

AI Agents for Science

Lecture 2, October 1: Frontiers of Language Models

Instructor: Ian Foster

TA: Alok Kamatar



Crescat scientia; vita excolatur

CMSC 35370 -- <https://agents4science.github.io>
<https://canvas.uchicago.edu/courses/67079>

Course Syllabus

[Jump to Today](#) Edit

An agentic **Scientific Discovery Platform** (SDP) is an integrated environment that combines reasoning-capable AI with scientific and engineering resources—such as literature collections, simulation codes, experimental platforms, and knowledge bases—to accelerate the pace of discovery. Recent advances in large language models (LLMs) and related technologies make it possible to build such platforms that can automate key aspects of scientific work: synthesizing information from the literature, generating and prioritizing hypotheses, designing and executing protocols, running simulations or experiments, and interpreting results.

To explore and advance these ideas, we are offering this intensive class in Fall 2025. The course is targeted at graduate students and advanced undergraduates eager to engage with this emerging field, but we also welcome postdocs, scientists, and faculty who are prepared to participate actively. The class emphasizes “learning by doing”: instructors will provide infrastructure, working examples, and state-of-the-art perspectives on agentic AI and scientific discovery, while students will design, build, and apply their own SDPs to tackle domain-relevant problems.

See <https://agents4science.github.io/>  for more information, including a draft curriculum.

See [Accessing the Argonne inference service](#) for details on how to access models at Argonne.

Curriculum and Materials

9/29: [Module 1](#) has slides and readings

10/1: [Module 2](#) has readings

Lecture 1 reading

☰ ▼ Module 1

☰  [Lecture_1.pdf](#)

☰  [EXPLORING LARGE LANGUAGE MODEL BASED INTELLIGENT AGENTS.pdf](#)

☰  [Artificial intelligence and illusions of understanding in scientific research.pdf](#)

☰  [The Shift from Models to Compound AI Systems](#)

EXPLORING LARGE LANGUAGE MODEL BASED INTELLIGENT AGENTS: DEFINITIONS, METHODS, AND PROSPECTS

“We define AI as the study of agents that receive percepts from the environment and perform actions.”

— *Artificial Intelligence: A Modern Approach*,
Stuart Russell &
Peter Norvig (2003).

Yuheng Cheng¹ * Ceyao Zhang¹ * Zhengwen Zhang¹ *
Xiangrui Meng¹ Sirui Hong² Wenhao Li¹ Zihao Wang³ Zekai Wang⁴
Feng Yin¹ Junhua Zhao¹ † Xiuqiang He⁵

¹The Chinese University of Hong Kong, Shenzhen

²DeepWisdom ³Peking University ⁴Yantu.ai ⁵FiT, Tencent

ABSTRACT

Intelligent agents stand out as a potential path toward artificial general intelligence (AGI). Thus, researchers have dedicated significant effort to diverse implementations for them.

Benefiting from recent progress in large language models (LLMs), LLM-based agents that use universal natural language as an interface exhibit robust generalization capabilities across various applications – from serving as autonomous general-purpose task assistants to applications in coding, social, and economic domains, LLM-based agents offer extensive exploration opportunities.

This paper surveys current research to provide an in-depth overview of LLM-based intelligent agents within single-agent and multi-agent systems. It covers their definitions, research frameworks, and foundational components such as their composition, cognitive and planning methods, tool utilization, and responses to environmental feedback. We also delve into the mechanisms of deploying LLM-based agents in multi-agent systems, including multi-role collaboration, message passing, and strategies to alleviate communication issues between agents. The discussions also shed light on popular datasets and application scenarios. We conclude by envisioning prospects for LLM-based agents, considering the evolving landscape of AI and natural language processing.

Jan 2024

<https://arxiv.org/html/2401.03428v1>

LLM = Large Language Model
LMM = Large Multi-modal Model

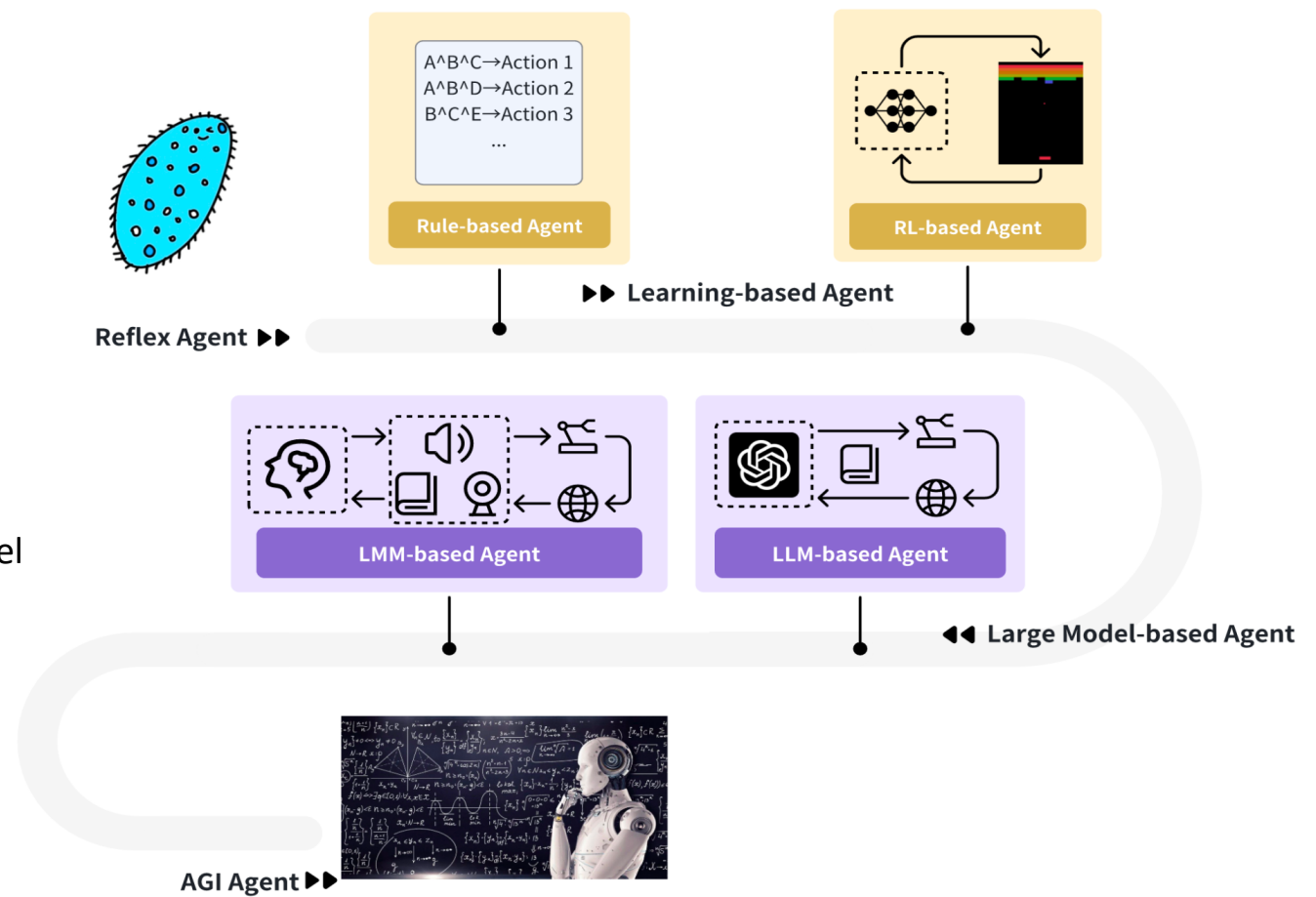


Figure 1: Roadmap of Intelligent Agents Development

<https://arxiv.org/html/2401.03428v1>

ICL=In-context learning

Table 1: List of LLM-based Single-Agent System.

ID	Name	Field	Training Environment		Data	Evaluation	Modality	Feedback	Tool	Planning	Review
1	Out of One[29]	Sociology	No	Text	No	with human	Text	None	None	None	None
2	Horton [30]	Sociology	No	Text	No	with human	Text	None	None	None	None
3	Park et al. [31]	Sociology	No	Text	No	with human	Text	None	None	None	None
4	Social AI School[32]	Sociology	Yes	Simulation environment	Yes	Comparison available	Image to text	self-feedback	None	None	RL
5	S ³ [33]	Sociology	No	Text	No	with humans	Text	environmental	None	None	None
6	Li et al. [34]	Sociology	No	Text	No	None	Text	None	Yes	None	None
7	Li et al. [35]	Sociology	No	Text	None	with human	Text	None	None	None	None
8	ChatLaw[36]	Law	Yes	Text	Yes	with models	Text	None	None	None	None
9	[37]	Law	Yes	Text	Yes	with models	Text	None	None	None	None
10	ChemCrow[38]	Chemistry	Yes	Experimental environment	Yes	LLM and expert assessment	Multimodal	self-feedback	Yes	ICL	ICL
11	ChatMOF[39]	Material Science	No	Experimental environment	Yes	success rate available	Multimodal	self-feedback	Yes	ICL	ICL
12	Mathagent[40]	Mathematics	Yes	Text	Yes	None	Text	None	None	None	None
13	[41]	Mathematics	Yes	Text	Yes	with models	Text	None	None	None	None
14	IGLU[42]	Collaborative	Yes	IGLU competition	No	with models	Text	humanfeedback	None	None	RL
15	GPTEngineer[43]	Code	No	Code Environment	No	None	Text	self-feedback	Yes	ICL	ICL
16	SmolModels[44]	Code	No	Code Environment	No	None	Text	self-feedback	Yes	ICL	ICL
17	DemoGPT[45]	Code	Support	Code Environment	Support	None	Text	self-feedback	Yes	ICL	ICL
18	IELLM[46]	Industrial Engineering	No	None	None	None	None	No	No	No	No
19	DialogueShaping[47]	Game	No	Game environment	No	with agents	Text	environmental	No	ICL	RL
20	DECKARD[48]	Game	No	Game environment	No	ablation study	Text	environmental	No	Multi-stage	RL
21	TaPA[49]	Embodied agents	No	Visual perception environment	No	with models	Visual	environmental	No	ICL	ICL
22	Voyager[50]	Game	No	Minecraft game	No	with models	Text	environmental	No	ICL	ICL
23	GITM[51]	Game	No	Minecraft game	No	with models	Text	environmental	No	ICL	ICL
24	LLM4RL[52]	Embodied agents	Yes	Different embodied environments	No	with baseline	Multi	environmental	No	None	RL
25	PET[53]	Embodied agents	Yes	AlfWorld interactive environment	Yes	with models	Text	self-feedback	No	Multi-stage	None
26	REMEMBERER[54]	None	No	Text	No	with models	Text	No	Yes	None	RL
27	UnifiedAgent[55]	Embodiedagents	Yes	Robot simulation environment	Yes	ablation study	Multi	None	None	None	RL
28	SayCan[56]	Embodied agents	No	Robot simulation environment	Yes	ablation study	Multi	No	None	External Method	RL
29	Allegion[57]	Universal	No	No	No	None	Multi	None	Yes	ICL	ICL
30	AGiXT[58]	Universal	No	No	No	None	Multi-modal	None	Yes	ICL	ICL
31	BabyAGI[59]	Embodied agents	No	No	No	None	Multi	None	Yes	ICL	ICL
32	LoopGPT[60]	Universal	None	None	None	None	Multi-modal		Yes	ICL	ICL
33	GPTresearcher[61]	Research	None	None	None	None	Multi	No	Yes	External Method	None
34	SuperAGI[62]	Universal	None	None	None	None	Multi-modal	None	Yes	ICL	ICL

Table 2: List of LLM-based Multi-Agent System.

ID	Name	Field	Training Environment		Data	Evaluation	Modality	Tool	Planning Review		
1	Generative Agents [63]	Sociology	No	GameText	No	None	Text	No	ICL	reflection	Cooperative CPDE
2	Williams et al. [64]	Epidemiology research	No	Text	None	None	Text	None	None	None	Cooperative CPDE
3	Boiko et al. [65]	Chemistry	No	Experimental environment	No	None	Multi	Yes	ICL	ICL	Hierarchical DPDE
4	ChatDev [66]	Software Development	No	Software Development	No	on dataset available	Text	Yes	ICL	ICL	Cooperative DPDE
5	MetaGPT [67]	Code	Support	Code Environment	Support	with models	Multi	Support	ICL	ICL	Cooperative DPDE
6	SCG [68]	Code	No	Code Environment	No	with models	Text	Yes	ICL	ICL	Cooperative DPDE
7	GPT4IA [69]	Industrial environment	Yes	Engineering environment	No	None	Multi	Yes	ICL	ICL	Cooperative DPDE
8	Planner-Actor-Reporter [70]	Embodied environments	Support	A 2D partially observable grid-world	No	task success rate	Image plus text	No	ICL	ICL	Cooperative DPDE
9	ProAgent [71]	Universal	No	Text	No	with models	Multi	No	None	None	Hierarchical DPDE
10	SAMA [72]	Game	No	Game	No	with models	GameText	No	ICL	ICL and RL	Cooperative DPDE
11	Co-LLM-Agents [73]	Game	No	Game	No	with models	GameText	No	None	None	Cooperative DPDE

Multi-Role Coordination: Cooperative, Competitive, Mixed, and Hierarchical

Planning Type:

- Centralized Planning Decentralized Execution (CPDE)
- Decentralized Planning Decentralized Execution (DPDE).

