Language Model Programming: Themes and Prospects

Kyle Richardson

University of Tübingen

November 2026

Philosophische Fakultät |
Seminar für Sprachwissenschaft

TüLLI-AI
Language, Learning & Inference

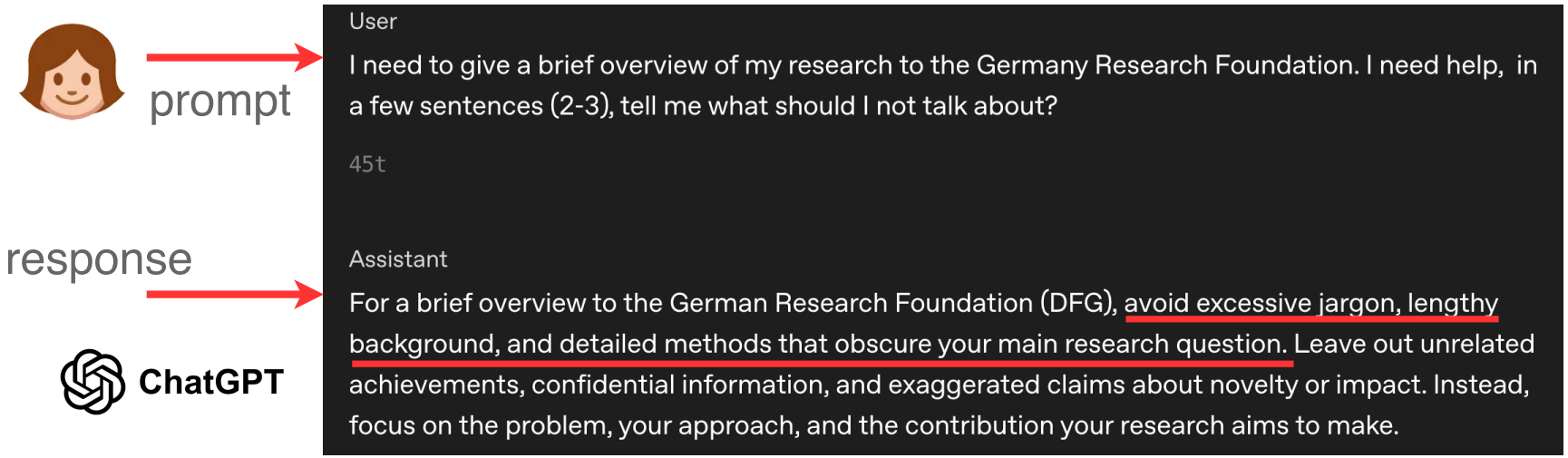
machine learning
new perspectives
for science

I work on **language models**

I work on **language models**

. . . and for years, I could never explain what that meant.

General purpose large language models (LLMs)



- **General purpose models**: at the foundation of today's generative AI, used for a wide range of problems.

General purpose large language models (LLMs)

prompt

User

I need to give a brief overview of my research to the Germany Research Foundation. I need help, in a few sentences (2-3), tell me what should I not talk about?

response

Assistant

For a brief overview to the German Research Foundation (DFG), avoid excessive jargon, lengthy [illegible] leave out unrelated achievements, confidential information, and exaggerated claims about novelty or impact. Instead, focus on the problem, your approach, and the contribution your research aims to make.

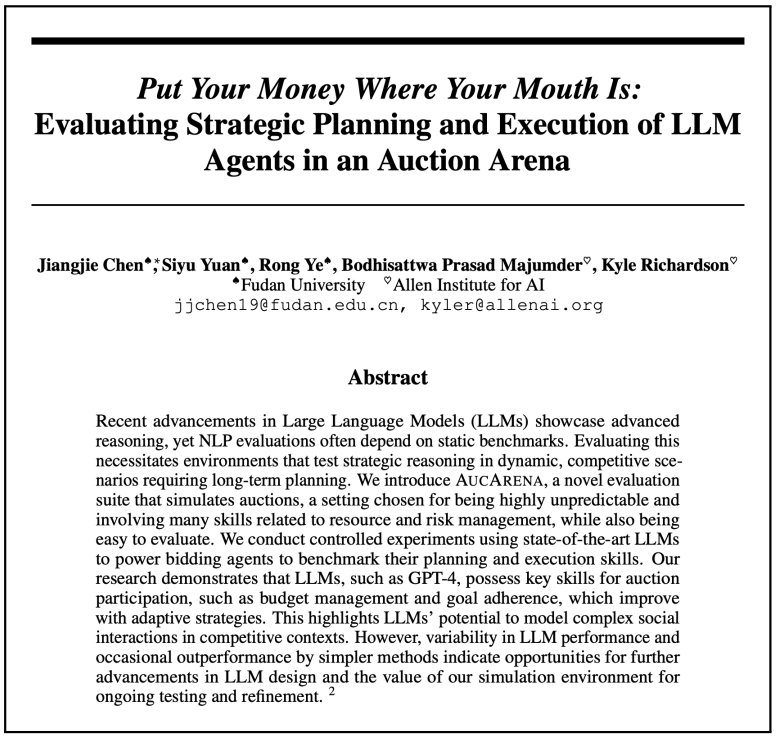
ChatGPT

Models have far exceeded expectations

opening up new applications and new research questions

- **General purpose models**: at the foundation of today's generative AI, used for a wide range of problems.

Language models as agent simulators



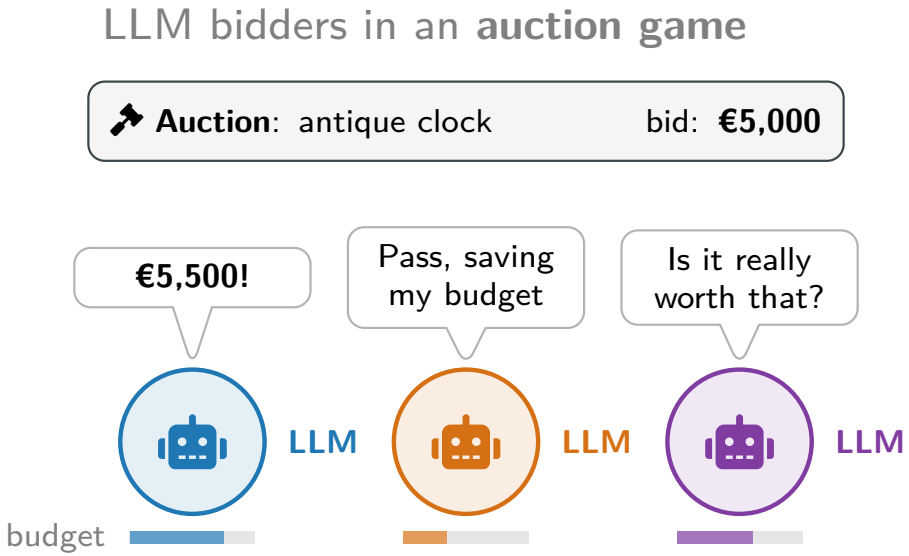
Put Your Money Where Your Mouth Is:
Evaluating Strategic Planning and Execution of LLM Agents in an Auction Arena

Jiangjie Chen[♠]*, **Siyu Yuan**[♠], **Rong Ye**[♠], **Bodhisattwa Prasad Majumder**[♡], **Kyle Richardson**[♡]
[♠]Fudan University [♡]Allen Institute for AI
jjchen19@fudan.edu.cn, kyler@allenai.org

Abstract

Recent advancements in Large Language Models (LLMs) showcase advanced reasoning, yet NLP evaluations often depend on static benchmarks. Evaluating this necessitates environments that test strategic reasoning in dynamic, competitive scenarios requiring long-term planning. We introduce AUCARENA, a novel evaluation suite that simulates auctions, a setting chosen for being highly unpredictable and involving many skills related to resource and risk management, while also being easy to evaluate. We conduct controlled experiments using state-of-the-art LLMs to power bidding agents to benchmark their planning and execution skills. Our research demonstrates that LLMs, such as GPT-4, possess key skills for auction participation, such as budget management and goal adherence, which improve with adaptive strategies. This highlights LLMs' potential to model complex social interactions in competitive contexts. However, variability in LLM performance and occasional outperformance by simpler methods indicate opportunities for further advancements in LLM design and the value of our simulation environment for ongoing testing and refinement. [2]

(Chen et al.2023)



- Do LLM agents behave like people? Can we simulate complex social dynamics? (Chen et al.2023; Zhang et al.2024; Yang et al.2025)

Language models as agent simulators

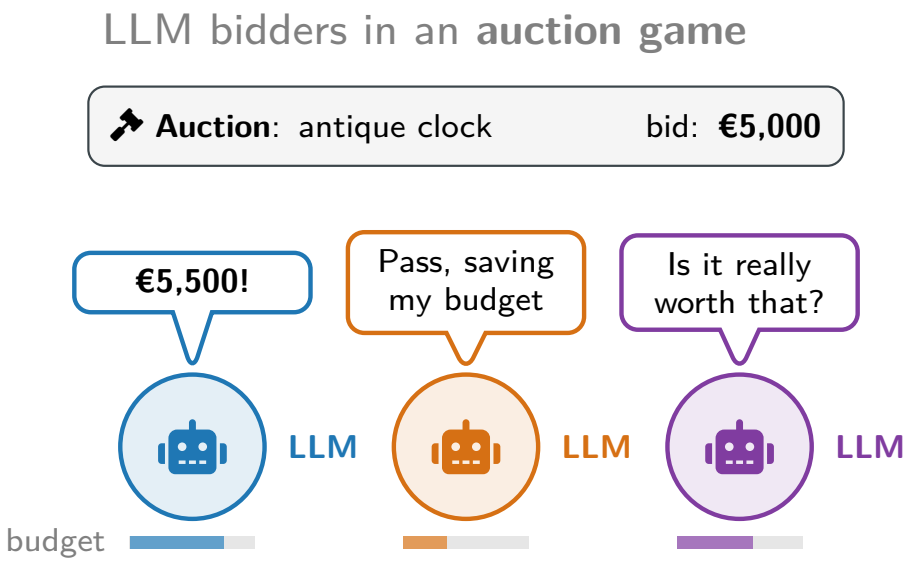
Put Your Money Where Your Mouth Is:
Evaluating Strategic Planning and Execution of LLM Agents in an Auction Arena

Jiangjie Chen[♠], **Siyu Yuan**[♠], **Rong Ye**[♠], **Bodhisattwa Prasad Majumder**[♡], **Kyle Richardson**[♡]
[♠]Fudan University [♡]Allen Institute for AI
jjchen19@fudan.edu.cn, kyler@allenai.org

Abstract

Recent advancements in Large Language Models (LLMs) showcase advanced reasoning, yet NLP evaluations often depend on static benchmarks. Evaluating this necessitates environments that test strategic reasoning in dynamic, competitive scenarios requiring long-term planning. We introduce AUCARENA, a novel evaluation suite that simulates auctions, a setting chosen for being highly unpredictable and involving many skills related to resource and risk management, while also being easy to evaluate. We conduct controlled experiments using state-of-the-art LLMs to power bidding agents to benchmark their planning and execution skills. Our research demonstrates that LLMs, such as GPT-4, possess key skills for auction participation, such as budget management and goal adherence, which improve with adaptive strategies. This highlights LLMs' potential to model complex social interactions in competitive contexts. However, variability in LLM performance and occasional outperformance by simpler methods indicate opportunities for further advancements in LLM design and the value of our simulation environment for ongoing testing and refinement. [2]

(Chen et al.2023)



How do they **compete & strategize**?
How do they **manage resources**?
How do they reason **under uncertainty**?

- Do LLM agents behave like people? Can we simulate complex social dynamics? (Chen et al.2023; Zhang et al.2024; Yang et al.2025)

Language models as agent simulators

Put Your Money Where Your Mouth Is:
Evaluating Strategic Planning and Execution of LLM Agents in an Auction Arena

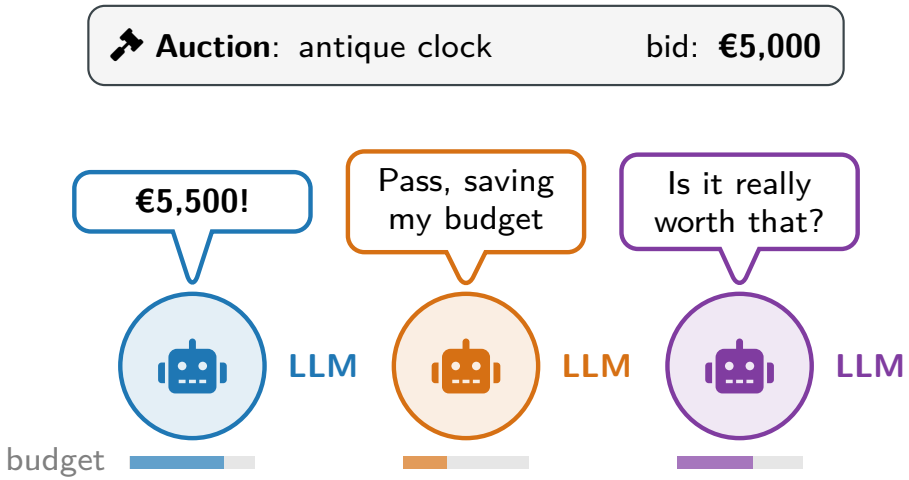
Jiangjie Chen[♠], **Siyu Yuan**[♠], **Rong Ye**[♠], **Bodhisattwa Prasad Majumder**[♡], **Kyle Richardson**[♡]
[♠]Fudan University [♡]Allen Institute for AI
jjchen19@fudan.edu.cn, kyler@allenai.org

Abstract

Recent advancements in Large Language Models (LLMs) showcase advanced reasoning, yet NLP evaluations often depend on static benchmarks. Evaluating this necessitates environments that test strategic reasoning in dynamic, competitive scenarios requiring long-term planning. We introduce AUCARENA, a novel evaluation suite that simulates auctions, a setting chosen for being highly unpredictable and involving many skills related to resource and risk management, while also being easy to evaluate. We conduct controlled experiments using state-of-the-art LLMs to power bidding agents to benchmark their planning and execution skills. Our research demonstrates that LLMs, such as GPT-4, possess key skills for auction participation, such as budget management and goal adherence, which improve with adaptive strategies. This highlights LLMs' potential to model complex social interactions in competitive contexts. However, variability in LLM performance and occasional outperformance by simpler methods indicate opportunities for further advancements in LLM design and the value of our simulation environment for ongoing testing and refinement. [2]

(Chen et al.2023)

