Introduction to Reinforcement Learning

A mini course @ HCMUS, Vietnam

Lectures 7-9

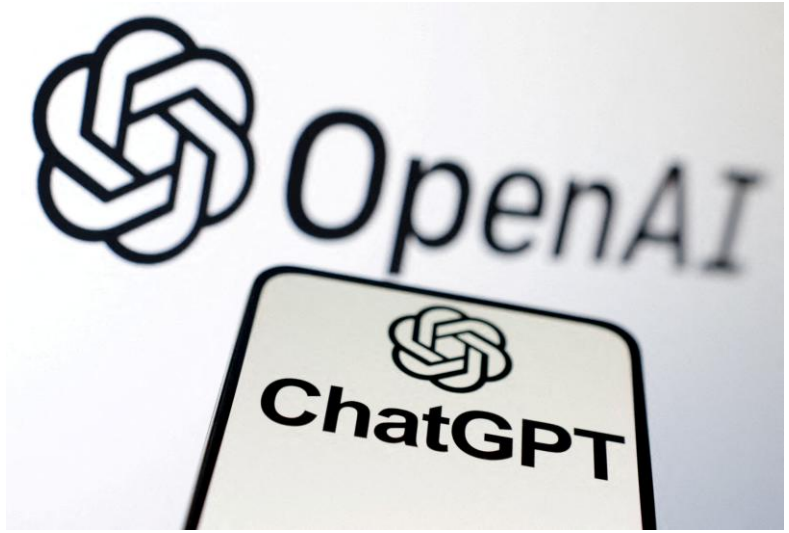
Long Tran-Thanh

long.tran-thanh@warwick.ac.uk

University of Warwick, UK

RLHF + LLM Agents

A Brave New World (of AI)



https://medium.com/enrique-dans/google-launches-gemini-but-again-fails-to-impress-anyone-a5a9bf7dd73d

100 million users after 2 months after launch - fastest-growing consumer application in history

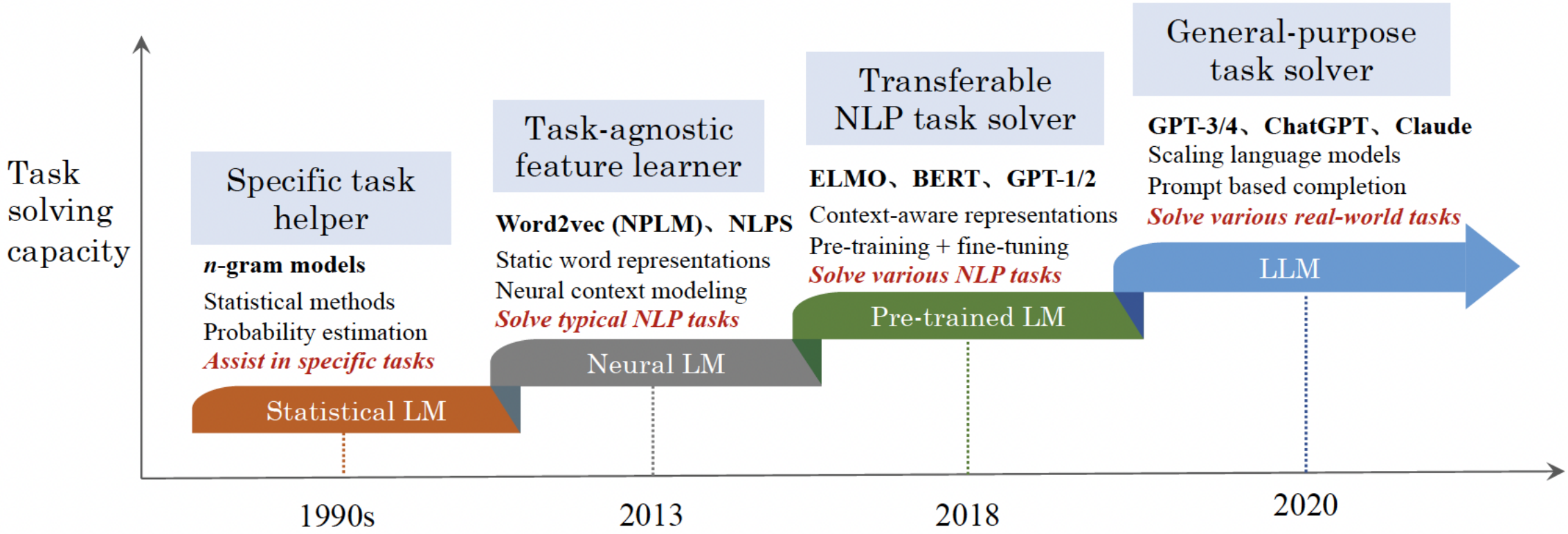
Disruptive in many areas of our lives:

- First book completely written by ChatGPT
- Solving maths problems
- Summarising large volume of text
- Translator

Newer heroes: Mistral, DeepSeek, Manus,…

The Evolution of LLMs

Large Language Model (LLM): large deep neural network (e.g., GPT-4 has 1.7 trillion parameters), pre-trained on very large data



Zhao *et al.* (2023): https://arxiv.org/pdf/2303.18223.pdf

Application 1: LLMs in Education

Image source: ChatGPT in education | Turing College

- Can pass many university exams with > 90% success rate
- Can write many Python/Matlab codes
- Can produce circuits diagrams
- Write philosophical essays

