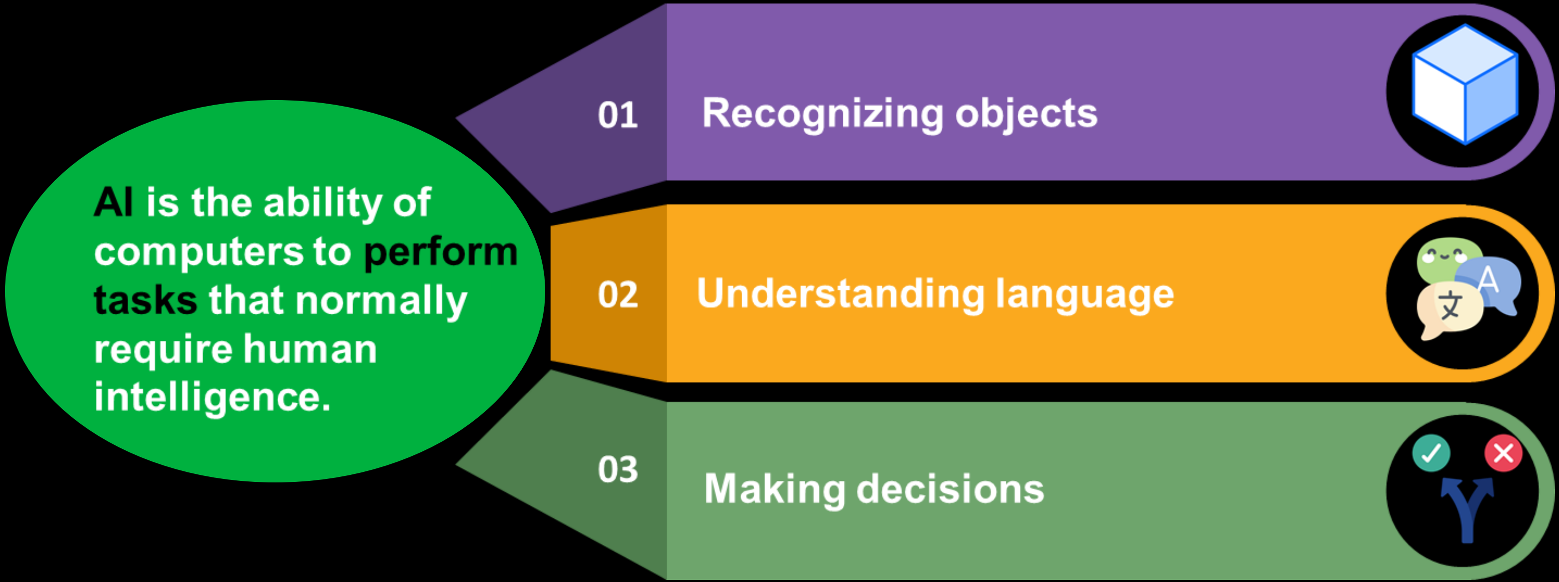
REPLY

REPLY STUDENT CLASH TECH CASE STUDY

WHAT IS ARTIFICIAL INTELLIGENCE?



AI is the ability of computers to **perform tasks** that normally require human intelligence.

01 Recognizing objects

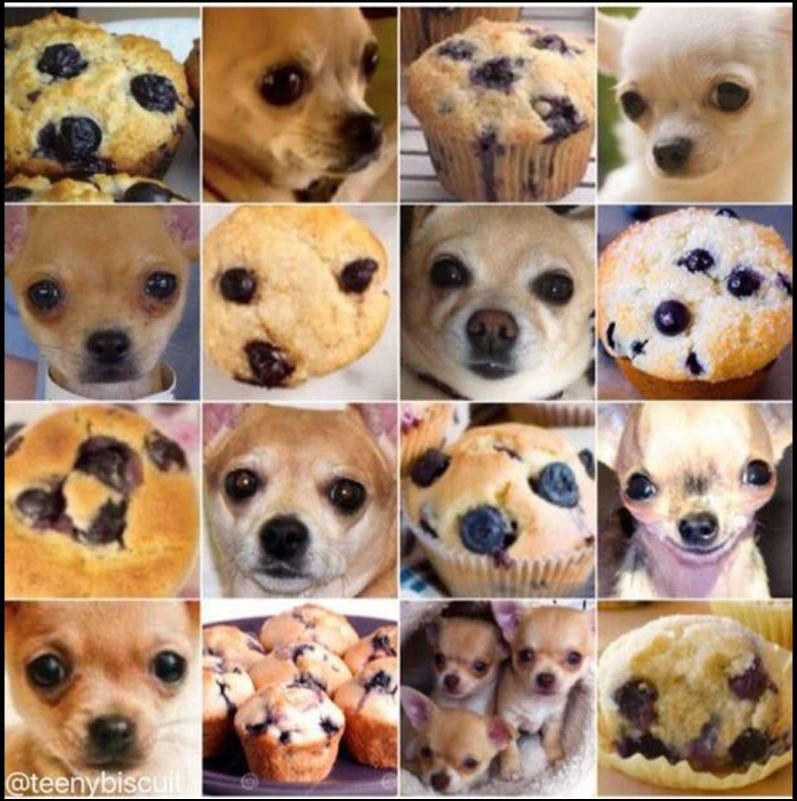
02 Understanding language

03 Making decisions

AI is used in everyday life, like virtual assistants (e.g., Siri, Alexa), spam filters in emails, and recommendations on Netflix.

WHAT IS ARTIFICIAL INTELLIGENCE?

What do you see?



FOUNDATION TECHNOLOGIES

Computer Vision

Enables agents to interpret visual information from the environment.

Natural Language Processing

Allows agents to understand and respond to human language effectively.

Knowledge Representation

Structures information for agents to reason and understand.

Machine Learning

Enables agents to learn and improve from data using various algorithms.

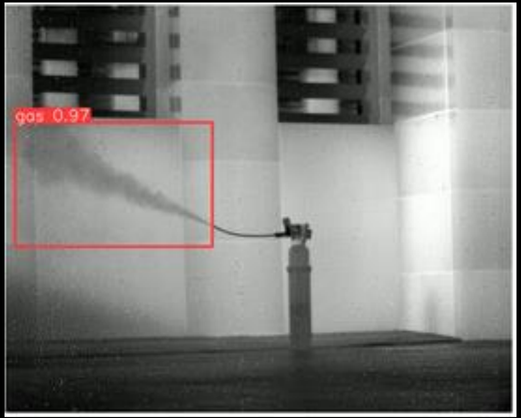
Robotics

Integrates agents with physical systems to perform real-world tasks.

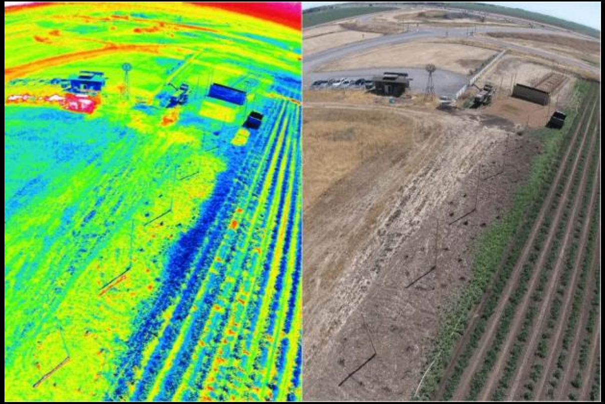
WHERE AI MEETS ROBOTICS

- AI is playing a pivotal role in enhancing the capabilities of robotic systems, particularly in surgical, diagnostic, and rehabilitation applications.
- By enabling robots to process large amounts of data, learn from experience, and adapt to complex situations, AI is improving the precision, safety, and efficiency of medical procedures.