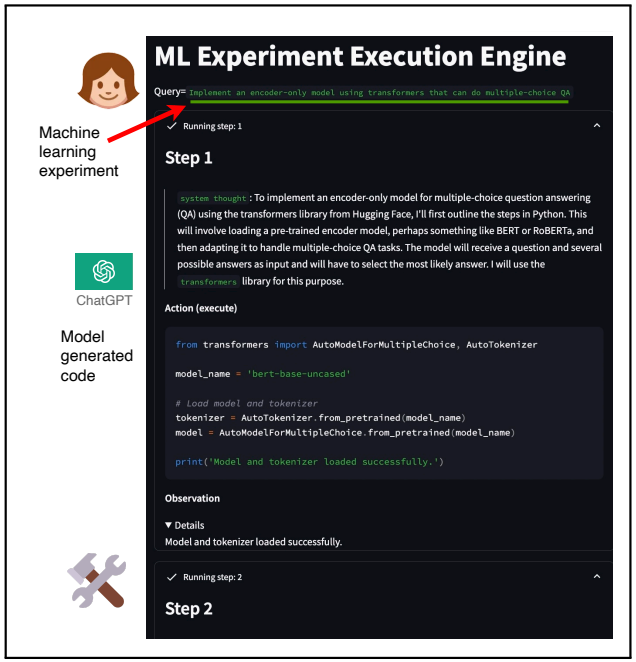
LLM bidders in an **auction game**

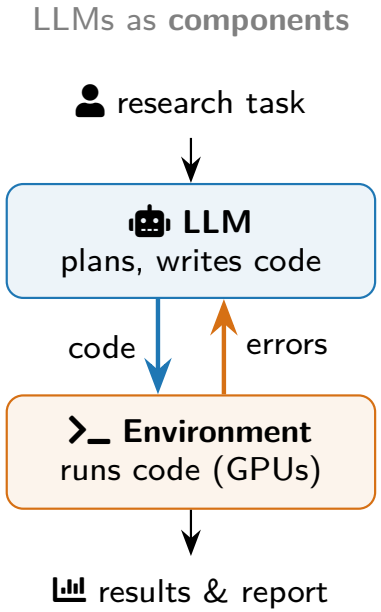


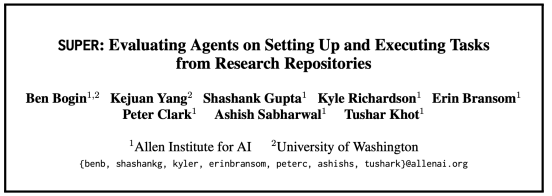
How do they **compete & strategize**?
How do they **manage resources**?
How do they reason **under uncertainty**?

Why it matters: LLM agents will become foundational to how we **simulate** and **study** society.

Language models as part of complex systems







SUPER: Evaluating Agents on Setting Up and Executing Tasks from Research Repositories

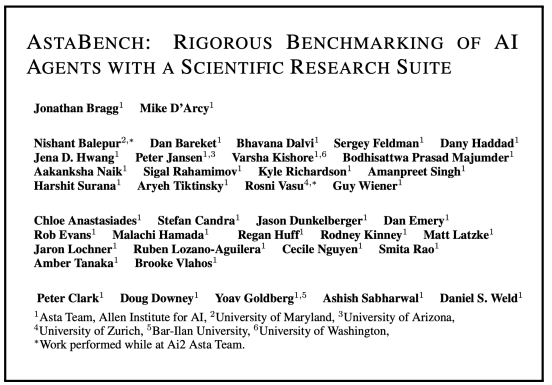
Ben Bogin[1,2] Kejuan Yang[2] Shashank Gupta[1] Kyle Richardson[1] Erin Bransom[1] Peter Clark[1] Ashish Sabharwal[1] Tushar Khot[1]

[1]Allen Institute for AI [2]University of Washington

{benb, shashankg, kyler, erinbransom, peterc, ashishs, tushark}@allenai.org

Executing **research repositories**

(Bogin et al.2024)



ASTABENCH: RIGOROUS BENCHMARKING OF AI AGENTS WITH A SCIENTIFIC RESEARCH SUITE

Jonathan Bragg[1] Mike D'Arcy[1]

Nishant Balepur[2,*] Dan Bareket[1] Bhavana Dalvi[1] Sergey Feldman[1] Dany Haddad[1] Jena D. Hwang[1] Peter Jansen[1,3] Varsha Kishore[1,6] Bodhisattwa Prasad Majumder[1] Aakanksha Naik[1] Sigal Rahamimov[1] Kyle Richardson[1] Amanpreet Singh[1] Harshit Surana[1] Aryeh Tiktinsky[1] Rosni Vasu[4,*] Guy Wiener[1]

Chloe Anastasiades[1] Stefan Candra[1] Jason Dunkelberger[1] Dan Emery[1] Rob Evans[1] Malachi Hamada[1] Regan Huff[1] Rodney Kinney[1] Matt Latzke[1] Jaron Lochner[1] Ruben Lozano-Aguilera[1] Cecile Nguyen[1] Smita Rao[1] Amber Tanaka[1] Brooke Vlahos[1]

Peter Clark[1] Doug Downey[1] Yoav Goldberg[1,5] Ashish Sabharwal[1] Daniel S. Weld[1]

[1]Asta Team, Allen Institute for AI, [2]University of Maryland, [3]University of Arizona, [4]University of Zurich, [5]Bar-Ilan University, [6]University of Washington, [*]Work performed while at Ai2 Asta Team.

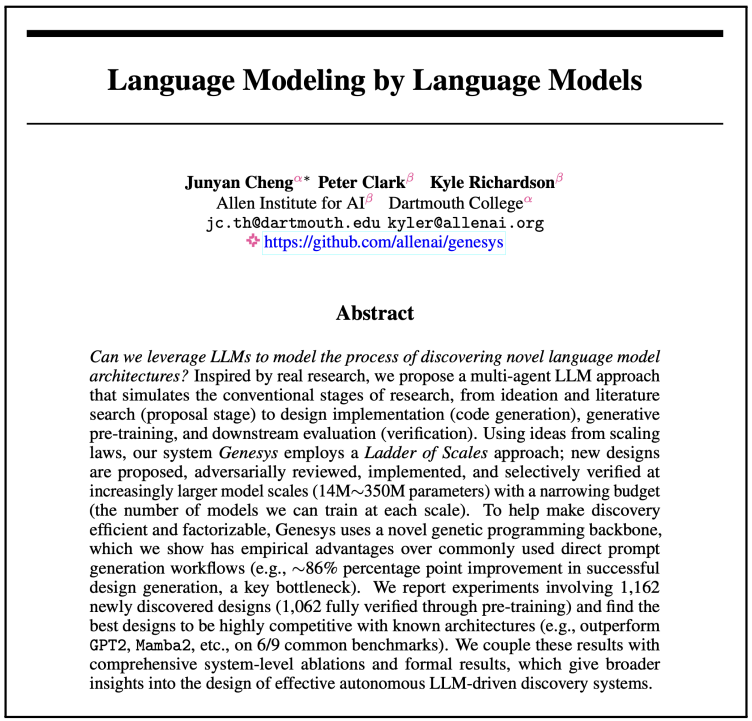
Benchmarking **science agents**

(Bragg et al.2026)

- **Agent development and benchmarking**: building agents that write, run and debug code, and benchmarks to measure them.

Language models as part of complex systems

Can LLM agents **discover new language model architectures** on their own?

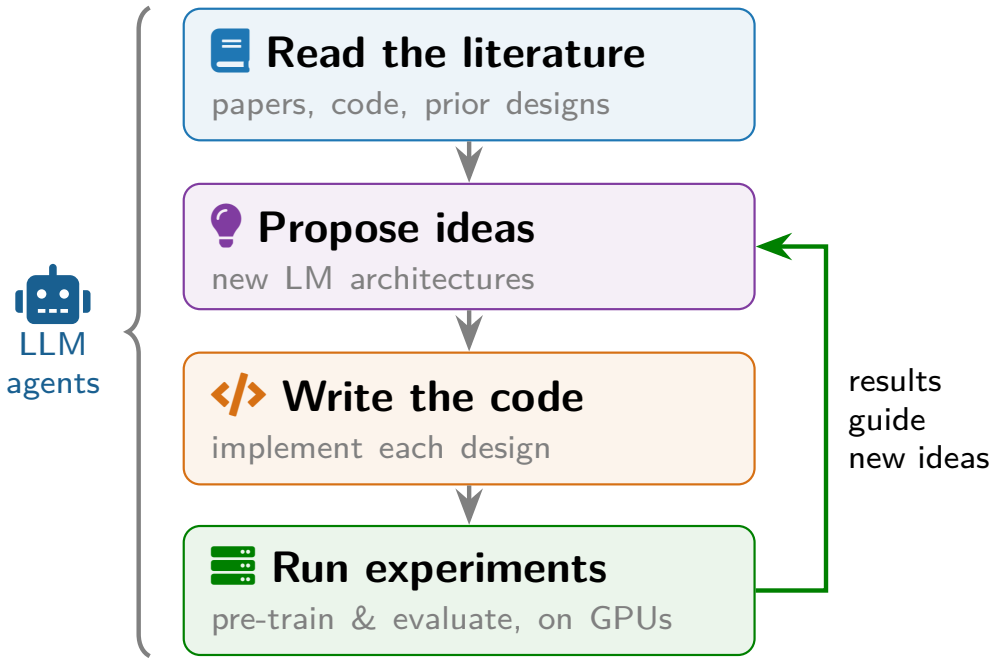


Language Modeling by Language Models

Junyan Cheng[α]* **Peter Clark**[β] **Kyle Richardson**[β]
Allen Institute for AI[β] Dartmouth College[α]
jc.th@dartmouth.edu kyler@allenai.org
https://github.com/allenai/genesys

Abstract

Can we leverage LLMs to model the process of discovering novel language model architectures? Inspired by real research, we propose a multi-agent LLM approach that simulates the conventional stages of research, from ideation and literature search (proposal stage) to design implementation (code generation), generative pre-training, and downstream evaluation (verification). Using ideas from scaling laws, our system *Genesys* employs a *Ladder of Scales* approach; new designs are proposed, adversarially reviewed, implemented, and selectively verified at increasingly larger model scales (14M∼350M parameters) with a narrowing budget (the number of models we can train at each scale). To help make discovery efficient and factorizable, Genesys uses a novel genetic programming backbone, which we show has empirical advantages over commonly used direct prompt generation workflows (e.g., ∼86% percentage point improvement in successful design generation, a key bottleneck). We report experiments involving 1,162 newly discovered designs (1,062 fully verified through pre-training) and find the best designs to be highly competitive with known architectures (e.g., outperform GPT2, Mamba2, etc., on 6/9 common benchmarks). We couple these results with comprehensive system-level ablations and formal results, which give broader insights into the design of effective autonomous LLM-driven discovery systems.



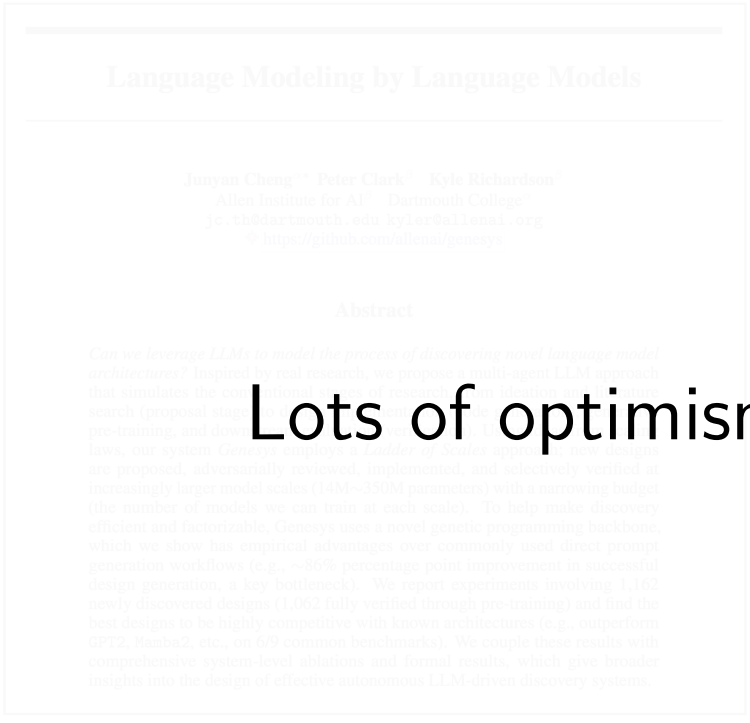
- **Automated research agents** for scientific discovery: from reading papers to running experiments, writing papers (Cheng et al.2025; Yu et al.2026).



Closely tied to the broader goals of Tübingen's **cluster of excellence**

Language models as part of complex systems

Can LLM agents **discover new language model architectures** on their own?

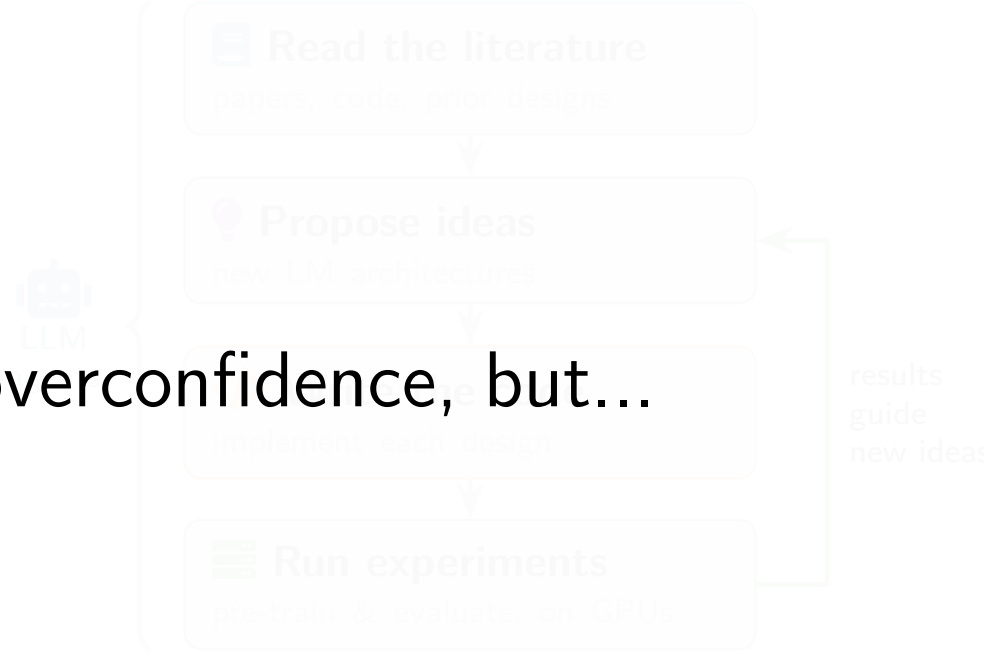


Language Modeling by Language Models

Junyan Cheng[1*] Peter Clark[2] Kyle Richardson[2]
Allen Institute for AI[2] Dartmouth College[1]
jc.th@dartmouth.edu ky1er@allenai.org
* https://github.com/allenai/genesys

Abstract

Can we leverage LLMs to model the process of discovering novel language model architectures? Inspired by real research, we propose a multi-agent LLM approach that simulates the conventional stages of research, from ideation and literature search (proposal stage) to design implementation and pre-training, and down to analysis and paper writing (verification stage). To scale up the search (proposal stage) and improve the success rate of implementation (verification stage), we propose adversarially reviewed, implemented, and selectively verified at increasingly larger model scales (14M~350M parameters) with a narrowing budget (the number of models we can train at each scale). To help make discovery efficient and factorizable, Genesys uses a novel genetic programming backbone, which we show has empirical advantages over commonly used direct prompt generation workflows (e.g., ~86% percentage point improvement in successful design generation, a key bottleneck). We report experiments involving 1,162 newly discovered designs (1,062 fully verified through pre-training) and find the best designs to be highly competitive with known architectures (e.g., outperform GPT2, Mamba2, etc., on 6/9 common benchmarks). We couple these results with comprehensive system-level ablations and formal results, which give broader insights into the design of effective autonomous LLM-driven discovery systems.