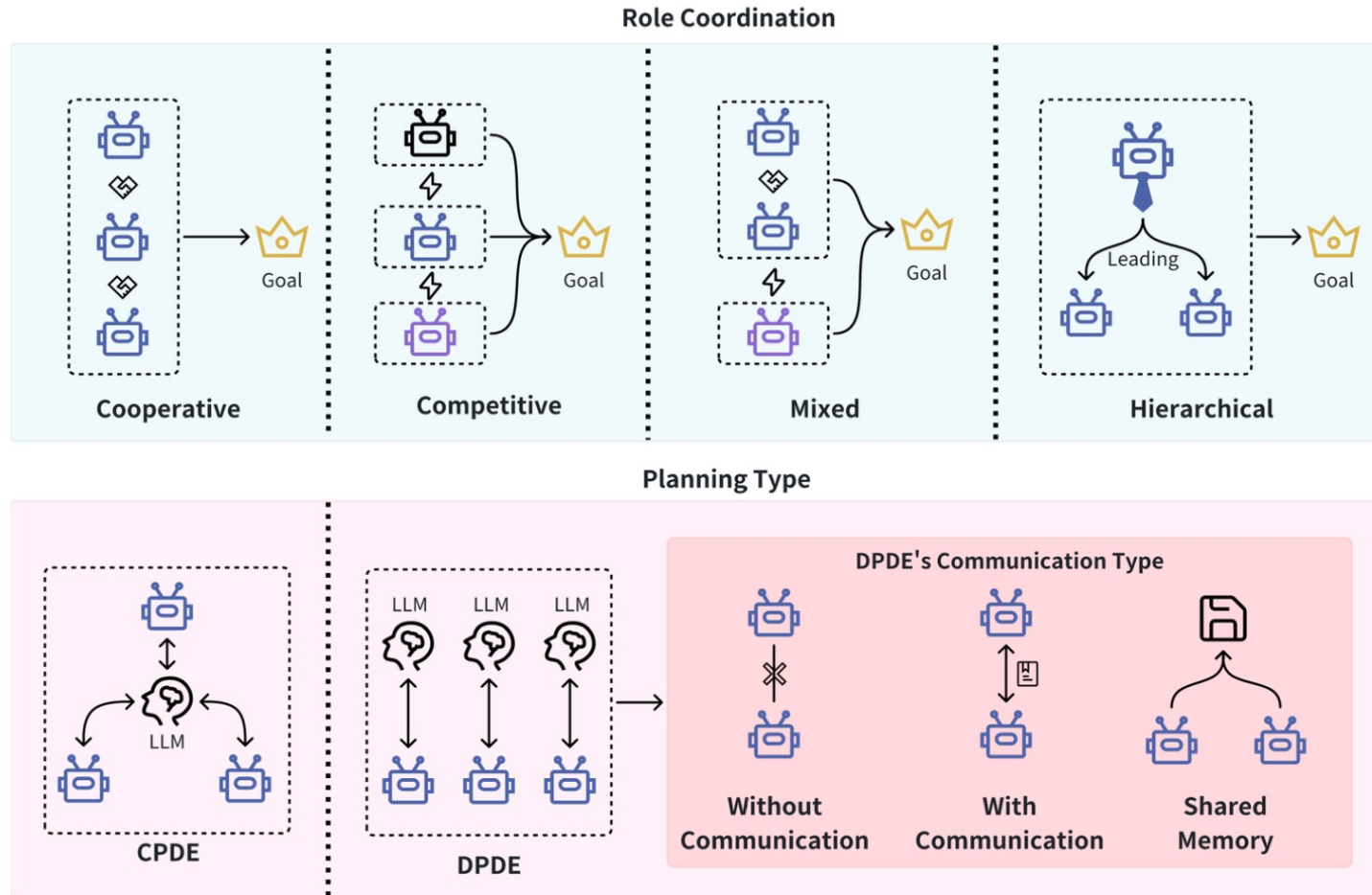


Figure 3: The Relationship between LLM-based agents

Artificial intelligence and illusions of understanding in scientific research

<https://doi.org/10.1038/s41586-024-07146-0>

Received: 31 July 2023

Accepted: 31 January 2024

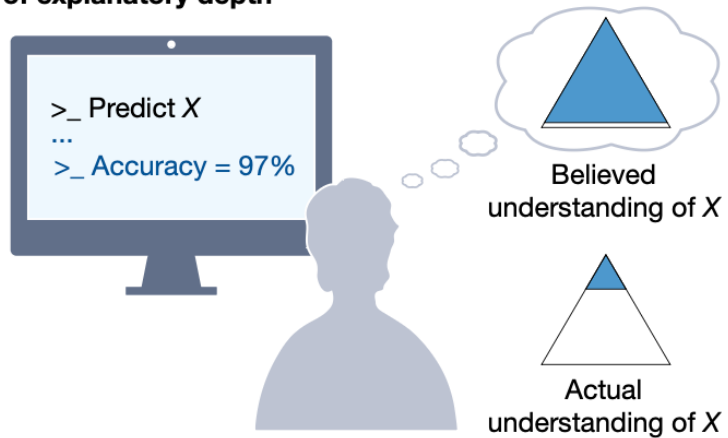
Published online: 6 March 2024

 Check for updates

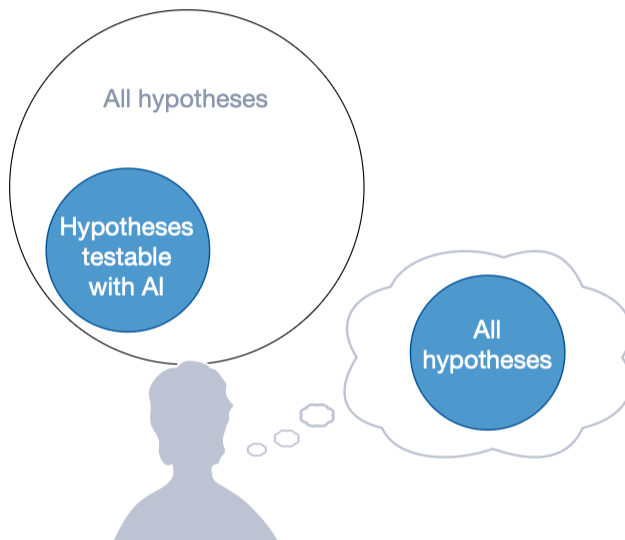
Lisa Messeri^{1,4}✉ & M. J. Crockett^{2,3,4}✉

Scientists are enthusiastically imagining ways in which artificial intelligence (AI) tools might improve research. Why are AI tools so attractive and what are the risks of implementing them across the research pipeline? Here we develop a taxonomy of scientists' visions for AI, observing that their appeal comes from promises to improve productivity and objectivity by overcoming human shortcomings. But proposed AI solutions can also exploit our cognitive limitations, making us vulnerable to illusions of understanding in which we believe we understand more about the world than we actually do. Such illusions obscure the scientific community's ability to see the formation of scientific monocultures, in which some types of methods, questions and viewpoints come to dominate alternative approaches, making science less innovative and more vulnerable to errors. The proliferation of AI tools in science risks introducing a phase of scientific enquiry in which we produce more but understand less. By analysing the appeal of these tools, we provide a framework for advancing discussions of responsible knowledge production in the age of AI.

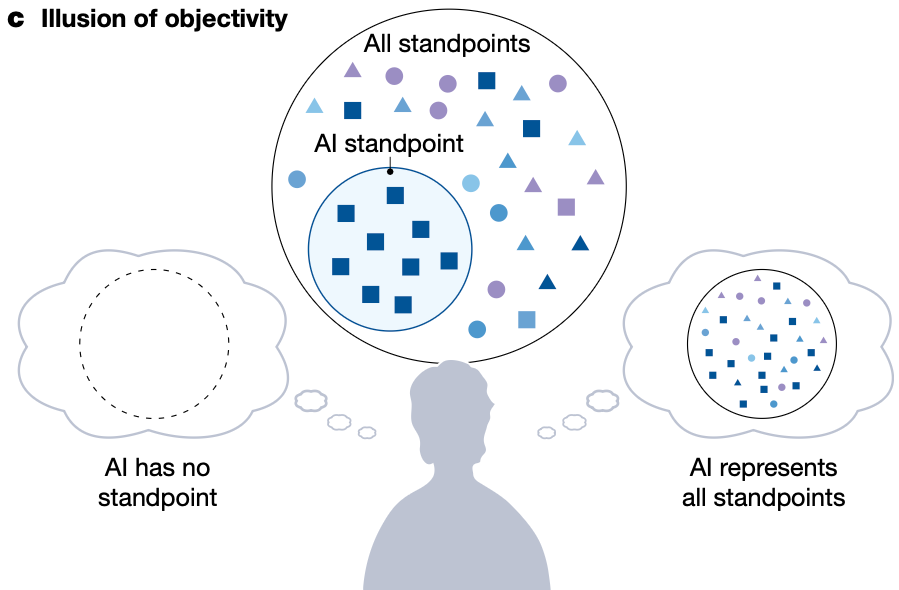
a Illusion of explanatory depth



b Illusion of exploratory breadth



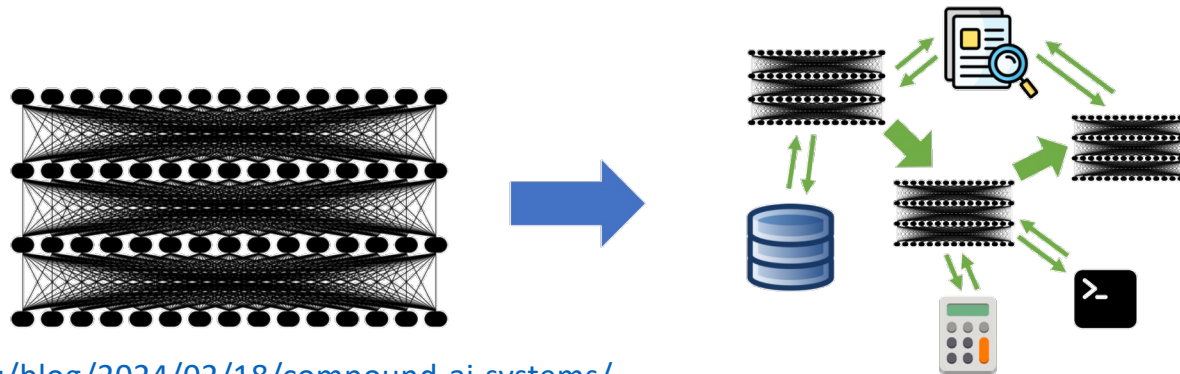
c Illusion of objectivity



The Shift from Models to Compound AI Systems

Matei Zaharia, Omar Khattab, Lingjiao Chen, Jared Quincy Davis, Heather Miller, Chris Potts, James Zou, Michael Carbin, Jonathan Frankle, Naveen Rao, Ali Ghodsi Feb 18, 2024

AI caught everyone's attention in 2023 with Large Language Models (LLMs) that can be instructed to perform general tasks, such as translation or coding, just by prompting. This naturally led to an intense focus on models as the primary ingredient in AI application development, with everyone wondering what capabilities new LLMs will bring. As more developers begin to build using LLMs, however, we believe that this focus is rapidly changing: **state-of-the-art AI results are increasingly obtained by compound systems with multiple components, not just monolithic models.**



<https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/>

Curriculum

1) Why AI agents for science?

AI agents and the sense-plan-act-learn loop. Scientific Discovery Platforms (SDPs): AI-native systems that connect reasoning models with scientific resources.

2) Frontiers of Language Models

Surveys frontier reasoning models: general-purpose LLMs (GPT, Claude), domain-specific foundation models (materials, bio, weather), and hybrids. Covers techniques for eliciting better reasoning: prompting, chain-of-thought, retrieval-augmented generation (RAG), fine-tuning, and tool-augmented reasoning.

3) Systems for Agents

Discusses architectures and frameworks for building multi-agent systems, with emphasis on inter-agent communication, orchestration, and lifecycle management.

4) Retrieval Augmented Generation (RAG) and Vector Databases

Covers how to augment reasoning models with external knowledge bases, vector search, and hybrid retrieval methods.

Lecture 2 reading

☰ ▼ Module 2

- ☰  [Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.pdf](#)
- ☰  [ReAct: Synergizing Reasoning and Acting in Language Models.pdf](#)
- ☰  [DeepSeek-R1 Incentivizing Reasoning Capability in LLMs via Reinforcement Learning.pdf](#)

Topics

- Quick LLM review
- Methods for improving LLM reasoning performance
 - Prompting
 - Chain-of-thought
 - Retrieval-augmented generation (RAG)
 - Fine-tuning
 - Tool-augmented reasoning
- Fine-tuning models
- Domain-specific LLMs: e.g., materials, biology, weather

A quick look at large language models (LLMs)

LLMs are just one thing after another:

Would → **you**

Would you → **tell**

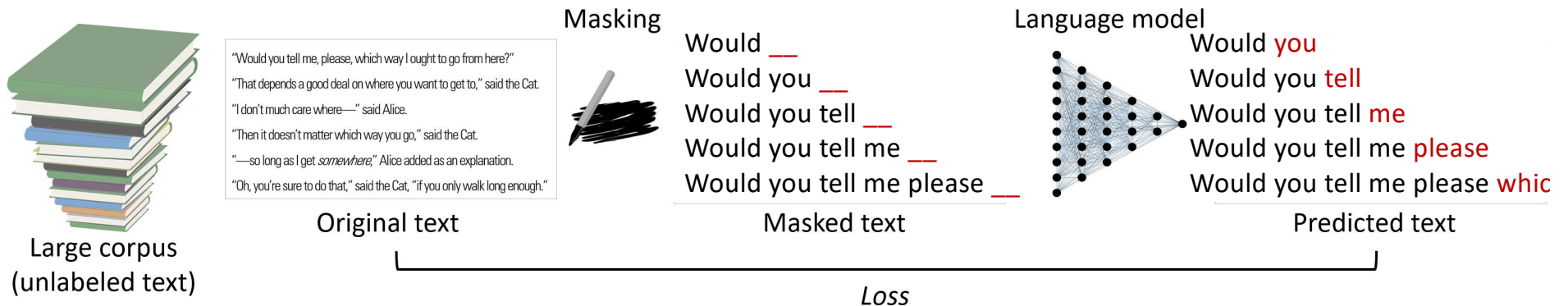
Would you tell → **me**

Would you tell me → **please**

Would you tell me please → **which**

LLMs are self-supervised - pretrained - autoregressive

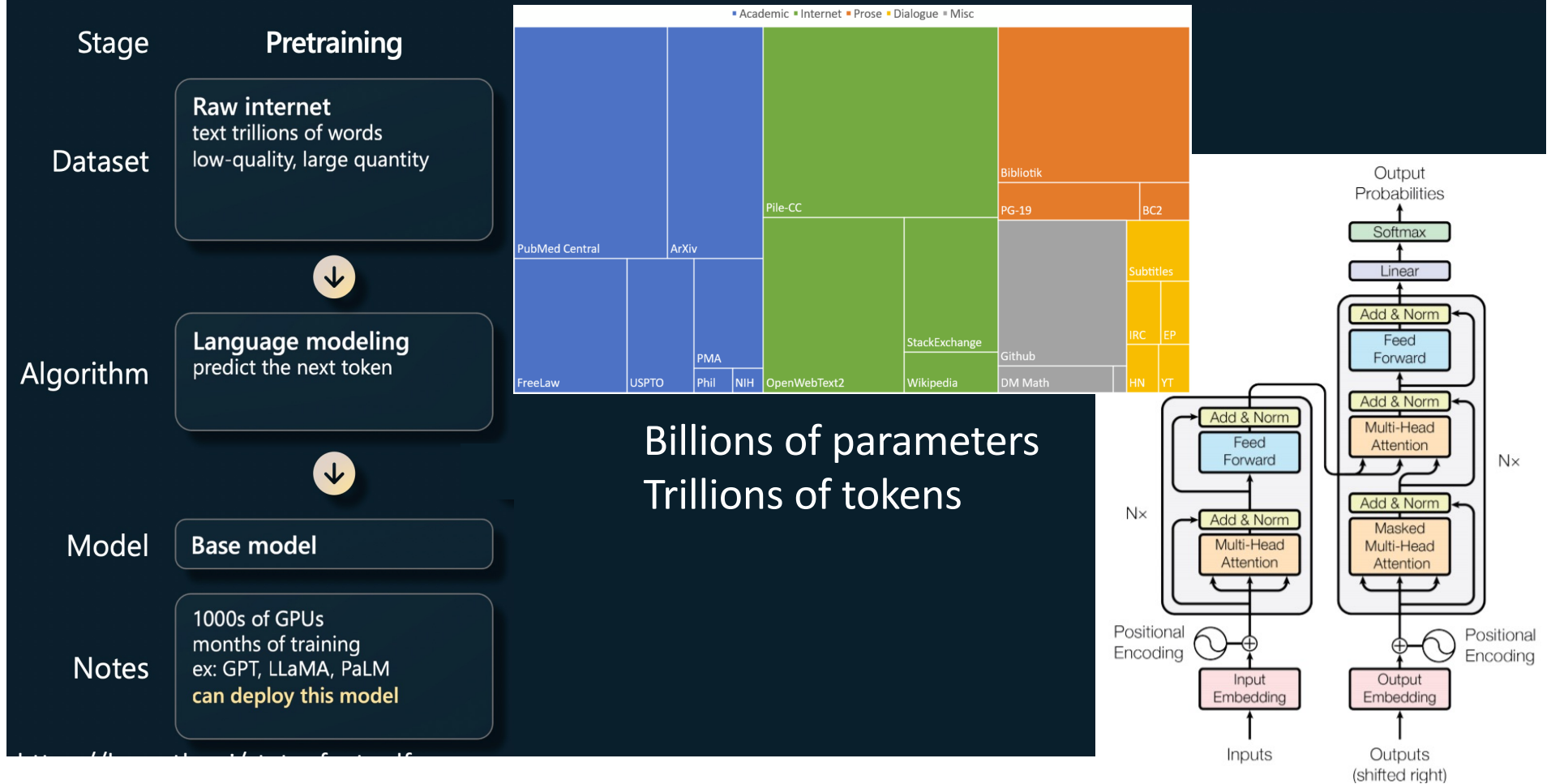
A Pretraining



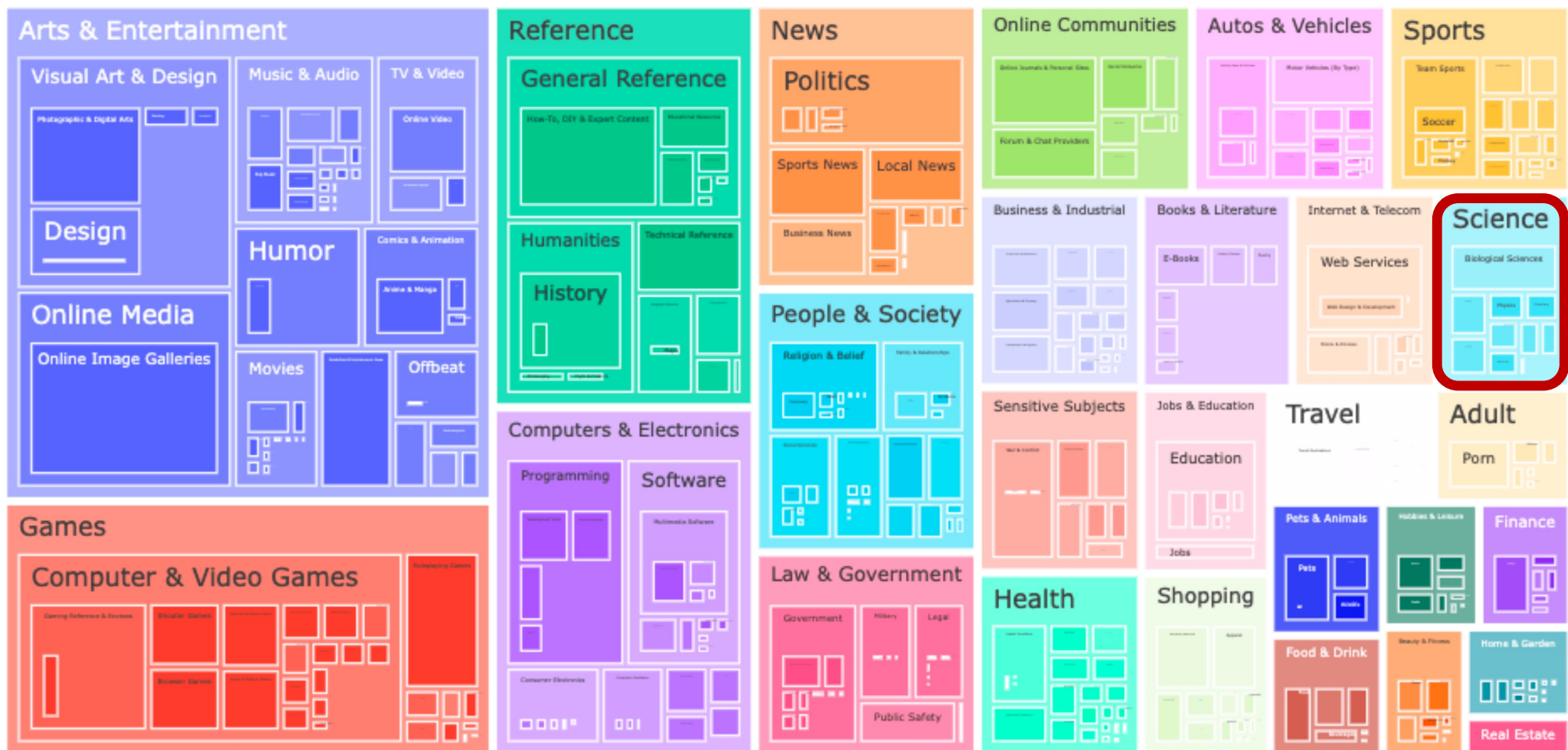
Alice: Would you tell me, please, which way I ought to go from here?
Cheshire Cat: That depends a good deal on where you want to get to.
Alice: I don't much care where.
Cheshire Cat: Then it doesn't much matter which way you go.
Alice: ... So long as I get somewhere.
Cheshire Cat: Oh, you're sure to do that, if only you walk long enough.

<https://doi.org/10.1016/j.csbj.2021.03.022>

GPT training pipeline



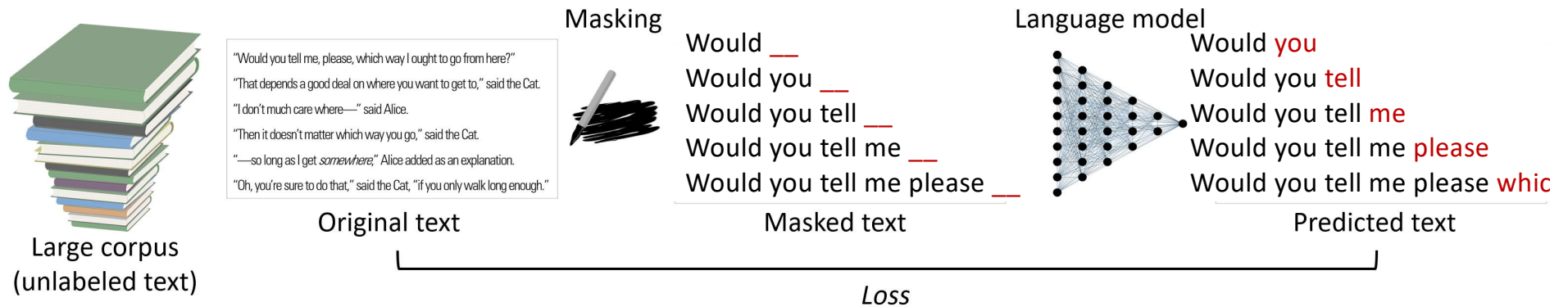
770B tokens used to train the PaLM-540B model



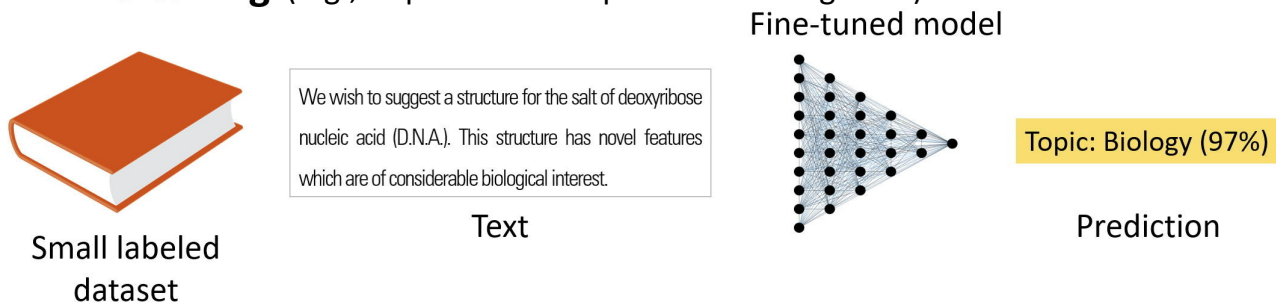
Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S. and Schuh, P., 2022. Palm: Scaling language modeling with pathways. arXiv preprint arXiv:2204.02311. Available at: <https://arxiv.org/abs/2204.02311>

LLMs can be fine-tuned for specific tasks

A Pretraining



B Fine-tuning (e.g., to predict the topic of a text fragment)



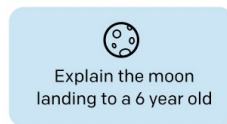
<https://doi.org/10.1016/j.csbj.2021.03.022>

Supervised fine-tuning (SFT)

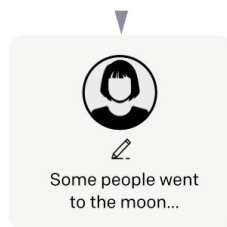
Step 1

**Collect demonstration data,
and train a supervised policy.**

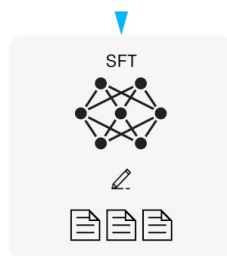
A prompt is
sampled from our
prompt dataset.



A labeler
demonstrates the
desired output
behavior.



This data is used
to fine-tune GPT-3
with supervised
learning.

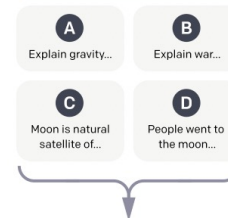
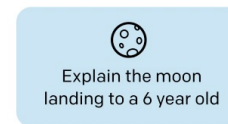


Reward model training

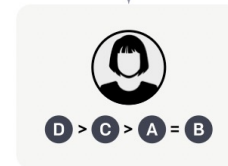
Step 2

**Collect comparison data,
and train a reward model.**

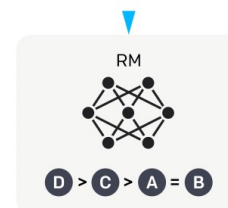
A prompt and
several model
outputs are
sampled.



A labeler ranks
the outputs from
best to worst.



This data is used
to train our
reward model.



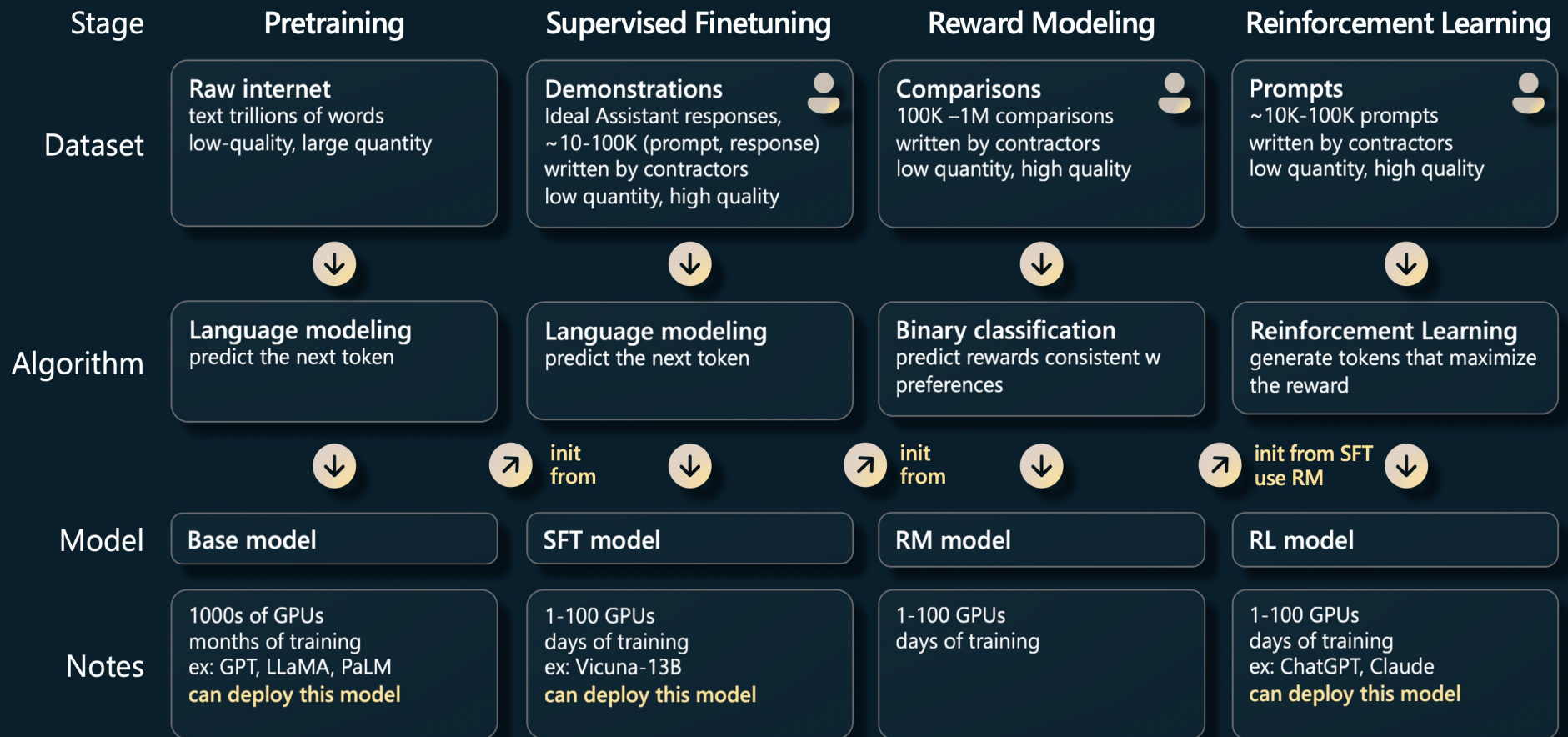
Reinforcement learning*

Step 3

**Optimize a policy against
the reward model using
reinforcement learning.**

* Via proximal policy
optimization on reward
model

GPT Assistant training pipeline



<https://karpathy.ai/stateofgpt.pdf>

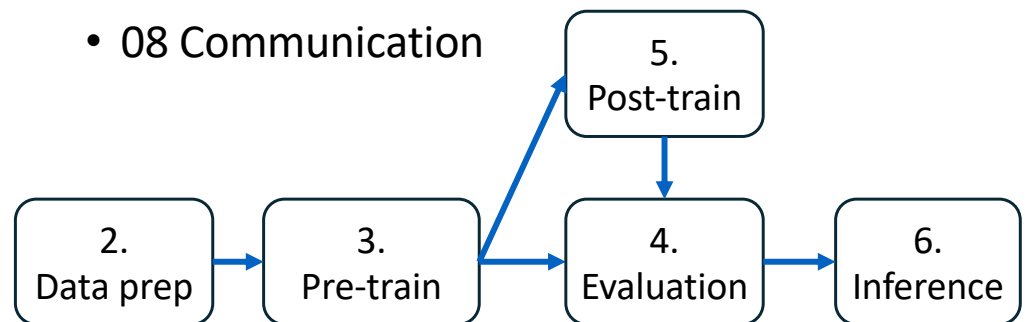
AuroraGPT*

Explore pathways towards a “Scientific Assistant” powered by Aurora supercomputer:

- Assemble high-quality **scientific datasets** for scientific FM training
- Adapt **FM development methods** to meet specialized needs of scientific FMs
- Assemble high-quality **benchmarks** to provide objective yardsticks for progress
- **Apply and evaluate methods** in areas important for DOE science

AuroraGPT project groups:

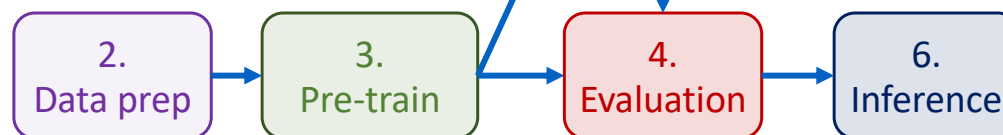
- 01 Planning
- 02 Data
- 03 Model training (pre-training)
- 04 Evaluation (skills, trustworthiness, safety)
- 05 Post-training (fine tuning, alignment)
- 06 Inference
- 07 Distribution
- 08 Communication



*named after the Leadership Class Supercomputer at Argonne that will be used for much of the research

AuroraGPT activities

- Large datasets of scientific text
- High-performance document parsing and de-duplication pipelines
- Synthetic data generation methods



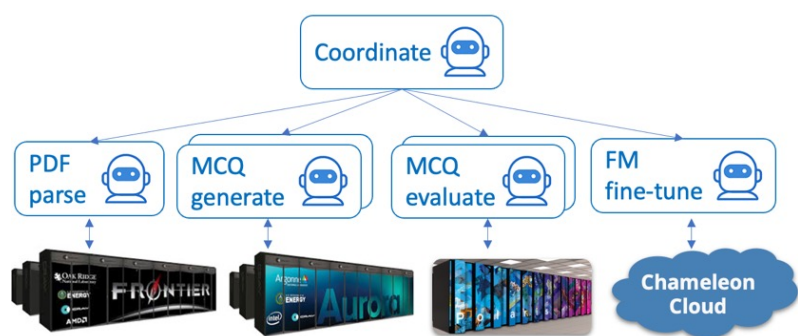
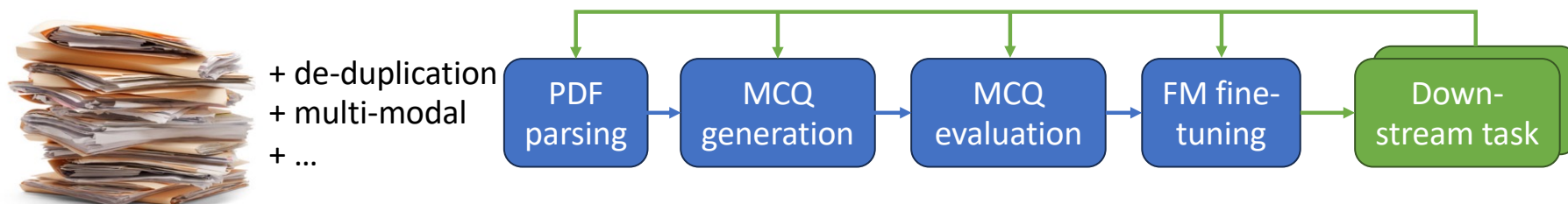
- Scalable pre-training pipelines for Polaris and Aurora
- Models trained with standard and enhanced datasets

