

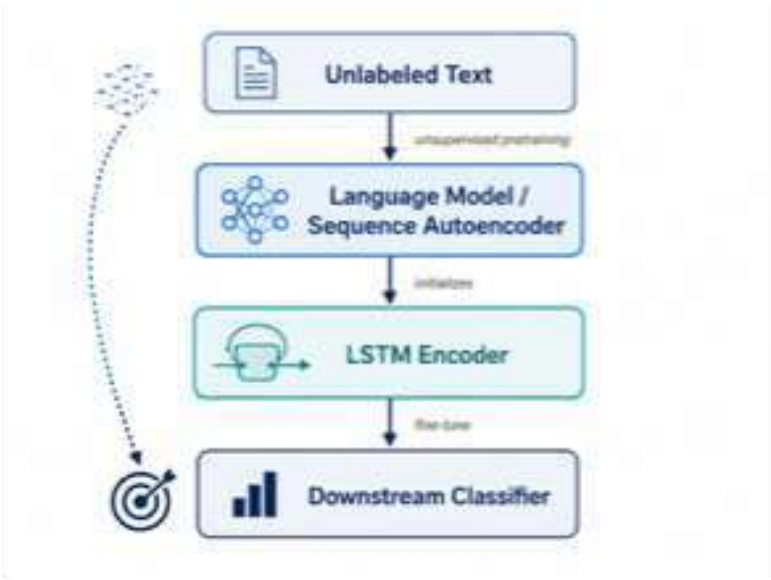
Jeff Pan

From Parameter Scaling to Agent Scaling

November 4, 2026

The Science behind the "Bigger is Better" Scaling Law

- **Kaplan et al.'s findings gave quantitative backing to the prevailing strategy of scaling**
- OpenAI's results with GPT-2 echoed the value of scale