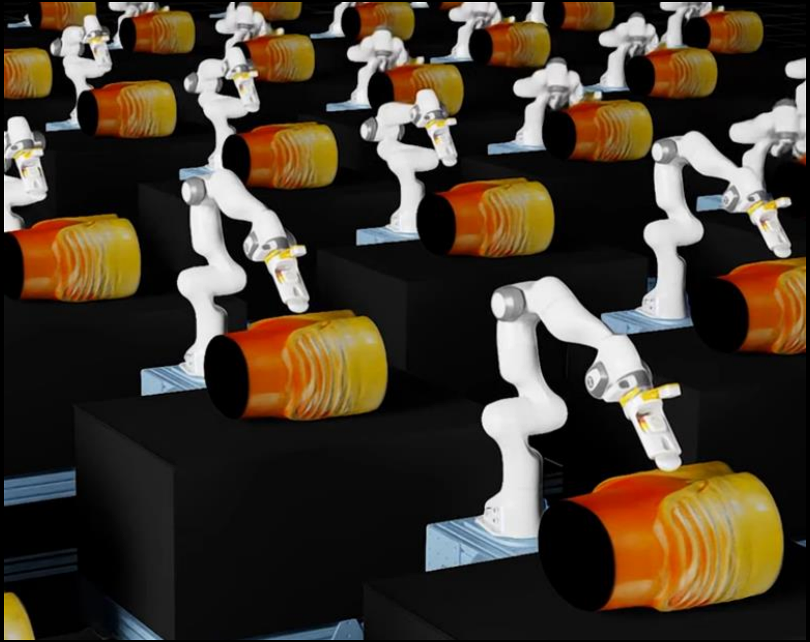
COMPUTER VISION ENABLES AI EYES

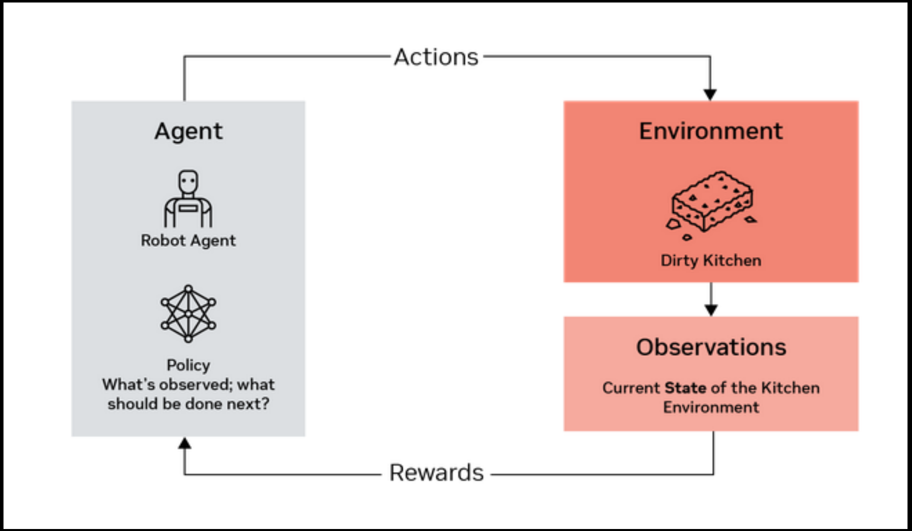


Computer vision is essential for enabling machines to perceive and interpret their surroundings, mimicking human visual capabilities. By integrating cameras with advanced image processing algorithms, robots can identify target objects in real-time, even in complex or hazardous environments.



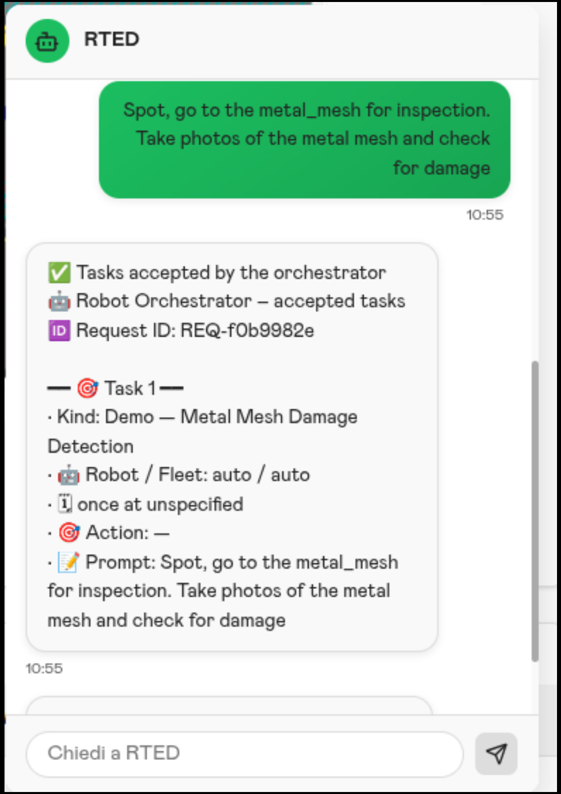
REINFORCEMENT LEARNING




Actions
Agent
Robot Agent
Policy
What's observed; what should be done next?
Environment
Dirty Kitchen
Observations
Current State of the Kitchen Environment
Rewards

RL enables robots to adapt and optimize their performance. Through an iterative procedure, an AI agent makes actions in the environment and collects rewards from it based on the result of the action. This process will be used to improve AI agent policies that provide ability to the robots to correctly perform a specific task.

ROBOTS INTERACTION THROUGH AI AGENT



Spot, go to the metal_mesh for inspection. Take photos of the metal mesh and check for damage

10:55

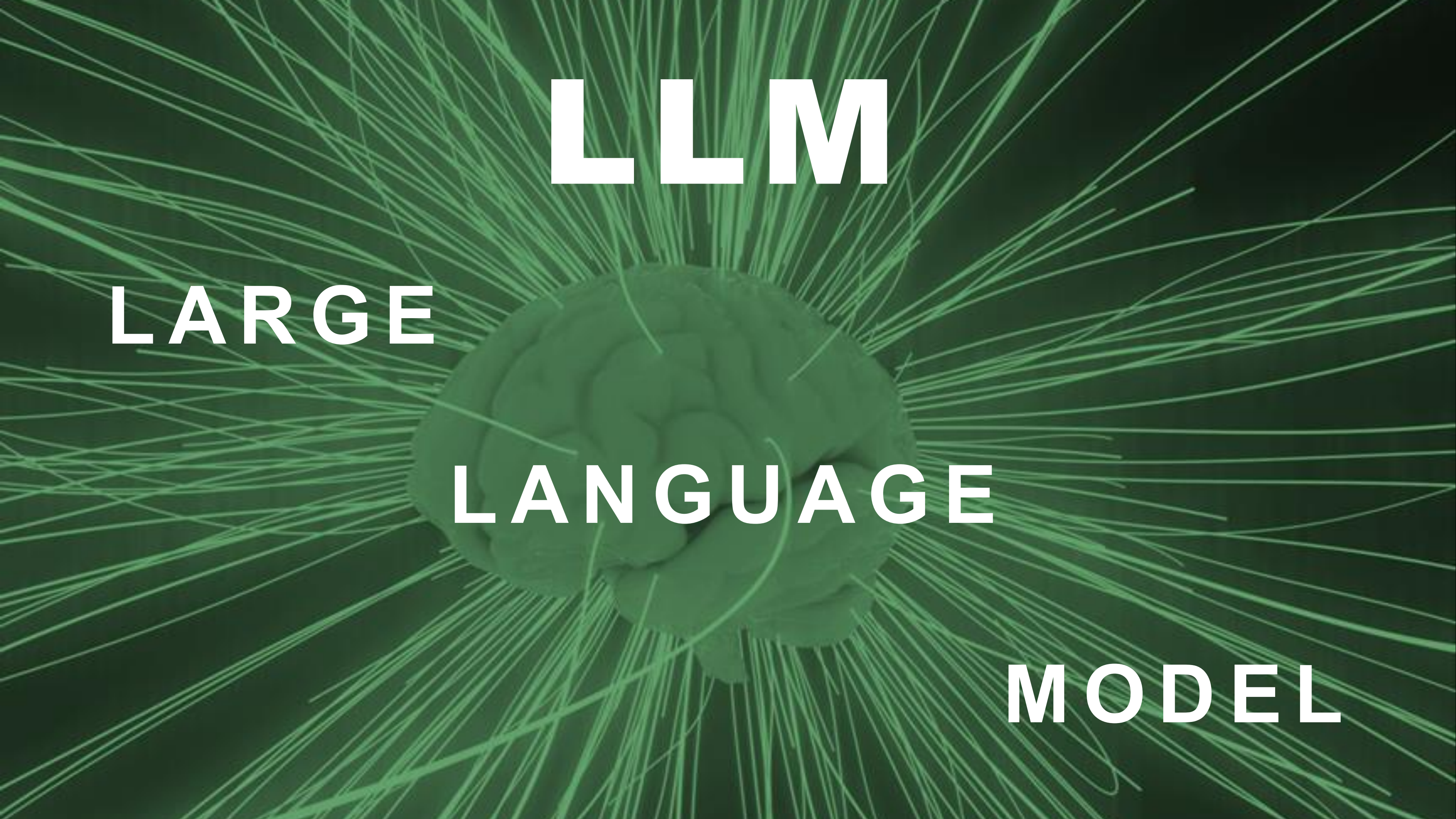
✅ Tasks accepted by the orchestrator
🤖 Robot Orchestrator – accepted tasks
🆔 Request ID: REQ-f0b9982e

— 🎯 Task 1 —
· Kind: Demo — Metal Mesh Damage Detection
· 🤖 Robot / Fleet: auto / auto
· 1️⃣ once at unspecified
· 🎯 Action: —
· 📝 Prompt: Spot, go to the metal_mesh for inspection. Take photos of the metal mesh and check for damage

10:55

- **Natural Language Control**: Chatbots powered by AI enable users to command and interact with robots using simple, conversational language, eliminating the need for complex programming.

- **Real-Time Feedback & Adaptation**: Through chatbot interfaces, robots can provide instant status updates and adapt their actions based on user input, creating a seamless human-robot collaboration experience.