Application 2: LLMs as Optimiser

Google DeepMind

LARGE LANGUAGE MODELS AS OPTIMIZERS

Chengrun Yang* **Xuezhi Wang** **Yifeng Lu** **Hanxiao Liu**
Quoc V. Le **Denny Zhou** **Xinyun Chen***
Google DeepMind * Equal contribution

ABSTRACT

Optimization is ubiquitous. While derivative-based algorithms have been powerful tools for various problems, the absence of gradient imposes challenges on many real-world applications. In this work, we propose Optimization by PROmpting (OPRO), a simple and effective approach to leverage large language models (LLMs) as optimizers, where the optimization task is described in natural language. In each optimization step, the LLM generates new solutions from the prompt that contains previously generated solutions with their values, then the new solutions are evaluated and added to the prompt for the next optimization step. We first showcase OPRO on linear regression and traveling salesman problems, then move on to prompt optimization where the goal is to find instructions that maximize the task accuracy. With a variety of LLMs, we demonstrate that the best prompts optimized by OPRO outperform human-designed prompts by up to 8% on GSM8K, and by up to 50% on Big-Bench Hard tasks. Code at `https://github.com/google-deepmind/opro`.

Prompt based search mechanism

Optimisation step:

- LLM generates new solutions from the prompt that contains previously generated solutions with their values
- new solutions are evaluated and added to the prompt for the next optimization step.

Application 3: LLMs as Virtual Assistant

https://www.searchunify.com/blog/introducing-searchunify-virtual-assistant-suva-an-llm-powered-chatbot-for-proactive-support/

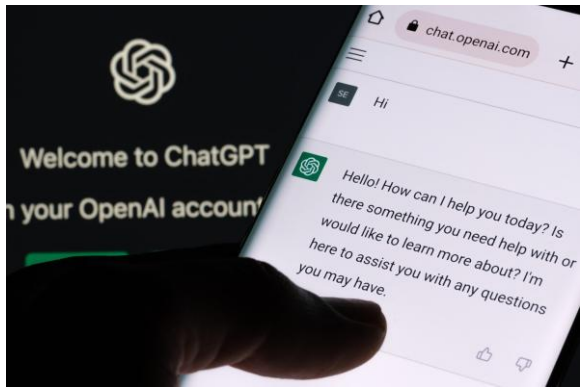
High level personalisation via prompt queries:

- Personal health assistant
- Customer support chatbot
- HR assistant

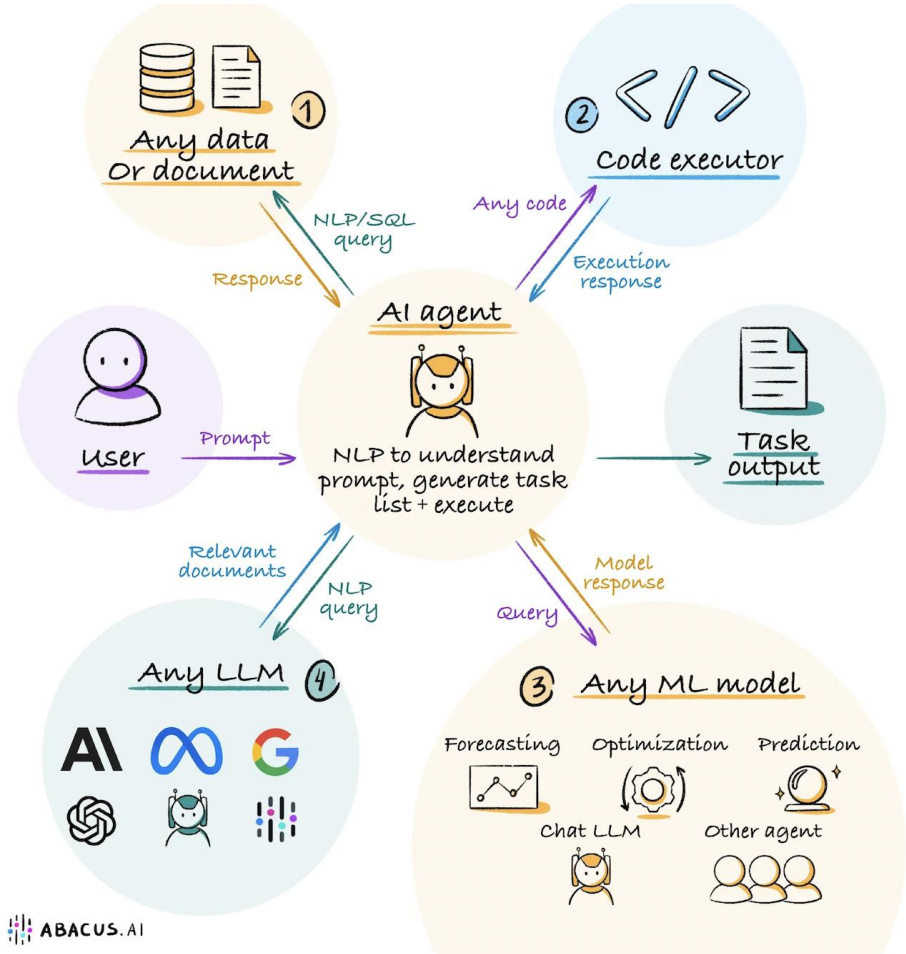
LLMs as Agents

From prompt engineering….

