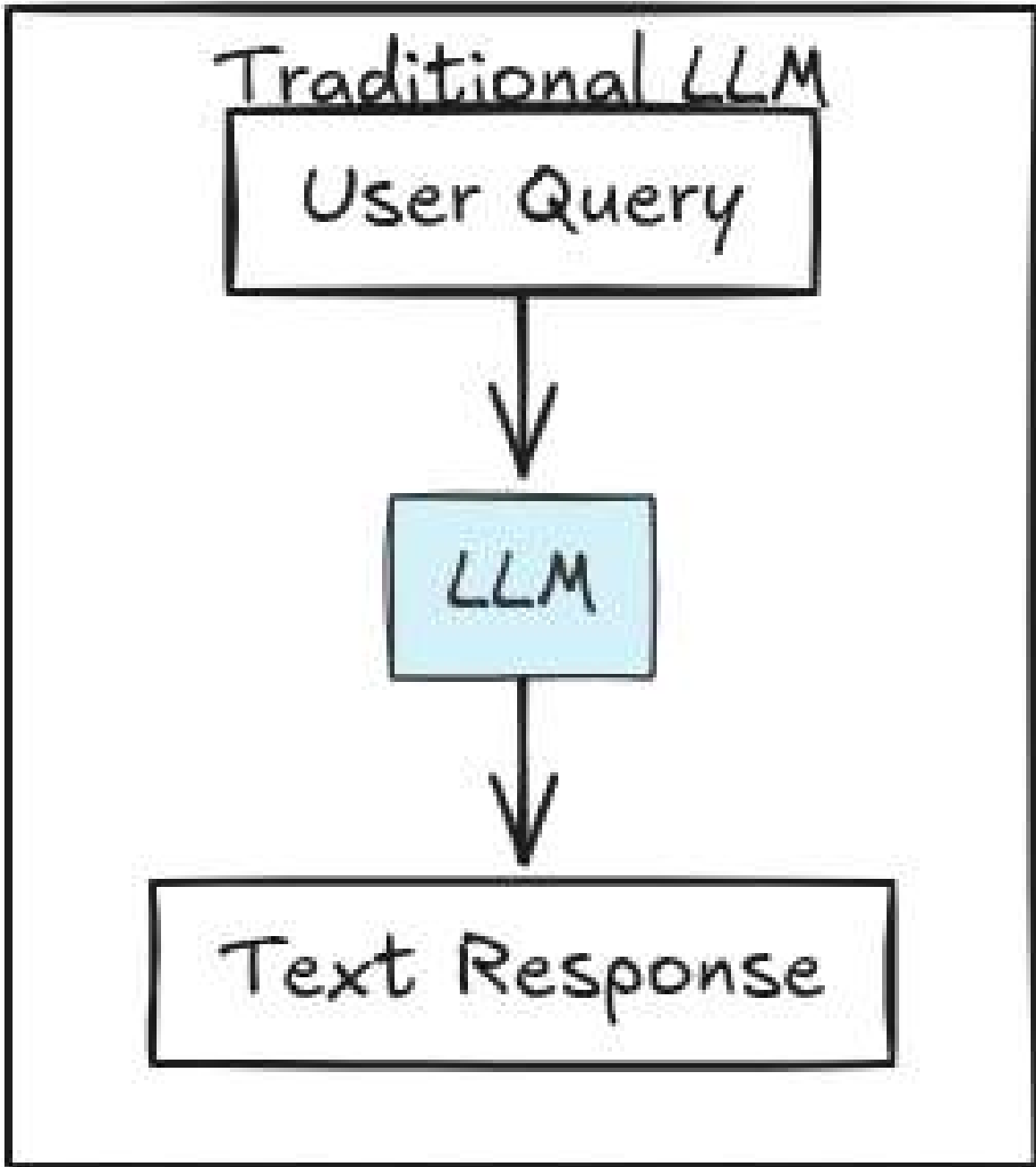


Orchestration Layer is
the **nervous system**

AI Agents: The Future

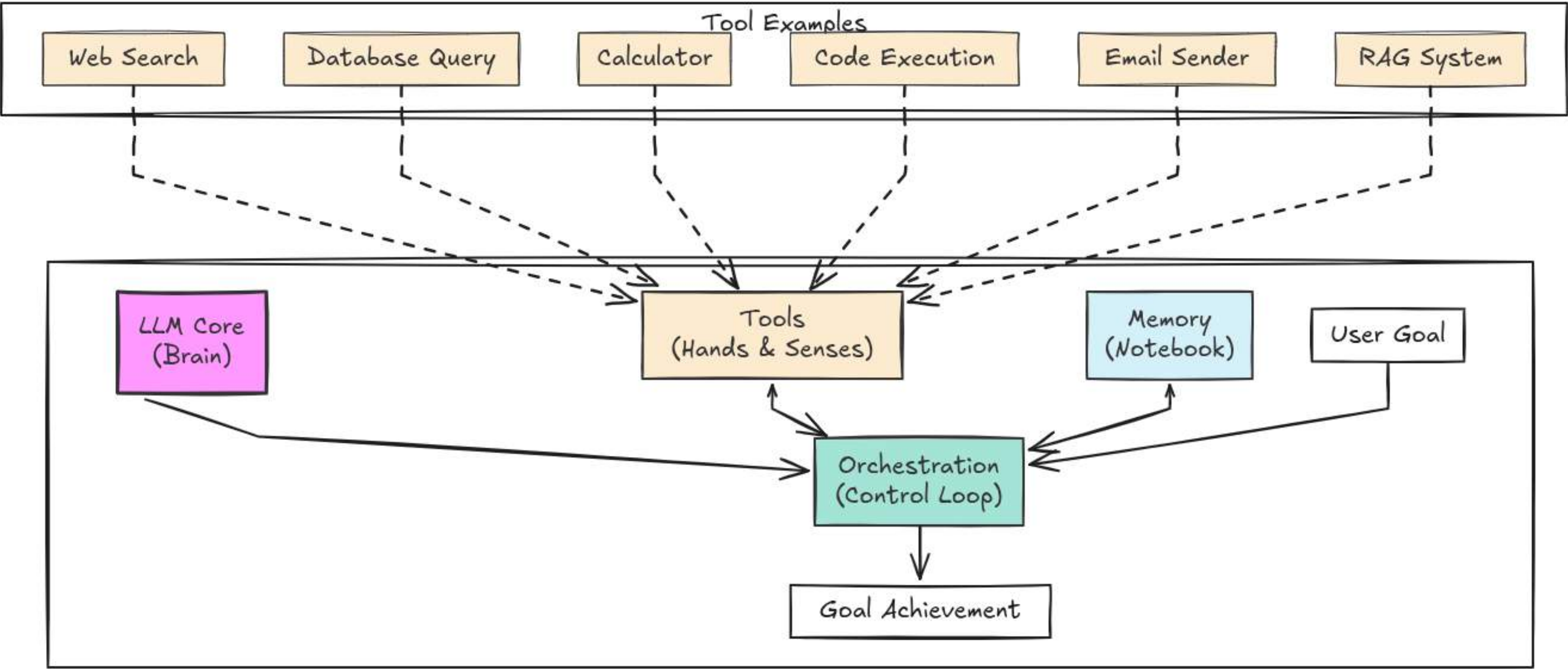


VS



AI Agents: The Futrure

Anatomy of an Agent: Core Components



AI Agents: The Futrure

AI Agents: The Future

Agent Reasoning: Planning & Decision Making

AI Agents: The Future

Agent Reasoning: Planning & Decision Making

The LLM Core doesn't just generate text; **it plans**.

Given a goal, it needs to decide:

- What **steps** are needed?
- Which **tool** (if any) is best for the current step?
- What **information** does that tool need?
- How should I interpret the tool's output?