LLM

LARGE

LANGUAGE

MODEL

LARGE LANGUAGE MODELS

An LLM, a large language model, is a neural network designed to understand, generate, and respond to human-like text

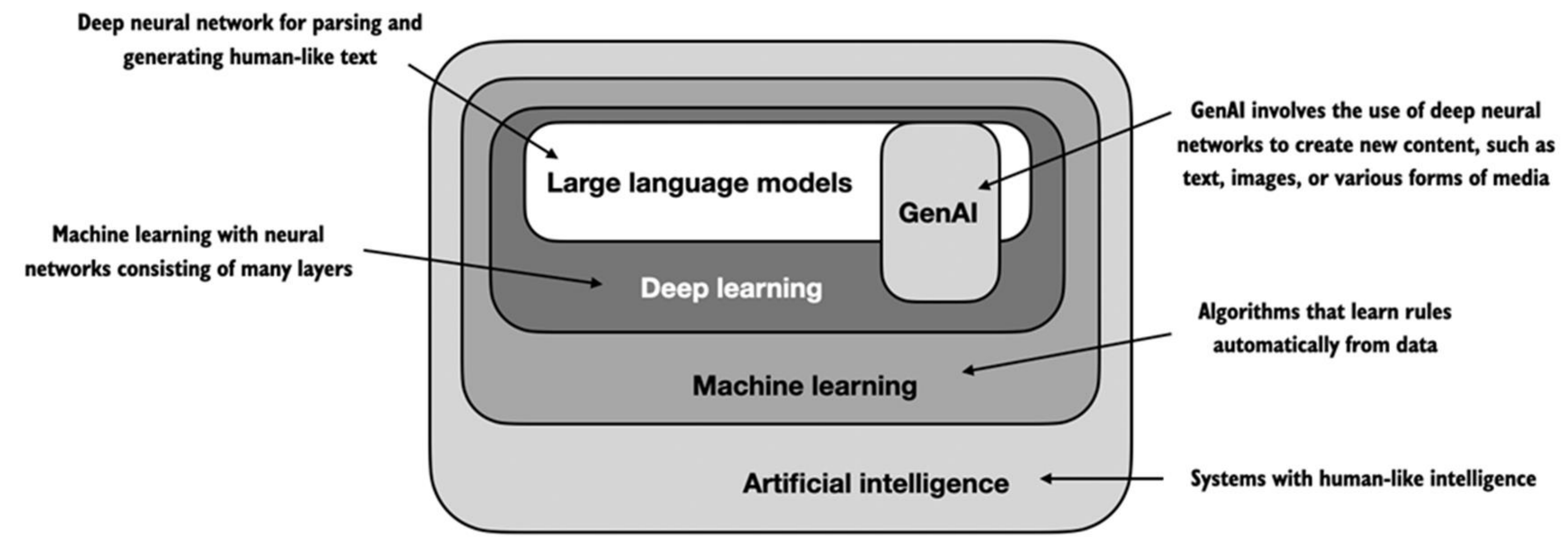
These models are deep neural networks trained on massive amounts of text data, sometimes encompassing large portions of the entire publicly available text on the internet.

LARGE LANGUAGE MODELS

LLMs have remarkable capabilities to understand, generate, and interpret human language

However, it's important to clarify that when we say language models "understand," we mean that they can process and generate text in ways that appear coherent and contextually relevant, not that they possess human-like consciousness or comprehension

LARGE LANGUAGE MODELS



Deep neural network for parsing and generating human-like text

Machine learning with neural networks consisting of many layers

GenAI involves the use of deep neural networks to create new content, such as text, images, or various forms of media

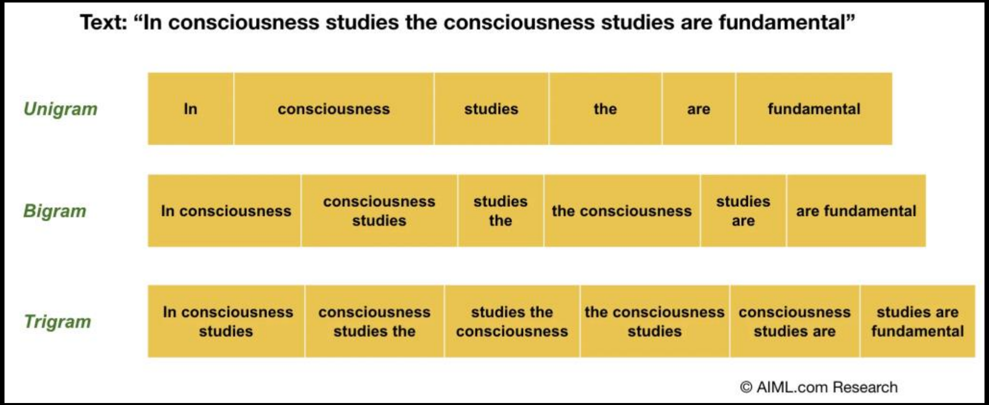
Algorithms that learn rules automatically from data

Systems with human-like intelligence

LANGUAGE MODELS

A language model is a machine learning model that estimates the probability of words or sequences of words, with the goal of predicting the next token in a text based on the preceding context.

A simple example is the n-gram model: this type of model estimates the probability of a word by considering only the last n−1 words in the context. For example, a bigram looks at just one previous word, while a trigram considers two.



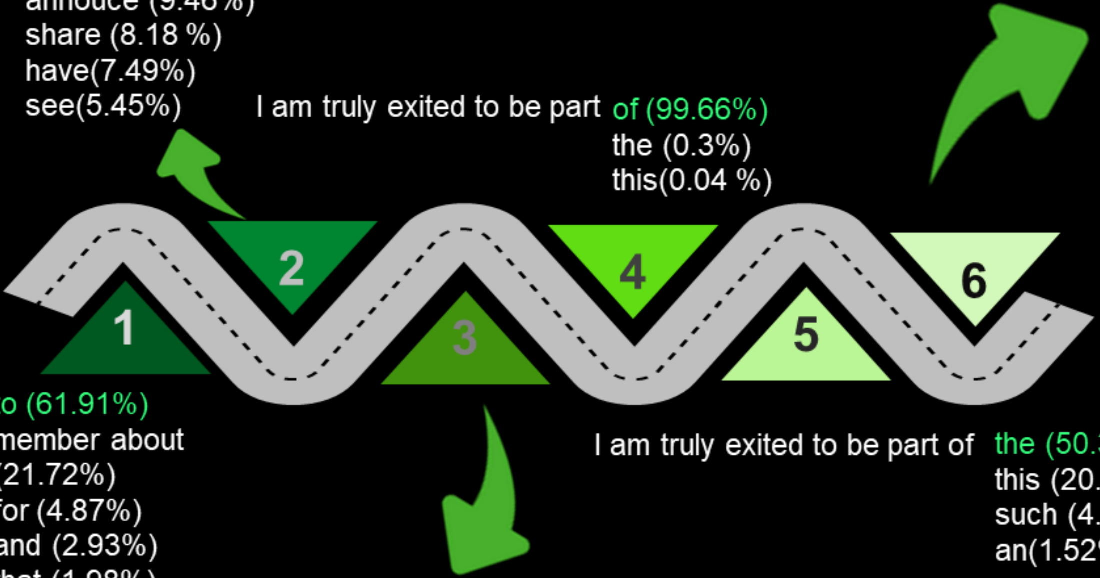
Text: “In consciousness studies the consciousness studies are fundamental”

Unigram	In	consciousness	studies	the	are	fundamental
Bigram	In consciousness	consciousness studies	studies the	the consciousness	studies are	are fundamental
Trigram	In consciousness studies	consciousness studies the	studies the consciousness	the consciousness studies	consciousness studies are	studies are fundamental

© AIML.com Research

HOW LLMS WORKS

PROBABILITY



I am truly exited to (61.91%)
member about (21.72%)
for (4.87%)
and (2.93%)
that (1.98%)

I am truly exited to be (25.43%)
annouce (9.46%)
share (8.18 %)
have(7.49%)
see(5.45%)

I am truly exited to be part (99.66%)
member (0.3%)
new(0.04 %)
Guest(0.88%)
Partner(0.34%)

I am truly exited to be part of (99.66%)
the (0.3%)
this(0.04 %)

I am truly exited to be part of the (50.32%)
this (20.92%)
such (4.73%)
an(1.52%)

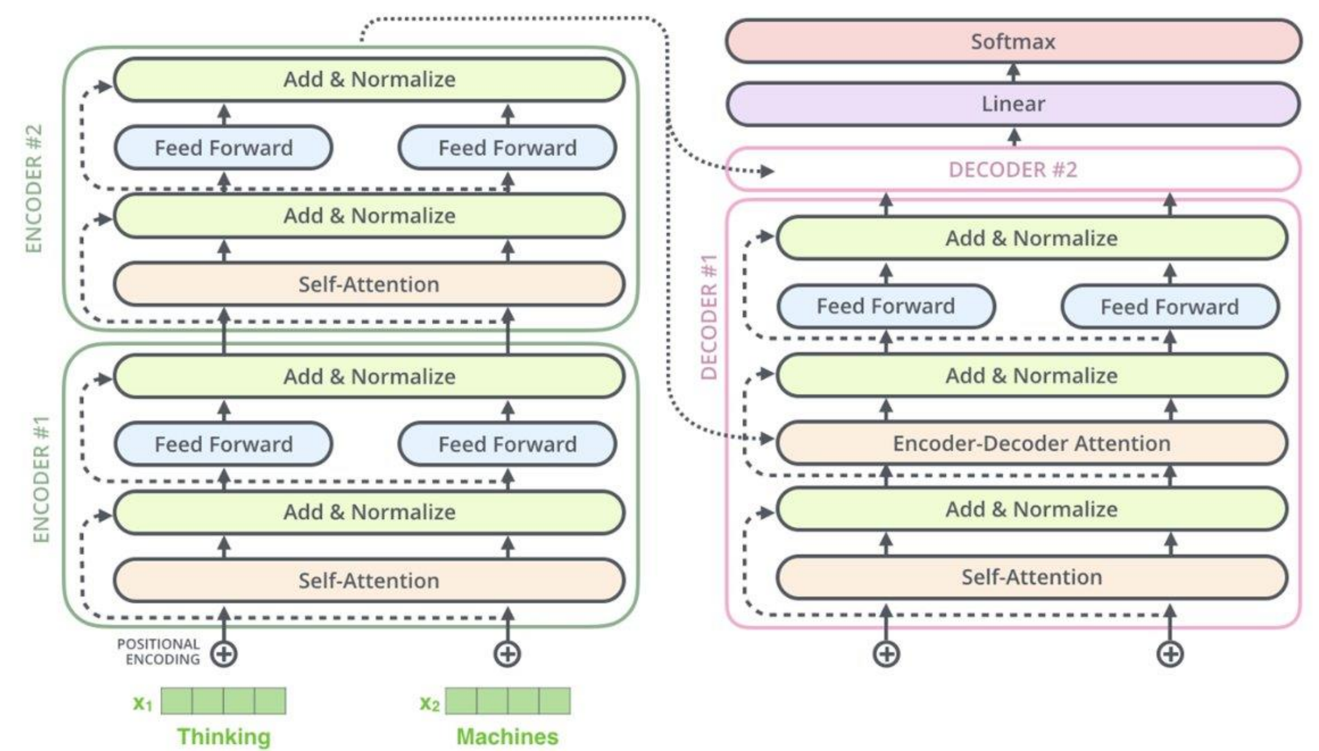
I am truly exited to be part of the team (96.8%)
amazing (1.81%)
new (0.83%)
community(0.61%)