Dai & Le. Semi-supervised Sequence Learning, Google 2015

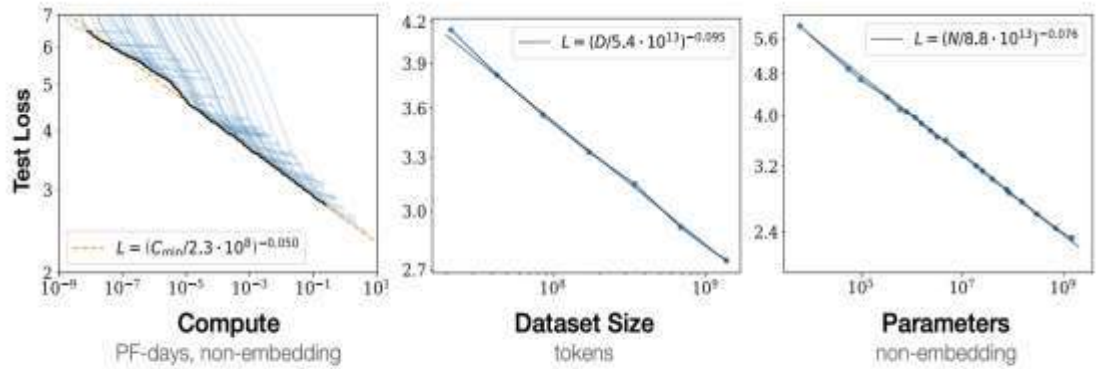
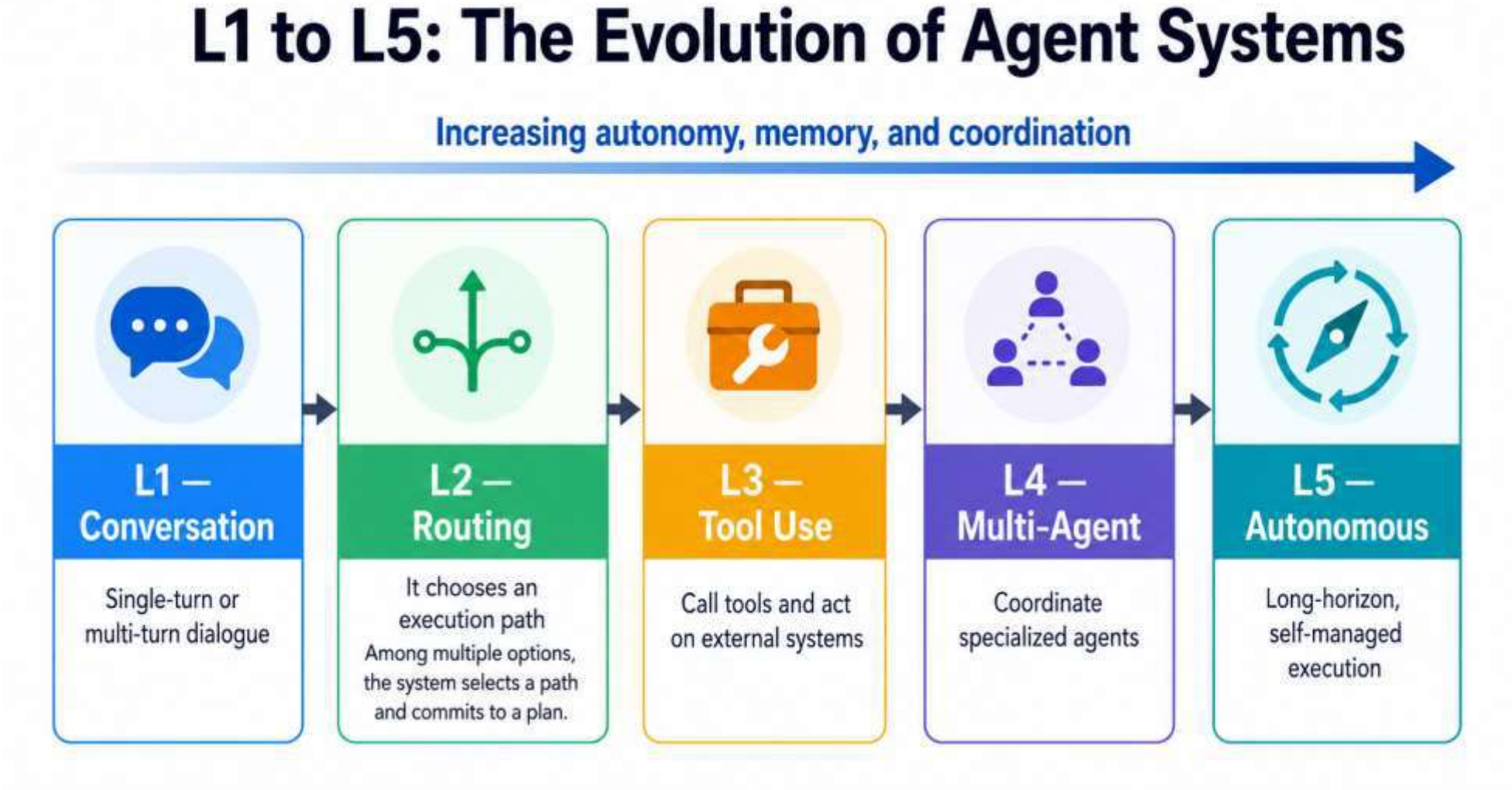