Common techniques guide this planning. ReAct , COT , TOT

AI Agents: The Future

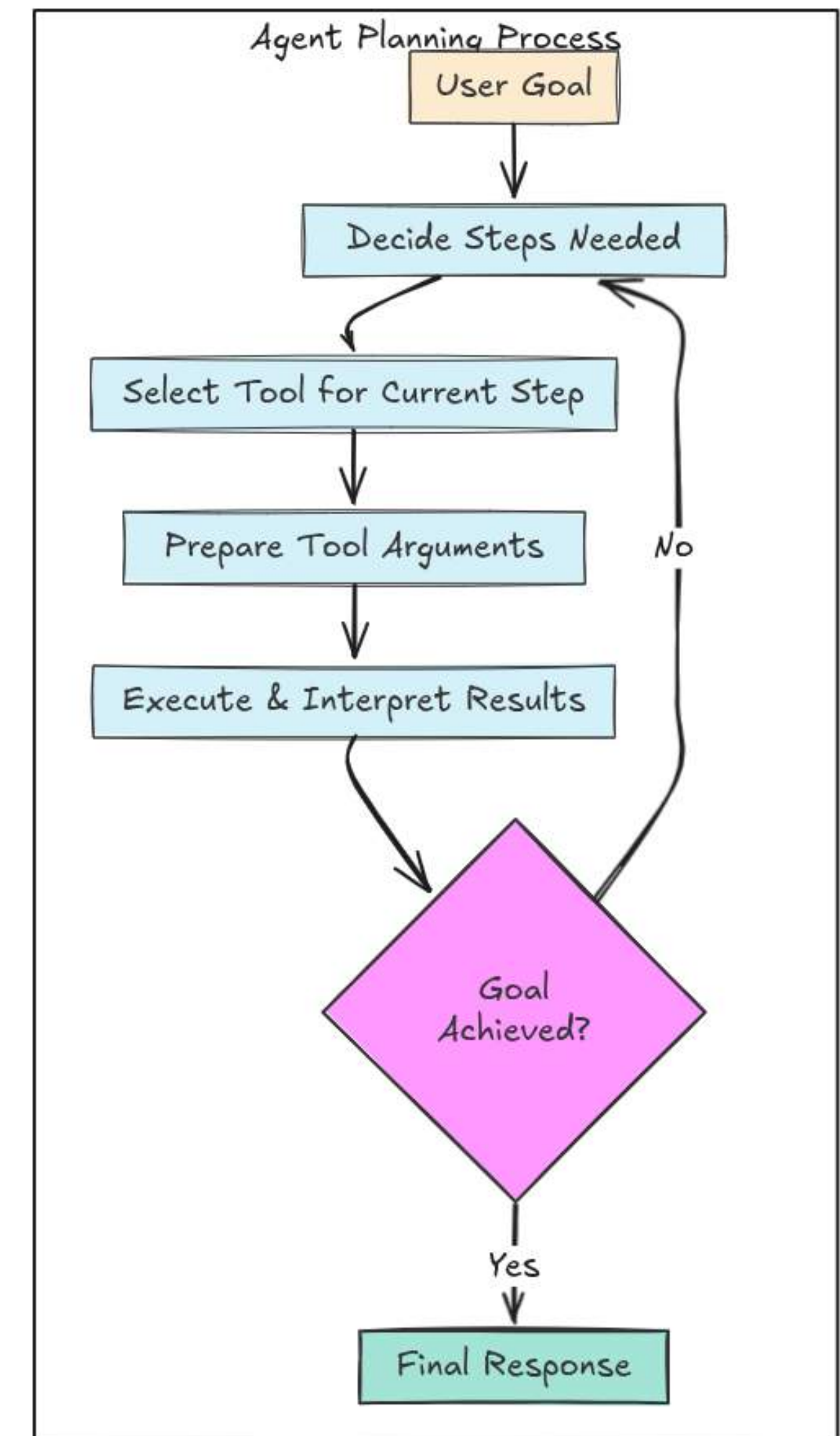
Agent Reasoning: Planning & Decision Making

The LLM Core doesn't just generate text; **it plans**.

Given a goal, it needs to decide:

- What **steps** are needed?
- Which **tool** (if any) is best for the current step?
- What **information** does that tool need?
- How should I interpret the tool's output?