I am truly exited to be part of the team

LARGE LANGUAGE MODELS

The success behind LLMs can be attributed to the **transformer** architecture which underpins many LLMs, and the **vast amounts** of data LLMs are trained on, allowing them to capture a wide variety of linguistic nuances, contexts, and patterns that would be challenging to manually encode

TRANSFORMER ARCHITECTURE



"ATTENTION IS ALL YOU NEED"

A key component of transformers and LLMs is the self-attention mechanism (not shown), which allows the model to weigh the importance of different words or tokens in a sequence relative to each other

This mechanism enables the model to capture long-range dependencies and contextual relationships within the input data, enhancing its ability to generate coherent and contextually relevant output

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

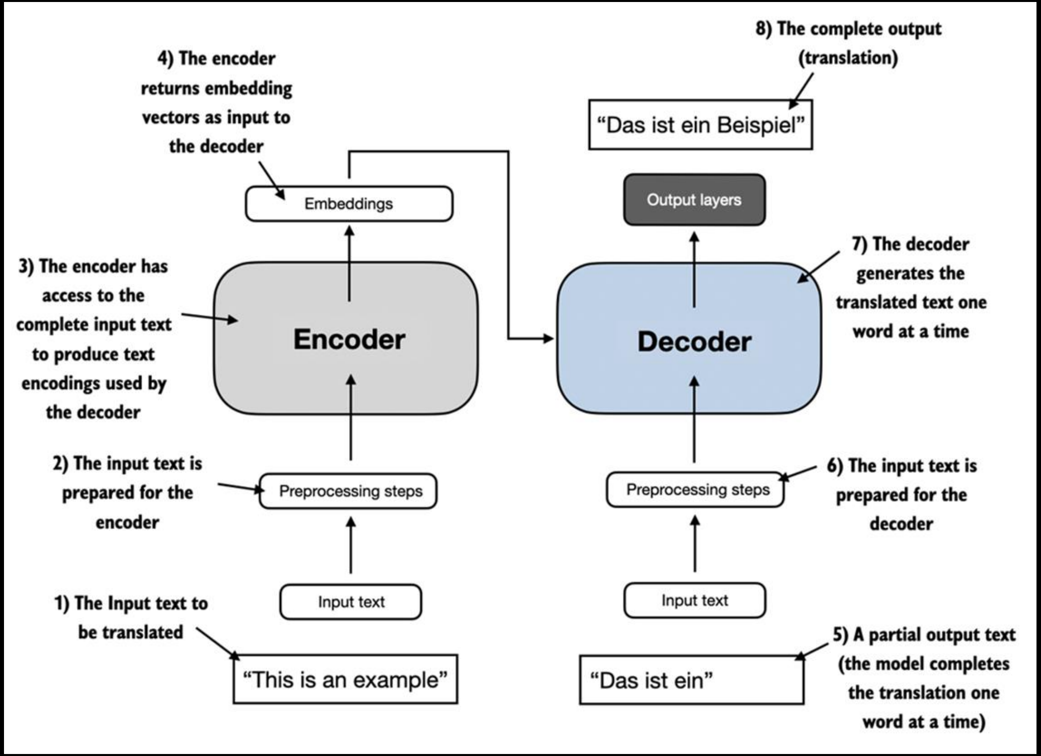
Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

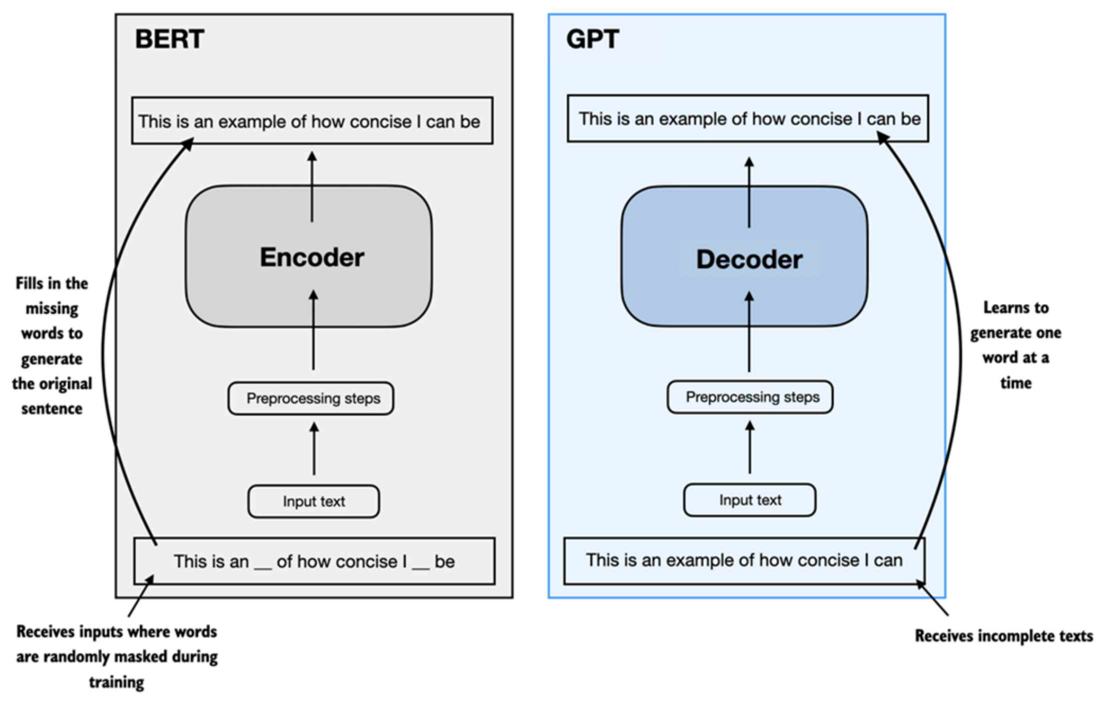
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

TRANSFORMER ARCHITECTURE



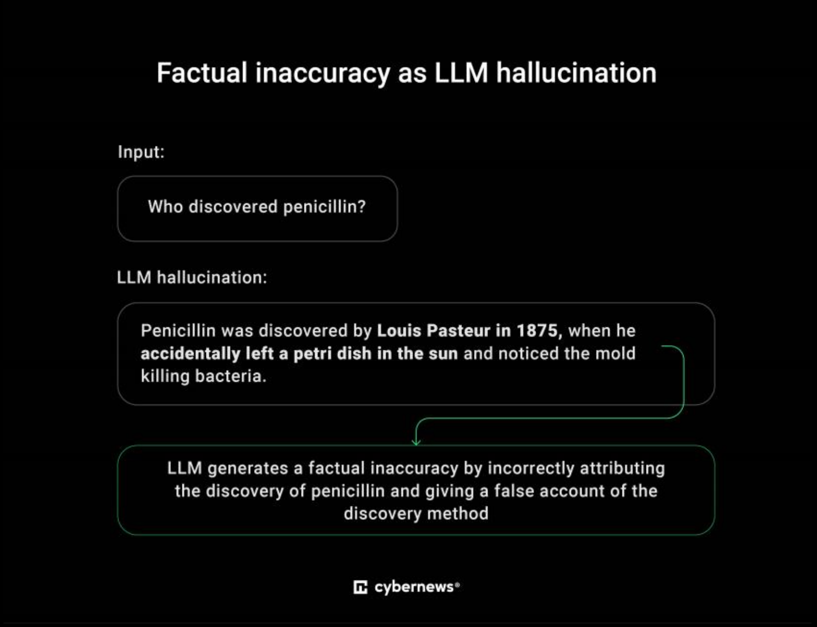
8) The complete output (translation)
"Das ist ein Beispiel"
4) The encoder returns embedding vectors as input to the decoder
Embeddings
Output layers
3) The encoder has access to the complete input text to produce text encodings used by the decoder
Encoder
Decoder
7) The decoder generates the translated text one word at a time
2) The input text is prepared for the encoder
Preprocessing steps
Preprocessing steps
6) The input text is prepared for the decoder
Input text
Input text
1) The Input text to be translated
"This is an example"
"Das ist ein"
5) A partial output text (the model completes the translation one word at a time)

TRANSFORMER ARCHITECTURE


BERT
This is an example of how concise I can be
Encoder
Preprocessing steps
Input text
This is an __ of how concise I __ be
Fills in the missing words to generate the original sentence
Receives inputs where words are randomly masked during training
GPT
This is an example of how concise I can be
Decoder
Preprocessing steps
Input text
This is an example of how concise I can
Learns to generate one word at a time
Receives incomplete texts

LLM: LIMITATIONS

HALLUCINATION



Factual inaccuracy as LLM hallucination

Input:

Who discovered penicillin?