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

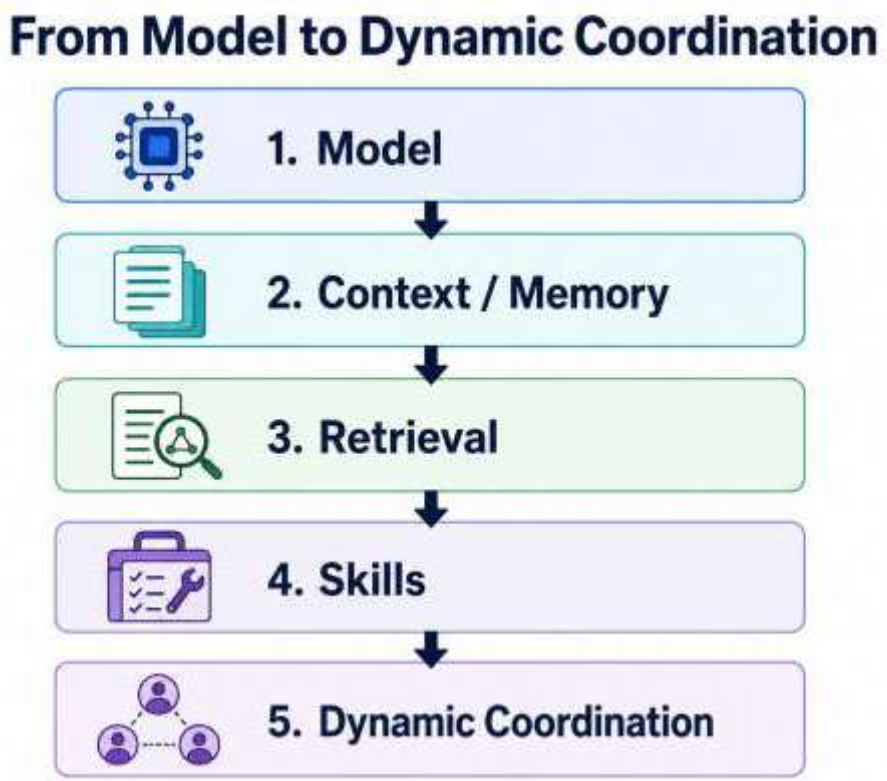
Kaplan et al. Scaling Laws for Neural Language Models, OpenAI, 2020.

The Evolution of Agent Systems



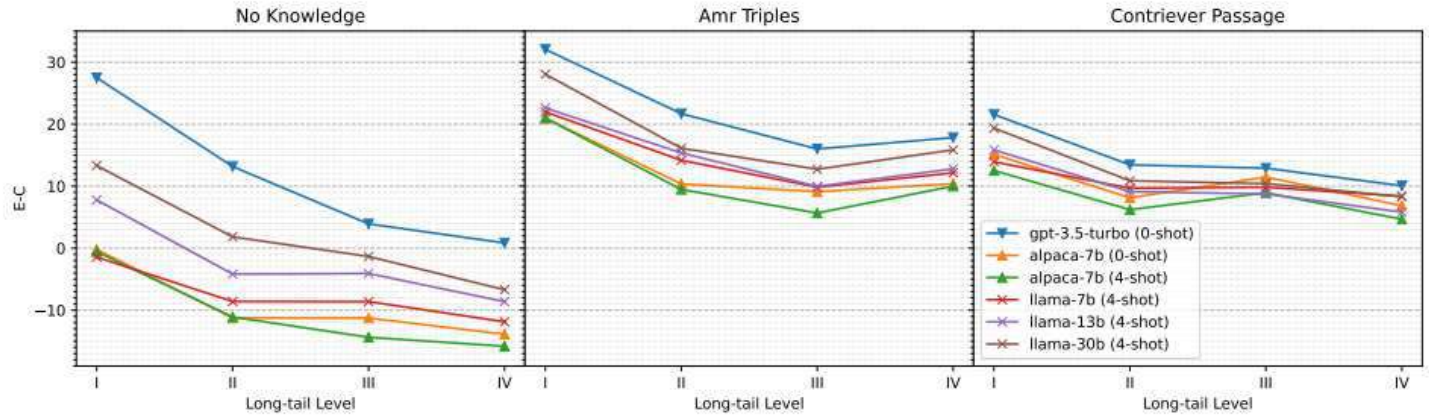
[Adapted from https://huggingface.co/blog/smolagents]

From Model Capability to System Capability



When LLMs enter the Agent phase:

- **Memory**: Agents need to work across contexts
- **Retrieval**: Agents need collect information from memory/external KBs
- **Skill:** Agents need to work repeatedly and reliably (potentially with tools)
- **Agent coordination:** Agents need to work across tasks



Increasing model size does not eliminate the long-tail knowledge problem. External knowledge remains valuable

Prompting Large Language Models with Knowledge Graphs for Question Answering involving Long-tail Facts, Knowledge-Based Systems