Common techniques guide this planning. ReAct , COT , TOT



AI Agents: The Futrure

AI Agents: The Future

The ReAct Framework: Reason + Act Cycle

AI Agents: The Future

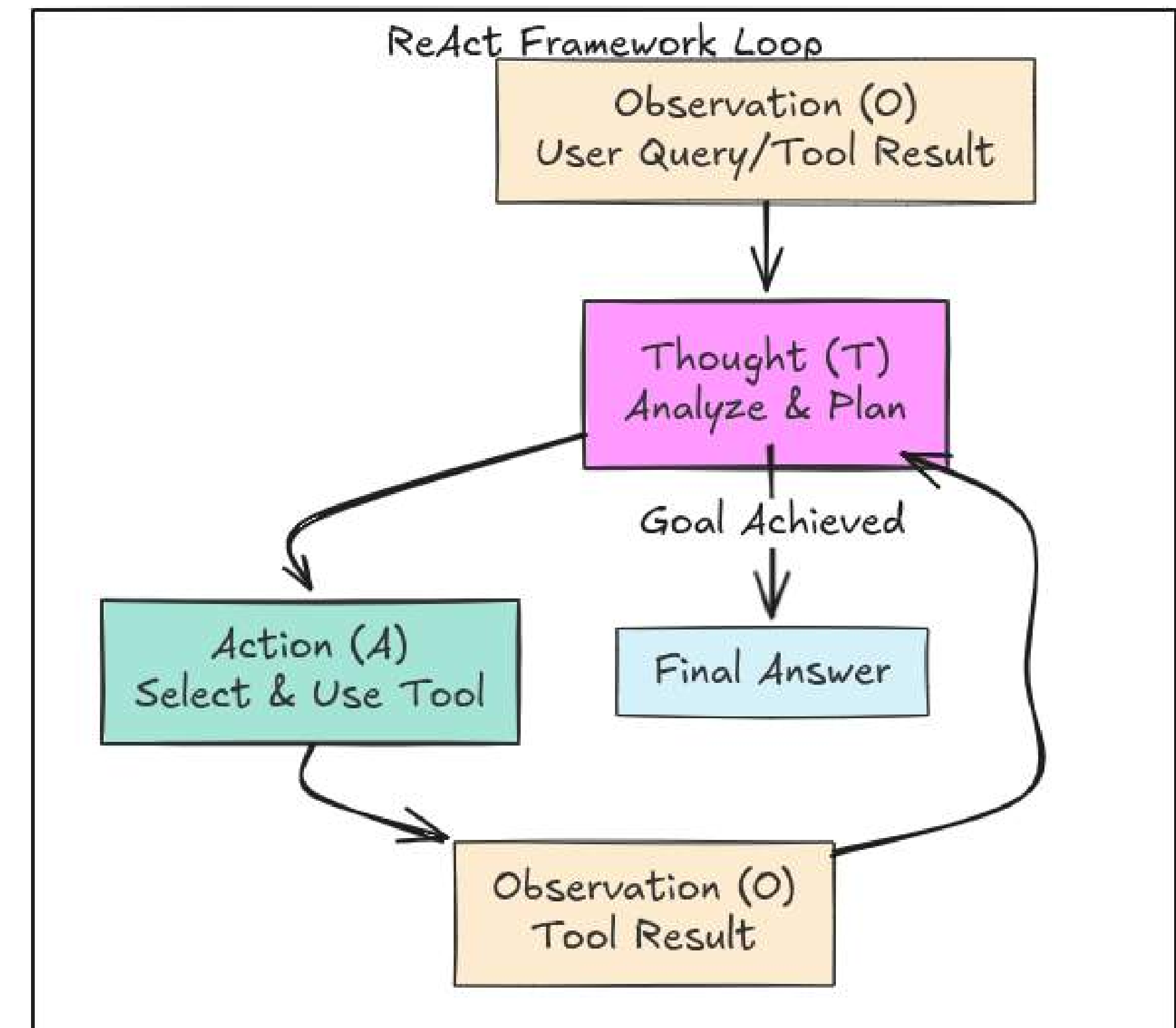
The ReAct Framework: Reason + Act Cycle

- a. Observation (O):** Initial query or result from previous action.
- b. Thought (T):** LLM analyzes Observation, considers the goal, plans the next step, and decides if a tool is needed. (Self-correction can happen here).
- c. Action (A):** LLM outputs the chosen Tool + Input arguments (e.g., Search("latest AI agent research")).
- d. Observation (O):** System executes the Action, gets the result (e.g., search results text), and feeds it back.
- e.** Repeat T-A-O until Thought determines the final answer is ready.