Lots of optimism, overconfidence, but...

- **Automated research agents** for scientific discovery: from reading papers to running experiments, writing papers (Cheng et al.2025; Yu et al.2026).

Problems

at the core of today's AI research

Problems

1. Scale and commercial dominance

Problems

2. Missing algorithmic and semantic foundations.

(i.e., we often don't know what we are doing)

1. Open source large language model development



: Accelerating the Science of Language Models

Dirk Groeneveld[α] Iz Beltagy[α]

Pete Walsh[α] Akshita Bhagia[α] Rodney Kinney[α] Oyvind Tafjord[α]

Ananya Harsh Jha[α] Hamish Ivison[α,β] Ian Magnusson[α] Yizhong Wang[α,β]

Shane Arora[α] David Atkinson[α] Russell Authur[α] Khyathi Raghavi Chandu[α]

Arman Cohan[γ,α] Jennifer Dumas[α] Yanai Elazar[α,β] Yuling Gu[α]

Jack Hessel[α] Tushar Khot[α] William Merrill[δ] Jacob Morrison[α]

Niklas Muennighoff Aakanksha Naik[α] Crystal Nam[α] Matthew E. Peters[α]

Valentina Pyatkin[α,β] Abhilasha Ravichander[α] Dustin Schwenk[α] Saurabh Shah[α]

Will Smith[α] Emma Strubell[α,μ] Nishant Subramani[α] Mitchell Wortsman[β]

Pradeep Dasigi[α] Nathan Lambert[α] Kyle Richardson[α]

Luke Zettlemoyer[β] Jesse Dodge[α] Kyle Lo[α] Luca Soldaini[α]

Noah A. Smith[α,β] Hannaneh Hajishirzi[α,β]

[α]Allen Institute for Artificial Intelligence

[β]University of Washington [γ]Yale University

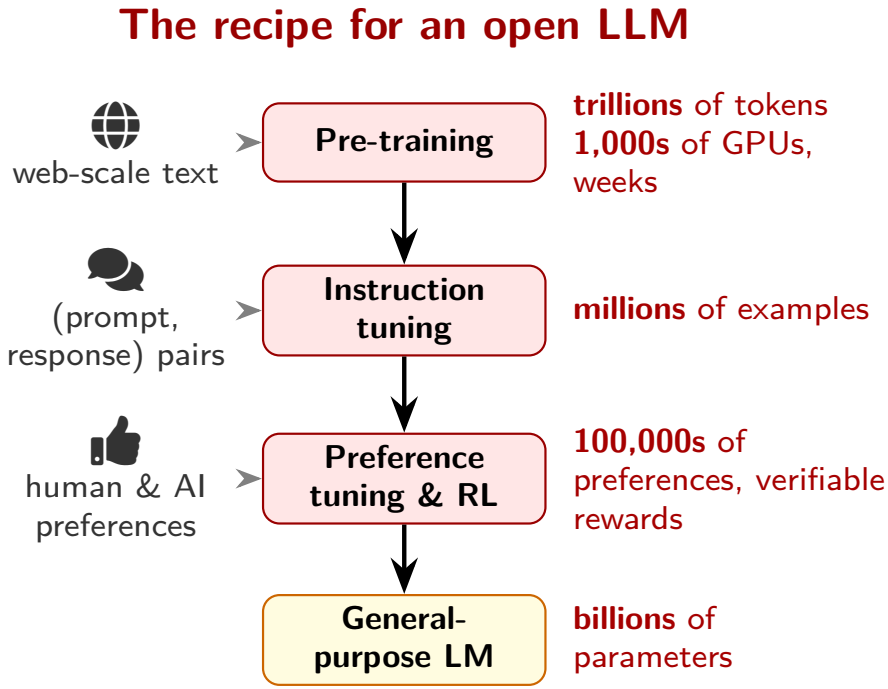
[δ]New York University [μ]Carnegie Mellon University

olmo@allenai.org

(Groeneveld et al.2024)

Olmo: fully open

data, code, checkpoints, logs



1. Open source large language model development



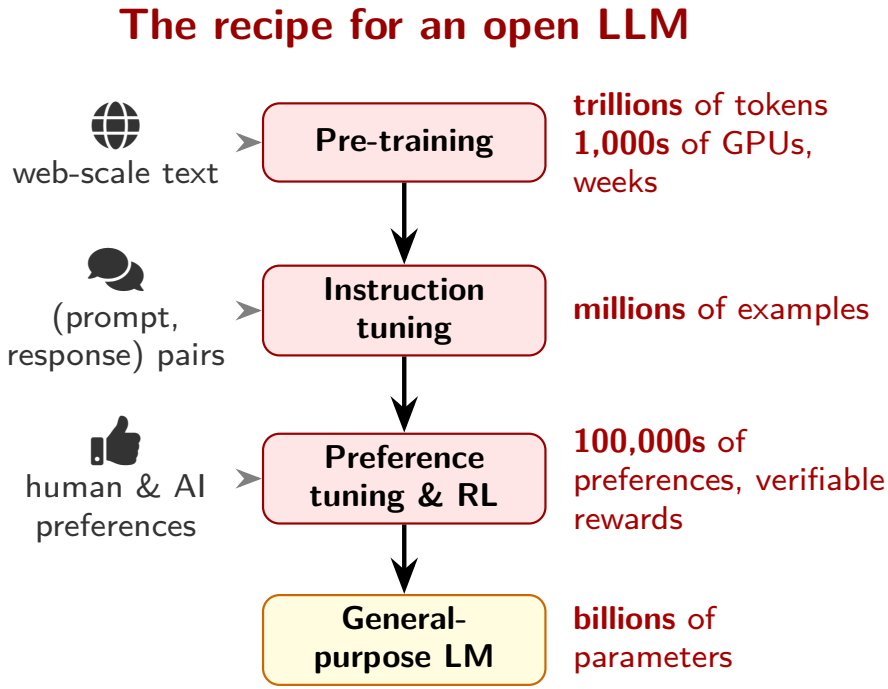
OLMo: Accelerating the Science of Language Models

Dirk Groeneveld[α] Iz Beltagy[α]
Pete Walsh[α] Akshita Bhagia[α] Rodney Kinney[α] Oyvind Tafjord[α]
Ananya Harsh Jha[α] Hamish Ivison[α,β] Ian Magnusson[α] Yizhong Wang[α,β]
Shane Arora[α] David Atkinson[α] Russell Authur[α] Khyathi Raghavi Chandu[α]
Arman Cohan[γ,α] Jennifer Dumas[α] Yanai Elazar[α,β] Yuling Gu[α]
Jack Hessel[α] Tushar Khot[α] William Merrill[δ] Jacob Morrison[α]
Niklas Muennighoff Aakanksha Naik[α] Crystal Nam[α] Matthew E. Peters[α]
Valentina Pyatkin[α,β] Abhilasha Ravichander[α] Dustin Schwenk[α] Saurabh Shah[α]
Will Smith[α] Emma Strubell[α,μ] Nishant Subramani[α] Mitchell Wortsman[β]
Pradeep Dasigi[α] Nathan Lambert[α] Kyle Richardson[α]
Luke Zettlemoyer[β] Jesse Dodge[α] Kyle Lo[α] Luca Soldaini[α]
Noah A. Smith[α,β] Hannaneh Hajishirzi[α,β]
[α]Allen Institute for Artificial Intelligence
[β]University of Washington [γ]Yale University
[δ]New York University [μ]Carnegie Mellon University
olmo@allenai.org

(Groeneveld et al.2024)

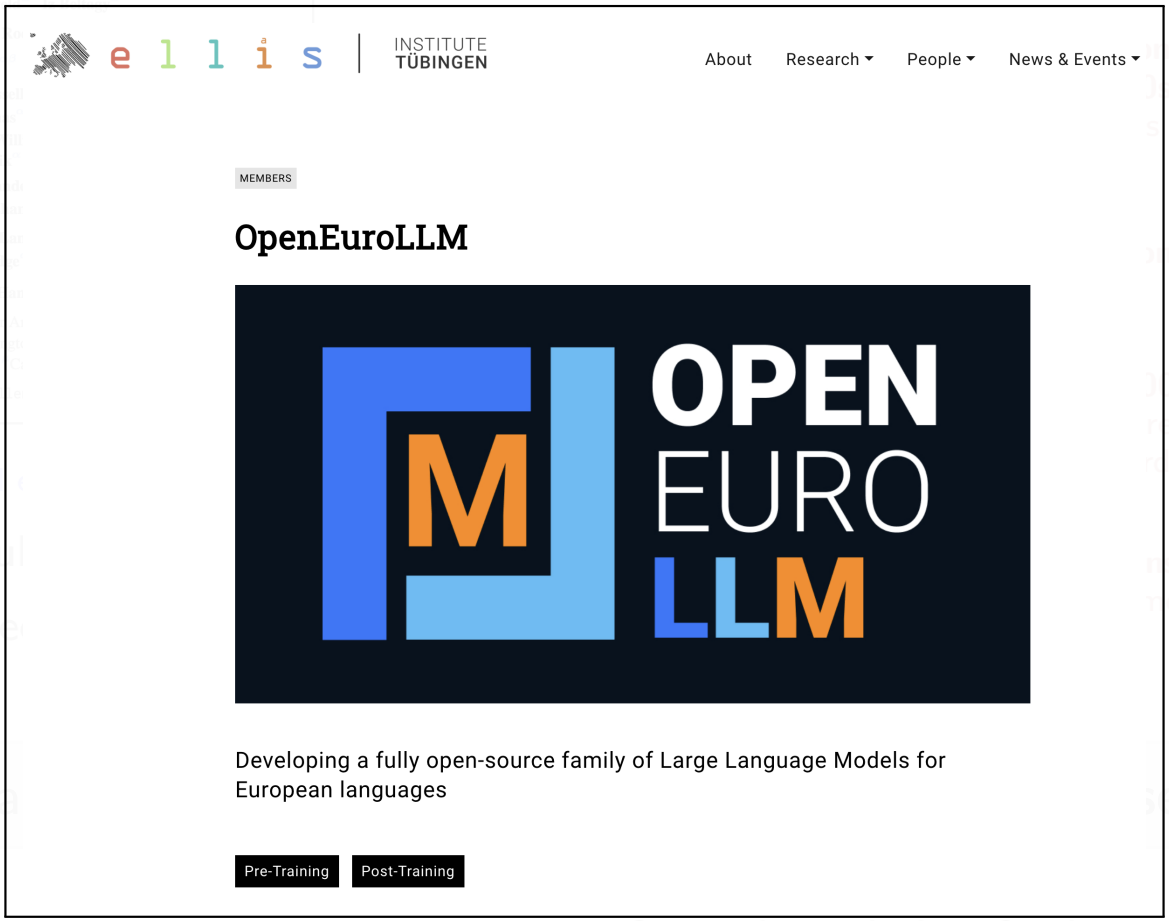
Olmo: fully open

data, code, checkpoints, logs



We can't understand LLMs unless we **build them ourselves**.

- Developing open source recipes, artifacts, knowledge for large language model development, targeted at the research community.