- Acquire and deploy wide variety of evaluation suites
- New evaluation methods specialized for science FMs

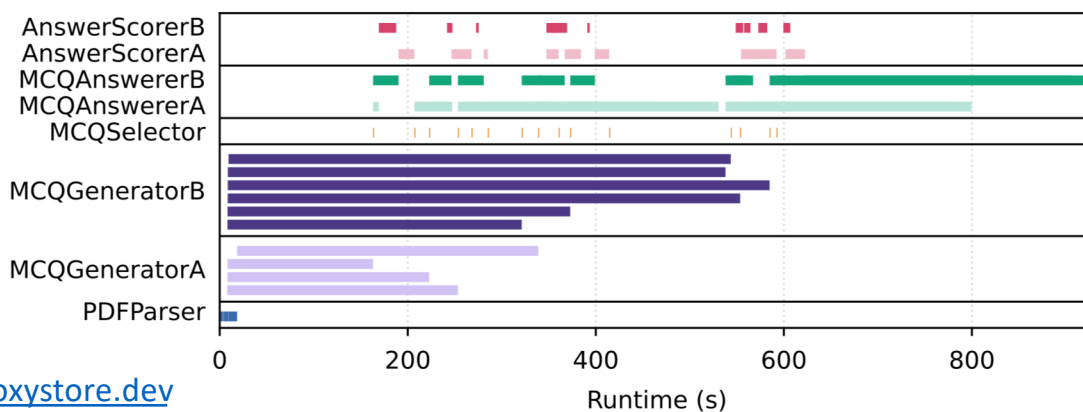
Scalable inference methods for use on ALCF and other supercomputers

Example: Adaptive fine tuning of text-based FM

“We need more documents on rare earths”



Using Academy agent framework: <https://academy.proxystore.dev>



EAIRA: Multi-faceted eval methodology End-to-End

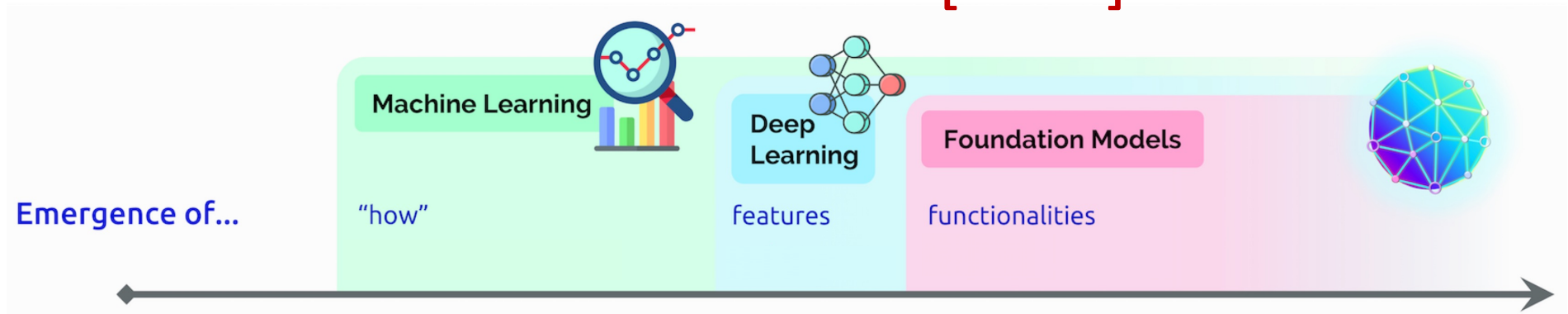
Proposed Methodology				
Techniques	MCQ Benchmarks	Open Response Benchmarks	Lab Style Experiments	In the Wild Field Style Experiments
Main Goal	Testing knowledge breadth, basic reasoning	Testing knowledge depth, planning, reasoning	Realistic testing	Realistic trend analysis and weakness diagnosis
Problem Type	Predetermined , Fixed Q&As with known solutions	Predetermined , Fixed Free-Response Problems with known solutions	Individual Defined Problems with unknown solutions	Individual Defined Problems with solutions
Verification	Automatic response verification	Automatic or Human response verification	Humans response analysis	Automatic summary of human response
Examples	Astro, Climate, AI4S (multi-domain), Existing Benchmarks	SciCode, ALDBench	see "lab style experiments"	see "field style experiments"
Cross Cutting Aspects	← Trust and Safety (ChemRisk), Uncertainty Quantification, Scalable Software Infrastructure (STAR) →			

4 complementary evaluation techniques to comprehensively assess the capabilities of LLMs as scientific assistants.

(Prior work by others, Prior work by authors, New work)

EAIRA: A Methodology for Evaluating AI Models as Scientific Research Assistants, <https://arxiv.org/pdf/2502.20309>.

The rise of foundation models [2022]

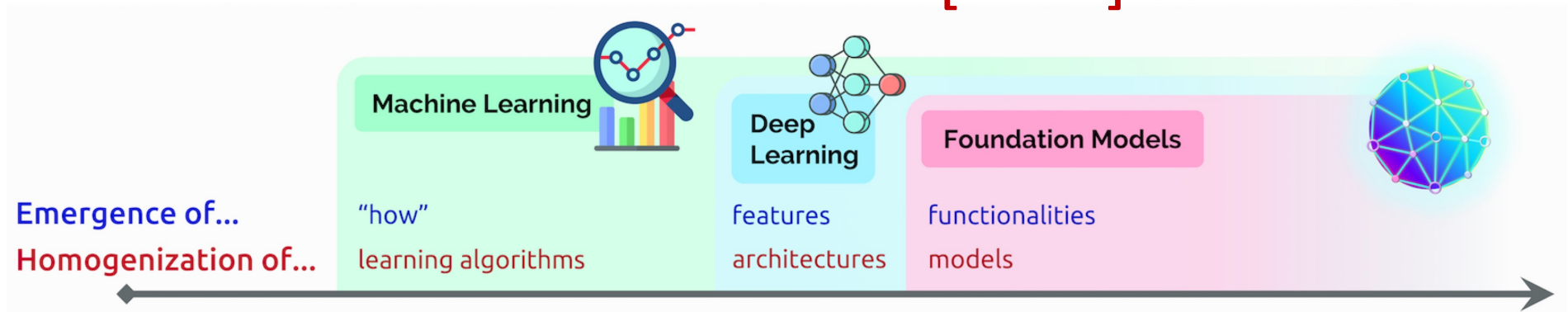


“The story of AI has been one of increasing **emergence**

With the introduction of **machine learning**, how a task is performed emerges (is inferred automatically) from examples; with **deep learning**, the high-level features used for prediction emerge; and with **foundation models**, even advanced functionalities such as in-context learning emerge.

<https://arxiv.org/pdf/2108.07258.pdf>

The rise of foundation models [2022]



“The story of AI has been one of increasing **emergence** and **homogenization**.

With the introduction of machine learning, how a task is performed emerges (is inferred automatically) from examples; with deep learning, the high-level features used for prediction emerge; and with foundation models, even advanced functionalities such as in-context learning emerge.

At the same time, **machine learning** homogenizes learning algorithms (e.g., logistic regression), **deep learning** homogenizes model architectures (e.g., Convolutional Neural Networks), and **foundation models** homogenizes the model itself (e.g., GPT-3).”

<https://arxiv.org/pdf/2108.07258.pdf>

Foundation models

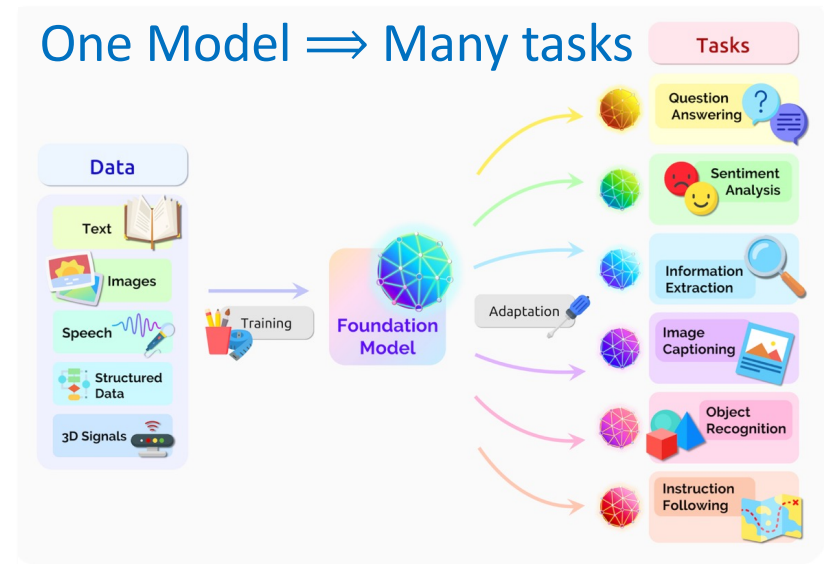
A **large model** trained on **large datasets** from **many sources**: Text, papers, datasets, code, molecules, etc.

Additional training to improve human interaction experience: E.g., ChatGPT-4

Exhibit emergent behaviors: Capable of tasks not originally trained to do

May have **100s of applications built on top**

<https://arxiv.org/pdf/2108.07258.pdf>



Trained on trillions of input “tokens” for many weeks on a big computer

SOTA models (GPT-4) have about 1 trillion parameters

Foundation Models (FMs): The starting point

Foundation models like GPT-3 and BERT are large-scale neural networks pretrained on vast corpora using **self-supervised learning**. Their power comes from:

- **Scale** (billions of parameters)
- **Data diversity**
- **Generalization** across tasks (zero/few-shot learning)

As **statistical language models**, they generate fluent output without robust internal reasoning, and often fail at multi-step logic

Emergence from scale and prompting

Foundation models have also led to surprising emergence which results from scale. For example, GPT-3, with 175 billion parameters compared to GPT-2's 1.5 billion, permits in-context learning, in which the language model can be adapted to a downstream task simply by providing it with a prompt (a natural language description of the task), an emergent property that was neither specifically trained for nor anticipated to arise.

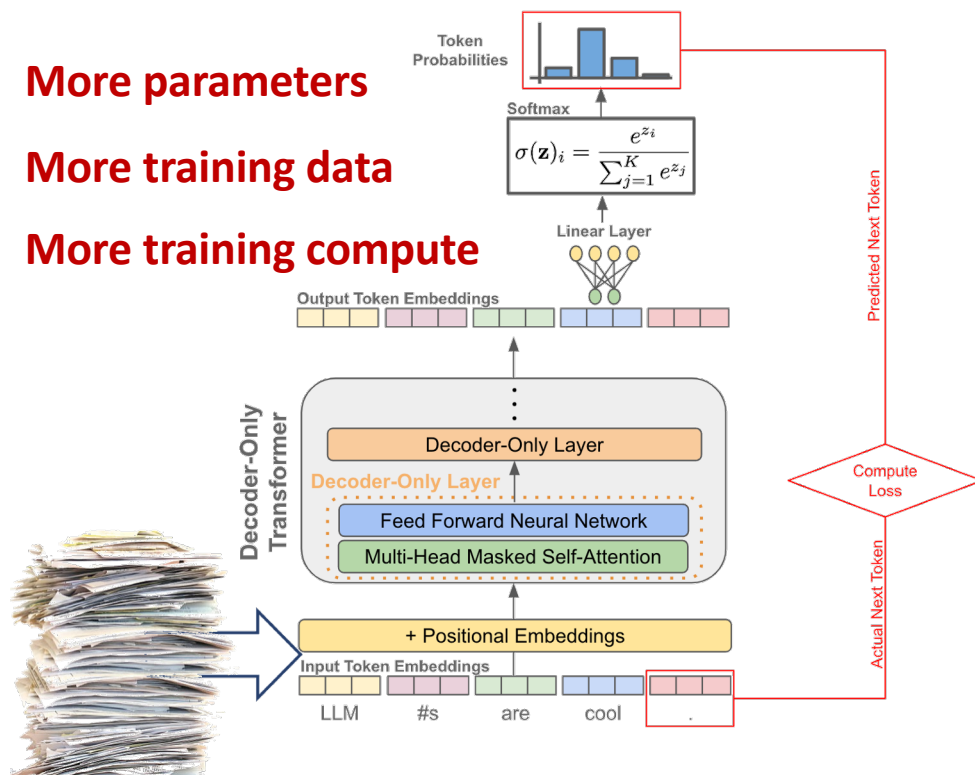
How foundation models are created and applied

Training: Learn parameter values that encode deep structure in data

More parameters

More training data

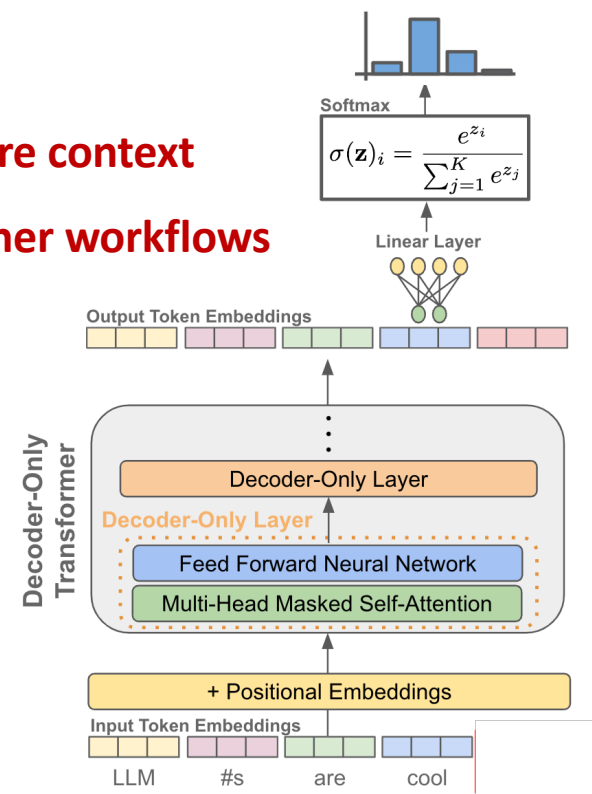
More training compute



Inference: Apply parameter values to predict next token

More context

Richer workflows



Method 1: Describe the task in the prompt

A) With GPT-4, 12 months ago

You are an expert chemist. Given the reactants SMILES, your task is to predict property of molecules using your experienced chemical Property Prediction knowledge.

Please strictly follow the format, no other information can be provided. Given the SMILES string of a molecule, the task focuses on predicting molecular properties, specifically penetration/non-penetration to the brain-blood barrier, based on the SMILES string representation of the molecule. Please answer with only Yes or No, indicating whether or not the molecule penetrates the blood-brain barrier.

SMILES: OCCN1CCN(CCCN2c3ccccc3Sc4ccc(Cl)cc24)CC1

Penetration: No

Congratulations! You are a prompt engineer. Except it got the answer wrong.

Method 1: Describe the task in the prompt

B) With GPT-5, today

You are an expert chemist. Given the reactants SMILES, your task is to predict property of molecules using your experienced chemical Property Prediction knowledge.