AI Agents: The Future

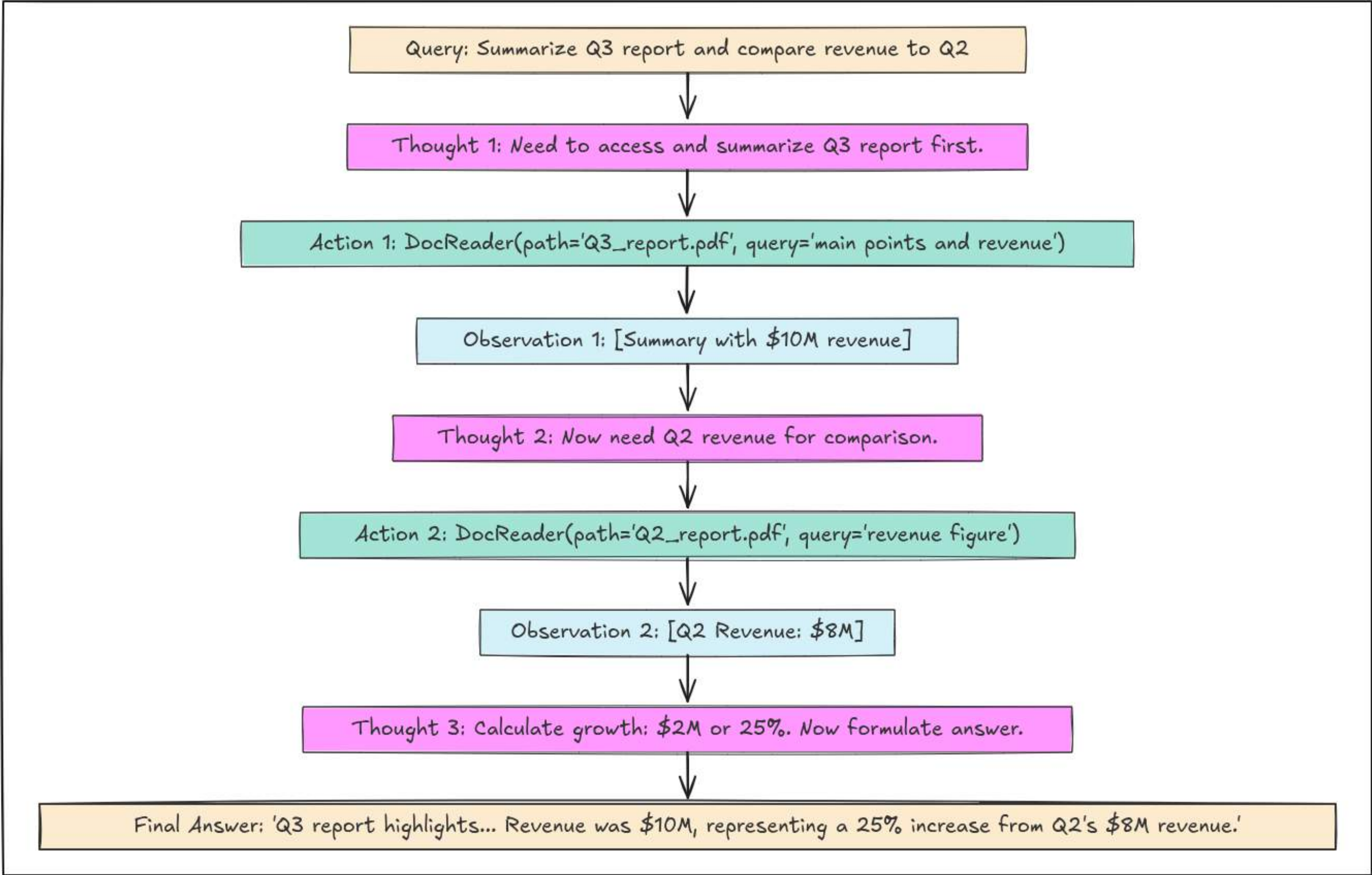
The ReAct Framework: Reason + Act Cycle

- a. **Observation (O)**: Initial query or result from previous action.
- b. **Thought (T)**: LLM analyzes Observation, considers the goal, plans the next step, and decides if a tool is needed. (Self-correction can happen here).
- c. **Action (A)**: LLM outputs the chosen Tool + Input arguments (e.g., Search("latest AI agent research")).
- d. **Observation (O)**: System executes the Action, gets the result (e.g., search results text), and feeds it back.
- e. Repeat T-A-O until Thought determines the final answer is ready.