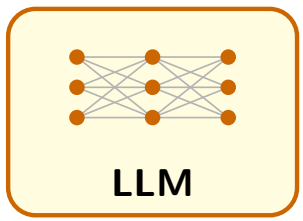


Connections to open model development in **Tübingen**

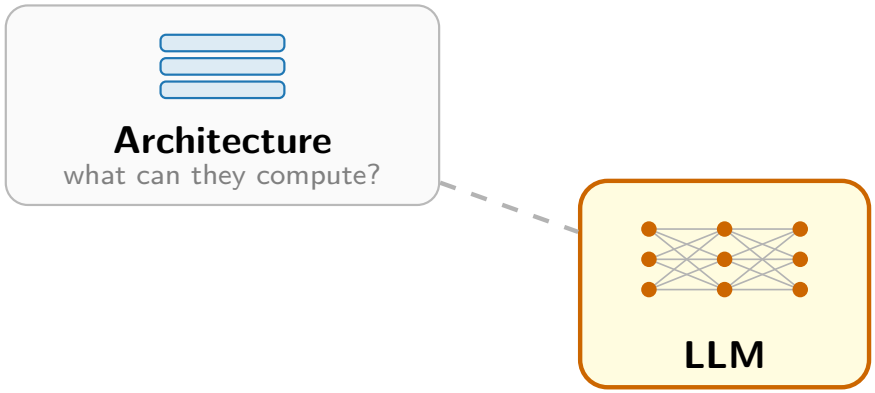
Problems

2. Missing algorithmic and semantic foundations.

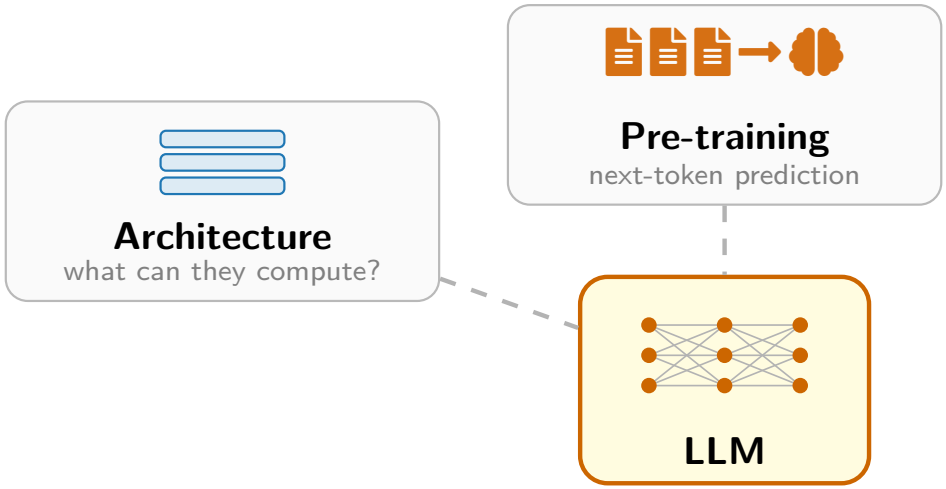
2. The landscape of modern AI research



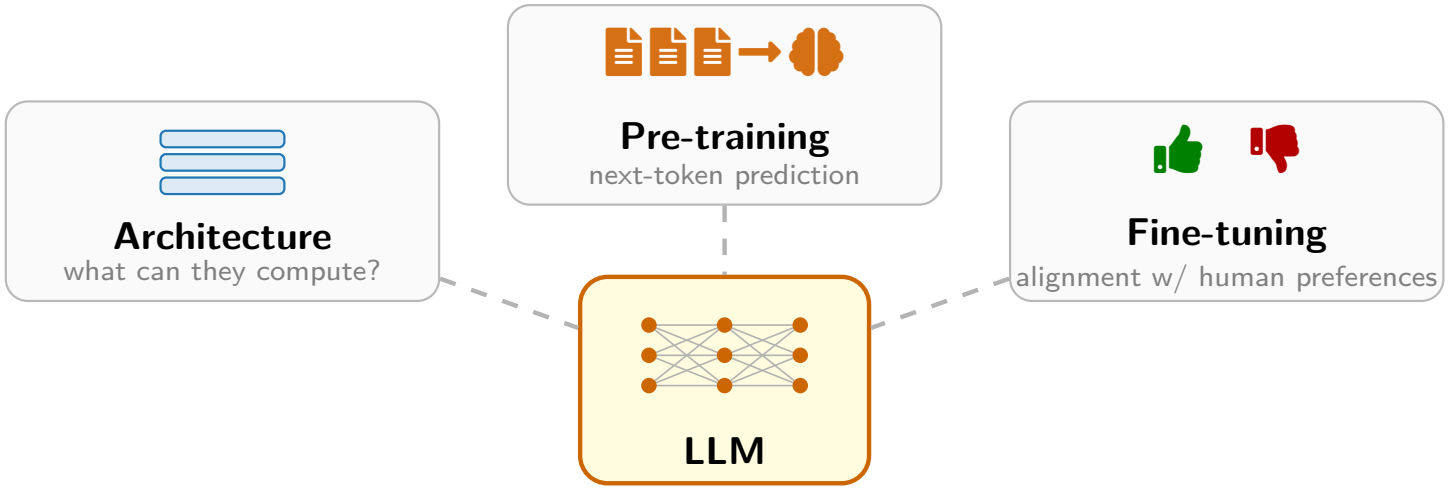
2. The landscape of modern AI research



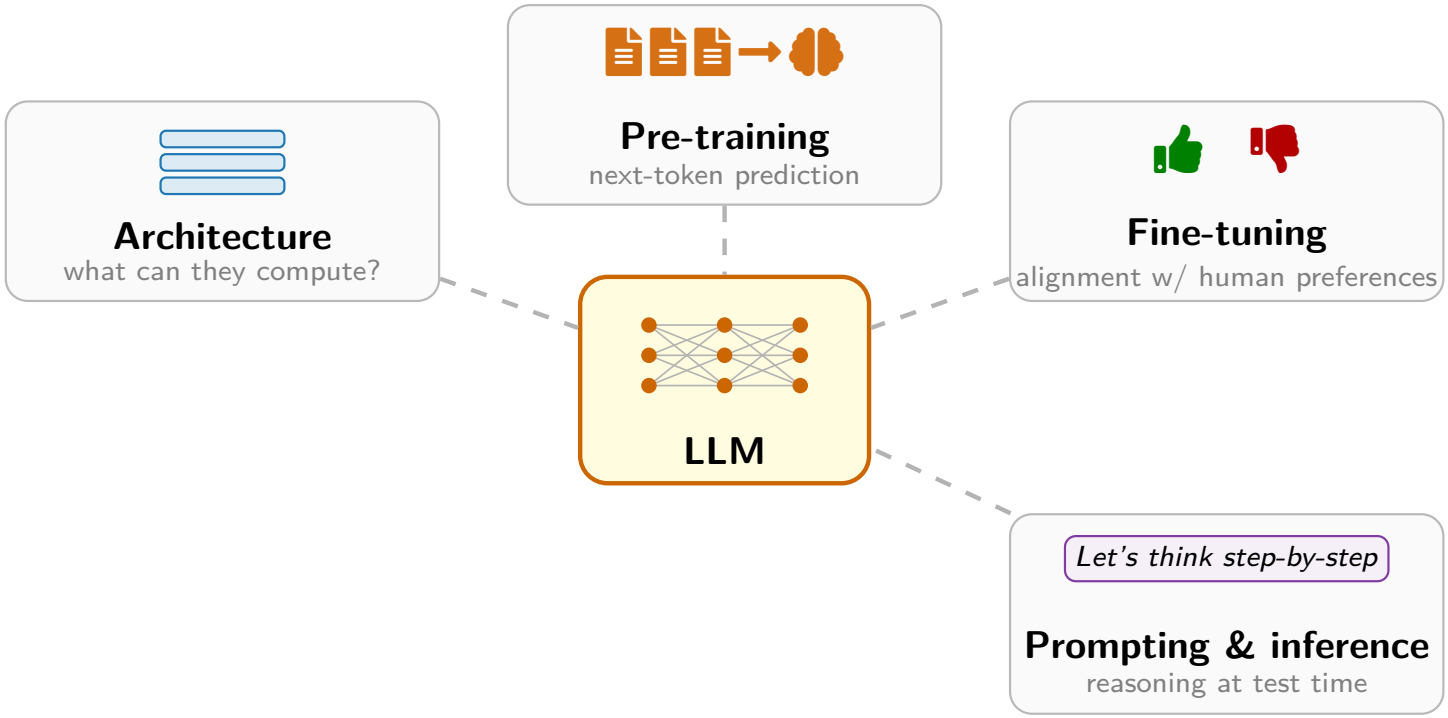
2. The landscape of modern AI research



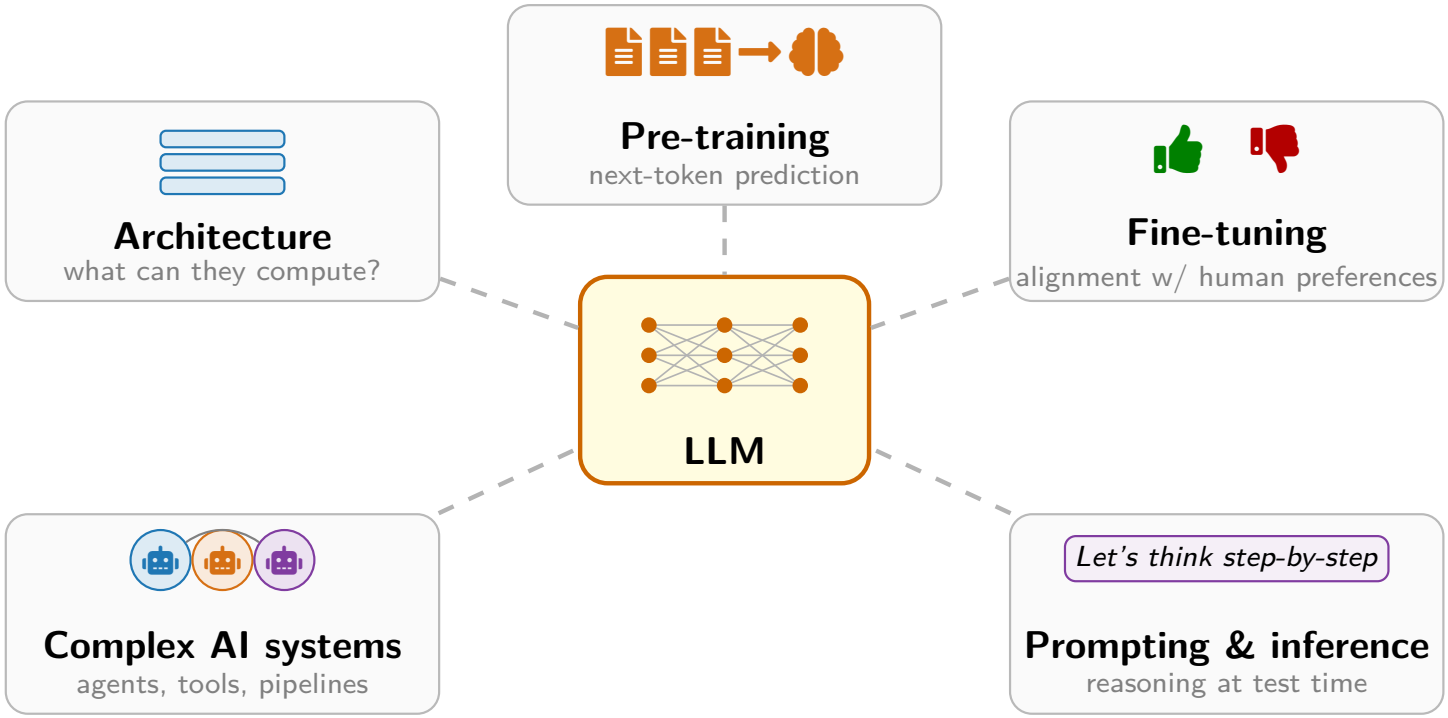
2. The landscape of modern AI research



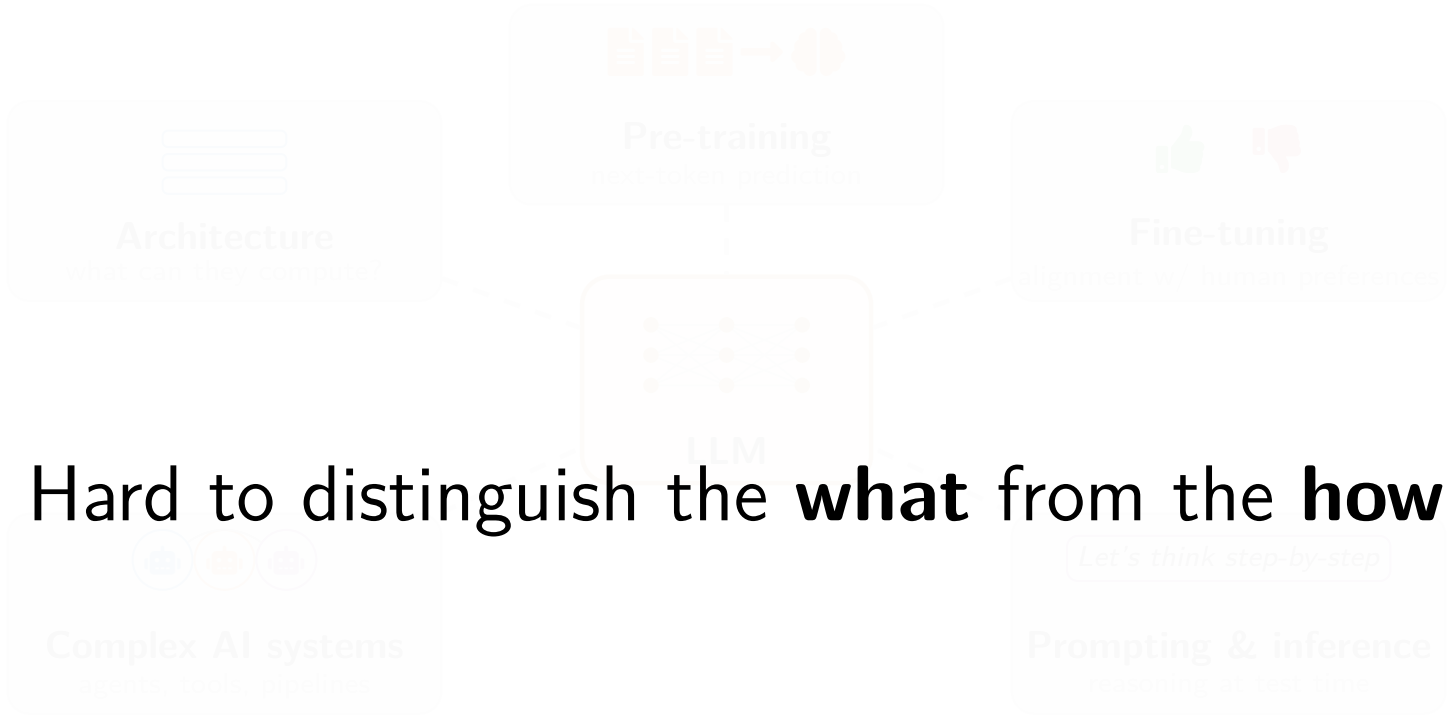
2. The landscape of modern AI research



2. The landscape of modern AI research

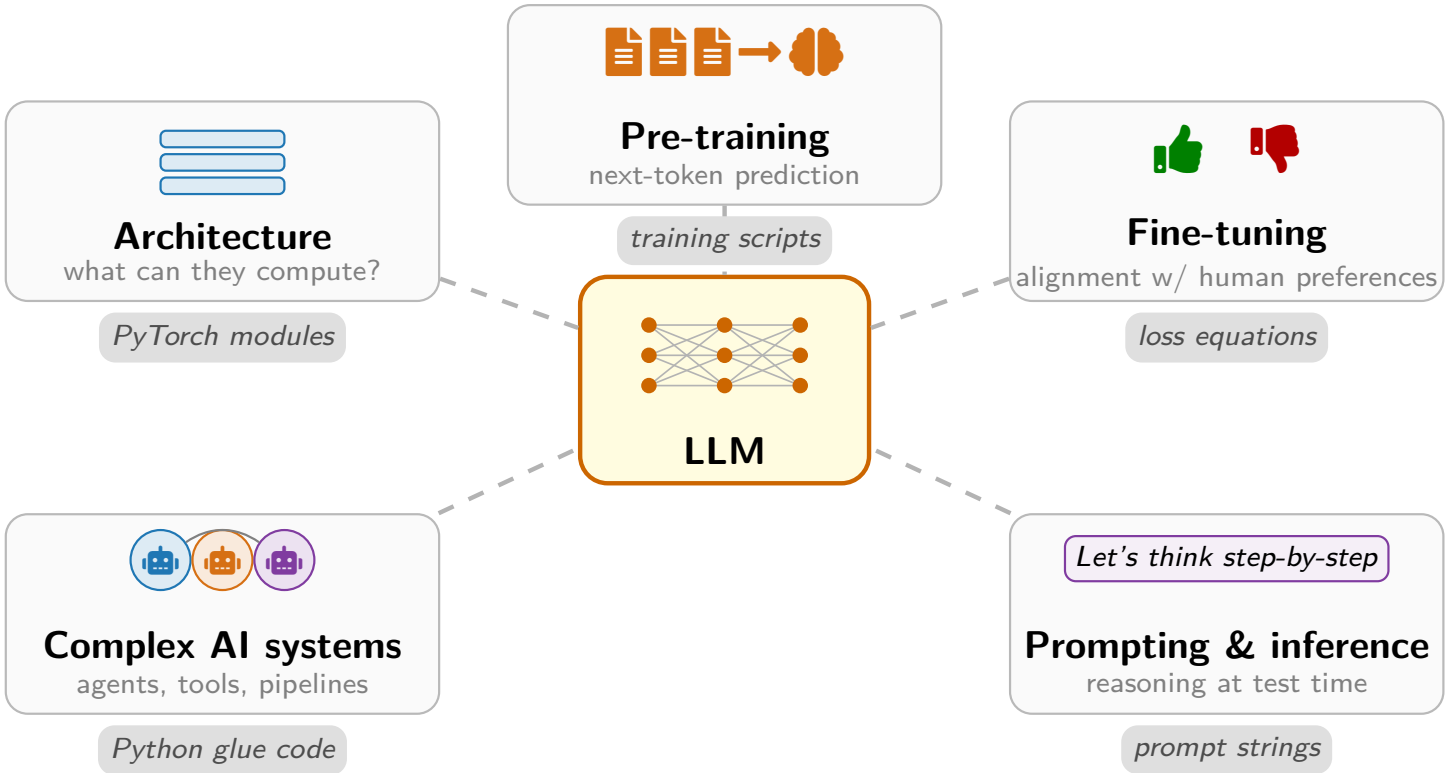


2. The landscape of modern AI research



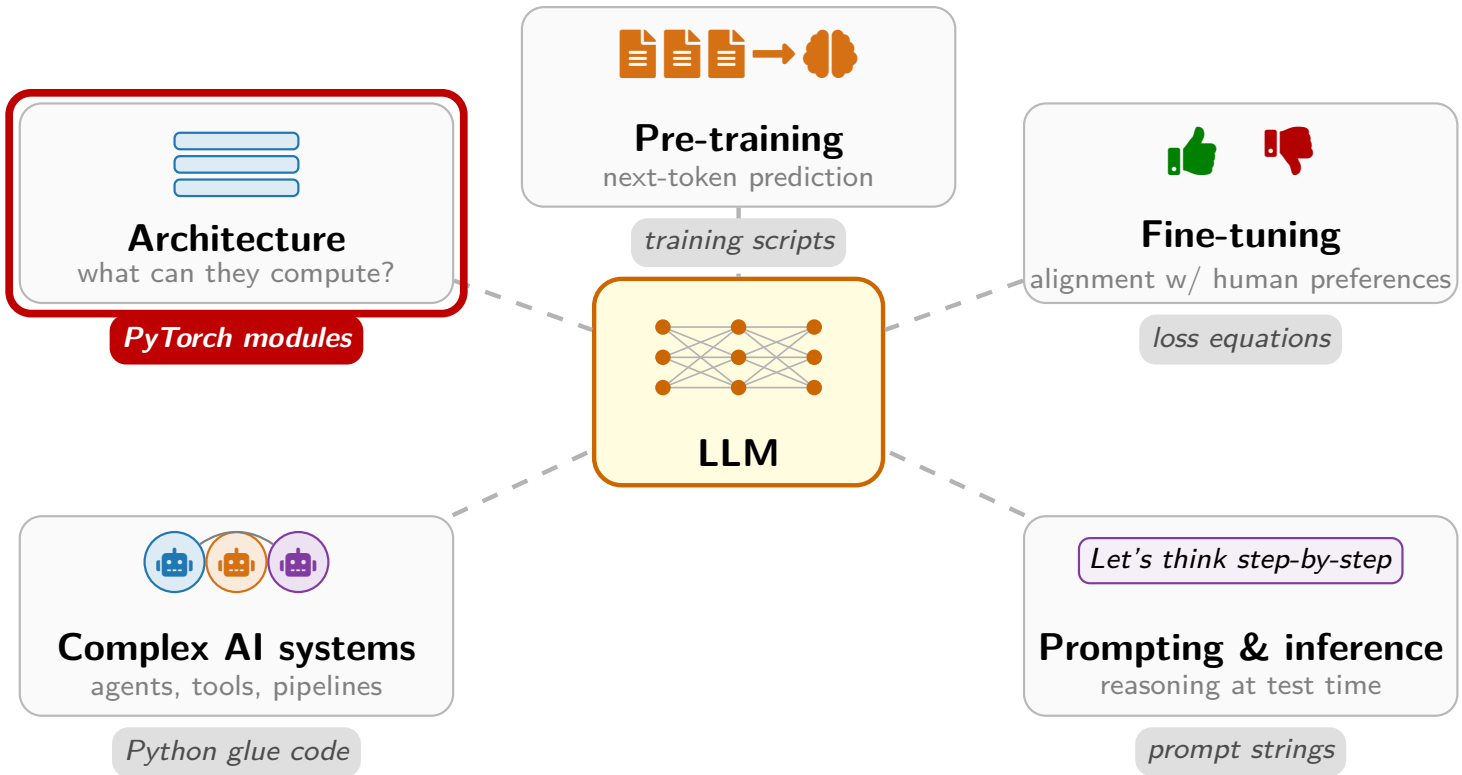
Hard to distinguish the **what** from the **how**

2. The landscape of modern AI research



- **Missing abstraction**: no rich modeling languages; making it very often hard to distinguish the *what* from the *how*.

2. The landscape of modern AI research



- **Missing abstraction**: no rich modeling languages; making it very often hard to distinguish the *what* from the *how*.

Architecture: PyTorch modules

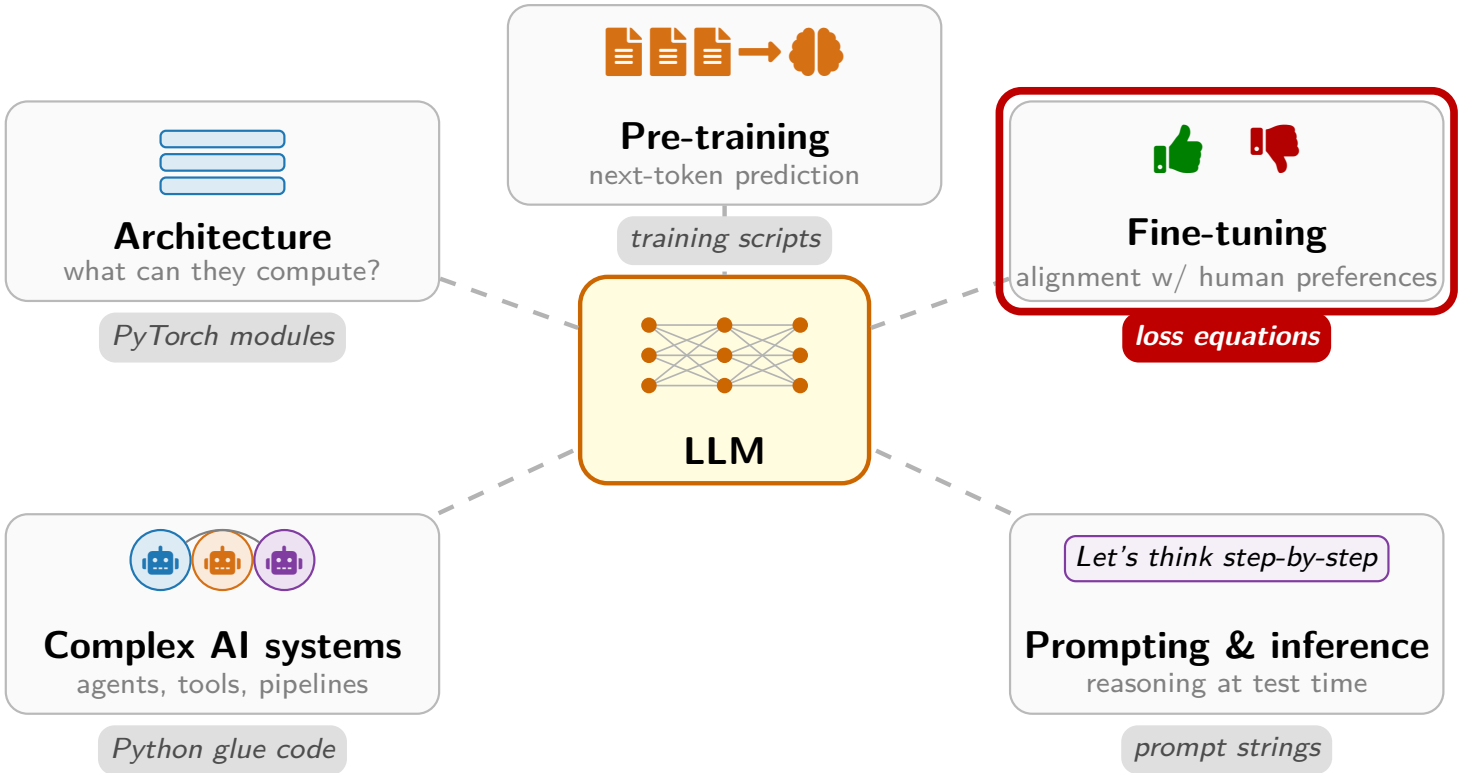