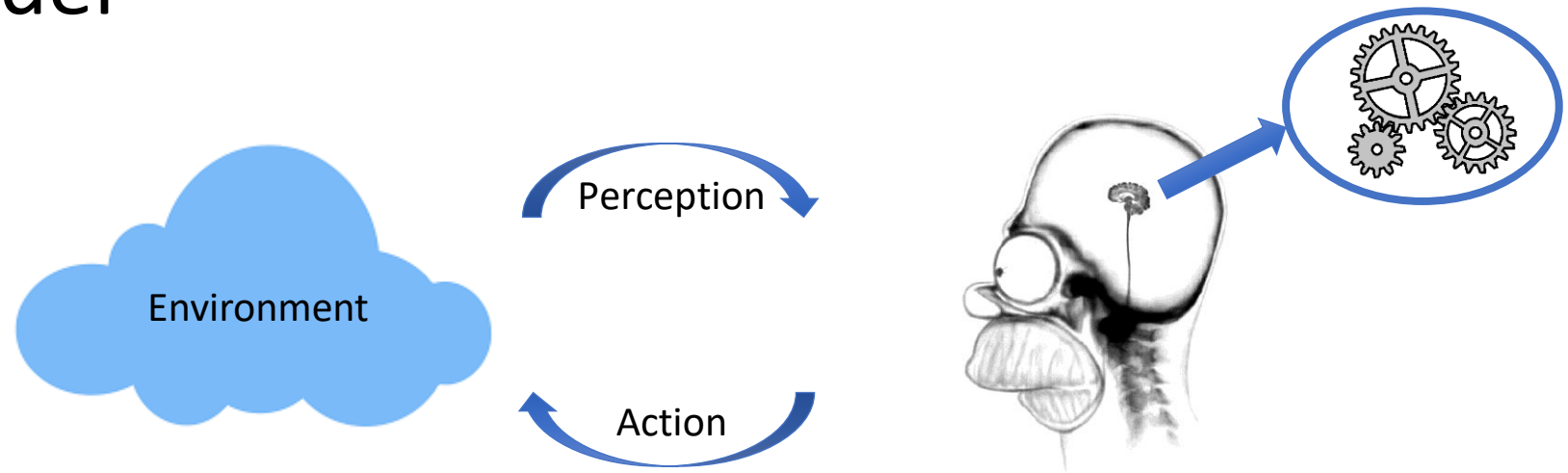
metamorworks/Getty Images

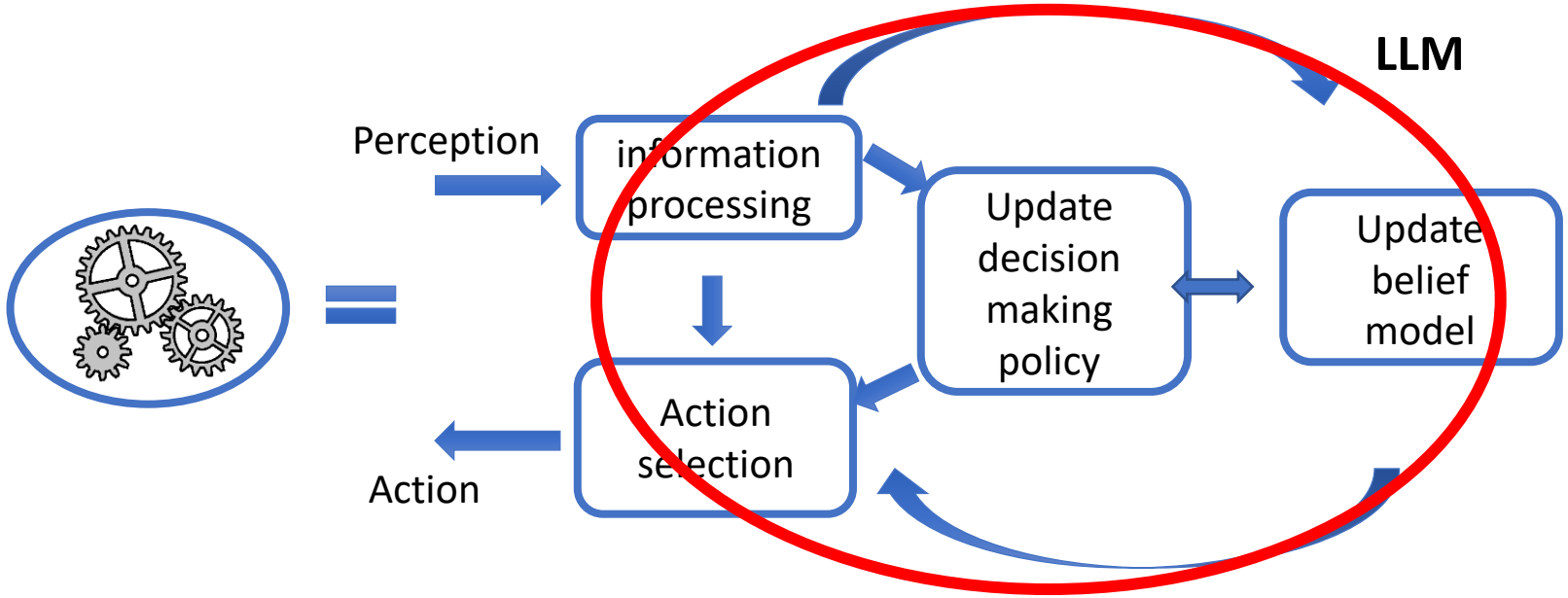


to autonomous LLM agents

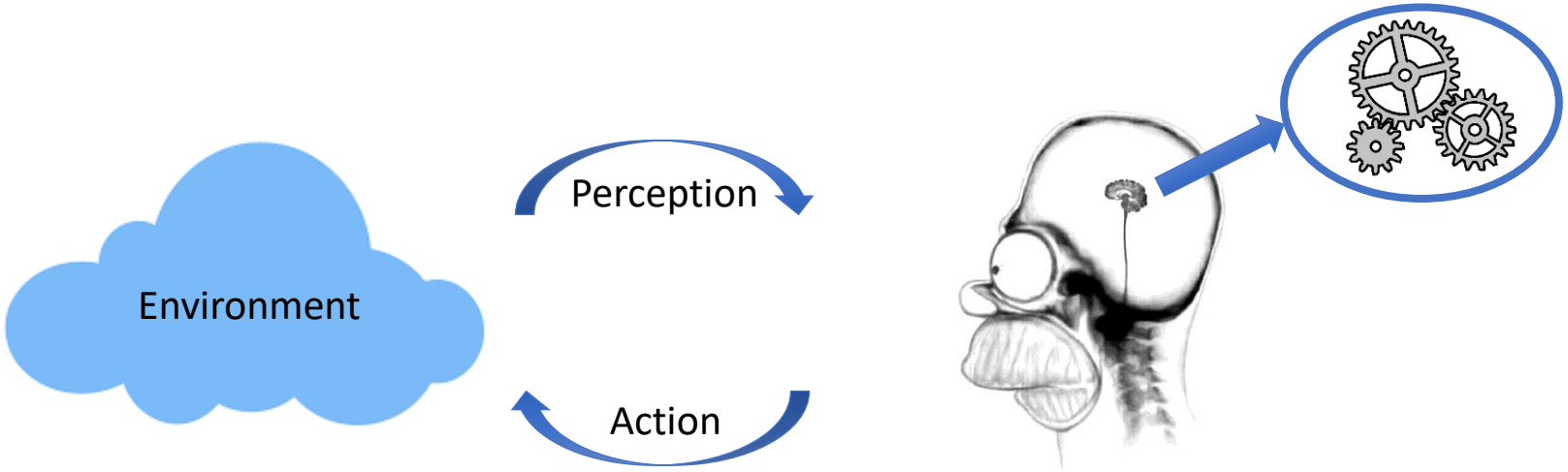