AI Agents: The Future

The ReAct Framework: Reason + Act Cycle



AI Agents: The Future

The ReAct Framework: Reason + Act Cycle

```
model_instructions = """
Solve a question answering task with interleaving Thought, Action, Observation steps. Thought can reason about the current situation,
Observation is understanding relevant information from an Action's output and Action can be one of three types:
(1) <search>entity</search>, which searches the exact entity on Wikipedia and returns the first paragraph if it exists. If not, it
    will return some similar entities to search and you can try to search the information from those topics.
(2) <lookup>keyword</lookup>, which returns the next sentence containing keyword in the current context. This only does exact matches,
    so keep your searches short.
(3) <finish>answer</finish>, which returns the answer and finishes the task.
"""
```

AI Agents: The Future

The ReAct Framework: Reason + Act Cycle

```
example1 = """Question
Musician and satirist Allie Goertz wrote a song about the "The Simpsons" character Milhouse, who Matt Groening named after who?

Thought 1
The question simplifies to "The Simpsons" character Milhouse is named after who. I only need to search Milhouse and find who it is named after.

Action 1
<search>Milhouse</search>

Observation 1
Milhouse Mussolini Van Houten is a recurring character in the Fox animated television series The Simpsons voiced by Pamela Hayden and created by Matt Groening.

Thought 2
The paragraph does not tell who Milhouse is named after, maybe I can look up "named after".

Action 2
<lookup>named after</lookup>

Observation 2
Milhouse was named after U.S. president Richard Nixon, whose middle name was Milhous.

Thought 3
Milhouse was named after U.S. president Richard Nixon, so the answer is Richard Nixon.

Action 3
<finish>Richard Nixon</finish>
"""
```