Causal self-attention in a transformer

```
def forward(self, x):
  B, T, C = x.size()
  q, k, v = self.c_attn(x).split(self.n_embd, dim=2)
  q = q.view(B, T, self.n_head, C // self.n_head).transpose(1, 2)
  k = k.view(B, T, self.n_head, C // self.n_head).transpose(1, 2)
  v = v.view(B, T, self.n_head, C // self.n_head).transpose(1, 2)
  att = (q @ k.transpose(-2, -1)) / math.sqrt(k.size(-1))
  att = att.masked_fill(self.mask[:, :, :T, :T] == 0, float('-inf'))
  att = F.softmax(att, dim=-1)
  y = (att @ v).transpose(1, 2).contiguous().view(B, T, C)
  return self.c_proj(y)
```

What can this *compute*? What *can't* it?

The algorithm is buried in tensor operations.

2. The landscape of modern AI research



- **Missing abstraction**: no rich modeling languages; making it very often hard to distinguish the *what* from the *how*.

Fine-tuning: loss equations

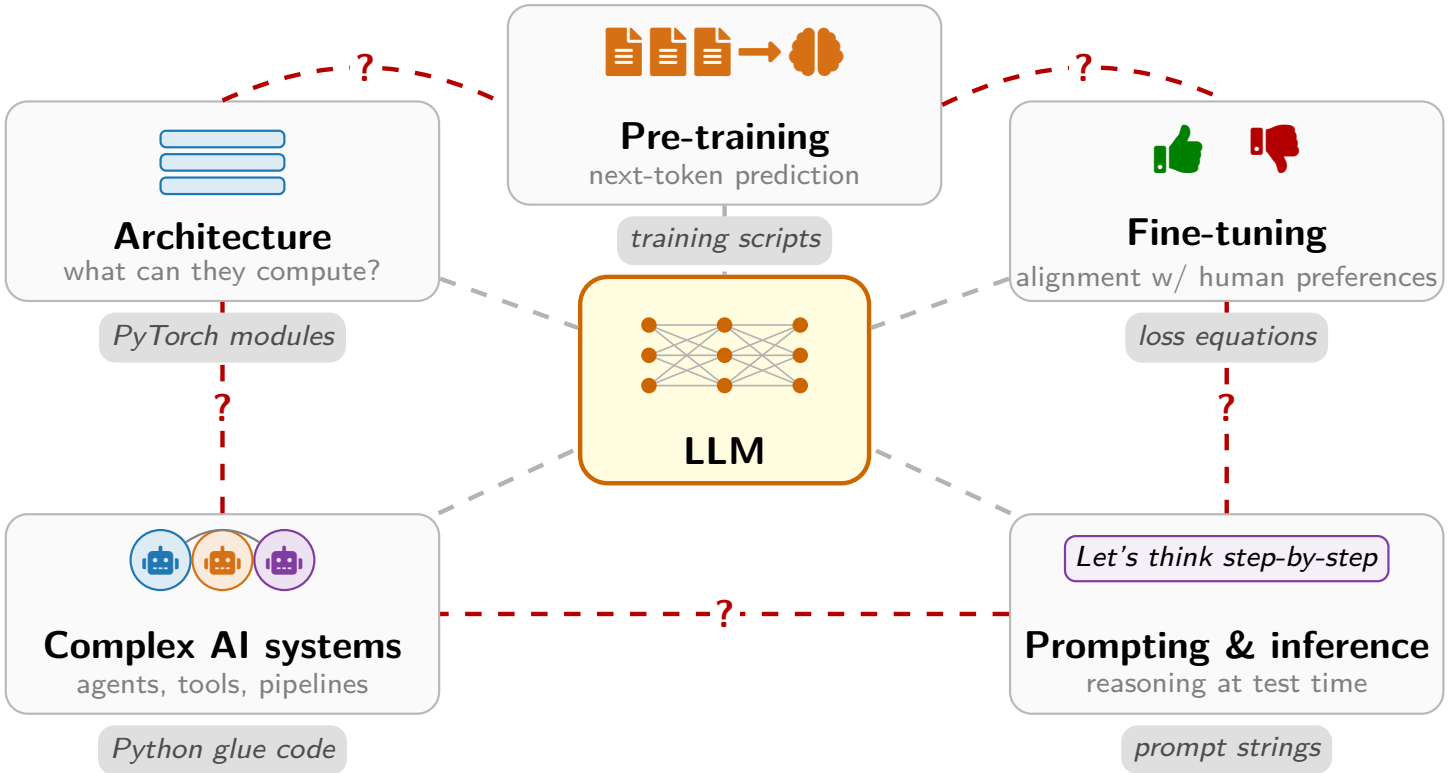
The DPO loss (Rafailov et al.2024)

$$\mathcal{L}_{\text{DPO}} = -\,\mathbb{E}_{(x,\,y_w,\,y_l)}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} \;-\; \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

What is this loss actually *saying*? How would we design a *new* one?

The algorithm is an equation, not a specification of what we want.