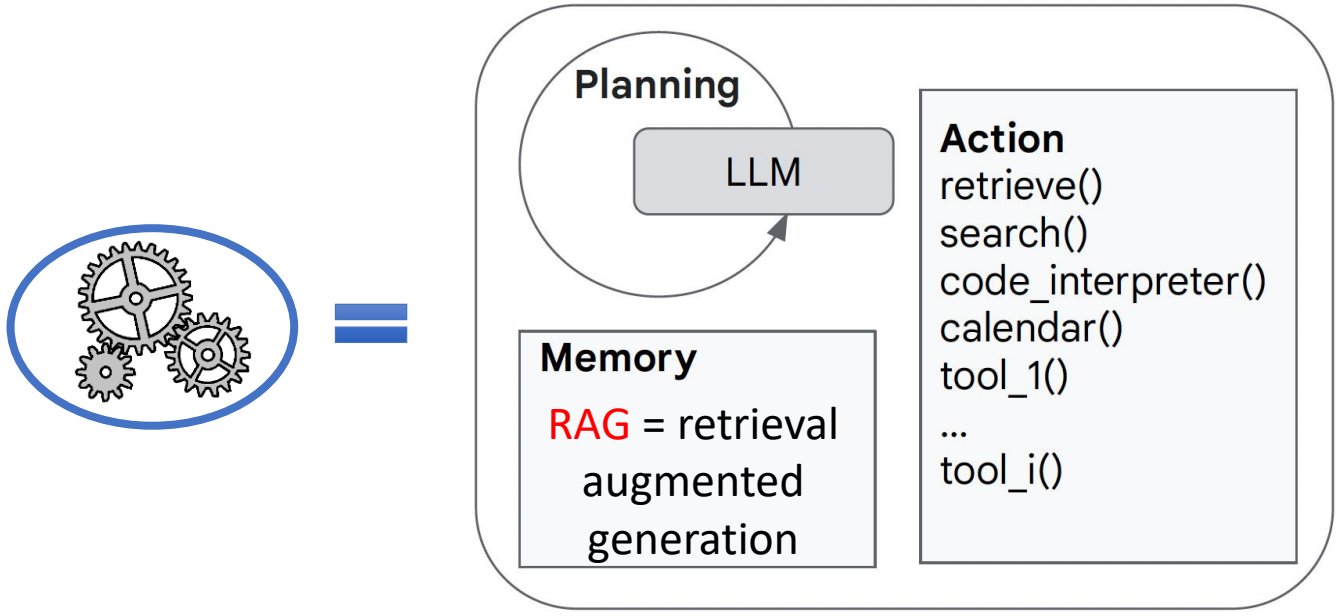
Agent Model


Environment
Perception
Action


LLM
Perception
information processing
Update decision making policy
Update belief model
Action selection
Action