AI Agents: The Futrure

AI Agents: The Future

Beyond ReAct

AI Agents: The Future

Beyond ReAct

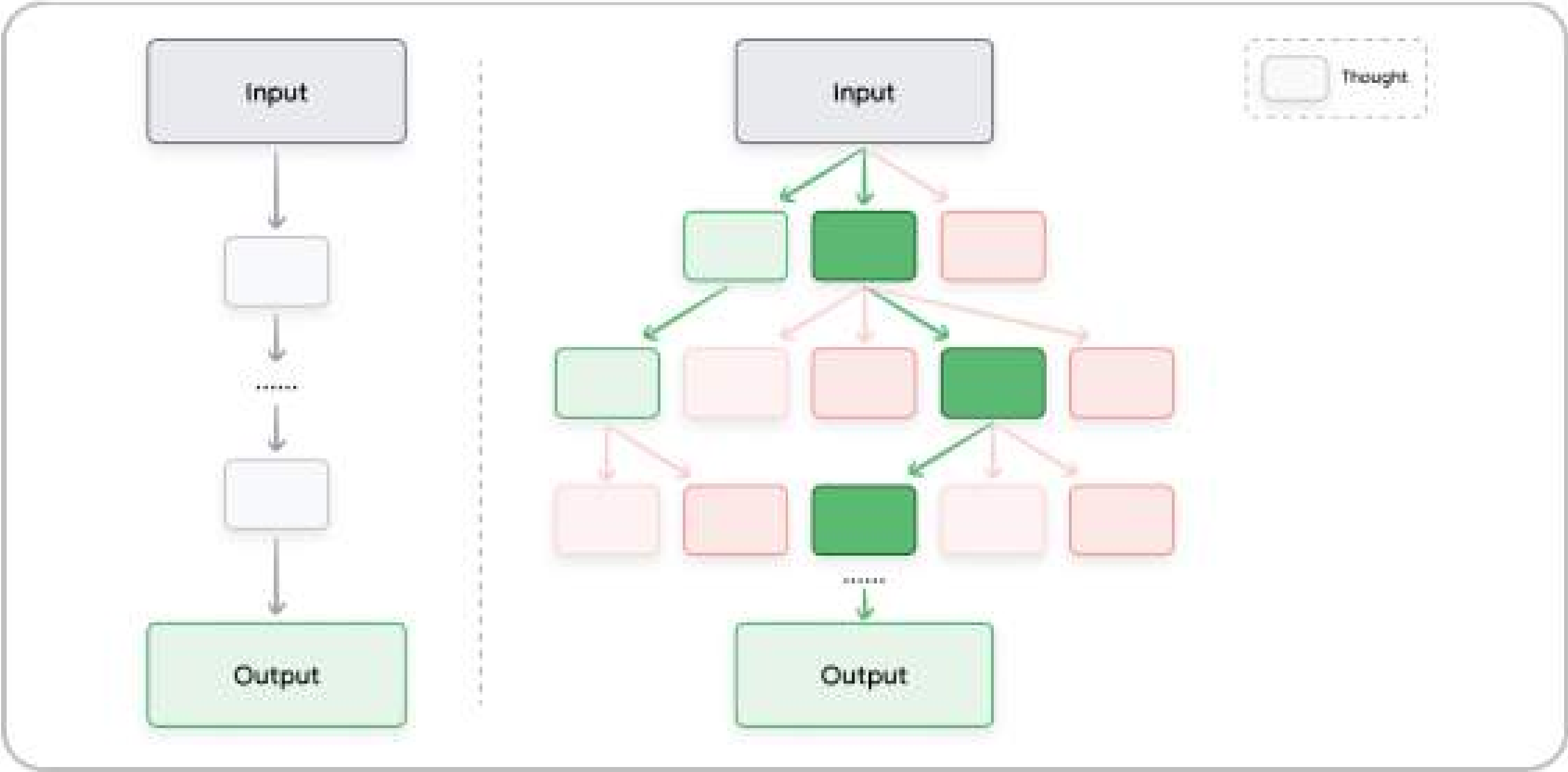


Figure 1. A visualization of chain of thought prompting on the left versus. Tree of Thoughts prompting on the right

AI Agents: The Futrure

Beyond ReAct

AI Agents: The Futrure

Beyond ReAct

Prompt	When I was 3 years old, my partner was 3 times my age. Now, I am 20 years old. How old is my partner?
Output	63 years old

Table 11. An example of a prompt which is trying to solve a mathematical problem

AI Agents: The Future

Beyond ReAct

Prompt	When I was 3 years old, my partner was 3 times my age. Now, I am 20 years old. How old is my partner?
Output	63 years old

Table 11. An example of a prompt which is trying to solve a mathematical problem

Prompt	When I was 3 years old, my partner was 3 times my age. Now, I am 20 years old. How old is my partner? Let's think step by step.
Output	1. Let's assume my current age is "x" years. 2. When I was 3 years old, my age was 3 years. 3. My partner's age at that time was 3 times my age, which means my partner's age was $3 * 3 = 9$ years. 4. Now, I am 20 years old, which means my age has increased by $20 - 3 = 17$ years since I was 3 years old. 5. Since my partner's age was 9 years when I was 3 years old, and my age has increased by 17 years, my partner's age must have also increased by 17 years. 6. Therefore, my partner's current age is $9 + 17 = 26$ years. So, my partner is 26 years old.

Table 12. An example of Chain of Thought prompting

AI Agents: The Future

Beyond ReAct

Prompt	When I was 3 years old, my partner was 3 times my age. Now, I am 20 years old. How old is my partner?
Output	63 years old

Table 11. An example of a prompt which is trying to solve a mathematical problem

Prompt	Q: When my brother was 2 years old, I was double his age. Now I am 40 years old. How old is my brother? Let's think step by step. A: When my brother was 2 years, I was $2 * 2 = 4$ years old. That's an age difference of 2 years and I am older. Now I am 40 years old, so my brother is $40 - 2 = 38$ years old. The answer is 38. Q: When I was 3 years old, my partner was 3 times my age. Now, I am 20 years old. How old is my partner? Let's think step by step. A:
Output	When I was 3 years old, my partner was $3 * 3 = 9$ years old. That's an age difference of 6 years and my partner is older. Now I am 20 years old, so my partner is $20 + 6 = 26$ years old. The answer is 26.