2. The landscape of modern AI research



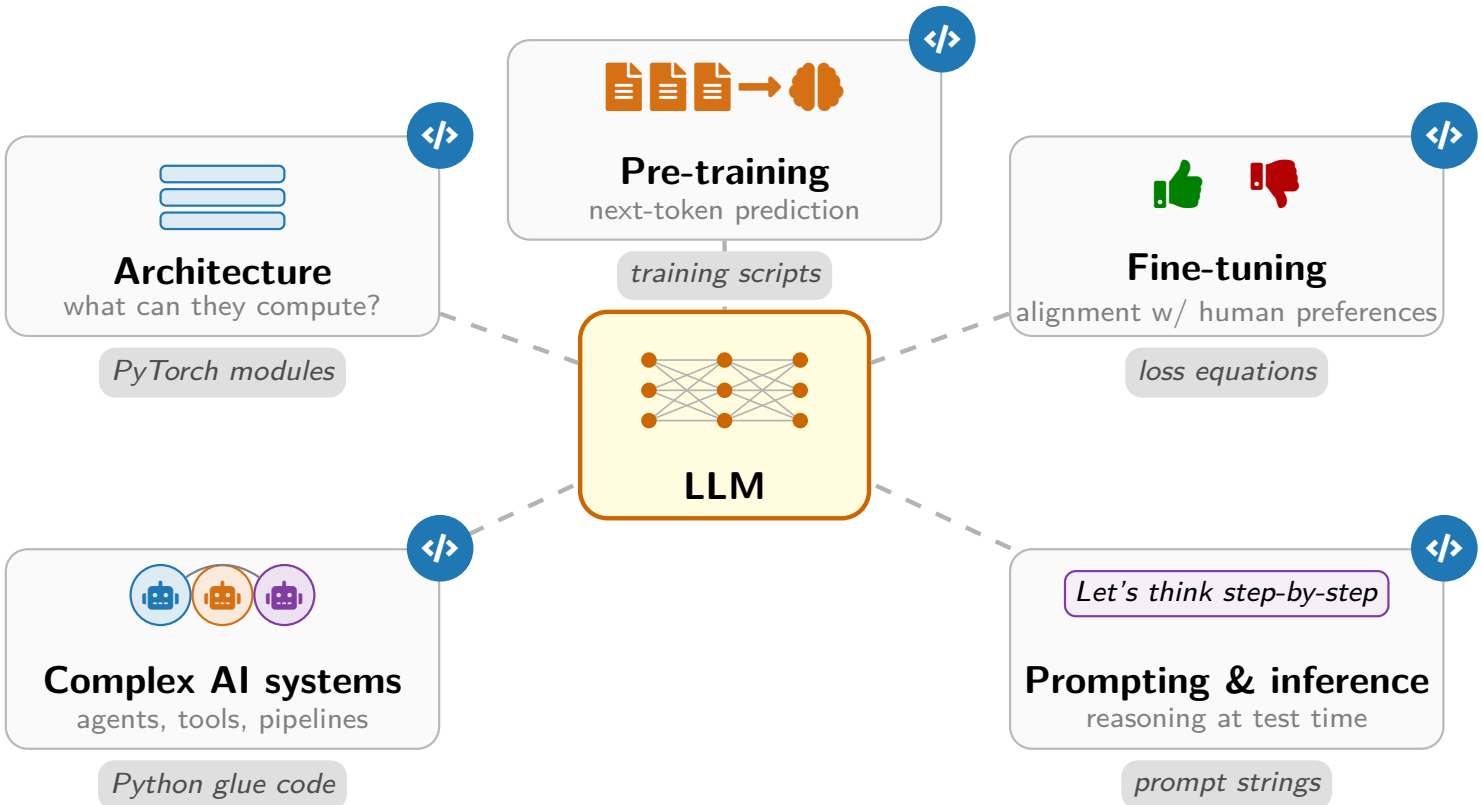
- **Disjointed**: each area has its own tools, notation and methods, hard to know how these different activities are related to one another...

2. The landscape of modern AI research



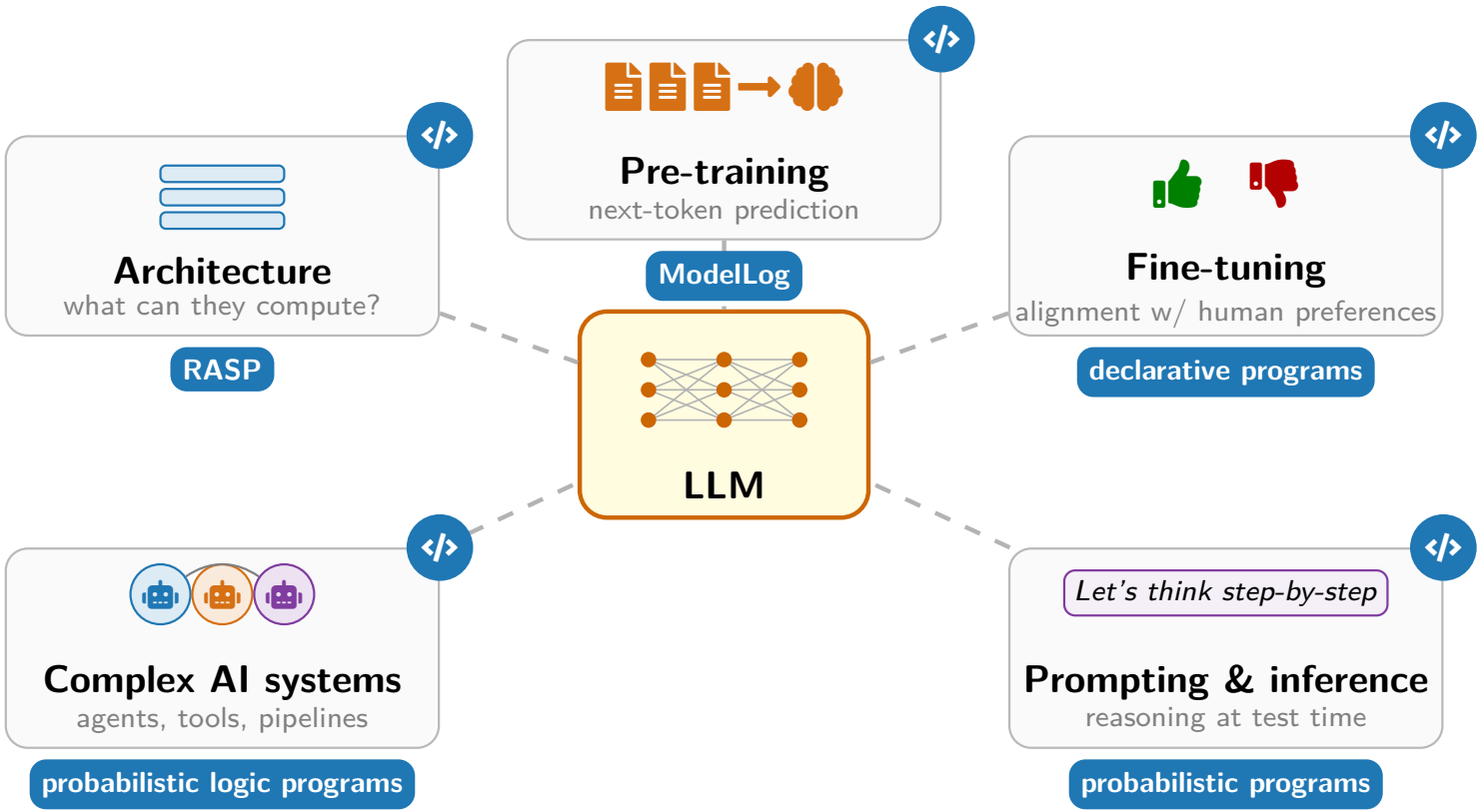
Is all of this just **programming**?

Is all of this just programming?



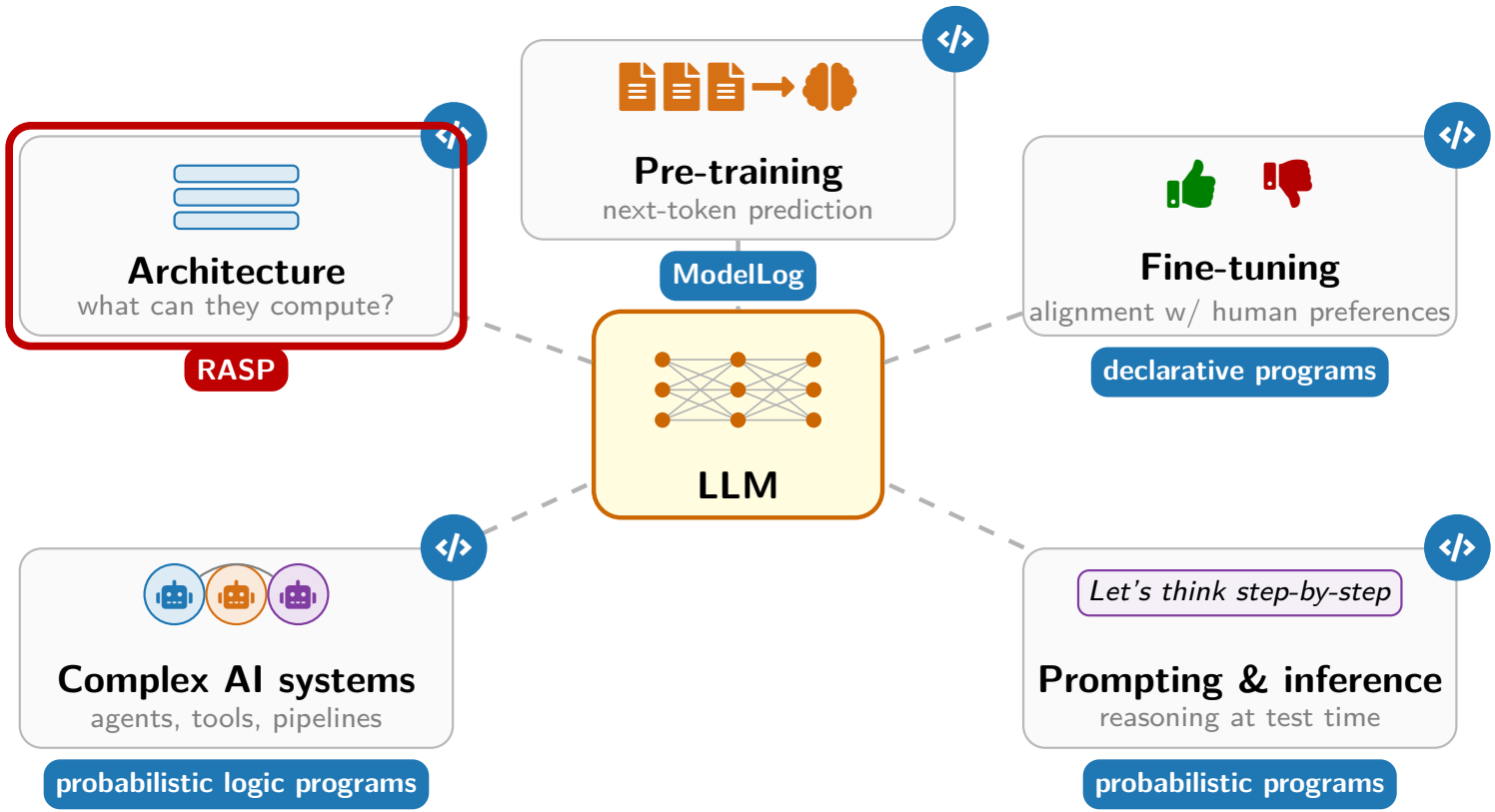
- All of this involves writing **high-level programs** on top of LLMs; What are the languages and styles of programming for these different areas?

Programming Languages for Modern AI



- Researchers in each area have started building their own **languages**, each aiming to solve different formal and practical problems.

Programming Languages for Modern AI



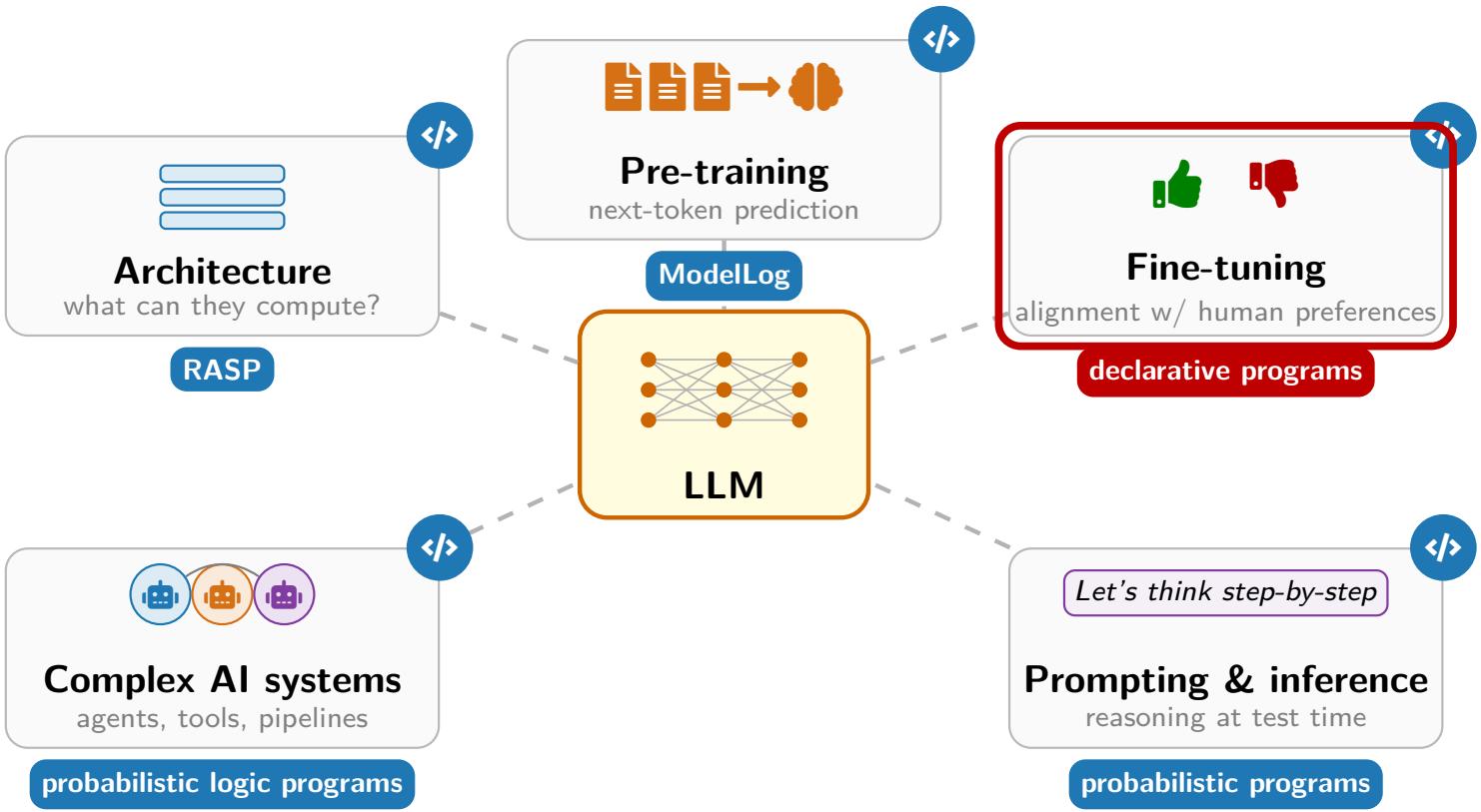
- Researchers in each area have started building their own **languages**, each aiming to solve different formal and practical problems.