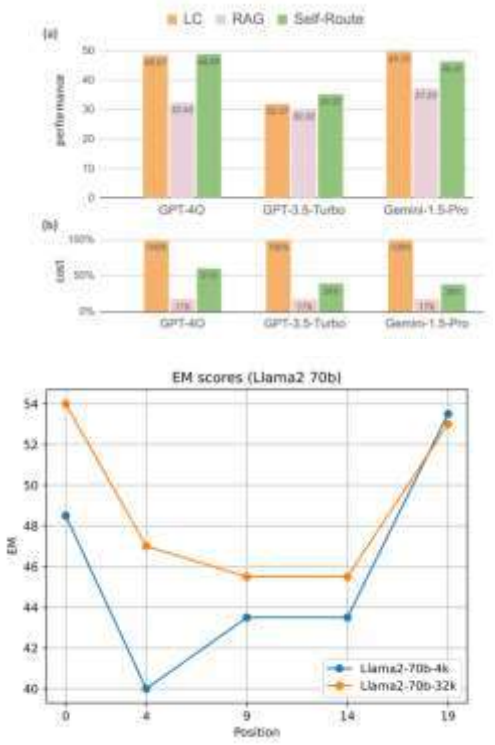
openJiuwen Targets the L3-to-L5 Gap

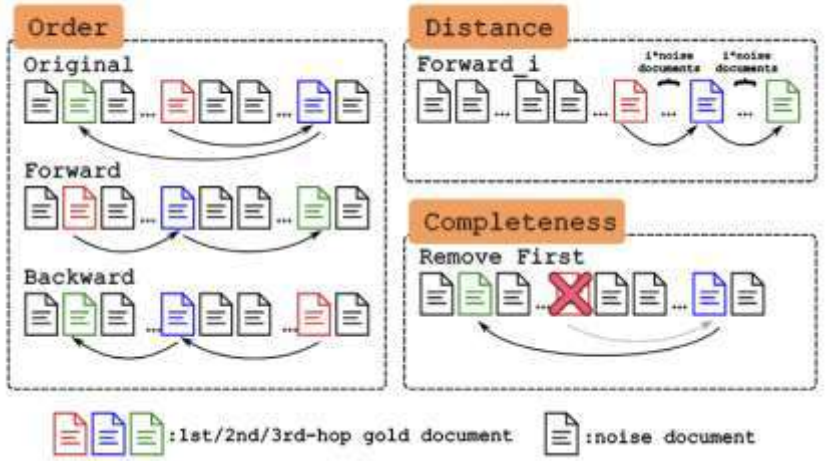
- **From tool access to evidence-grounded action: DeepSearch** connects agents to structured, traceable evidence.
- **From isolated workflows to task-aware collaboration**: **JiuwenSwarm** supports three execution modes: plan, performance and **swarm**

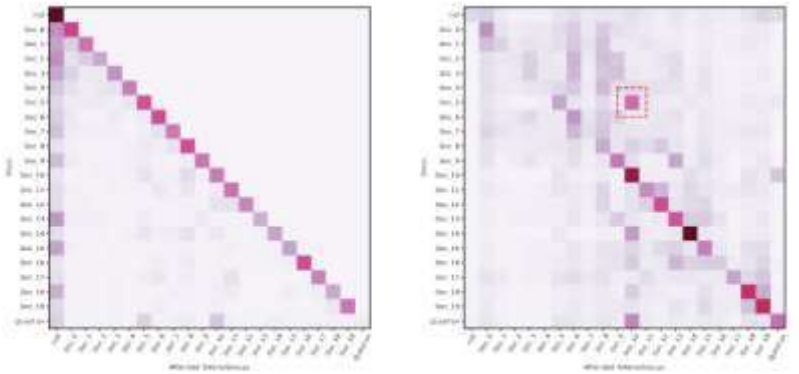


Why Context Scaling is Not Enough

- **More tokens ≠ better context**
- **Lost in the middle:** Relevant information placed in the middle of a long prompt receives less effective attention than information near the beginning or end
- **Lost in between:** For multi-hop tasks, success depends not only on retrieving the right documents, but on keeping the supporting documents close enough, ordered coherently, and connected through the reasoning chain