Table 13. An example of chain of thought prompting with a single-shot

AI Agents: The Futrure

AI Agents: The Future

When to use agents / when to avoid them

AI Agents: The Future

When to use agents / when to avoid them

Agents are useful when you need an LLM to determine the workflow of an app. But they're often **overkill**. If you know the exact workflow of your app , relying on an agentic system is a **mistake**

AI Agents: The Future

When to use agents / when to avoid them

Agents are useful when you need an LLM to determine the workflow of an app. But they're often **overkill**. If you know the exact workflow of your app, relying on an agentic system is a **mistake**

Example : you have a company, and in your website the users.

Want some knowledge on the trips? ⇒ **give them access to a search bar to search your knowledge base**

Wants to talk to sales? ⇒ **let them type in a contact form.**

AI Agents: The Future

When to use agents / when to avoid them

If that deterministic workflow fits all queries, by all means just **code everything!** This will give you a **100% reliable** system with no risk of error introduced by letting unpredictable LLMs meddle in your workflow.

AI Agents: The Future

When to use agents / when to avoid them

If that deterministic workflow fits all queries, by all means just **code everything!** This will give you a **100% reliable** system with no risk of error introduced by letting unpredictable LLMs meddle in your workflow.

Flexibility

AI Agents: The Future

When to use agents / when to avoid them

If that deterministic workflow fits all queries, by all means just **code everything!** This will give you a **100% reliable** system with no risk of error introduced by letting unpredictable LLMs meddle in your workflow.

Flexibility



AI Agents: The Future

When to use agents / when to avoid them

If that deterministic workflow fits all queries, by all means just **code everything**! This will give you a **100% reliable** system with no risk of error introduced by letting unpredictable LLMs meddle in your workflow.

Flexibility



AI agent

Evaluation, Limitations & Future

Evaluation, Limitations & Future

Evaluating Agents: Beyond Accuracy

Evaluation, Limitations & Future

Evaluating Agents: Beyond Accuracy

- More complex than evaluating simple LLM outputs.
- Goal Completion, Task Success Rate, Trajectory Quality (efficiency, correctness of steps/tool use), Robustness to errors, Cost.
- Requires benchmarks, human oversight, analysis of intermediate steps.

Evaluation, Limitations & Future

Evaluating Agents: Beyond Accuracy

- More complex than evaluating simple LLM outputs.
- Goal Completion, Task Success Rate, Trajectory Quality (efficiency, correctness of steps/tool use), Robustness to errors, Cost.
- Requires benchmarks, human oversight, analysis of intermediate steps.

Benchmarks : BFCL, τ -bench, PlanBench, AgentBench, DBAStep

Evaluation, Limitations & Future

Evaluating Agents: Beyond Accuracy

- More complex than evaluating simple LLM outputs.
- Goal Completion, Task Success Rate, Trajectory Quality (efficiency, correctness of steps/tool use), Robustness to errors, Cost.
- Requires benchmarks, human oversight, analysis of intermediate steps.

Benchmarks : BFCL, τ -bench, PlanBench, AgentBench, DBAStep

LLMs as judges



Frameworks

Frameworks



LangGraph



LlamaIndex



LangChain

And more ...



smolagents



Conclusion

The future is agentic — and you must be too!

What's left to cover

- Tool Selection & Execution
- Memory
- Multi-Agent Systems
- Agentic RAG
- AgentOps

Resources & Further Learning

- <https://langchain-ai.github.io/langgraph/tutorials/introduction/>
- <https://huggingface.co/learn/agents-course/unit0/introduction>
- <https://www.ibm.com/think/topics/parameter-efficient-fine-tuning>
- <https://www.linkedin.com/feed/update/urn:li:activity:7322560318465892352/>
- <https://github.com/Chamiln17/generative-ai-for-beginners/tree/main/01-introduction-to-genai>
- **and more ...**

Q&A