Architecture: RASP

$$
\begin{aligned}
A(i) &:= \blacktriangleright_j \ [M(i,j), S(i,j)] \ \ 1 : 0 \\
C_1(i) &:= \blacktriangleright_j \ [M(i,j), S(i,j)] \ \ V_1(i,j) : 0 \\
C_2(i) &:= \blacktriangleright_j \ [M(i,j), S(i,j)] \ \ V_2(i,j) : 0 \\
P(i) &:= (A(i) \wedge C_1(i) \wedge C_2(i)) \vee (\neg A(i) \wedge D(i))
\end{aligned}
$$

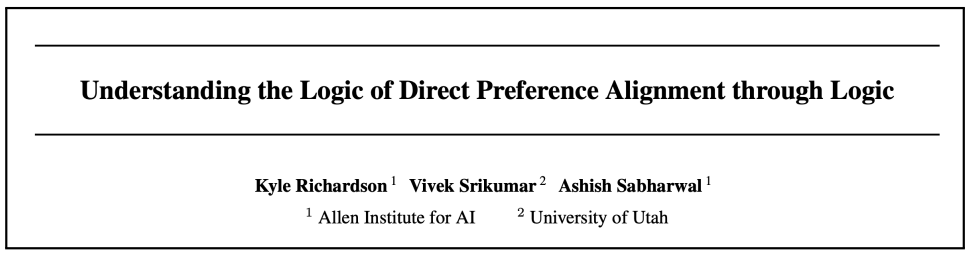
RASP: a programming language that mirrors how transformers compute. Programs compile into transformers, so we can ask precisely *what transformers can and cannot compute* (Weiss et al.2021; Yang et al.2024).

Programming Languages for Modern AI



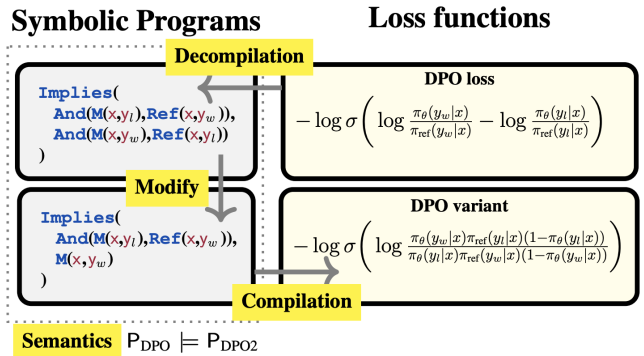
- Researchers in each area have started building their own **languages**, each aiming to solve different formal and practical problems.

Fine-tuning: declarative programs



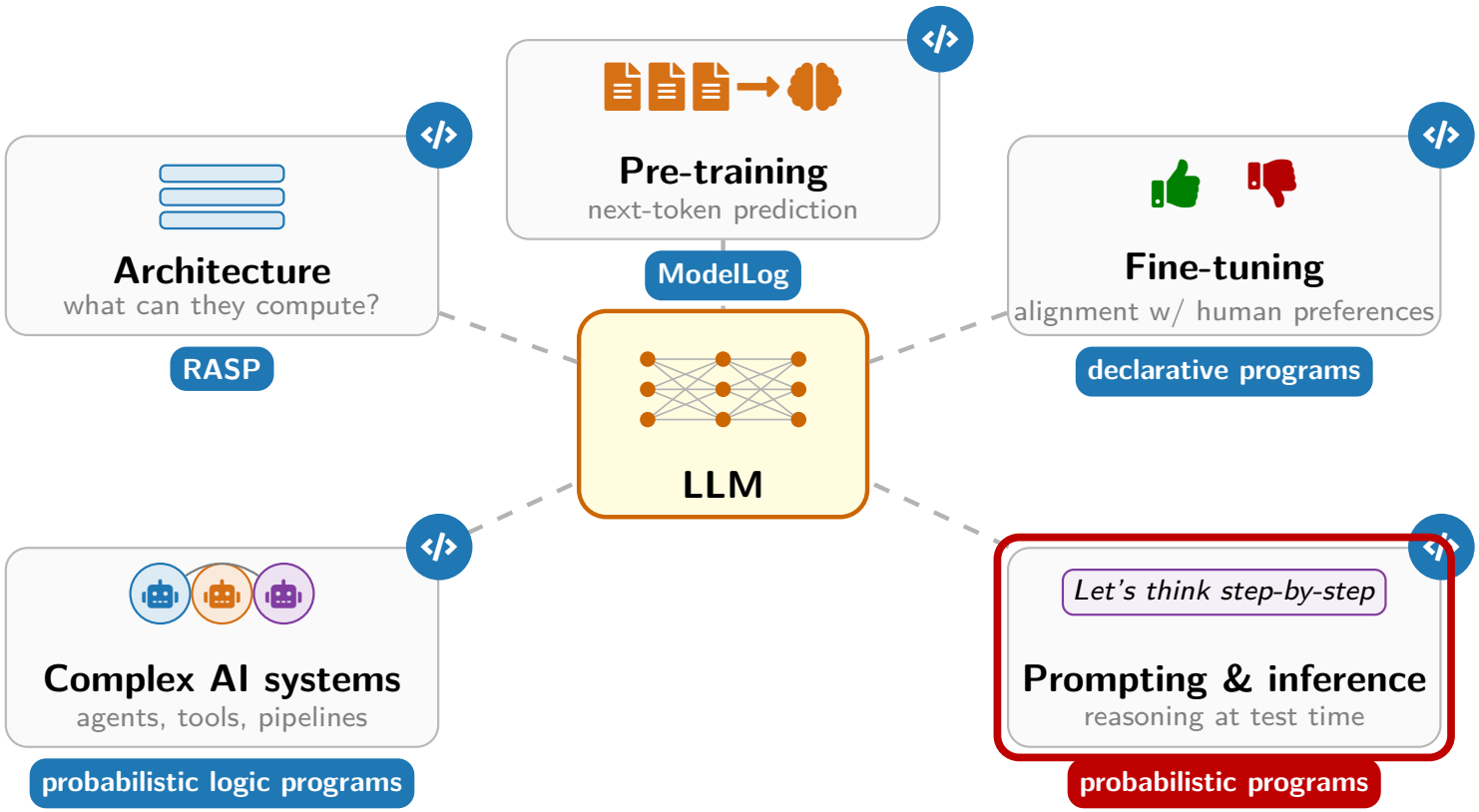
Understanding the Logic of Direct Preference Alignment through Logic

Kyle Richardson [1] Vivek Srikumar [2] Ashish Sabharwal [1]
[1] Allen Institute for AI [2] University of Utah



Declarative programs: losses such as DPO *decompile* into symbolic programs that we can read, modify and *compile* back into new losses (Richardson et al.2025).

Programming Languages for Modern AI

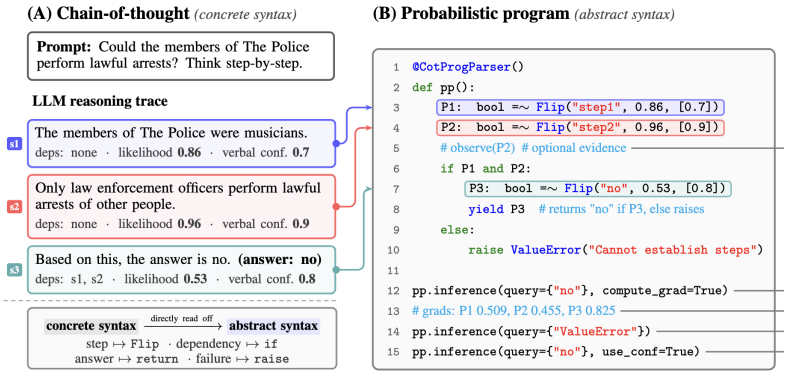


- Researchers in each area have started building their own **languages**, each aiming to solve different formal and practical problems.

Prompting: probabilistic programs

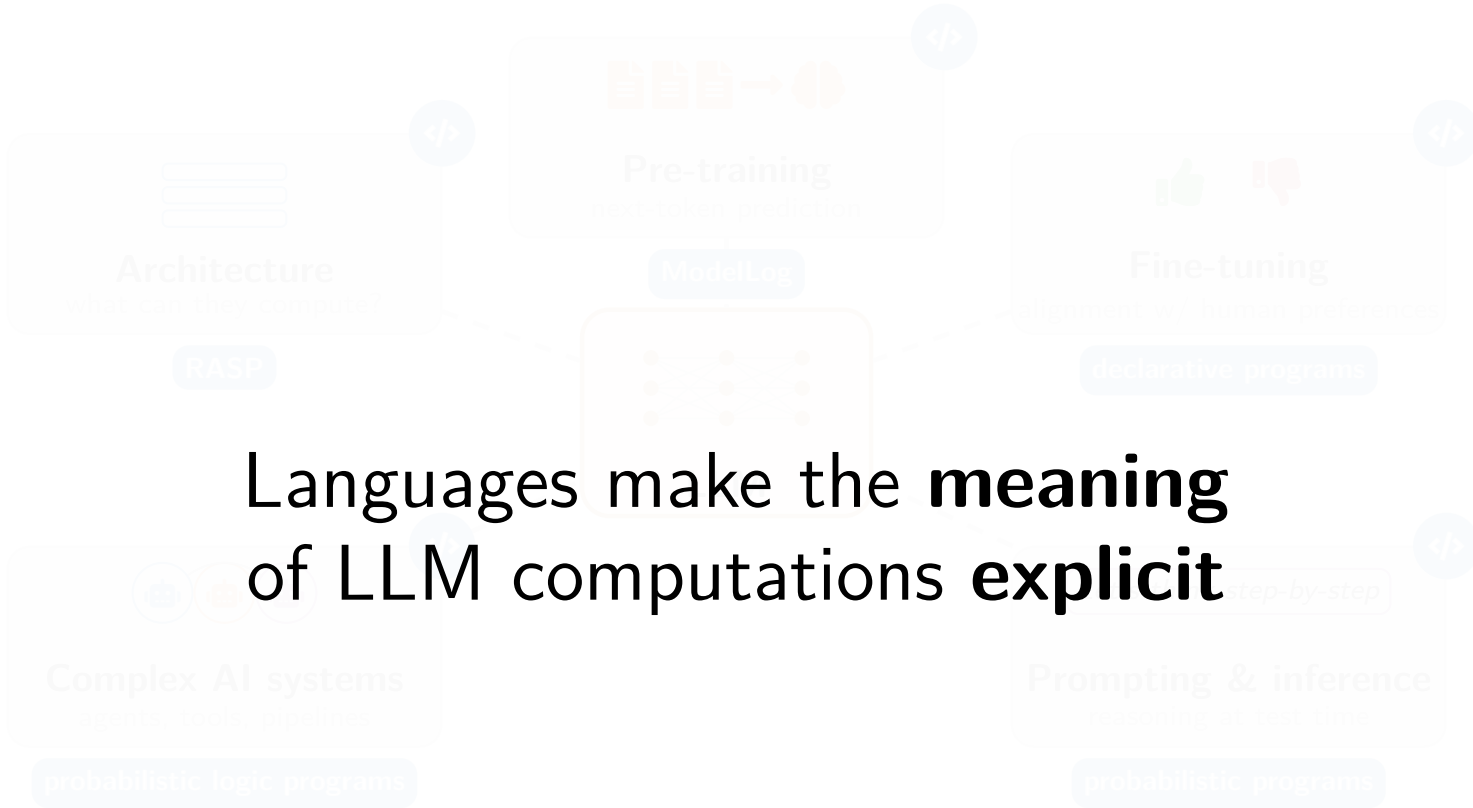
CoTs as Probabilistic Programs: A Programmatic View of Thinking Step-by-Step in Language Models

Kyle Richardson[1,2*] **Yu Feng**[3*] **Poorva Garg**[4]
Junyan Cheng[5] **Guy Van den Broeck**[4] **Dan Roth**[3,6]
[1]University of Tübingen [2]Allen Institute for AI [3]University of Pennsylvania
[4]University of California, Los Angeles [5]Dartmouth College [6]Oracle AI



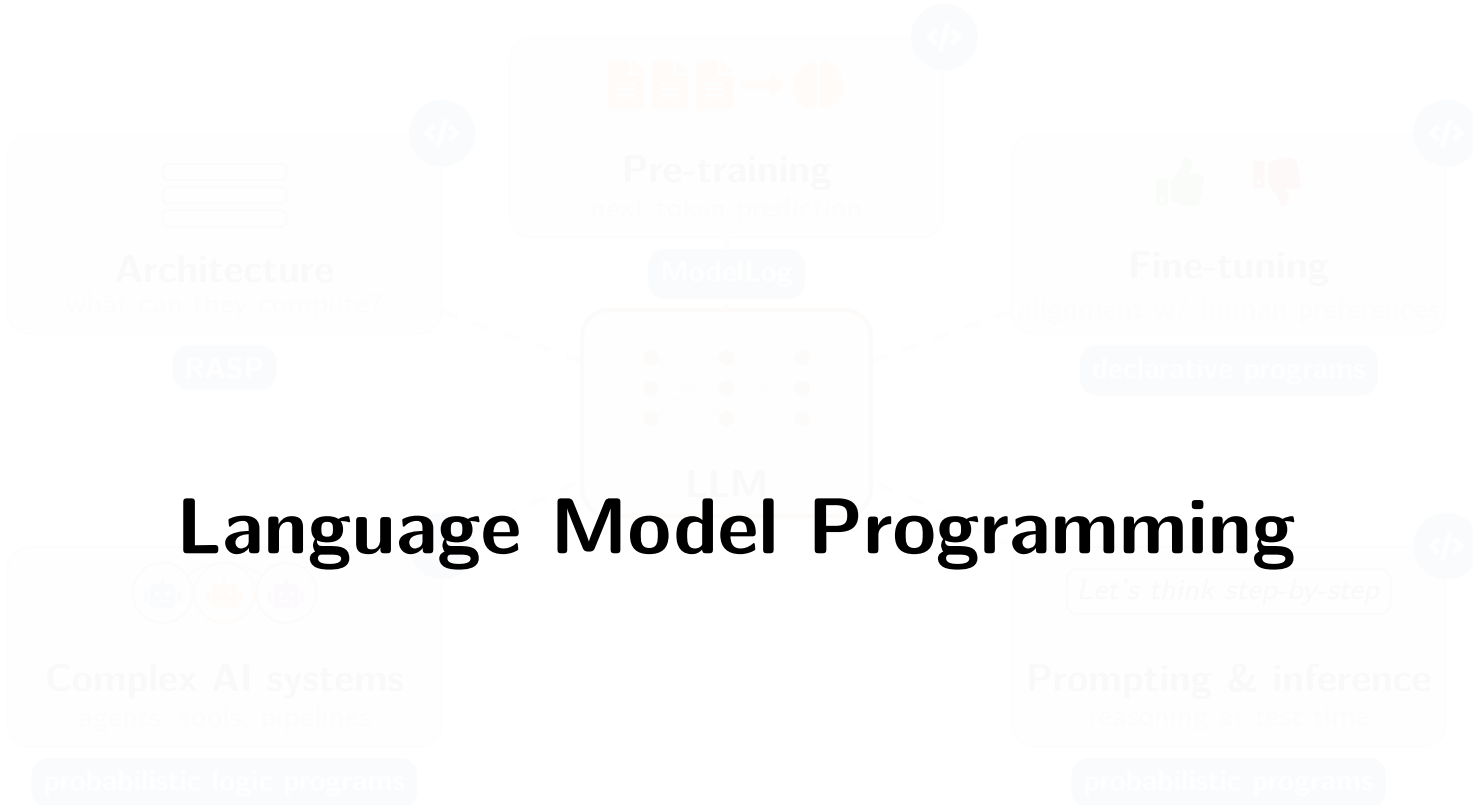
Probabilistic programs: a chain-of-thought trace is read off as a program, reducing CoT operations to standard probabilistic inference, designing new inference strategies (Richardson et al.2026).