LLM hallucination:

Penicillin was discovered by **Louis Pasteur in 1875**, when he **accidentally left a petri dish in the sun** and noticed the mold killing bacteria.

LLM generates a factual inaccuracy by incorrectly attributing the discovery of penicillin and giving a false account of the discovery method

cybernews

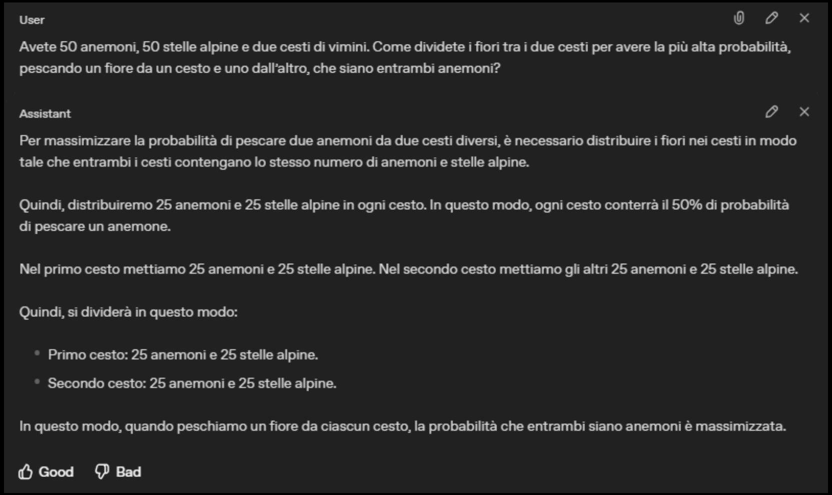
📌 Origin:

- Statistical generation: when certain knowledge is missing, the model fills in with what “sounds plausible,” without an internal verification mechanism.

📌 How it appears:

- False statements delivered with confidence
- invented citations and references
- distorted historical facts or misattributions.

LOGICAL INCAPACITY AND CALCULATION



User

Avete 50 anemoni, 50 stelle alpine e due cesti di vimini. Come dividete i fiori tra i due cesti per avere la più alta probabilità, pescando un fiore da un cesto e uno dall'altro, che siano entrambi anemoni?

Assistant

Per massimizzare la probabilità di pescare due anemoni da due cesti diversi, è necessario distribuire i fiori nei cesti in modo tale che entrambi i cesti contengano lo stesso numero di anemoni e stelle alpine.

Quindi, distribuiremo 25 anemoni e 25 stelle alpine in ogni cesto. In questo modo, ogni cesto conterrà il 50% di probabilità di pescare un anemone.

Nel primo cesto mettiamo 25 anemoni e 25 stelle alpine. Nel secondo cesto mettiamo gli altri 25 anemoni e 25 stelle alpine.

Quindi, si dividerà in questo modo:

- Primo cesto: 25 anemoni e 25 stelle alpine.
- Secondo cesto: 25 anemoni e 25 stelle alpine.

In questo modo, quando peschiamo un fiore da ciascun cesto, la probabilità che entrambi siano anemoni è massimizzata.

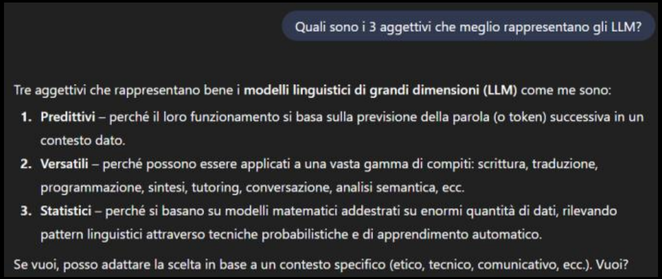
📌 Origin:

- Statistical generation: multi-step reasoning, complex relationships among multiple conditions, or uncommon mathematical calculations are difficult to handle.

📌 How it appears:

- Errors in chains of reasoning (missing steps, contradictions, incoherent conclusions)
- Numerical calculations are often wrong, especially those that are less frequent in training data.

STOCHASTICITY AND VARIABILITY

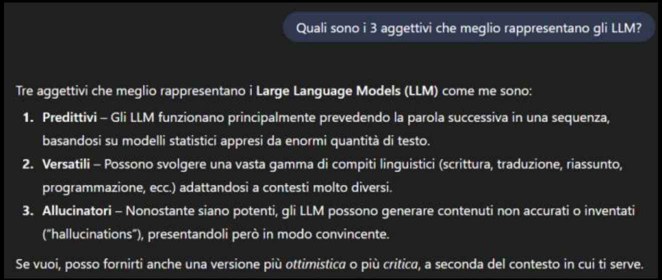


Quali sono i 3 aggettivi che meglio rappresentano gli LLM?

Tre aggettivi che rappresentano bene i **modelli linguistici di grandi dimensioni (LLM)** come me sono:

1. **Predittivi** – perché il loro funzionamento si basa sulla previsione della parola (o token) successiva in un contesto dato.
2. **Versatili** – perché possono essere applicati a una vasta gamma di compiti: scrittura, traduzione, programmazione, sintesi, tutoring, conversazione, analisi semantica, ecc.
3. **Statistici** – perché si basano su modelli matematici addestrati su enormi quantità di dati, rilevando pattern linguistici attraverso tecniche probabilistiche e di apprendimento automatico.

Se vuoi, posso adattare la scelta in base a un contesto specifico (etico, tecnico, comunicativo, ecc.). Vuoi?



Quali sono i 3 aggettivi che meglio rappresentano gli LLM?

Tre aggettivi che meglio rappresentano i **Large Language Models (LLM)** come me sono:

1. **Predittivi** – Gli LLM funzionano principalmente prevedendo la parola successiva in una sequenza, basandosi su modelli statistici appresi da enormi quantità di testo.
2. **Versatili** – Possono svolgere una vasta gamma di compiti linguistici (scrittura, traduzione, riassunto, programmazione, ecc.) adattandosi a contesti molto diversi.
3. **Allucinatori** – Nonostante siano potenti, gli LLM possono generare contenuti non accurati o inventati ("hallucinations"), presentandoli però in modo convincente.

Se vuoi, posso fornirti anche una versione più *ottimistica* o più *critica*, a seconda del contesto in cui ti serve.

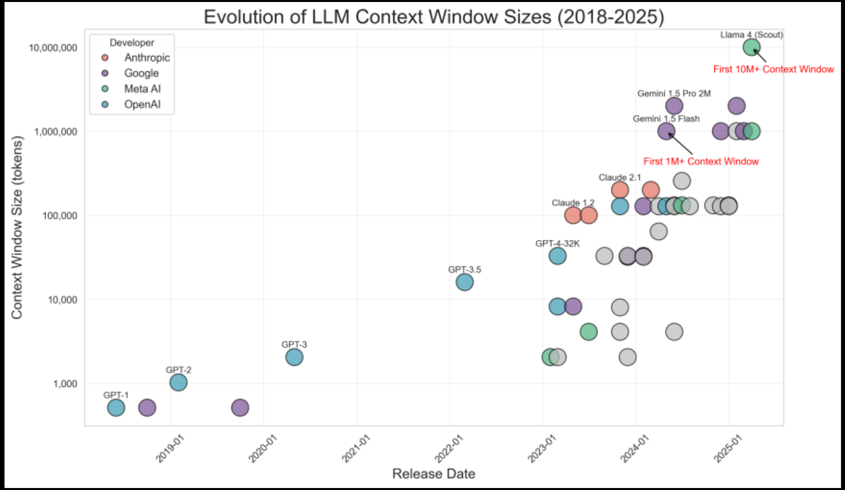
📌 Origin:

- Statistical generation: even with identical inputs, the model explores distributions of possible tokens. Mechanisms (sampling, temperature, random seed) mean it isn't always deterministic..

📌 How it appears:

- Different answers to the same input
- differences in phrasing, style, and detail between repeated identical questions
- difficulty guaranteeing consistency

MEMORY AND CONTEXT



📌 Origin:

- Intrinsic to the architecture: an LLM can consider only a finite number of tokens at a time (context + output). Compute resources and structural limits mean only a portion of the text is “visible” to the model at any moment.

📌 How it appears:

- The model “forgets” or loses information if the conversation becomes too long
- Difficulty maintaining coherence over extended dialogues.