Please strictly follow the format, no other information can be provided. Given the SMILES string of a molecule, the task focuses on predicting molecular properties, specifically penetration/non-penetration to the brain-blood barrier, based on the SMILES string representation of the molecule. Please answer with only Yes or No, indicating whether or not the molecule penetrates the blood-brain barrier.

SMILES: OCCN1CCN(CCCN2c3ccccc3Sc4ccc(Cl)cc24)CC1

Penetration: Yes

Congratulations! You are a prompt engineer. And you got the right answer.

Prompt engineering: Stimulating latent abilities

Researchers realized you could coax latent reasoning capabilities out of FMs by using clever prompts:

- **Few-shot prompting:** Show the model examples.
- **Chain-of-thought (CoT):** Encourage step-by-step thinking.
- **ReAct prompting:** Interleave reasoning and actions (e.g., tool use).

These techniques began to unlock reasoning without retraining

Method 2: Provide examples in the prompt

In-context learning is an LLM paradigm where predictions are based solely on contexts enriched with a few demonstration examples.

Also known as “few-shot” learning

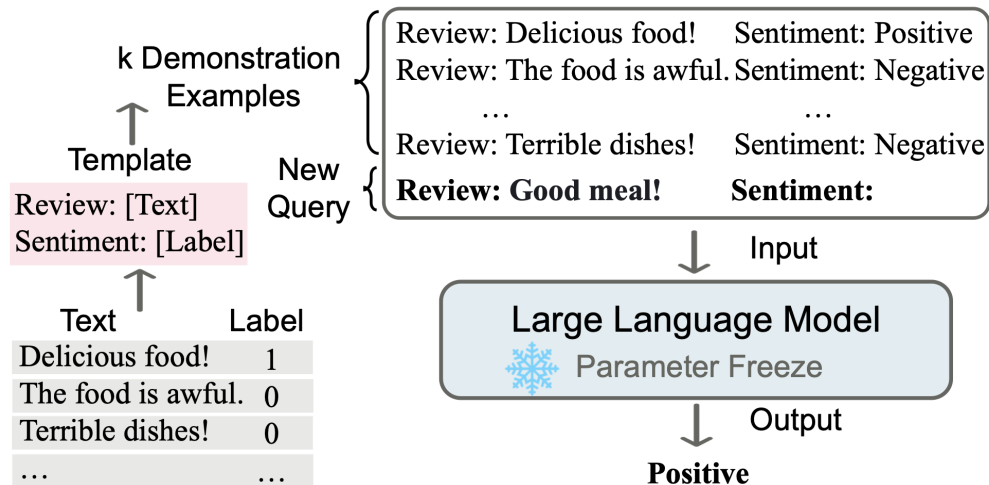


Figure 1: Illustration of in-context learning. ICL requires a prompt context containing a few demonstration examples written in natural language templates. Taking this prompt and a query as the input, large language models are responsible for making predictions.

<https://arxiv.org/abs/2301.00234> for survey, and <https://arxiv.org/pdf/2305.18365> for chemistry examples

In-context learning for our chemistry example

General Template

You are an expert chemist. Given the reactants SMILES, your task is to predict property of molecules using your experienced chemical Property Prediction knowledge.

Task-specific Template

Please strictly follow the format, no other information can be provided. Given the SMILES string of a molecule, the task focuses on predicting molecular properties, specifically penetration/non-penetration to the brain-blood barrier, based on the SMILES string representation of each molecule. You will be provided with several examples molecules, each accompanied by a binary label indicating whether it has penetrative property (Yes) or not (No). Please answer with only Yes or No.

ICL

SMILES: OCCN1CCN(CCCN2c3ccccc3Sc4ccc(Cl)cc24)CC1
Penetration: Yes
SMILES: [C@@]1([C@H](C2CCC1CC2)NC(C)C)(C3=CC(=C(C=C3)Cl)Cl)O
Penetration: Yes
SMILES: COC1=C(N3C(SC1)C(NC(=O)C(N)C2C=CCC=C2)C3=O)C(O)=O
Penetration: No
SMILES: CC1(C)N[C@@H](C(=O)N1[C@H]2[C@H]3SC(C)(C)[C@@H](N3C2=O)C(O)=O)c4ccccc4
Penetration: No

Question

SMILES: CC(C)[C@H](NC(=O)N(C)Cc1csc(n1)C(C)C)C(=O)N[C@H](C[C@H](O)[C@H](Cc2ccccc2)NC(=O)OCc3scnc3)Cc4ccccc4
Penetration:

Answer

Yes



<https://arxiv.org/pdf/2305.18365>

Method 3: Chain of Thought

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

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?

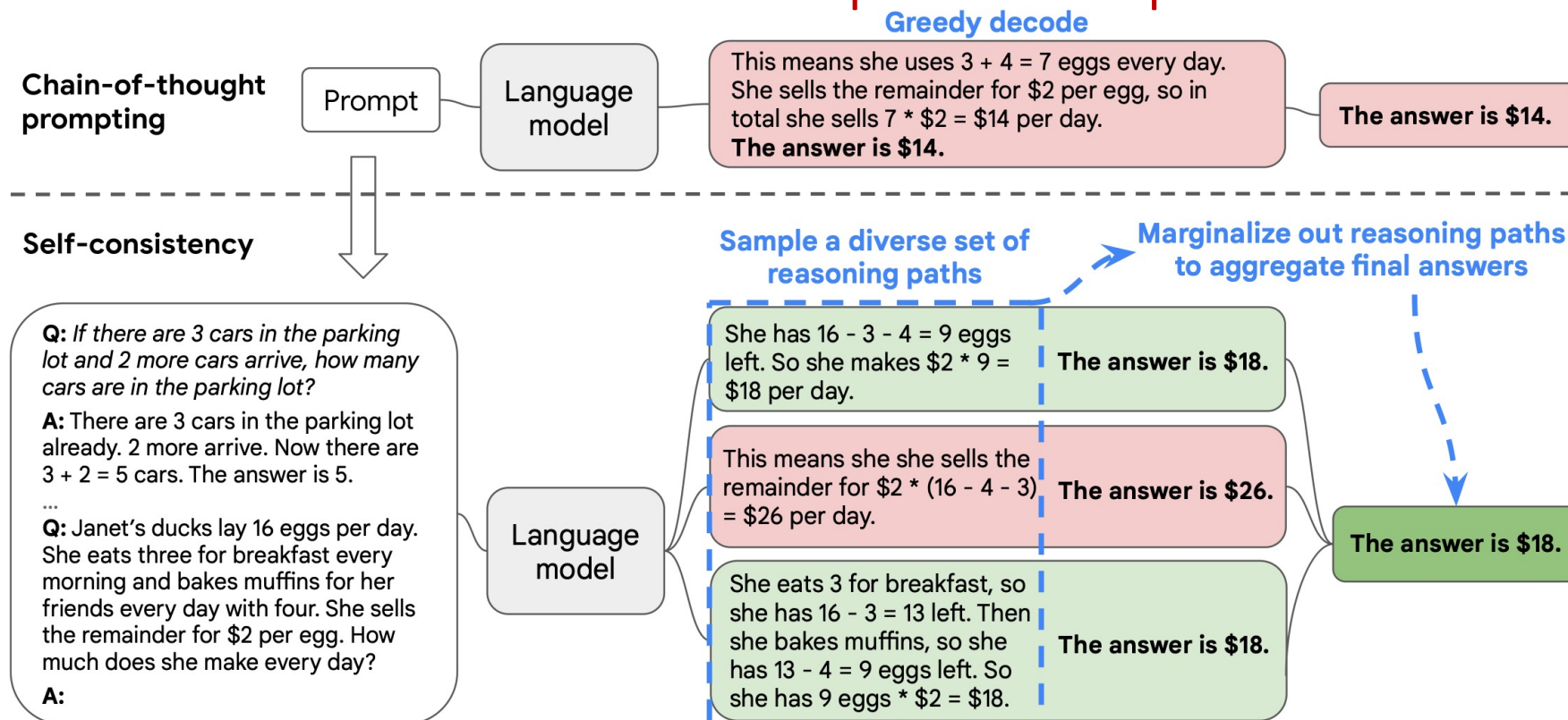
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

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. ✅

Method 4: Ensemble multiple attempts



<https://arxiv.org/pdf/2203.11171>

Self-consistency improves chain-of-thought reasoning in LLMs by sampling multiple diverse reasoning paths for a single prompt and then selecting the most frequent final answer, effectively acting as a self-ensemble

Self-Consistency with Chain of Thought

- An advanced prompting method that enhances a language model's reasoning capabilities by :
 - Generate multiple reasoning paths: The model is asked to answer a question multiple times, generating a different "chain of thought" or step-by-step reasoning process each time.
 - Sample diverse responses: The process uses varied reasoning paths, rather than a single greedy (most likely) one, to generate more diverse potential answers.
 - Vote for the most consistent answer: After multiple reasoning paths have been generated, the model selects the most consistent or frequently occurring final answer from the set. This "majority vote" reduces the chance of a single bad reasoning path leading to an incorrect conclusion.

1. Add “think step-by-step” to your original question
2. Ask the question repeatedly (n times) and collect the answers.
3. Apply voting to the collected answers to pick final answer.

Method 5: Ask for reflection

[Step 1] Solve the following question:

Q: Why does the moon cause tides?

A: The moon's gravity pulls on Earth's oceans...

[Step 2] Reflect on your answer. Are there mistakes, assumptions, or missing details?

Reflection: The answer does not mention the role of centrifugal force or the fact there are two tides per day.

[Step 3] Revise your answer using your reflection:

Revised Answer: The moon's gravity pulls on the oceans, creating a bulge. On the opposite side, inertia causes a second bulge...

Variants of reflection

Name	Description
Reflexion (Shinn et al., 2023)	The agent reflects on failures and updates its behavior in subsequent trials. Often used in RL or agentic settings.
Critique and Revise	Prompt the model to critique a draft answer, then rewrite it using that critique.
STaR (Self-Taught Reasoner)	Model generates reasoning chains, uses final answers to teach itself via self-consistency + refinement.
ReAct + Reflect	Combines reasoning and acting (ReAct) with a follow-up self-reflection step.
Tree of Thoughts (ToT)	After generating multiple thought branches, models reflect on paths and decide which to pursue further.

Method 6: Recreate our “System 2”

- Daniel Kahneman’s dual-system model (“Thinking: Fast and Slow”):
 - System 1 is fast, intuitive, and automatic (reflexive responses, heuristics)
 - System 2 is slow, deliberate, effortful, and logical (proofs, planning, debugging).
- Most LLM outputs are System 1-like by default: they produce fluent, plausible responses quickly, but may not reflect deep reasoning or error-checking
- “Recreating System 2” means guiding the LLM (or LLM + external components) to behave like a slow, deliberative thinker.

Method 6: Recreate our “System 2” (2/2)

Chain-of-Thought (CoT) Prompting

Encourages step-by-step reasoning.

- *"Let's think step-by-step."*
- Often improves math, logic, and science question accuracy.

Self-Consistency + Sampling

Generate many CoT completions, then **vote on final answers**.

- Emulates multiple passes through a problem.

Reflection Loops

Prompt the model to **critique or revise** its own answer.

- Can be repeated recursively.
- Mimics metacognition (a key trait of System 2).

Method 6: Recreate our “System 2” (1/2)

Verifier or Critique Models

Use a **second LLM** (or the same LLM with a critique prompt) to:

- Score different candidate answers; suggest corrections; choose between competing solutions

Tree of Thoughts (ToT) / Graph of Thoughts

Encodes deliberation as exploration:

- Model generates, evaluates selectively expands multiple thought branches
- Emulates "consider multiple alternatives before deciding"

Tool Use + Memory

Give the model access to:

- External calculators, planners, search tools; scratchpads / working memory to mimic how System 2 uses tools and notes to support slow thinking.

Tool-using and Interactive Models

Reasoning was extended by letting models:

- **Use tools** (APIs, calculators, databases)
- **Interact with environments** (web agents, code interpreters, robotic planners)

Examples include:

- **ReAct** (Yao et al.): interleaved thoughts and actions
- **MRKL systems**: Modular tool invocation
- **AutoGPT, AgentGPT**: Task-decomposing, memory-using agents

Here, models became **planners and orchestrators**, not just predictors

Method 7: Access external tools for tasks

Offload tasks that LLMs are not good at. But: They do not “know” what they are (not) good at.

JW

a brick weighs 30% of a brick plus five GPUs
there are 100 bricks and 200 GPUs in a metric
ton how much does a brick weigh?

Code Interpreter plugin



Let's call the weight of a brick "B" and the weight of a GPU "G". We can set up an equation based on the information given:

$$B = 0.3 * B + 5 * G$$

```
from sympy import symbols, Eq, solve

B, G = symbols('B G')
eq1 = Eq(B, 0.3 * B + 5 * G)
eq2 = Eq(100 * B + 200 * G, 1000)
solution = solve((eq1, eq2), (B, G))
solution[B]
```

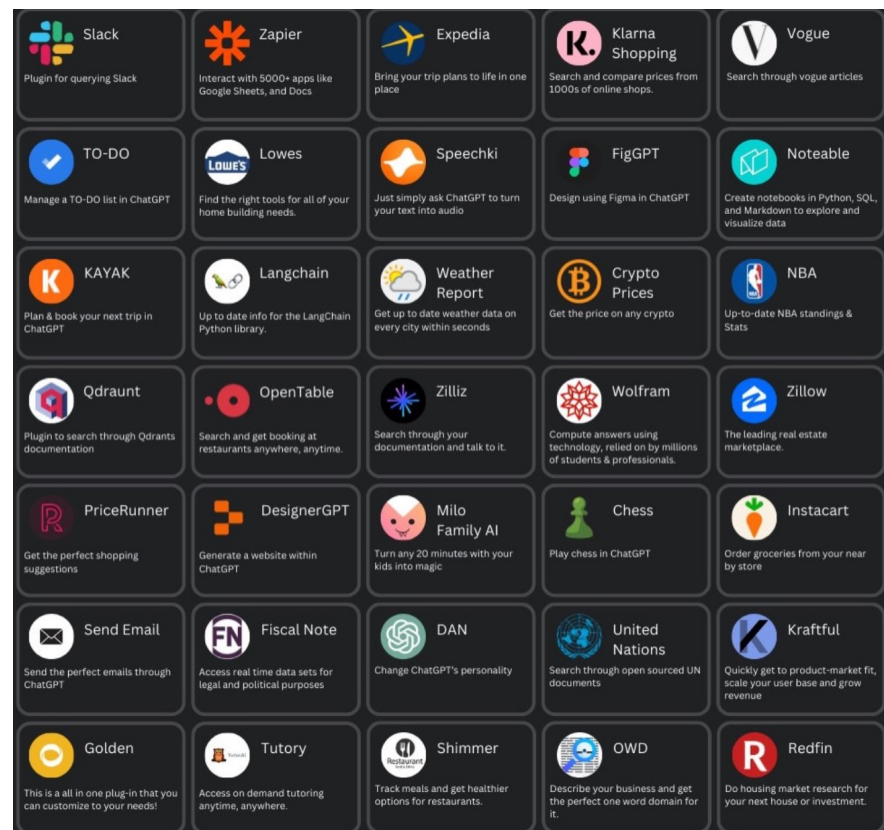
```
# Result
7.812500000000000
```

Finished calculating ^

The weight of a brick is 7.8125 kg.

on the
ing equal


d the
J (G).




ChatGPT
plugins

Modular Reasoning, Knowledge, and Language (MRKL) Systems


A set of modules (e.g., calculator, weather API, database) a router that decides how to 'route' incoming queries to appropriate module

 **Prompt**


What is 100×100 ?

 **AI Output**

`CALCULATOR[100*100]`

 **Prompt**

What is the weather like in New York?