Programming Languages for Modern AI



Languages make the **meaning** of LLM computations **explicit**

- Researchers in each area have started building their own **languages**, each aiming to solve different formal and practical problems.



Language Model Programming

- Researchers in each area have started building their own **languages**, each aiming to solve different formal and practical problems.

Different languages for different areas

Architecture: **RASP**

$$A(i) := \blacktriangleright_j \ [M(i,j), S(i,j)] \ \ 1 : 0$$
$$C_1(i) := \blacktriangleright_j \ [M(i,j), S(i,j)] \ \ V_1(i,j) : 0$$
$$C_2(i) := \blacktriangleright_j \ [M(i,j), S(i,j)] \ \ V_2(i,j) : 0$$
$$P(i) := (A(i) \wedge C_1(i) \wedge C_2(i)) \vee (\neg A(i) \wedge D(i))$$

(Weiss et al.2021; Yang et al.2024)

Post-training: **declarative programs**

```
Implies(
  And(M(x,y_l), Ref(x,y_w)),
  And(M(x,y_w), Ref(x,y_l)))
```

⇝ compiles to the **DPO** loss

(Richardson et al.2025)

Prompting: **probabilistic programs**

```
P1 ~ Flip("step1", 0.86)
P2 ~ Flip("step2", 0.96)
if P1 and P2:
  P3 ~ Flip("no", 0.53)
  yield P3
else:  raise ValueError
```

(Richardson et al.2026)

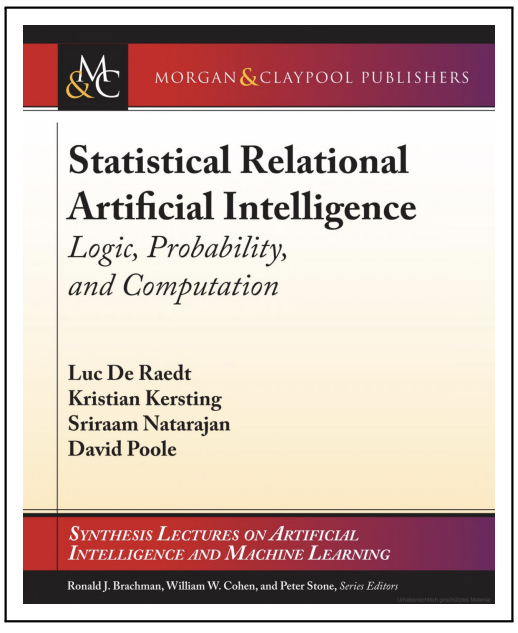
Agents: **probabilistic logic programs**

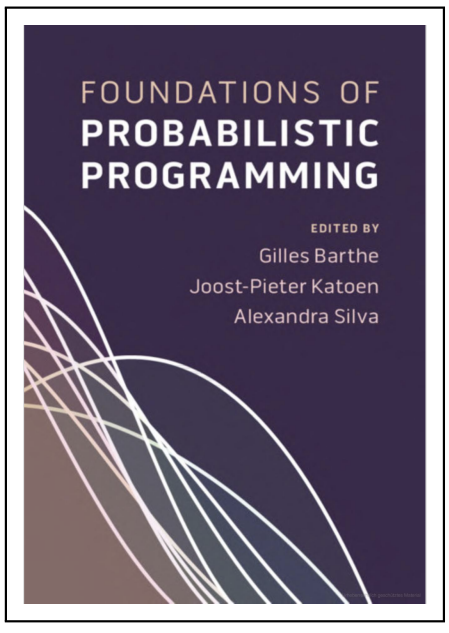
```
## evidence scores
s(c1)::c1.  s(c2)::c2.  s(c3)::c3.
β_0^r::br0; β_1^r::br1; β_2^r::br2.
r :- br0.  r :- br1, p1.
r :- br2, p2.
## subtrees for p1, p2 ...
query(r).
```

(Cheng et al.2026)

Language model programming: the study and design of languages for **specifying**, **composing** and **reasoning about** LLM computations.

We borrow from established fields





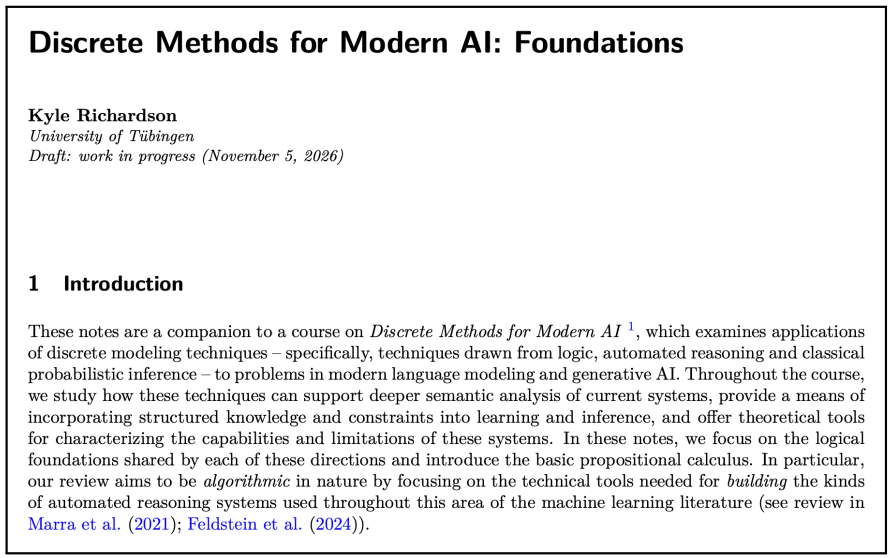
The Elements of Differentiable Programming

Mathieu Blondel
Google DeepMind
mblondel@google.com

Vincent Roulet
Google DeepMind
vroulet@google.com

Statistical relational learning, **differentiable** and **probabilistic programming**: decades of work on programs that combine logic, probability and learning

New foundations and tools



Discrete Methods for Modern AI: Foundations

Kyle Richardson
University of Tübingen
Draft: work in progress (November 5, 2026)

1 Introduction

These notes are a companion to a course on *Discrete Methods for Modern AI* [1], which examines applications of discrete modeling techniques – specifically, techniques drawn from logic, automated reasoning and classical probabilistic inference – to problems in modern language modeling and generative AI. Throughout the course, we study how these techniques can support deeper semantic analysis of current systems, provide a means of incorporating structured knowledge and constraints into learning and inference, and offer theoretical tools for characterizing the capabilities and limitations of these systems. In these notes, we focus on the logical foundations shared by each of these directions and introduce the basic propositional calculus. In particular, our review aims to be *algorithmic* in nature by focusing on the technical tools needed for *building* the kinds of automated reasoning systems used throughout this area of the machine learning literature (see review in Marra et al. (2021); Feldstein et al. (2024)).

formal foundations

```
In [1]: @Parser(language="constraint")
        def program():
            E: 0.3; C: 0.2; N: 0.5   ## LLM scores
            E or C or N              ## one label
            not (E and C); not (E and N); not (C and N)

        program.inference().inference
```

```
Out[1]: tensor(0.4700)
```

```
In [2]: P = program.inference(symbolic=True).inference; P
```

Out[2]: $N(1-C)(1-E) + (1-N)\big(C(1-E) + E(1-C)\big)$

```
In [3]: program.grad(symbolic=True)["weights"]  ## gradients
```

Out[3]:
$$\begin{bmatrix} -N(1-E) + (1-2E)(1-N) \\ -N(1-C) + (1-2C)(1-N) \\ -C(1-E) - E(1-C) + (1-C)(1-E) \end{bmatrix}$$

```
$ pip install symprog
```

github.com/TuLLI-AI/symprog

My work on establishing new **formal foundations** for these languages, and **tools** for implementing new modeling languages.

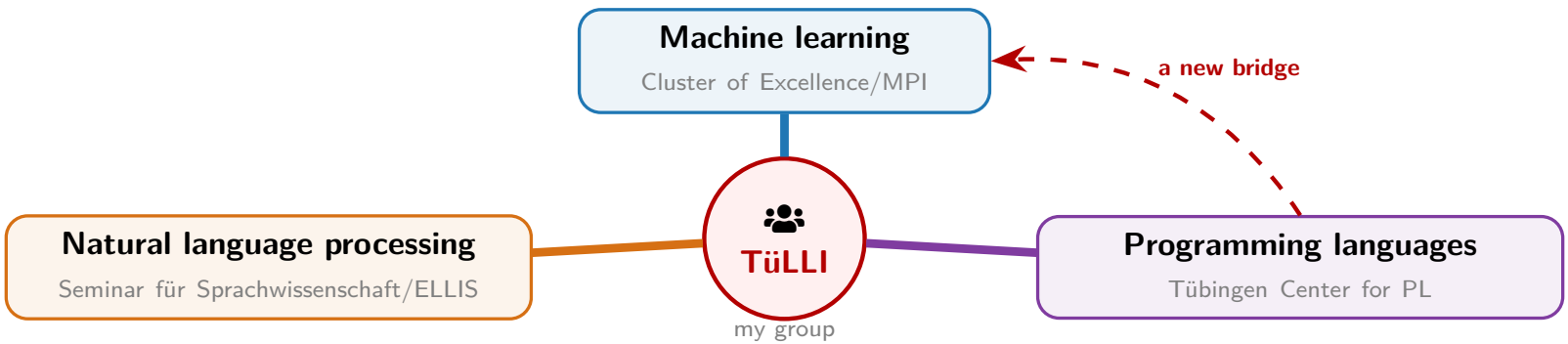
Programming languages in Tübingen



We plan to work closely with the **programming languages** groups here: developing our **formal models**, and building the **next generation of AI programming languages**.

Conclusion and Looking Ahead

- **LLM agents**: the future of how we simulate society and do research.
- **Open models**: we can't understand LLMs unless we build them ourselves.
- **Language model programming**: foundations and tools for LLM computations.



I hope to serve as a **bridge** between different parts of the University.

References I

Bogin, B., Yang, K., Gupta, S., Richardson, K., Bransom, E., Clark, P., Sabharwal, A., and Khot, T. (2024). Super: Evaluating agents on setting up and executing tasks from research repositories. *Proceedings of EMNLP*.

Bragg, J., D'Arcy, M., Balepur, N., Bareket, D., Dalvi, B., Feldman, S., Haddad, D., Hwang, J. D., Jansen, P., Kishore, V., Majumder, B. P., Naik, A., Rahamimov, S., Richardson, K., Singh, A., Surana, H., Tiktinsky, A., Vasu, R., Wiener, G., Anastasiades, C., Candra, S., Dunkelberger, J., Emery, D., Evans, R., Hamada, M., Huff, R., Kinney, R., Latzke, M., Lochner, J., Lozano-Aguilera, R., Nguyen, C., Rao, S., Tanaka, A., Vlahos, B., Clark, P., Downey, D., Goldberg, Y., Sabharwal, A., and Weld, D. S. (2026). Astabench: Rigorous benchmarking of ai agents with a scientific research suite. *Proceedings of ICLR*.

Chen, J., Yuan, S., Ye, R., Majumder, B. P., and Richardson, K. (2023). Put your money where your mouth is: Evaluating strategic planning and execution of llm agents in an auction arena. *arXiv preprint arXiv:2310.05746*.

Cheng, J., Clark, P., and Richardson, K. (2025). Language modeling by language models. *Proceedings of Neurips*.

Cheng, J., Richardson, K., and Chin, P. (2026). Analytica: Soft propositional reasoning for robust and scalable llm-driven analysis. *Proceedings of ICLR*.

Groeneveld, D., Beltagy, I., Walsh, P., Bhagia, A., Kinney, R., Tafjord, O., Jha, A. H., Ivison, H., Magnusson, I., Wang, Y., et al. (2024). OLMo: Accelerating the science of language models. In *Proceedings of ACL*.

References II

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. (2024). Direct preference optimization: Your language model is secretly a reward model. *Proceedings of Neurips*.

Richardson, K., Feng, Y., Garg, P., Cheng, J., and Roth, D. (2026). CoTs as Tractable Probabilistic Programs. *Proceedings of TPM*.

Richardson, K., Srikumar, V., and Sabharwal, A. (2025). Understanding the logic of direct preference alignment through logic. *Proceedings of ICML*.

Weiss, G., Goldberg, Y., and Yahav, E. (2021). Thinking like transformers. In *International Conference on Machine Learning*, pages 11080–11090. PMLR.

Yang, A., Chiang, D., and Angluin, D. (2024). Masked hard-attention transformers recognize exactly the star-free languages. *Advances in Neural Information Processing Systems*, 37:10202–10235.

Yang, R., Chen, J., Zhang, Y., Yuan, S., Chen, A., Richardson, K., Xiao, Y., and Yang, D. (2025). Selfgoal: Your language agents already know how to achieve high-level goals. *Proceedings of NAACL*.

Yu, H., You, J., Clark, P., Majumder, B. P., and Richardson, K. (2026). Artifactlinker: Linking scientific artifacts for automatic state-of-the-art discovery. *Proceedings of COLM*.

Zhang, Y., Yuan, S., Hu, C., Richardson, K., Xiao, Y., and Chen, J. (2024). Timearena: Shaping efficient multitasking language agents in a time-aware simulation. *Proceedings of ACL*.