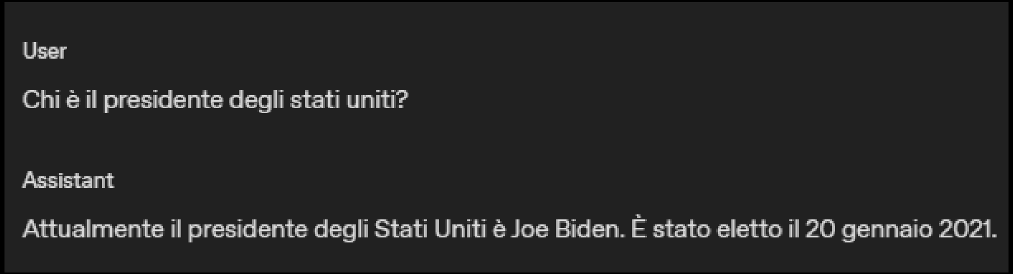
KNOWLEDGE CUT-OFF

📌 **Origin:**

- LLMs are trained on data collected up to a certain date. After that date, they do not autonomously acquire new events or discoveries unless updated with a new dataset.

📌 **How it appears:**

- Outdated responses regarding facts after the cut-off
- Lack of knowledge of recent technologies, regulations, events. [1]



User

Chi è il presidente degli stati uniti?

Assistant

Attualmente il presidente degli Stati Uniti è Joe Biden. È stato eletto il 20 gennaio 2021.

BEYOND THE LIMITS OF LLMS

PROMPT ENGINEERING

Prompt engineering means designing and refining prompts to communicate effectively with AI models, improving the quality, relevance, and accuracy of their responses.

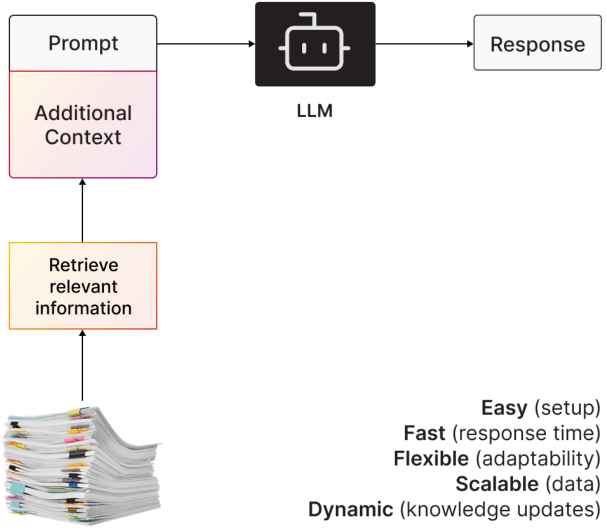
A prompt does not yield a single deterministic answer, but induces a probability distribution over generated tokens: crafting the prompt means shifting the density of this distribution toward the desired outputs.

Thanks to prompt engineering we can:

- Improve the accuracy and relevance of responses
- Control the tone, style, and format of outputs
- Produce more creative results

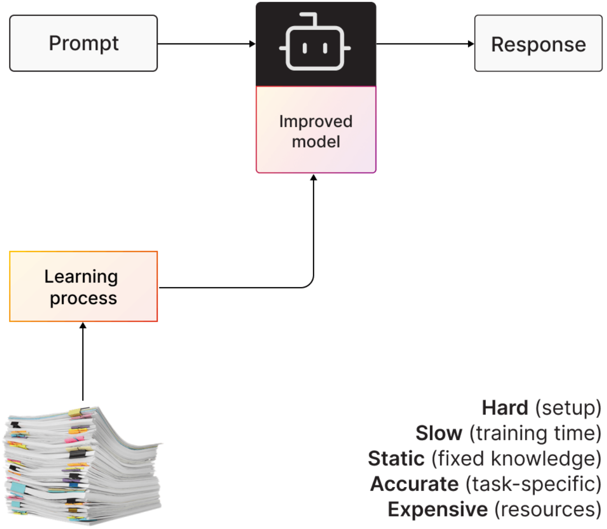
RAG VS FINE-TUNING

Retrieval-augmented generation (RAG)



Easy (setup)
Fast (response time)
Flexible (adaptability)
Scalable (data)
Dynamic (knowledge updates)

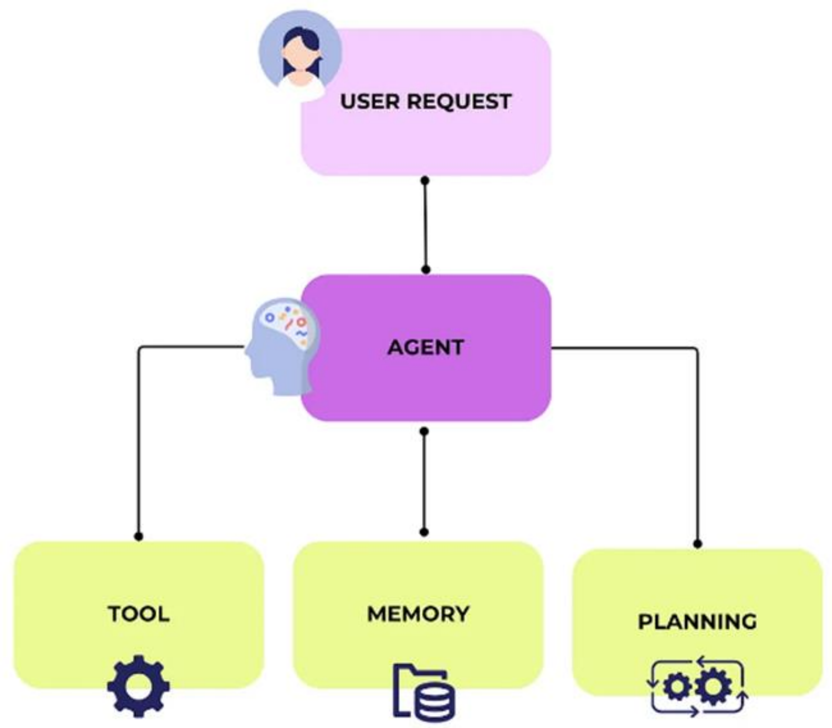
Fine-tuning the Model



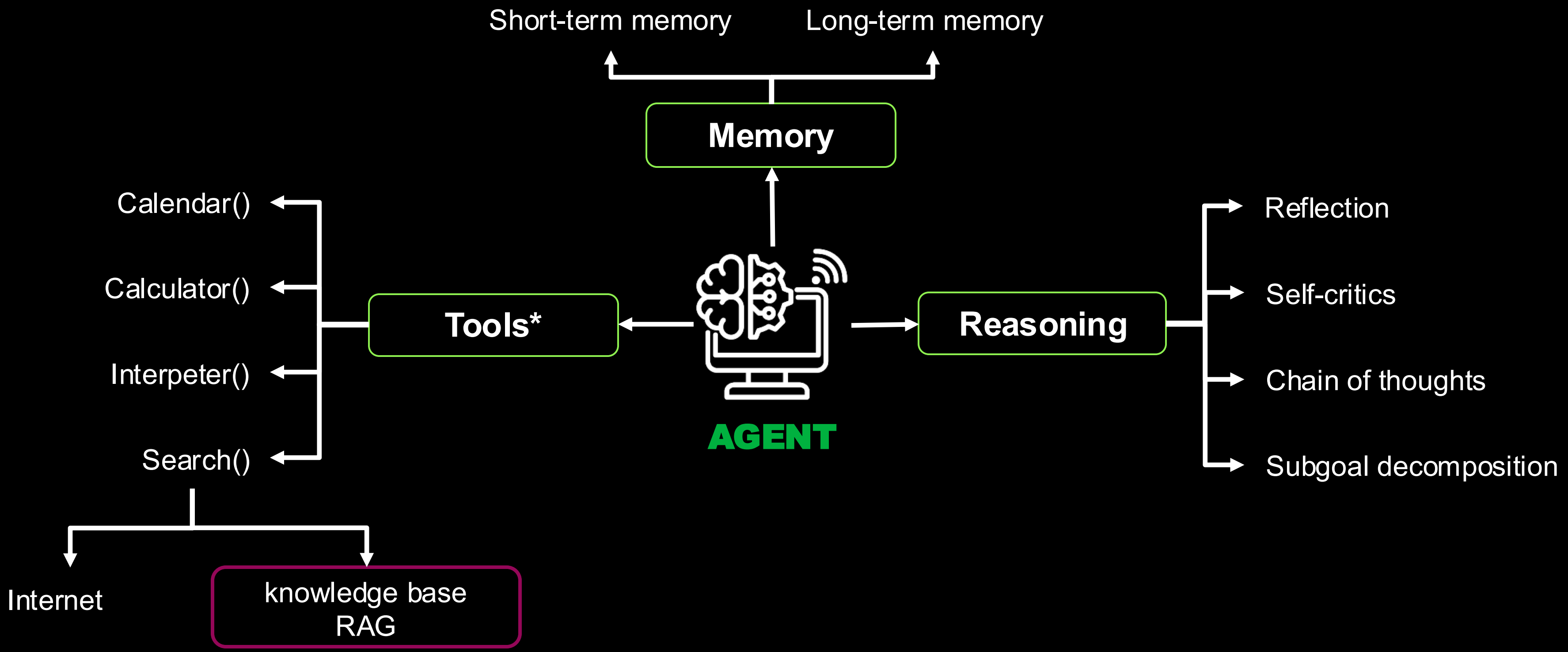
Hard (setup)
Slow (training time)
Static (fixed knowledge)
Accurate (task-specific)
Expensive (resources)

WHAT IS AN AGENT?