LLM Agent Model

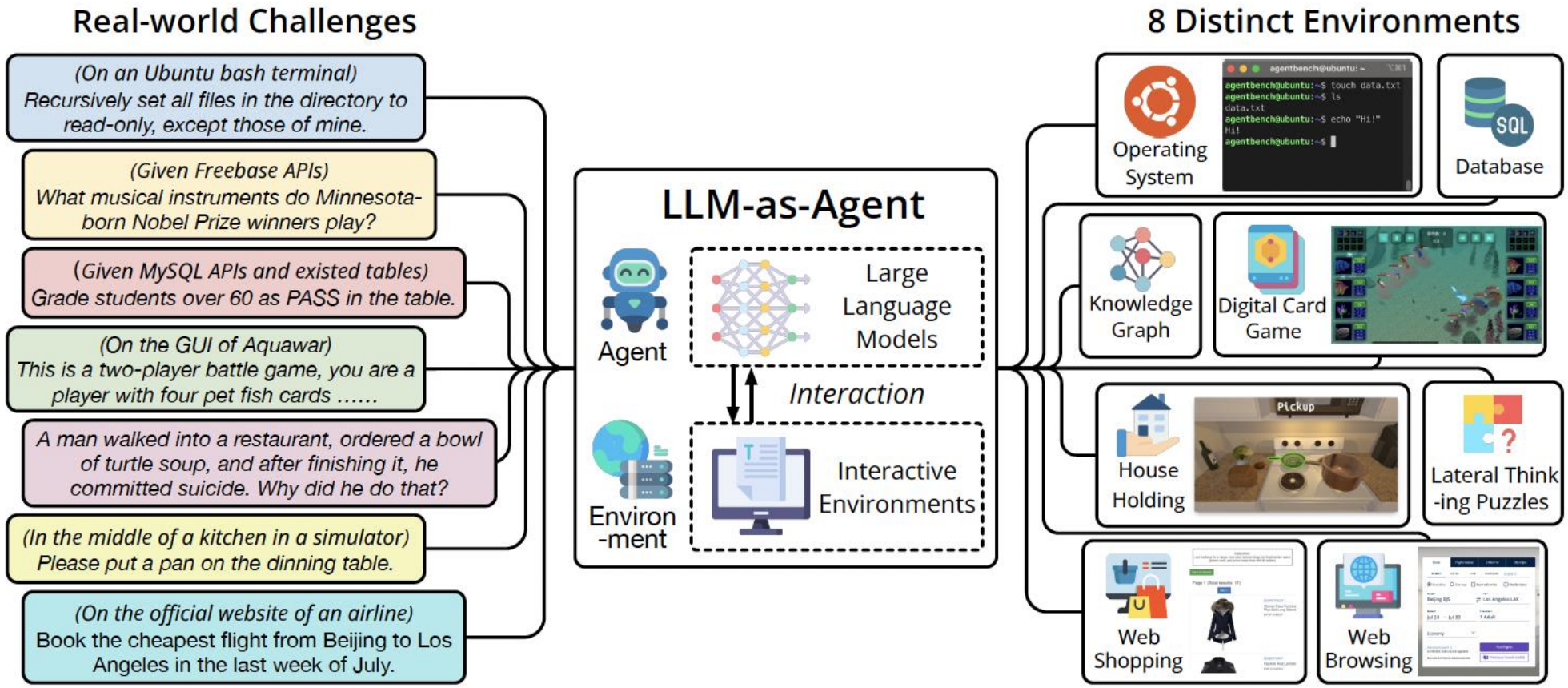




(Courtesy of Nora Kassner)

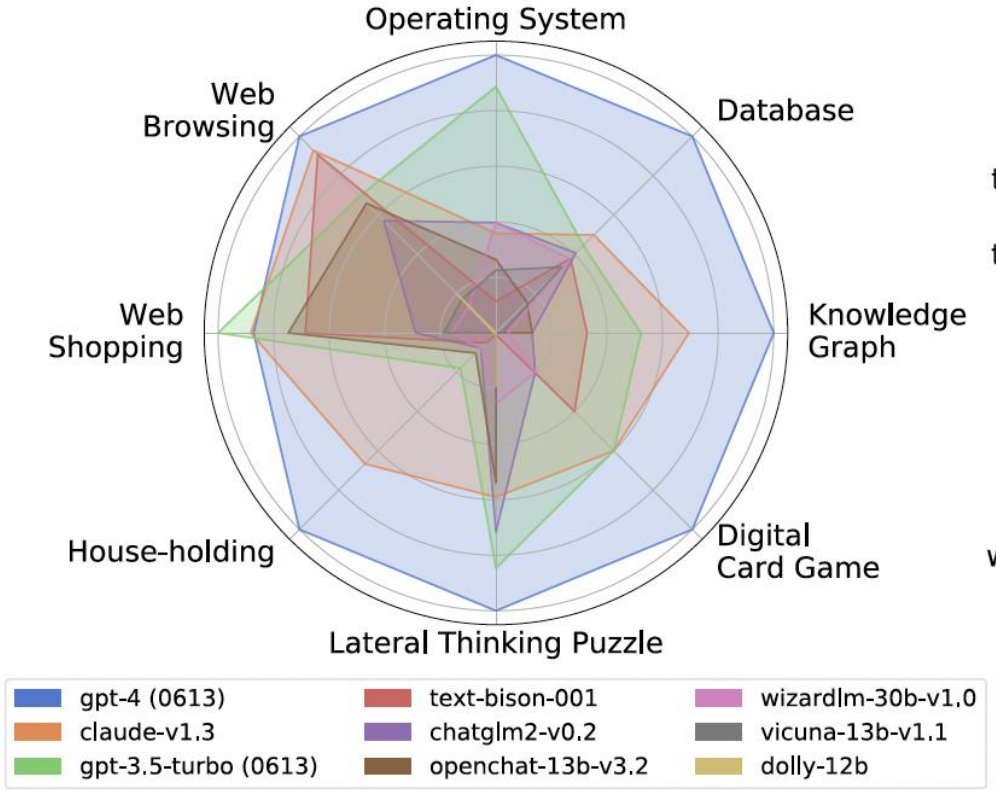
LLM Agent Benchmark Frameworks

AgentBench: Liu *et al.* (2023)