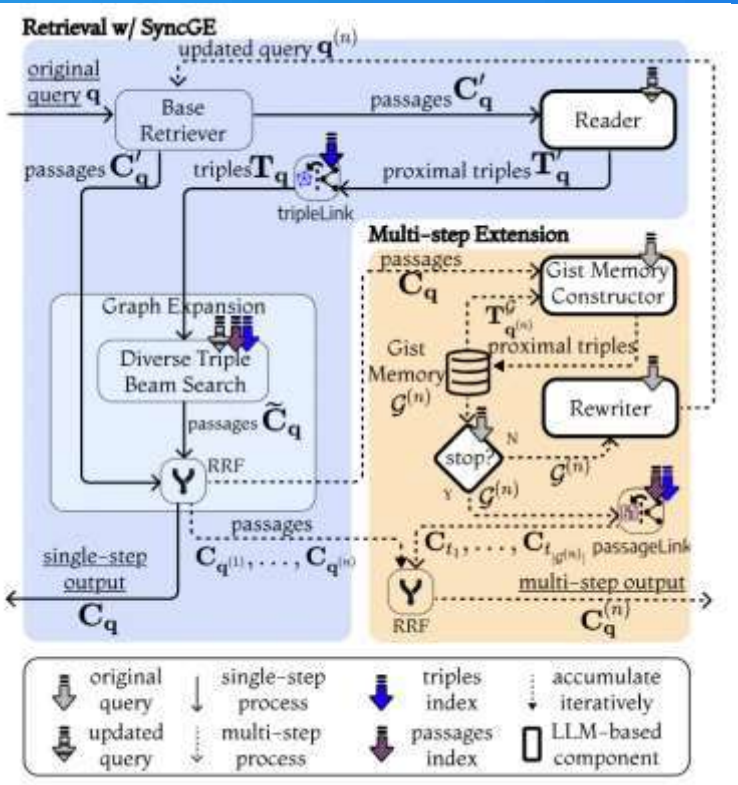


(a) Qwen 2.5, head 3 (b) FlanT5, head 1

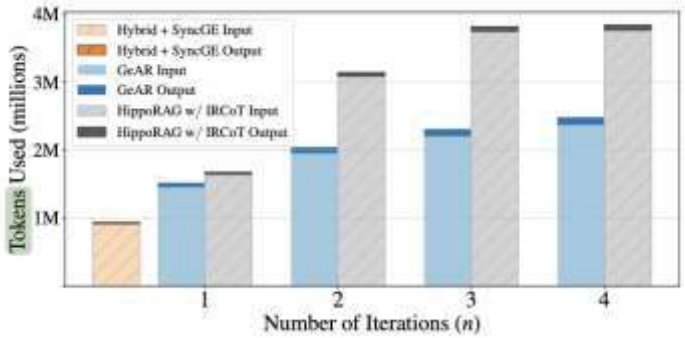
Lost in the Middle: How Language Models Use Long Contexts, TACL, 2024. (Stanford)
Masking in Multi-hop QA: An Analysis of How Language Models Perform with Context Permutation, ACL, 2025. (Edinburgh)

How can Agents Discover Connected, Multi-hop Evidence

- **DeepSearch is the layer that turns a tool-using agent from an API caller into an evidence-grounded decision-maker**
- **GeAR** supports the technical idea behind DeepSearch: graph expansion plus multi-step agentic retrieval improves multi-hop retrieval