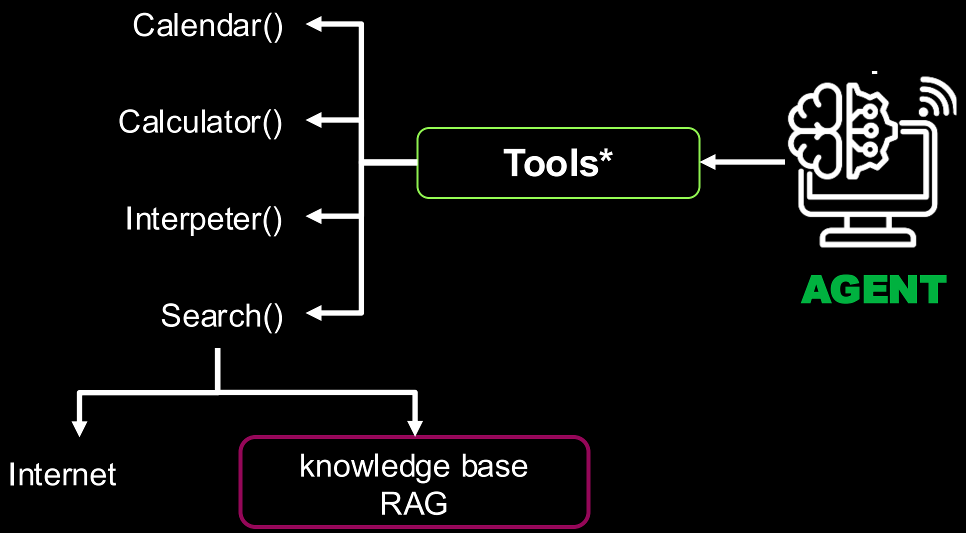
- An **agent** is a system programmed to independently accomplish tasks on your behalf.
- It mimics **human decision-making** and it can be in the form of software programs, AI systems, or even autonomous machines.
- When we talk about workflow execution, we refer to **agentic systems**, where tasks are carried out sequentially to achieve the agent's goal.
- An agent can recognize when the workflow is complete or **correct its actions** in case of errors, or simply hand back control to the user.
- Thanks to LLM's we are now in the era of **LLM – based agents.**



HOW DO AGENTS WORK?



HOW DO AGENTS WORK?



TOOLS are resources that allow the LLM to interact with the external world, such as knowledge bases, websites, APIs, and much more.

The agent can autonomously select the most useful tool based on the current stage of the workflow.

SINGLE-AGENT ARCHITECTURES

HAVA A SINGLE AI AGENT THAT INDIPENDETLY RESOLVES TASKS

Strengths

- Low complexity and thus easier to develop and manage
- No coordination between multiple agents required.
- May require fewer computational resources for a single powerful agent than multiple less powerful agents.

Weaknesses

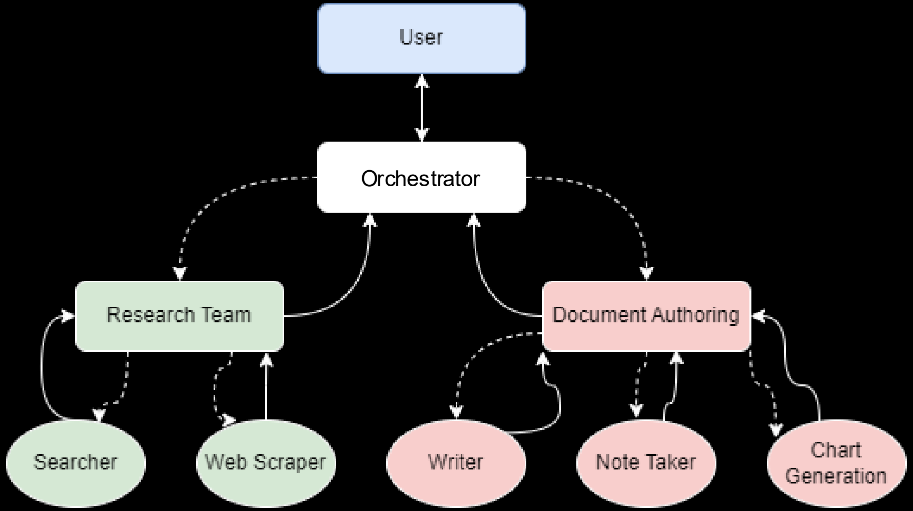
- May struggle with complex or dynamic tasks.
- Limited in handling tasks that require collaboration or diverse expertise
- Agent can get confused and use incorrect tool call arguments if the agent has too many different tool options available.
- May require a larger, more expensive model to handle multiple reasoning steps.

MULTI-AGENT SYSTEM

A **Multi-Agent System (MAS)** is a system where multiple agents interact in a shared environment to reach their goals.

Why Multi-agents?

- **Collaboration**: lead to better outcomes
- **Scalability**: Single agents are limited in handling complex tasks and don't scale with many tools and capabilities
- **Specialization:** Different agents can specialize in different tasks



MULTI-AGENTS

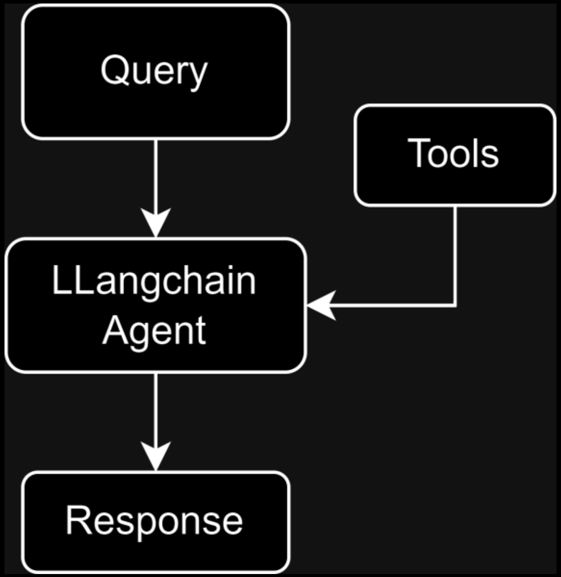
A **multi-agentic system (MAS)** is ideal when dealing with complex decision-making, difficult-to-maintain rules, or reliance on unstructured data.

Agents can be **characterised** by different guardrails, different models, different tools, and so on.

In the most common MAS pattern, there is an **orchestrator**—an agent and tool itself—whose purpose is to coordinate the tasks within the multi-agentic system. It is responsible for enforcing rules on which agent should carry out which task.

BUILDING AGENTS IN PYTHON: LANGCHAIN

LangChain is one of the most used libraries in Python for the creation of LLM agents. It streamlines the code to the bare minimum.



```python
from langchain.agents import initialize_agent, load_tools
from langchain.llms import OpenAI
from langchain.memory import ConversationBufferMemory

# Initialize the LLM and load the tools
llm = OpenAI(api_key="YOUR_API_KEY")
memory = ConversationBufferMemory( memory_key="chat_history")
tools = load_tools(["ddg-search"], llm=llm)

# Initialize the agent and run a query
agent = initialize_agent(tools, llm,
agent_type="conversational-react-description", memory=memory,
verbose=True)

response = agent.run("What's the capital of France?")
```

MULTI-AGENT ARCHITECTURES

Strengths

- Parallelism & scalability: more requests without bottlenecks
- Specialization & modularity: components can be updated and tested individually
- Resilience: one agent fails, the others keep running
- Adaptability: tasks can be reassigned, self-correction enabled

Weaknesses

- Complex coordination (consistent context and state)
- Overhead in calls/latency/costs
- Harder debugging (non-deterministic behaviors)

MULTI-AGENT

✅ PROS

- Parallelism & scalability: more requests without bottlenecks
- Specialization & modularity: components can be updated and tested individually
- Resilience: one agent fails, the others keep running
- Adaptability: tasks can be reassigned, self-correction enabled
- Fine-grained observability: per-agent logs/metrics, clear audit

⚠ LIMITATIONS

- Complex coordination (consistent context and state)
- Overhead in calls/latency/costs
- Harder debugging (non-deterministic behaviors)

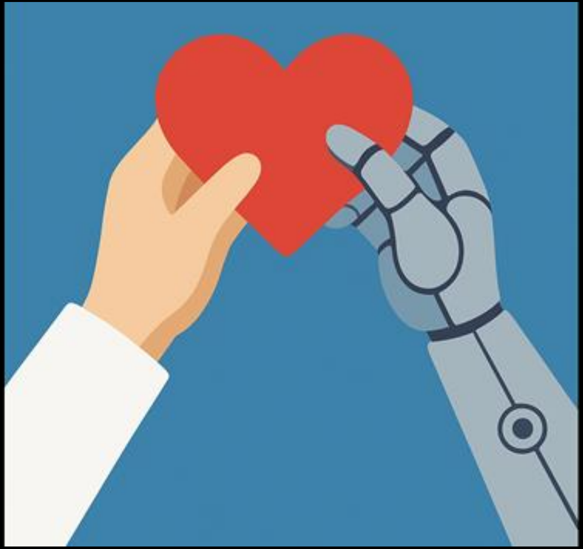
ETHICS & PRIVACY

ETHICS & PRIVACY

WHY DO YOU NEED TO CONSIDER THEM

👉 **Ethics** allows us to reflect on “doing the right thing”.
It’s about whether it is fair, lawful and appropriate to use AI
to improve care, without creating injustice or unnecessary risks.