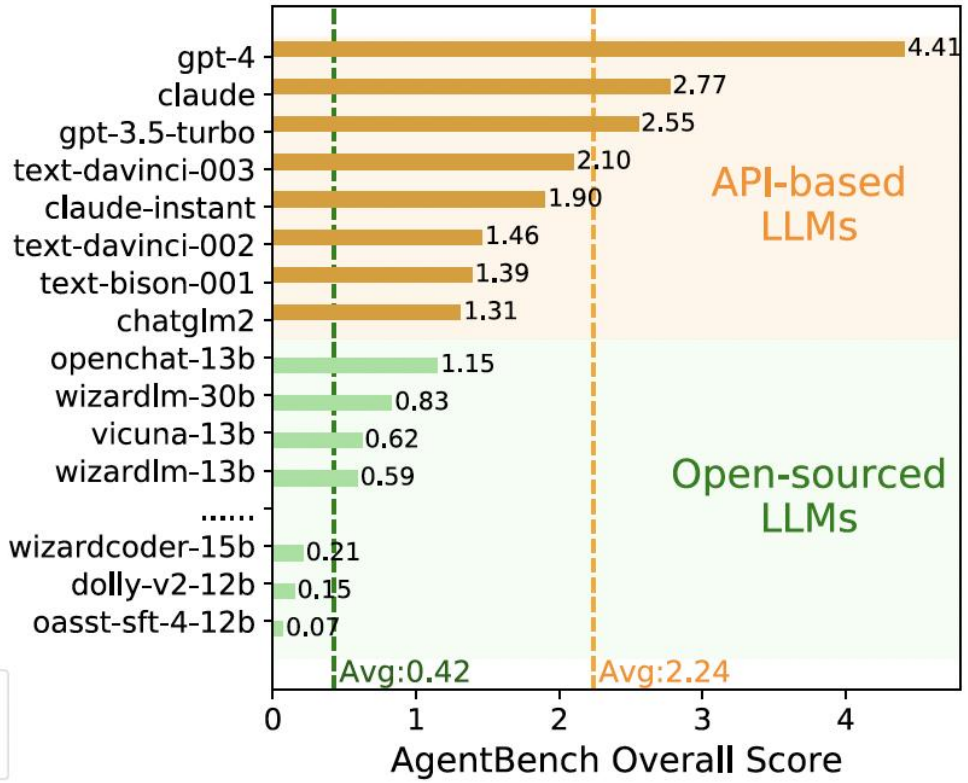


LLM Agent Benchmark Frameworks (cont'd)

AgentBench: Liu *et al.* (2023)



(a) Typical LLMs' AgentBench performance (relative) against the best in each environment.



(b) Overall scores of AgentBench across 8 environ -ments. Dashed lines for two LLM types' average.

The 2025 AI Agent Ecosystem v2

AI agents will become the dominant entities using the internet, apps, and enterprise software, disrupting established business models.

Ecosystem Layer:

Agent Marketplaces:

-Foundational Models (OpenAI GPT as a precursor)
-Enterprise (Agent.ai by Hubspot. Salesforce AgentForce)
-Big Tech (Amazon, Google, Microsoft, Meta, Apple)
-Startups: (MulitOn agent leaderboard)

Application Layer:

Agent Apps:

-Largest sector, thousands of companies to emerge
-Multimodal: input/sensors, knowledge data, output
-No-code agents. future: dynamically generated

Connect to LLMs

Agent Platforms:

Enable developers to build agents:
(MultiOn/Please, Adept, CrewAI, Lyzr, LangGraph)

Management Layer:

Agent Permissions/Security:

-Agent authentication (KYA)
-Tiered credentials
-Agent capabilities

Management:

-Orchestration: observability (AgentOps), compliance, swarm management
-Arbitration: which agent gets priority in network
-Payments: agent to tech (Skyfire), agent to human (Payman), human to agent
-Improvement: metacognition, simulation, reflection, eval, self-healing

Data Layer:

Exclusive/Private Data:

-RAG/Access to enterprise, gov, personal data, agent data

Open Data:

-Public data, (Dendrite)
-Data Providers, scraping services

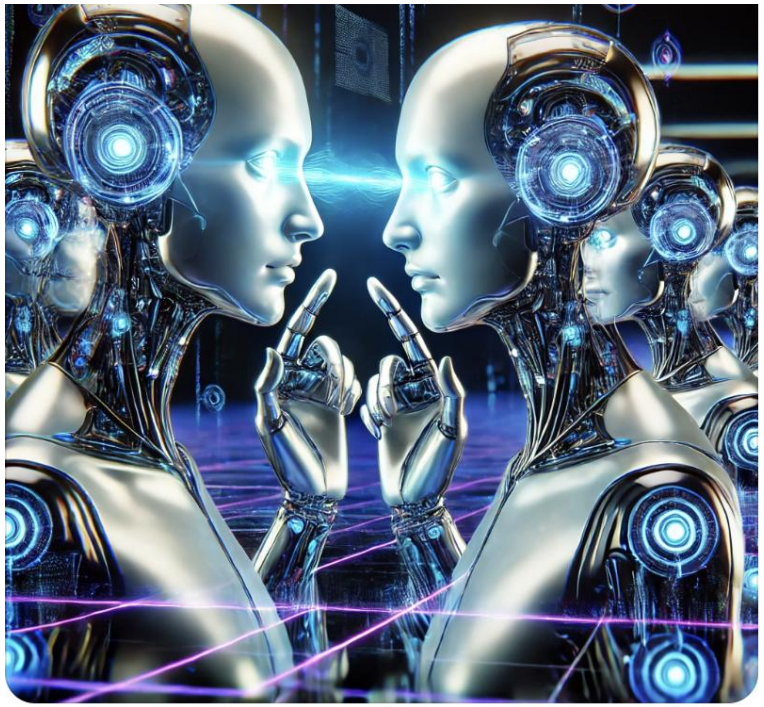
Unified APIs:

- Fast info or transactions; no imitating click path

BLITZSCALING VENTURES

By Jeremiah Owyang | Sep 2024. Interviews with: Chris Yeh, Div Garg, Omar Shaya, Sarah Allali, Scott Johnson, Jeff Abbott, Parth, Craig DeWitt, Amir Sarhangi, Mikhil Raja, Alex Reibman, Christopher Kauffman, David Schatsky, Chris Saad, Charles Maddock, Ori Neidich, Siva Surendira, Matt Schlicht, Ben Parr, João Moura, Sharad Sundararaja, Satya Nitta, Brooke Hopkins, and more.

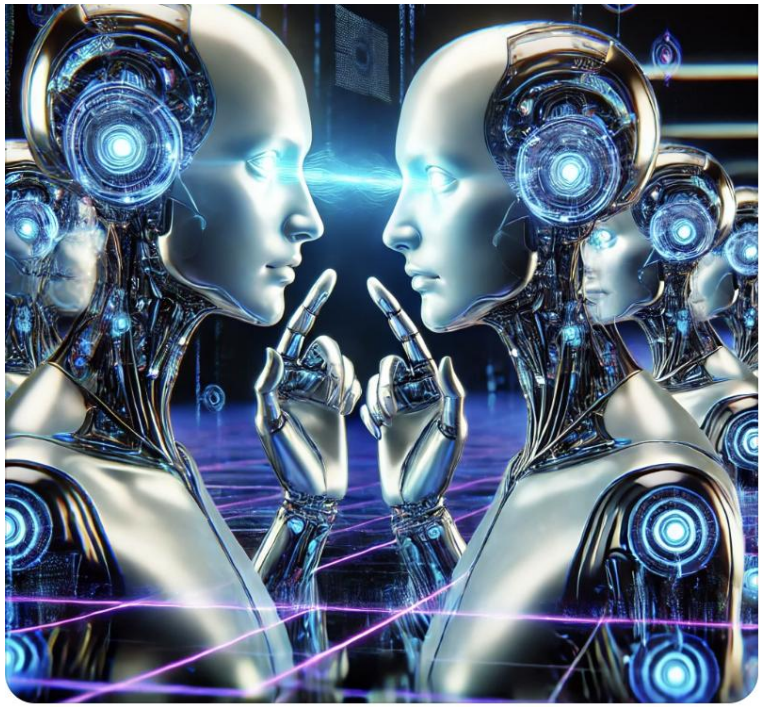
LLM Agents in the Wild 1



Exploit each other:

- Deceive their human users/fellow agents (e.g., to steal their money) (Scheurer *et al.*, ICLR workshop'24)
- Without being caught
  (Wang *et al.*, ICML workshop'24)

LLM Agents in the Wild 2



Selfish behaviour causes system collapse:

- Piatti *et al.*, Arxiv'24 (e.g., financial system's collapse)

Need for Aligning AI Agent Behaviour to Human Values

The Alignment Problem

Prompt: *"Explain quantum computing simply."*

✅ *"Quantum computing uses qubits to process information via superposition and entanglement, enabling faster calculations for certain problems."*

🟡 *"Quantum computers are like super-fast computers but use physics stuff. They're not like regular PCs."*