 **AI Output**

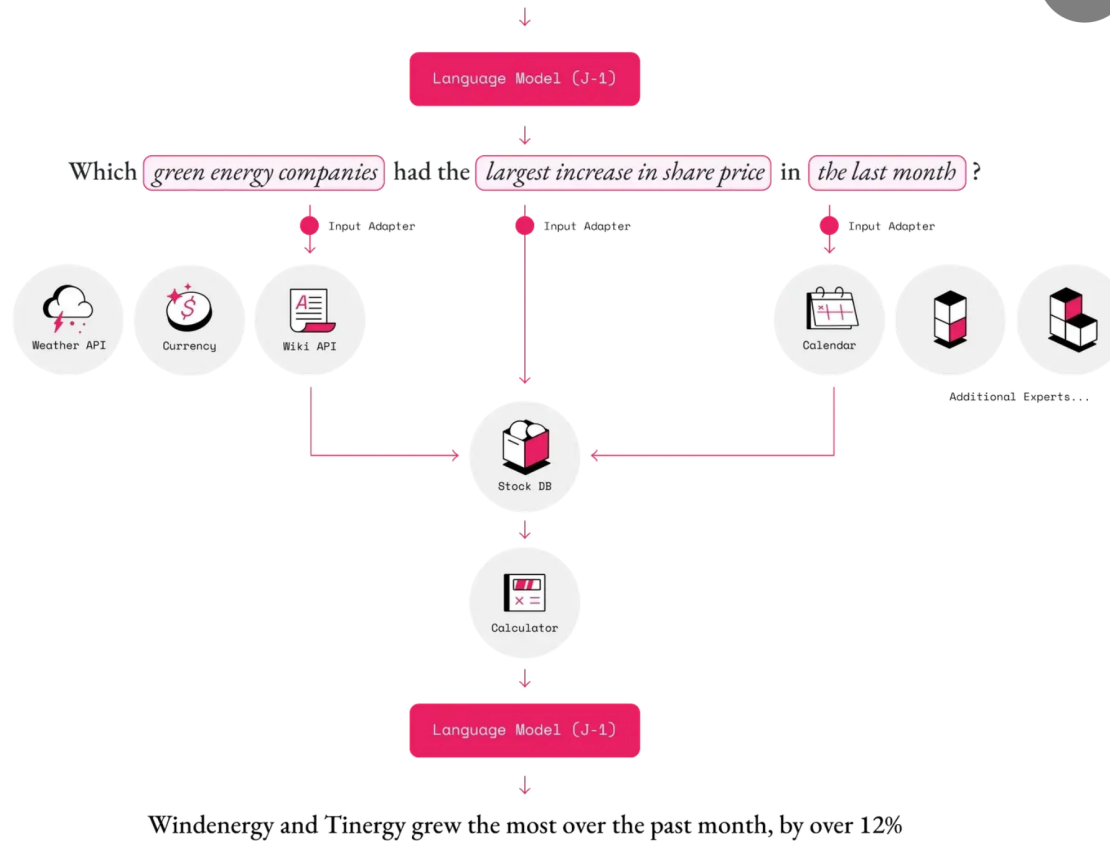
The weather is `WEATHER_API[New York]`.

<https://arxiv.org/pdf/2205.00445>

The router

- Once the router has made the decision of which module to call upon, it needs to pass the right information to it. The router is a specialized neural net and therefore invoking a neural module is easy since the neural-to-neural interface is natural.
- However, when a neural network needs to access a database, make an API call, or invoke another symbolic computation, it must extract from the text discrete parameters required by the module.
- The main message here is that there is no free lunch, but that lunch is in many cases affordable.
- The cost is in training the router to extract the arguments reliably, which must be done rigorously.

Which green energy companies had the largest increase in share price in the last month?



<https://arxiv.org/pdf/2205.00445>

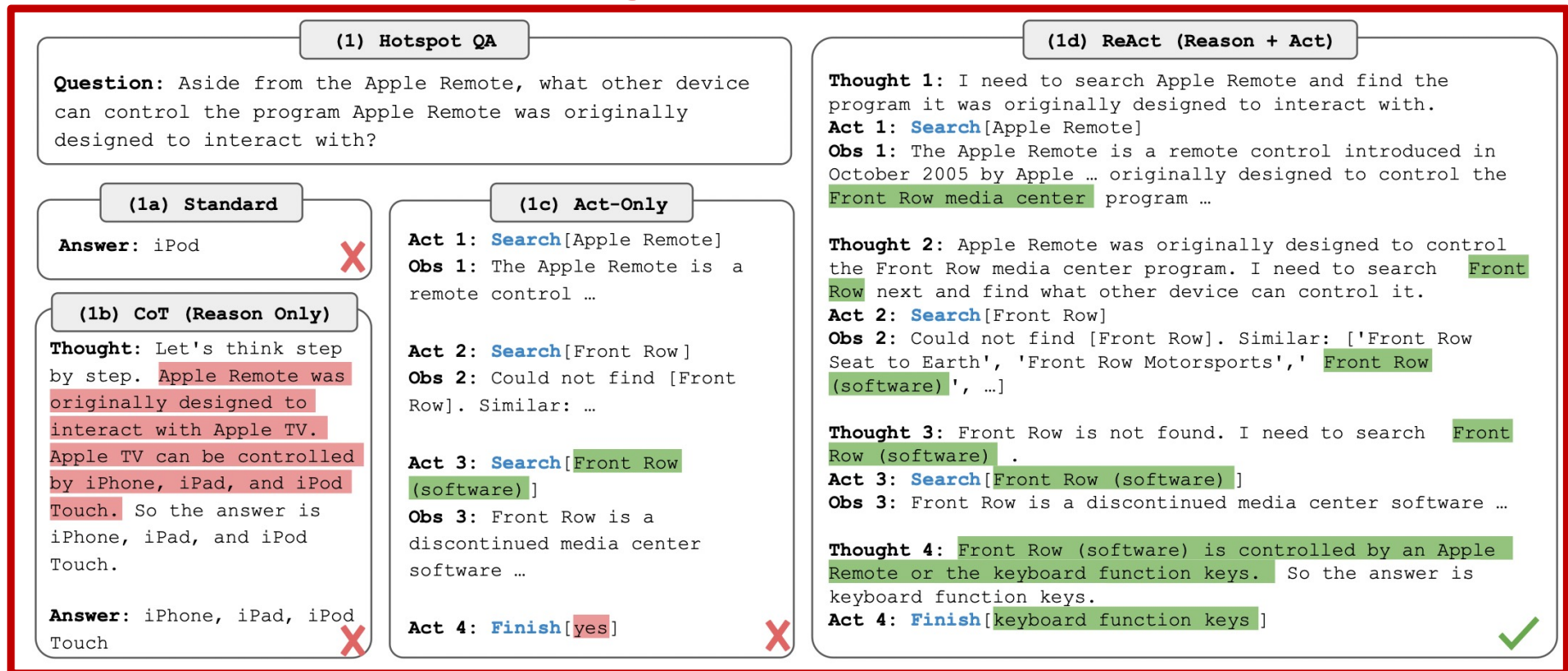
Example MRKL System (AI21)

Method 8: Chains/agents – ReAct (Reason + Act)

ReAct: Use LLM to generate both reasoning traces and task-specific actions in an interleaved manner, allowing for greater synergy between the two:

- **reasoning traces** help the model induce, track, and update action plans as well as handle exceptions, while
- **actions** allow it to interface with and gather additional information from external sources such as knowledge bases or environments

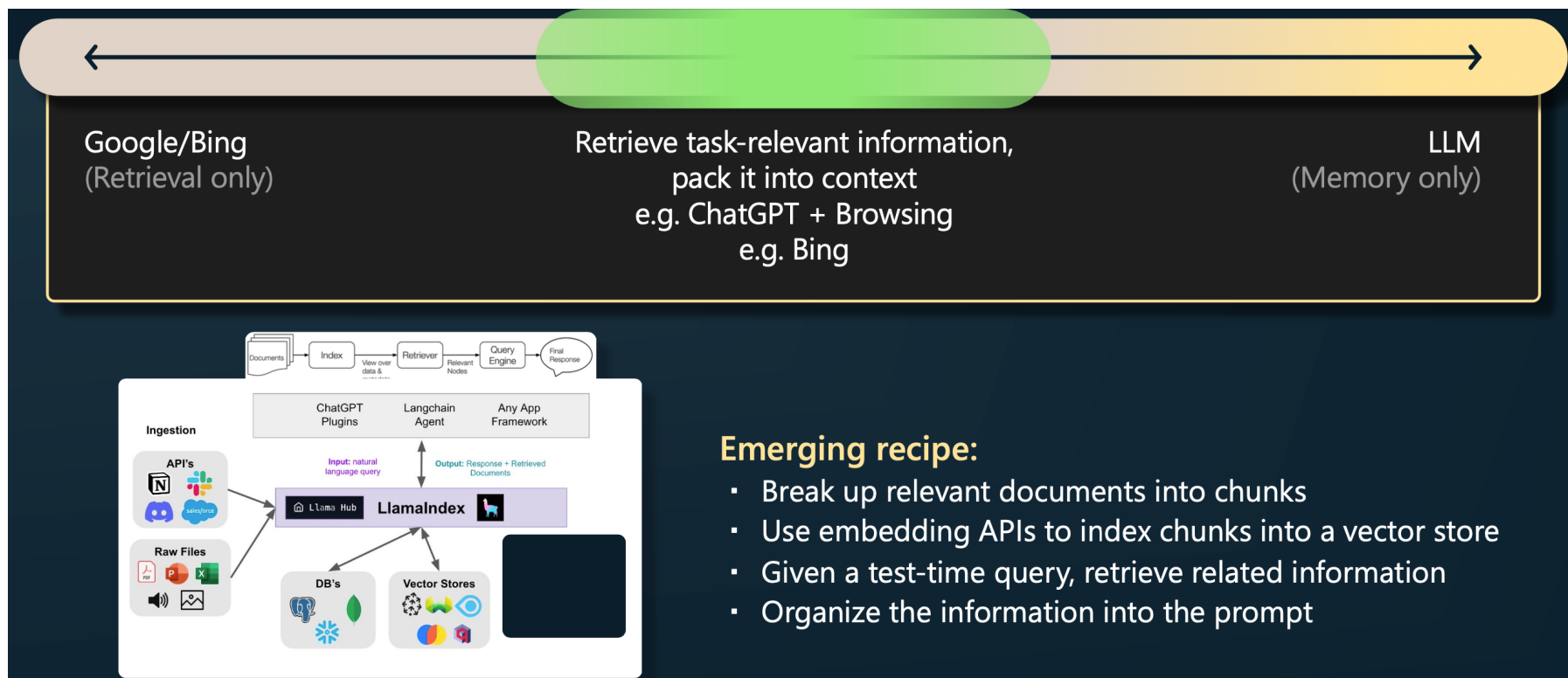
Method 8: Chains/agents – ReAct (Reason + Act)



4 prompting methods solving a HotpotQA question. We omit in-context examples in prompt, and only show task-solving trajectories generated by model (Act, Thought) and environment (Obs).

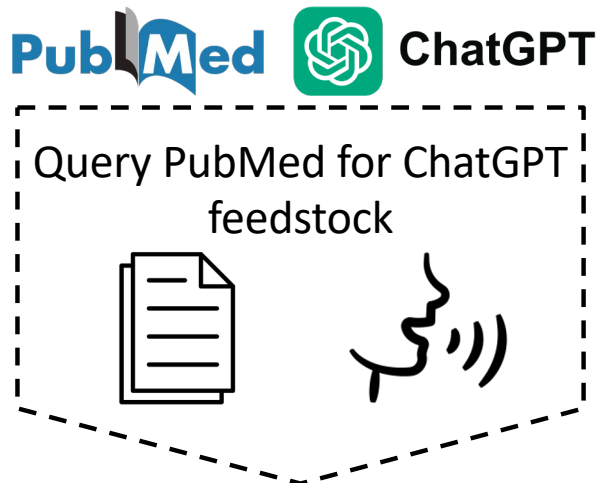
Method 9: Retrieval augmented generation

Load related context/information into “working memory” context window



Recall example: A peptide expert

(Prototyped with PubMed and ChatGPT)



Retrieve abstracts **A** from PubMed that reference specified **peptide**

Use ChatGPT to build hypotheses by using retrieval-augmented generation: e.g.:

“Given **A**, on which organism is {**peptide**} acting?”

Self-improvement and feedback loops

The next leap involved turning the FM onto itself:

- **Self-consistency**: Generate multiple CoT responses and vote on final answers.
- **Reflexion & Reflection**: Ask the model to critique and revise its own output.
- **STaR (Self-Taught Reasoner)**: Use its own outputs to fine-tune itself.

These methods moved models from **single-shot output** to **deliberative computation**, aligning with "System 2"-like behavior.

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

Lecture 2 Reading

DeepSeek-AI

`research@deepseek.com`

Abstract

We introduce our first-generation reasoning models, DeepSeek-R1-Zero and DeepSeek-R1. DeepSeek-R1-Zero, a model trained via large-scale reinforcement learning (RL) without supervised fine-tuning (SFT) as a preliminary step, demonstrates remarkable reasoning capabilities. Through RL, DeepSeek-R1-Zero naturally emerges with numerous powerful and intriguing reasoning behaviors. However, it encounters challenges such as poor readability, and language mixing. To address these issues and further enhance reasoning performance, we introduce DeepSeek-R1, which incorporates multi-stage training and cold-start data before RL. DeepSeek-R1 achieves performance comparable to OpenAI-o1-1217 on reasoning tasks. To support the research community, we open-source DeepSeek-R1-Zero, DeepSeek-R1, and six dense models (1.5B, 7B, 8B, 14B, 32B, 70B) distilled from DeepSeek-R1 based on Qwen and Llama.

<https://arxiv.org/pdf/2501.12948>

Deepseek-R1 architecture

- **Mixture of Experts (MoE)**: 671B parameters are partitioned among expert sub-models, each with 47B parameters, specialised in e.g., mathematical computation, logical deduction, language generation
- **Reinforcement Learning (RL)**: Model learns by receiving rewards or penalties based on its actions, improving through trial and error
 - E.g., if training on a prompt like "2 + 2 =", the model might receive a reward of +1 for outputting "4" and a penalty of -1 for any other answer
- Employs **Group Relative Policy Optimization (GRPO)** RL to avoid need for [large amounts of] human-labeled data
 - Score LLM moves over multiple rounds by using predefined rules like coherence and/or fluency