👉 **Privacy** allows us to “protect sensitive data” that fuel
algorithms
(records, reports, images, messages...)
If this data is leaked, misused, or collected without consent, trust
collapses.
And trust is essential.



ETHICS & PRIVACY

ETHICS: KEY PRINCIPLES

- ⚖️ **Fairness**: AI should minimize bias and promote equitable access for all users.
- 🏛️ **Accountability**: AI outcomes must be overseen by qualified human experts.
- 👁️ **Transparency**: AI systems should be clear about their capabilities and limitations.
- 💬 **Explainability**: AI decisions must be easily understandable.
- 💖 **Beneficence**: AI must not cause harm and should respect human rights.
- 🛡️ **Protection**: AI must ensure privacy safeguards and data protection.
- 🤝 **Autonomy**: AI should support (not replace) human decision-making.

ETHICS & PRIVACY

PRIVACY: KEY PRINCIPLES

In recent years, governments have introduced regulations to ensure responsible AI use
One of the most important examples is the **EU AI Act,** adopted in **2024.**
Although it’s European, it will influence companies and developers all over the world.

The regulation introduces:

- Common rules for the placing on the market and use of AI algorithms and models within the EU
- Limitations and prohibitions on practices considered unacceptable
- Requirements and obligations for providers of high-risk AI algorithms and models
- Transparency rules for certain AI models with specific content
- Objectives and rules for general-purpose AI models
- Supervision, governance, and enforcement mechanisms
- Measures to support innovation, with a focus on SMEs and startups

CASE STUDY

AGENTIC AI FOR HEALTH & WELLNESS

The healthcare and wellness sector offers countless opportunities for innovation through Agentic AI.

Think creatively and design an **AI-based Wellness Agent** with a clear goal and the autonomy to achieve it.

REPLY STUDENT CLASH

CASE STUDY

The case study is organised in levels:
each team can accept the challenge and progress
to the level they are capable of, based on skills and time constraints.

1. IDENTIFY A PROBLEM AND DEVELOP YOUR SOLUTION
2. DEVELOPE A WORKING PROTOTYPE
3. IMPROVE YOUR SOLUTION

LET THE WORK START!

DOWNLOAD THE FULL CASE STUDY

WWW.REPLY.COM

REPLY

DELIVERY

UPLOAD YOUR PROJECT HERE

EXTRA MATERIALS

HOW TO WRITE GOOD PROMPTS

CLARITY AND PRECISION

- Use clear and specific language to guide the model's response.
- Let's suppose we want to generate a Python function that computes the factorial of a number.

Incomplete Prompt:

Write a function for factorial

Problems:

- The programming language is not specified
- It's not clear whether we prefer recursion or iteration
- There are no details about input validation

CLARITY AND PRECISION

- Use clear and specific language to guide the model’s response.
- Let’s suppose we want to generate a Python function that computes the factorial of a number.

Improved Prompt:

Write a Python function called factorial that takes a non-negative integer n as input and returns its factorial. Implement it using recursion and raise ValueError if n is negative.

Components

- Specific language
- Define the function name
- State input constraints
- Define the methodology
- Handle the error

CONTEXT AND CONSTRAINTS

- Include context and constraints that steer responses toward the output we want.
- Let's suppose we want to generate a Python function that sorts a list of numbers.

Incomplete Prompt:

Write a function to sort a list

Problems:

- Which language?
- Which sorting algorithm?
- Ascending or descending order?
- How are special cases handled?
- …

CONTEXT AND CONSTRAINTS

- Include context and constraints that steer responses toward the output we want.
- Let’s suppose we want to generate a Python function that sorts a list of numbers.

Improved Prompt:

Write a Python function custom_sort that takes a list of integers and sorts them in ascending order using the Merge Sort algorithm. The function should be efficient for large lists and should preserve duplicate values. If an empty list is given, return an empty list.

Components

- Context: Python, sort integers in ascending order
- Constraints: Use Merge Sort, handle large lists, preserve duplicates, and handle edge cases.

ROLE-BASED PROMPTING

***Role-Based Prompting** is a prompt engineering technique that explicitly assigns the model a role, persona, or profession (e.g., “teacher,” “legal consultant,” “food critic”) so that the style, tone, and content of the response are shaped by that role, making the output more relevant, specialized, and aligned with the context.*

Prompt without a role:

Explain this python function: …

Problems

- The model doesn’t know your level of experience
- The model returns generic, shallow answers

ROLE-BASED PROMPTING

Role-Based Prompting *is a prompt engineering technique that explicitly assigns the model a role, persona, or profession (e.g., "teacher," "legal consultant," "food critic") so that the style, tone, and content of the response are shaped by that role, making the output more relevant, specialized, and aligned with the context.*

Role-Based Prompt:

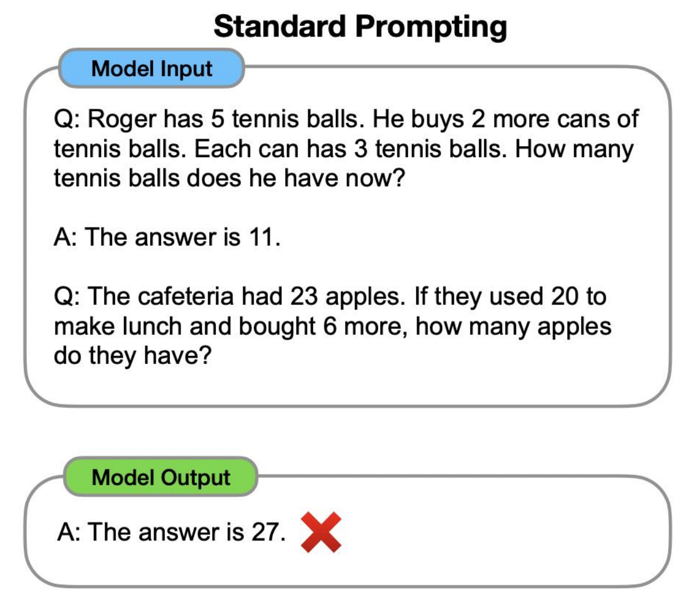
You are a senior python developer reviewing code for a junior colleague. Explain the following function with clear comments and suggestions for improvement: …

Components

- Set the tone
- Steer the output toward a complete and structured answer
- Make the contents of the answer explicit

CHAIN-OF-THOUGHT PROMPTING

*Il **chain-of-thought (CoT) prompting** is a technique that pushes the model to spell out a sequence of intermediate reasoning steps (step-by-step) before the final answer, improving performance on arithmetic, logical, and commonsense reasoning tasks; it can be induced by showing a few examples of reasoning chains or, in some cases, with zero-shot instructions like “let’s think step by step”.*



Standard Prompting

Model Input

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

A: The answer is 11.

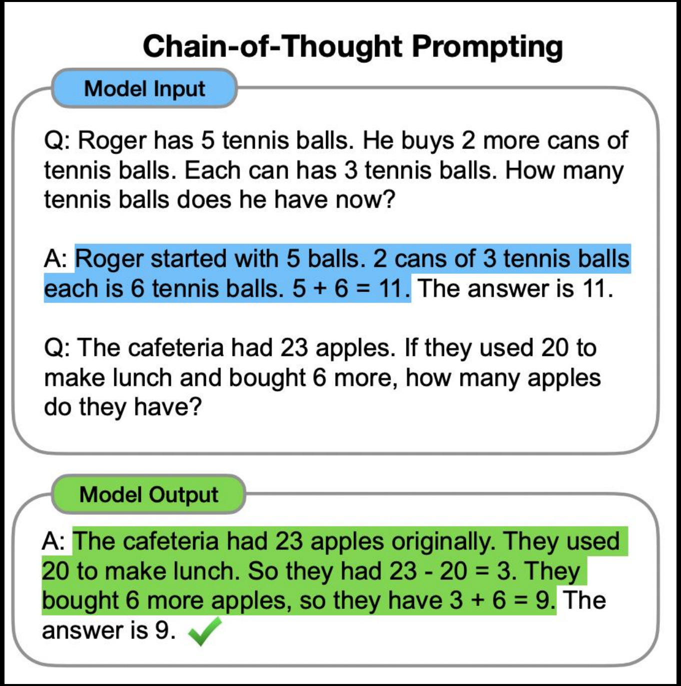
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

CHAIN-OF-THOUGHT PROMPTING

*Il **chain-of-thought (CoT) prompting** is a technique that pushes the model to spell out a sequence of intermediate reasoning steps (step-by-step) before the final answer, improving performance on arithmetic, logical, and commonsense reasoning tasks; it can be induced by showing a few examples of reasoning chains or, in some cases, with zero-shot instructions like “let’s think step by step”.*



Chain-of-Thought Prompting

Model Input

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

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

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

Model Output

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

OTHER TECHNIQUES

- **Prompt Chaining:** break a complex task into subtasks, building a sequence of prompts where each output feeds the next, for greater control and accuracy.
- **Self-Consistency:** generate multiple reasoning paths (e.g., with few-shot CoT), then choose the final answer that is most consistent among them.
- **Least-to-Most Prompting:** start from simple subproblems and build the solution up to more complex ones, leveraging previous solutions to reach the final goal.
- For a complete guide: promptingguide.ai/it