❌ *"Quantum is a rock band from the 1970s. Computing is math."*

Target: chose the most human-like/ethical/fair/etc answer

Enter the RLHF

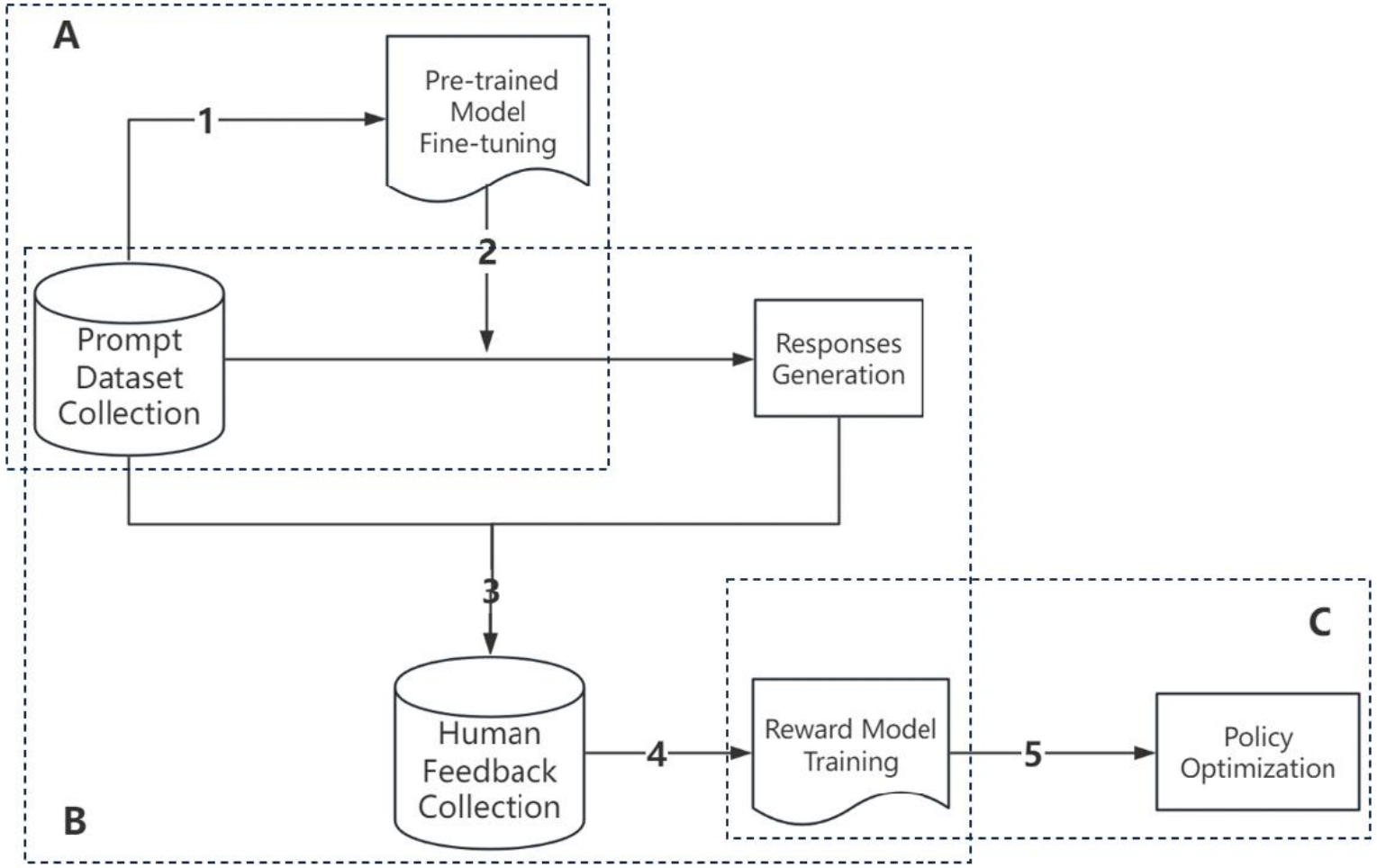
RLHF – Reinforcement Learning with Human Feedback

High level idea:

- Dialog with LLM = RL
- (prompt, answer) = (state, action) // Why?
- Policy = LLM
- Alignment with human behaviour = train on data of human feedback

- Learning optimal policy = training LLM with alignment

Enter the RLHF



Key components:

- A: Supervised fine-tuning
- B: reward model training
- C: RL policy optimization

Steps in RLHF

A: Supervised fine-tuning

- Get a (pre-trained) LLM model (e.g., GPT, Llama, etc)
- Fine-tune on a domain specific dataset

B: Learning the reward model

✅ *"Quantum computing uses qubits to process information via superposition and entanglement, enabling faster calculations for certain problems."*

🟡 *"Quantum computers are like super-fast computers but use physics stuff. They're not like regular PCs."*

❌ *"Quantum is a rock band from the 1970s. Computing is math."*

- Idea: to learn the human users' preferences when choosing the most appropriate answers

Reward Model (the Human Feedback part)

B: Learning the reward model

✅ *"Quantum computing uses qubits to process information via superposition and entanglement, enabling faster calculations for certain problems."*

🟡 *"Quantum computers are like super-fast computers but use physics stuff. They're not like regular PCs."*

❌ *"Quantum is a rock band from the 1970s. Computing is math."*

- High level mathematical model: for each prompt p, assume there's an underlying function r (reward function) such that for 2 answers x_1, x_2 :

$$r(p, x_1) > r(p, x_2) \Leftrightarrow \text{pref}(x_1) > \text{pref}(x_2)$$

- How to learn this r function?
- Step 1: get data samples via human feedback: $D = (p_i, x_i^{chosen}, x_i^{rejected})$
- Step 2: train on this dataset. Typical training methos:
  - Smaller Pre-trained LLMs (usually 10x smaller ones, e.g., OpenAI uses GPT-3)
  - Contrastive learning (Claude from Anthropic)

Fine-tuning the LLM with RL

C: RL policy optimization

- Recall policy = LLM
- Use PPO (or GRPO) to optimise
- Compare with base LLM to anchor the optimisation process (so the optimised one will not deviate too much)
- This is done by incorporating a penalty component into the final reward value