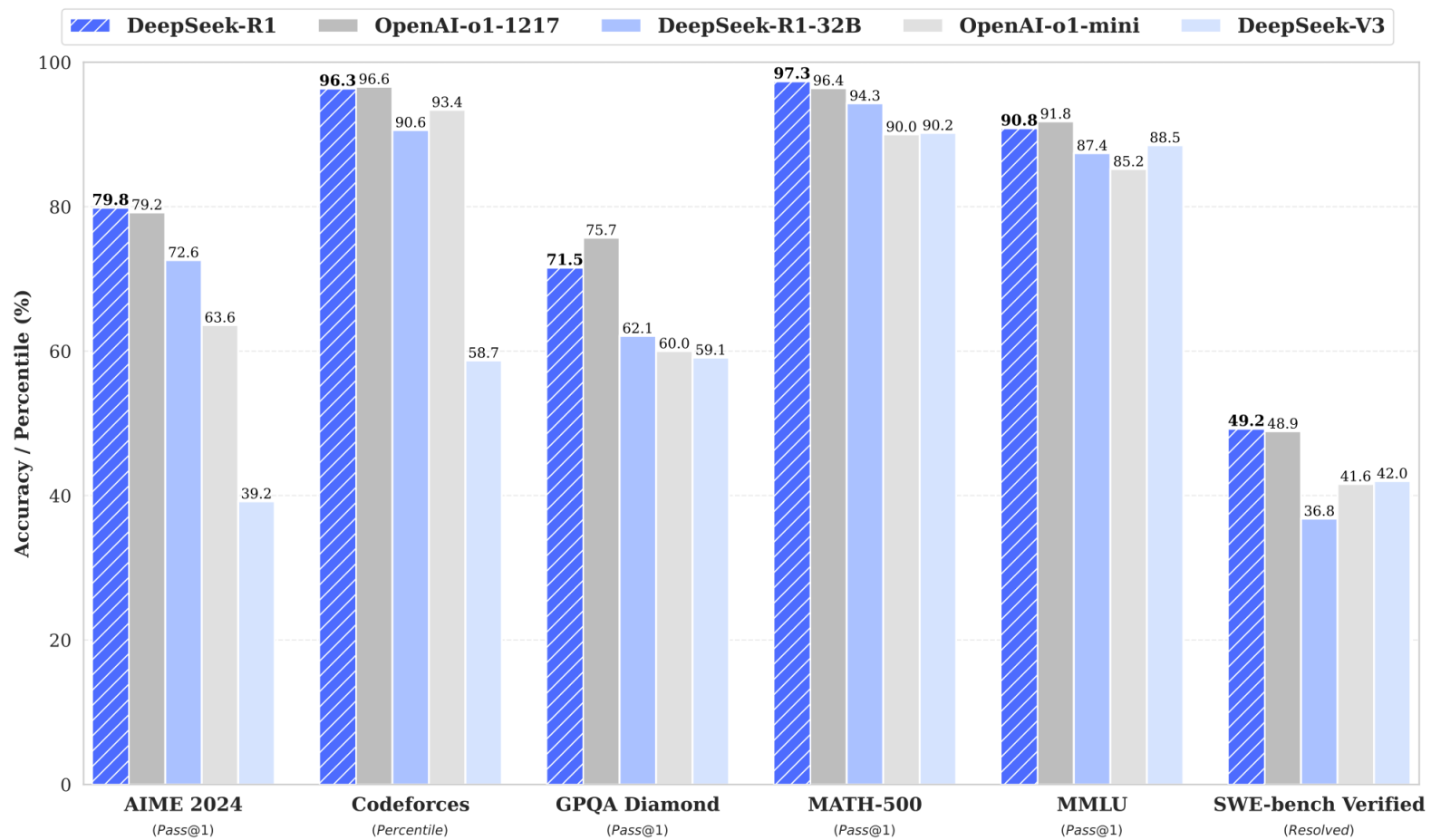
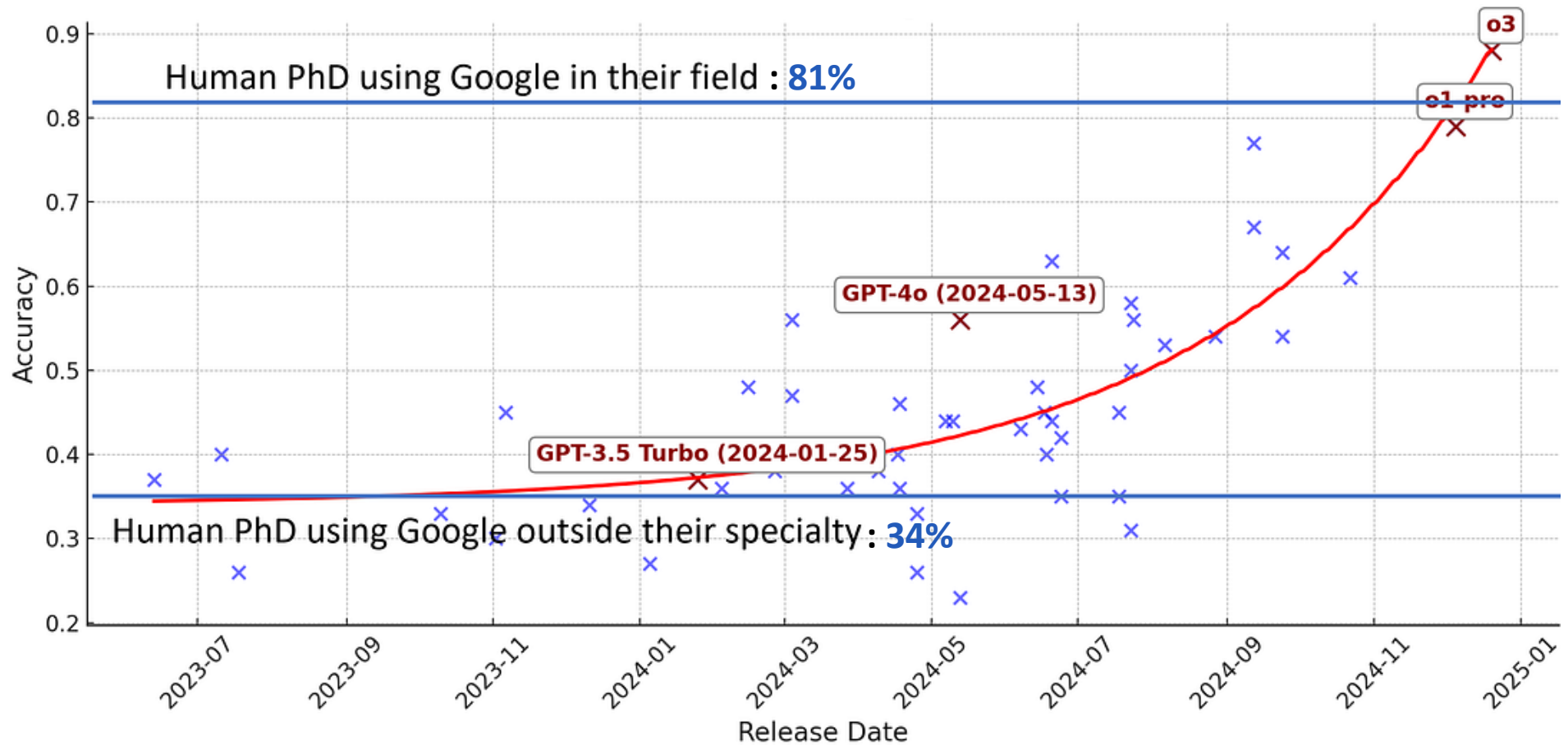


Figure 1 | Benchmark performance of DeepSeek-R1.

Graduate-Level Google-Proof Q&A test (GPQA), Diamond problems



<https://arxiv.org/pdf/2311.12022>

<https://epoch.ai/data/ai-benchmarking-dashboard>

Example GPQA problems

Genetics

If a sperm from species A is injected into an egg from species B and both species have the same number of chromosomes, what would be the main cause of the resulting zygote mortality?

- A) Species specific zona pellucida proteins on the egg cannot bind sperms from a different species.
- B) Epistatic interactions between the genes of different species
- C) Chromosomal incompatibilities will cause failure of meiosis leading to death of zygote.
- D) Chromosomal recombination will not occur in different species.

Quantum Mechanics

Suppose we have a depolarizing channel operation given by $E(\rho)$. The probability, p , of the depolarization state represents the strength of the noise. If the Kraus operators of the given state are $A_0 = \sqrt{1 - \frac{3p}{4}}$, $A_1 = \sqrt{\frac{p}{4}}X$, $A_2 = \sqrt{\frac{p}{4}}Y$, and

$A_3 = \sqrt{\frac{p}{4}}Z$. What could be the correct Kraus Representation of the state $E(\rho)$?

- A) $E(\rho) = (1 - p)\rho + \frac{p}{3}X\rho X + \frac{p}{3}Y\rho Y + \frac{p}{3}Z\rho Z$
- B) $E(\rho) = (1 - p)\rho + \frac{p}{3}X\rho^2 X + \frac{p}{3}Y\rho^2 Y + \frac{p}{3}Z\rho^2 Z$
- C) $E(\rho) = (1 - p)\rho + \frac{p}{4}X\rho X + \frac{p}{4}Y\rho Y + \frac{p}{4}Z\rho Z$
- D) $E(\rho) = (1 - p)\rho^2 + \frac{p}{3}X\rho^2 X + \frac{p}{3}Y\rho^2 Y + \frac{p}{3}Z\rho^2 Z$

AIME benchmark

American Invitational Mathematics Examination

Please reason step by step, and put your final answer within `\boxed{}`

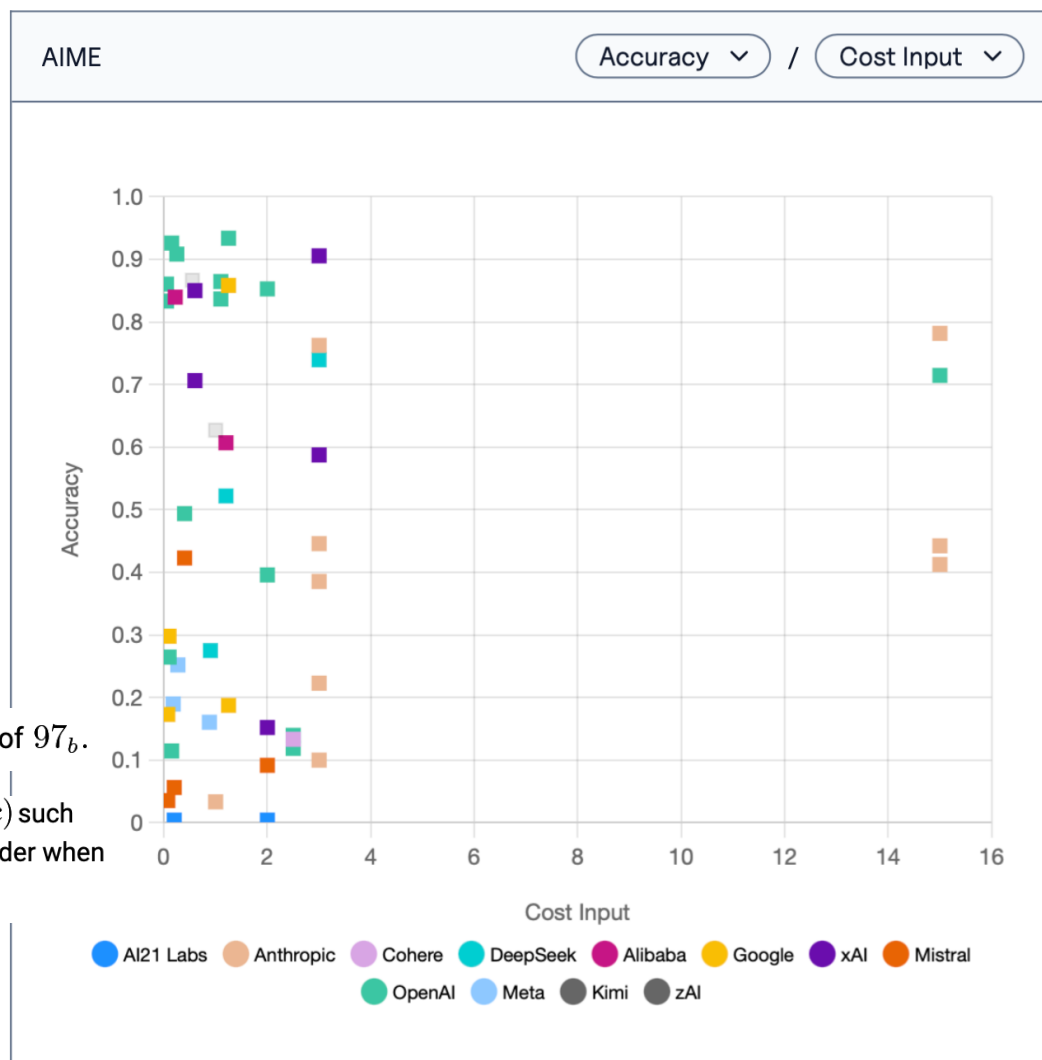
{Question}

Example problems:

Find the sum of all integer bases $b > 9$ for which 17_b is a divisor of 97_b .

Let N denote the number of ordered triples of positive integers (a, b, c) such that $a, b, c \leq 3^6$ and $a^3 + b^3 + c^3$ is a multiple of 3^7 . Find the remainder when N is divided by 1000.

<https://www.vals.ai/benchmarks/aime-2025-09-08>



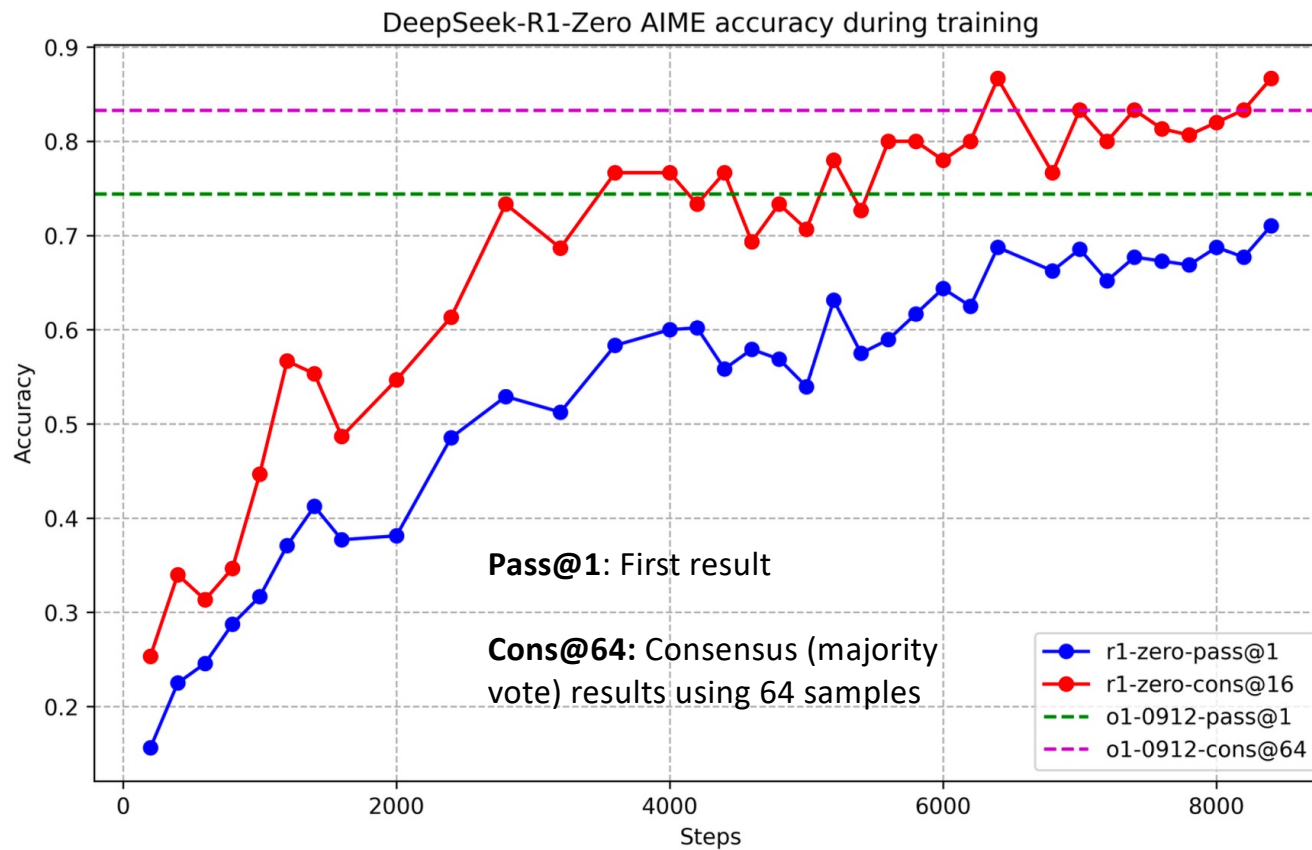


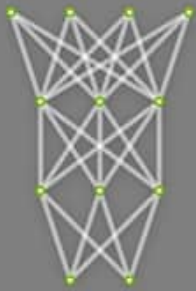
Figure 2 | AIME accuracy of DeepSeek-R1-Zero during training. For each question, we sample 16 responses and calculate the overall average accuracy to ensure a stable evaluation.

DEEP LEARNING

TRAINING

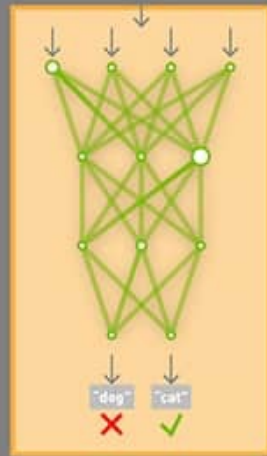
Learning a new capability
from existing data

Untrained
Neural Network
Model



Deep Learning
Framework

TRAINING
DATASET



Trained Model
New Capability



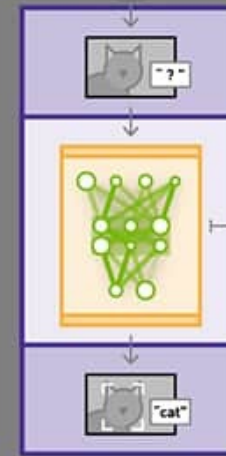
INFERENCE

Applying this capability
to new data

NEW
DATA



App or Service
Featuring Capability



Trained Model
Optimized for
Performance

<https://blogs.nvidia.com/blog/difference-deep-learning-training-inference-ai/>

Scaling test-time compute

- Traditional FMs: Much time **training** deep neural network, then rapid (relatively low-cost) **inference** at **test-time** to answer questions
- Recently: Make use of additional computation at test time so as to improve the accuracy of their response

Question: How to balance time spent on training (e.g., by training a bigger model or with more data) vs. test-time compute

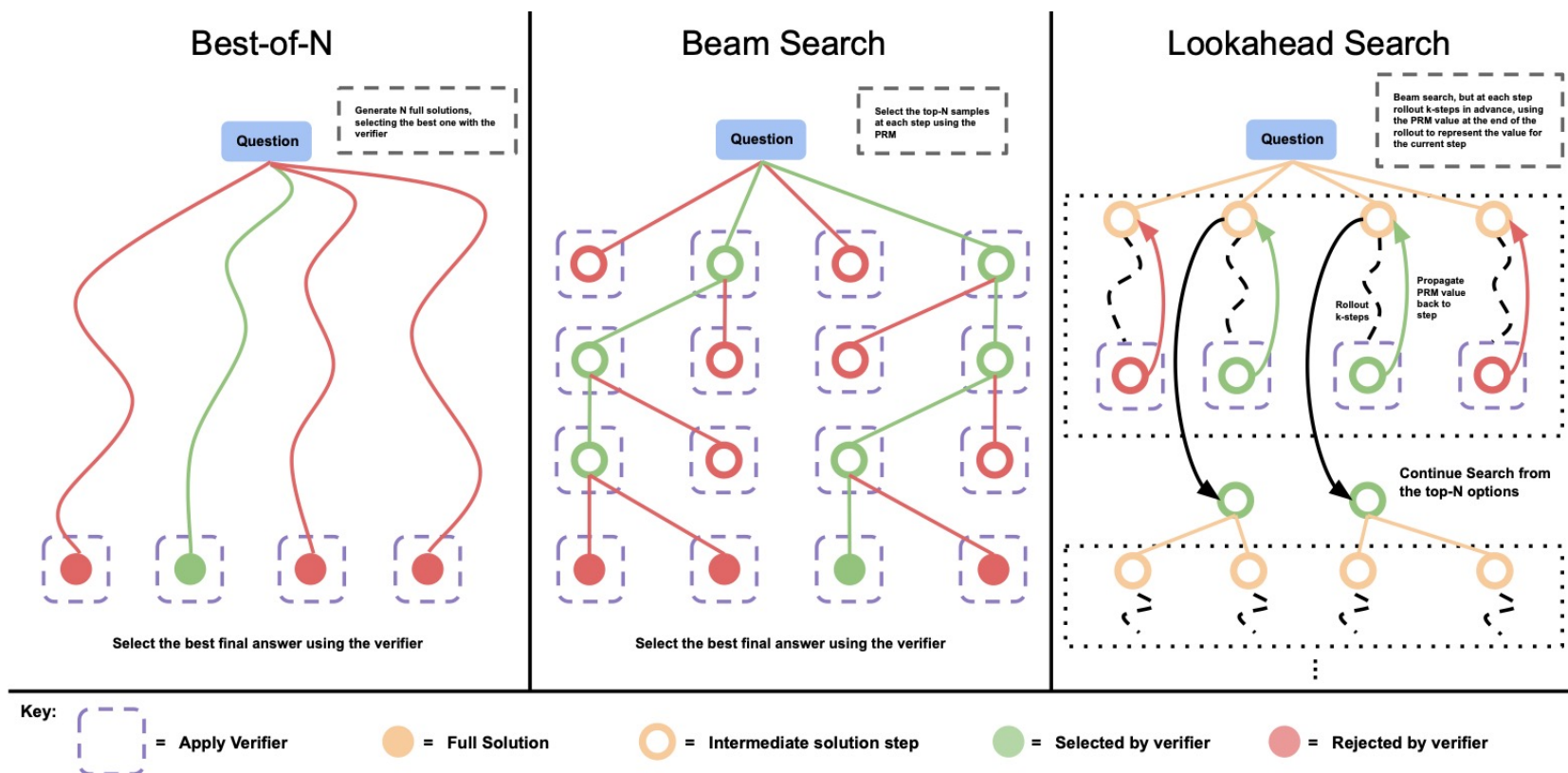


Figure 2 | Comparing different PRM search methods. **Left:** Best-of-N samples N full answers and then selects the best answer according to the PRM final score. **Center:** Beam search samples N candidates at each step, and selects the top M according to the PRM to continue the search from. **Right:** lookahead-search extends each step in beam-search to utilize a k-step lookahead while assessing which steps to retain and continue the search from. Thus lookahead-search needs more compute.

<https://arxiv.org/pdf/2408.03314>

PRM = process-based reward model

Summary: Emergence of reasoning models (1/2)

Layered on top of or evolved from foundation models

1) Prompt-Based Reasoners (zero retraining)

- **CoT, ReAct, SC-CoT, Few-shot reasoning**
- Extend base LLMs without fine-tuning
- Fast iteration, but limited consistency

3) Fine-Tuned Reasoners

- **Flan-T5, WizardLM, Mistral-Instruct, OpenChat**
- Trained on curated reasoning datasets (math, logic, instruction tuning)
- More robust, but limited to their finetuned domains

Examples of fine-tuned reasoners

- **Flan-T5** is a powerful and versatile, open-source Large Language Model (LLM) developed by Google, based on the T5 (Text-To-Text Transfer Transformer) architecture. It excels at a wide range of natural language processing (NLP) tasks due to its "instruction finetuning" on a massive, diverse dataset of over a thousand tasks, allowing it to understand and execute complex, zero-shot, and few-shot prompts.
- **WizardLM** is a family of advanced Large Language Models (LLMs) developed by Microsoft AI that excels in complex reasoning, chat, multilingual tasks, and agentic use cases. The models are trained using the Evol-Instruct method, which uses existing LLMs to automatically generate increasingly complex instructions and high-quality instruction-following data, leading to enhanced performance compared to traditional instruction fine-tuning methods.

Summary: Emergence of reasoning models (2/2)

3) Agentic Reasoners

- ReAct agents (planner + tool invoker)
- AutoGPT-like frameworks (multi-step task planners)
- Often combine memory + long-term planning, tool use, self-evaluation or self-reflection

4) Modular Reasoners

- Graph of Thoughts, Tree of Thoughts (expand multiple options before committing)
- Use LLMs as nodes in a higher-level cognitive architecture
- Inspired by human deliberation and planning algorithms

5) Multimodal Reasoning Models

- GPT-4V, LLaVA, Kosmos, IDEFICS
- Perform reasoning over text + images (and sometimes audio/video)
- Used in scientific discovery, robotics, and perception-heavy tasks

OpenAI models: What we know

- Reasoning models are designed to “think before answering” by generating internal chains of thought or planning tokens internally before producing a final answer
- The GPT-5 API has a parameter **reasoning_effort** that lets you adjust how much internal “thinking” the model does before giving its answer
- The GPT-5 system is described as having a router that decides whether to use a “fast” model or a “deeper reasoning” model depending on task complexity
- For reasoning tasks, the reasoning models pay extra compute/latency to generate more structured or deliberative responses

What GPT o5 likely does

1.Parallel Reasoning at Test-Time

- Run multiple internal reasoning paths in parallel, then aggregate or vote among them before deciding on a final answer. (A pattern seen in many top-tier reasoning systems.)

2.Adaptive Reasoning Effort

- Dynamically scale the amount of internal compute (thinking steps) based on the complexity of the prompt — either automatically via the system, or via a parameter like `reasoning_effort`.

3.Routing Layer

- The model might route between lighter vs heavier reasoning modes. For easier queries, it may use a simpler path; for complex mathematics, it may escalate to deeper reasoning steps. This is consistent with the GPT-5 architecture.
- The model likely can call tools, perform code execution, search, or retrieve external knowledge as part of its reasoning pipeline.

5.Longer Context / Memory Handling

- To sustain reasoning over long chains or multi-step problems, it may maintain internal memory or context beyond just the prompt tokens.

6.Higher Accuracy / Reduced Hallucination via Verification

- It may include internal verification steps: after generating a candidate answer, re-evaluate or cross-check with alternative reasoning paths to ensure consistency.

7.User-Controlled Verbosity / Effort Trade-Offs

Provide knobs (parameters) so the user can dial down reasoning effort for speed, or dial it up for precision.

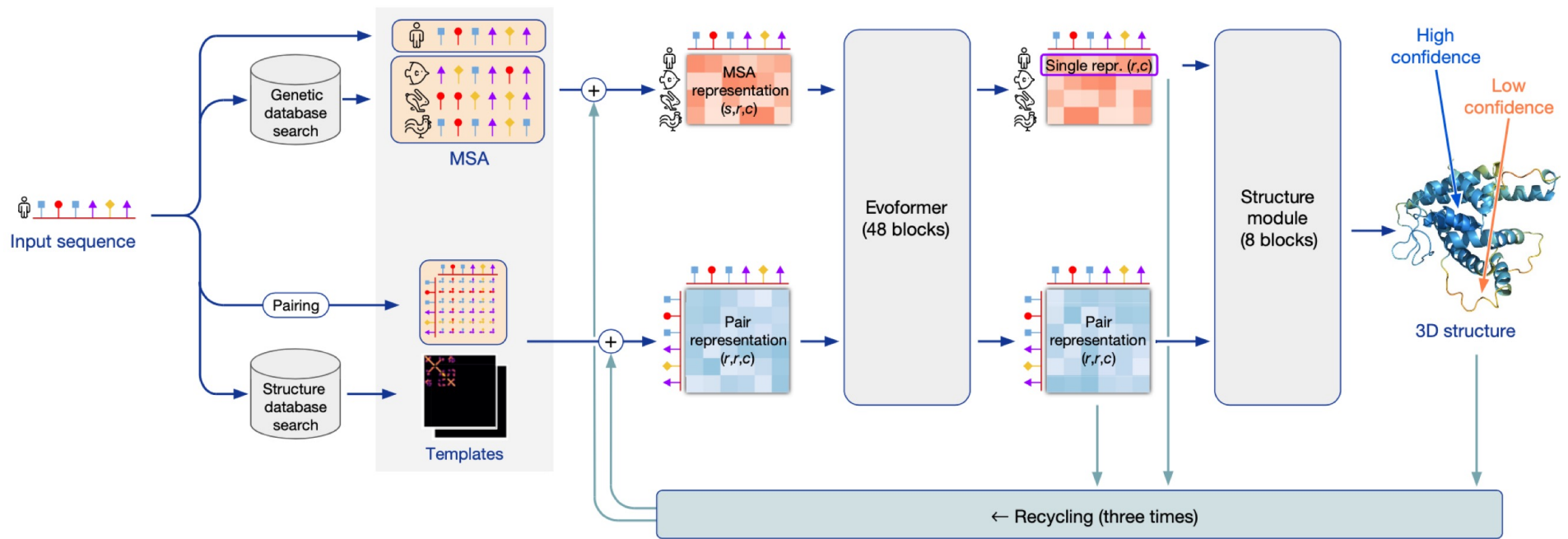
Domain-specific foundation models (DSFMs)

- LLMs are trained on natural language sequences
- DSFMs are **large pretrained models**, often transformer-based, that are tailored to the data distributions, task formats, and reasoning patterns of a scientific or technical domain. They may:
 - Encode **domain-specific priors** (e.g., molecular structure, protein folding rules, physics laws)
 - Leverage **structured data formats** (e.g., SMILES, PDB, netCDF, crystal graphs)
 - Support **inference + generation + simulation** (beyond natural language)

DSFM landscape

Domain	Model Types	Key Traits
Biology	Sequence, Structure, Docking	Pretrained on PDB, UniProt, SMILES
Materials	Graphs, Equivariant GNNs	Crystal symmetry, DFT alignment
Climate	Spatiotemporal Transformers	CMIP, ERA5, nowcasting
Physics	PINNs, Operators, Solvers	Embeds physical laws
Math	Formal logic, Step-by-step solvers	Symbolic & neural proof assistants
Multimodal	Vision–language–structure	SEM, histology, spectra, graphs

Example: AlphaFold



Jumper et al, Nature, 2021

Example: Stormer

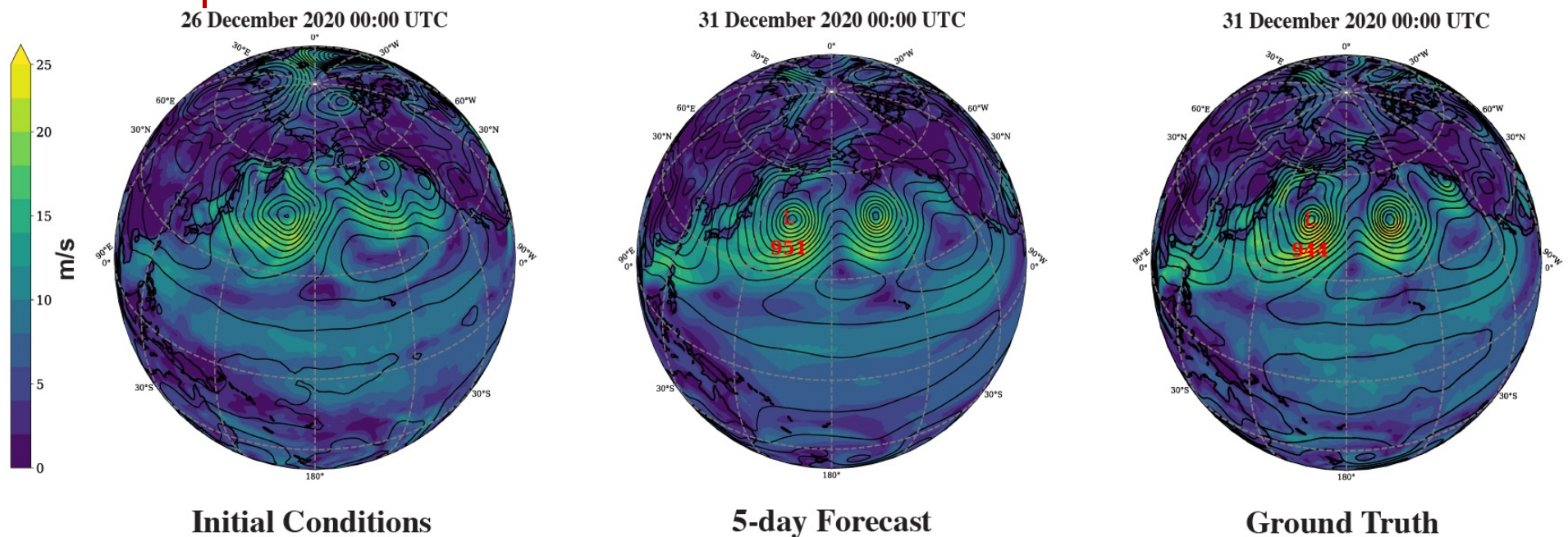


Fig 1. Illustration of an example 5-day forecast of near-surface wind speed (color-fill) and mean sea level pressure (contours). On December 31, 2020, an extratropical cyclone impacted Alaska setting a new North Pacific low-pressure record. Here, we evaluate the ability of Stormer to predict this record-breaking event 5 days in advance. Using initial conditions from 0000 UTC, 26 December 2011, Stormer was able to successfully forecast both the location and strength of this extreme event with great accuracy.

<https://arxiv.org/abs/2312.03876>