	Retriever	MuSiQue			2Wiki			HotpotQA		
		R@5	R@10	R@15	R@5	R@10	R@15	R@5	R@10	R@15
Single-step Retrieval	ColBERTv2	39.4	44.8	47.7	59.1	64.3	66.2	79.3	87.1	90.1
	HippoRAG	41.0	47.0	51.4	**75.1**	**83.2**	**86.4**	79.8	89.0	92.4
	BM25	33.8	38.5	41.3	59.5	62.7	64.1	74.2	83.6	86.3
	+ NaiveGE	37.5	45.5	48.4	65.0	70.7	71.8	79.1	89.1	91.9
	+ SyncGE	44.7	52.6	57.4	70.5	76.1	79.3	87.4	93.0	94.0
	SBERT	31.1	37.9	41.6	41.2	48.1	51.5	72.1	79.3	84.0
	+ NaiveGE	32.2	41.4	45.4	45.1	54.0	57.3	76.1	84.7	88.8
	+ SyncGE	41.6	51.3	54.2	54.8	64.9	70.7	84.1	89.6	92.8
	Hybrid	39.9	46.3	49.1	60.0	65.8	66.6	77.8	85.8	89.7
	+ NaiveGE	41.8	49.4	53.0	63.0	70.8	72.6	80.6	89.4	92.7
	+ SyncGE	**48.7**	**57.7**	**61.2**	72.6	80.9	82.4	**87.4**	**93.3**	**95.2**
Multi-step Retrieval	IRCoT (BM25)	46.1	54.9	57.9	67.9	75.5	76.1	87.0	92.6	92.9
	IRCoT (ColBERTv2)	47.9	54.3	56.4	60.3	86.6	69.7	86.9	92.5	92.8
	HippoRAG w/ IRCoT	48.8	54.5	58.9	82.9	90.6	93.0	90.1	94.7	95.9
	GEAR	**58.4**	**67.6**	**71.5**	**89.1**	**95.3**	**95.9**	**93.4**	**96.8**	**97.3**



GeAR: Graph-enhanced Agent for Retrieval-augmented Generation. ACL, 2025.

Hidden Bottleneck in Coding Agents: Finding the Right Code

- **Problem**: Coding agents must identify the relevant parts of a repository before they can reliably fix an issue
- **In the past, we often considered if the model was powerful enough for coding agents**
- As coding agents move into large-scale repositories, **the challenge extends beyond their ability to write or fix code: Can they identify the right code to examine?**

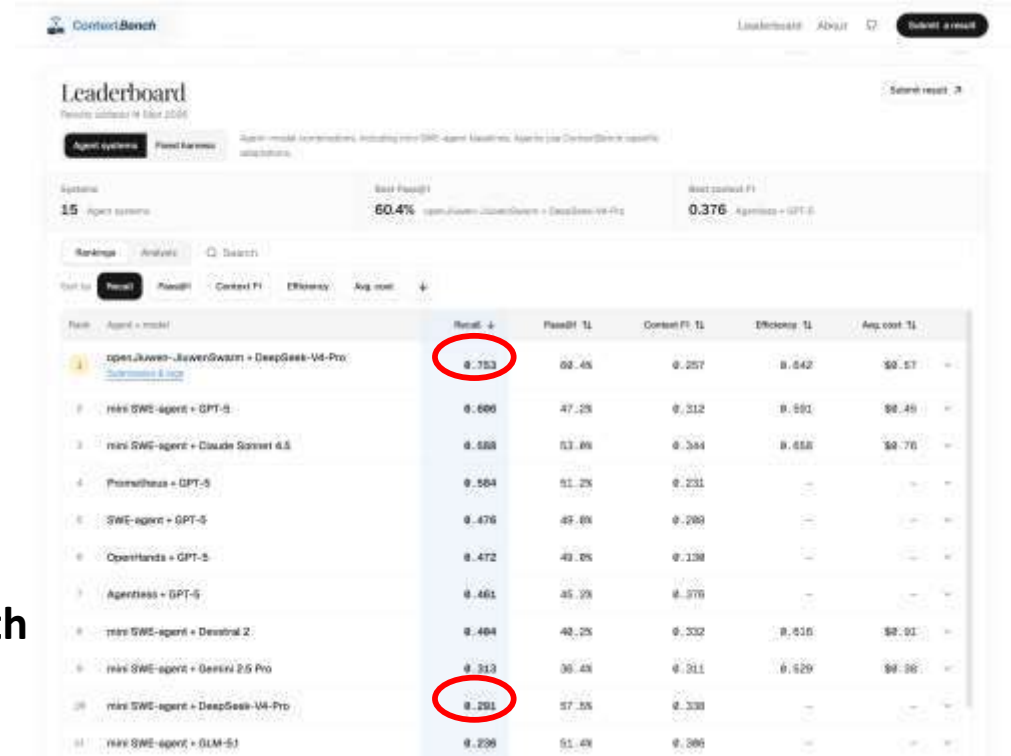
Repository-context retrieval is a major coding-agent bottleneck

ContextBench provides a particularly useful research trigger.

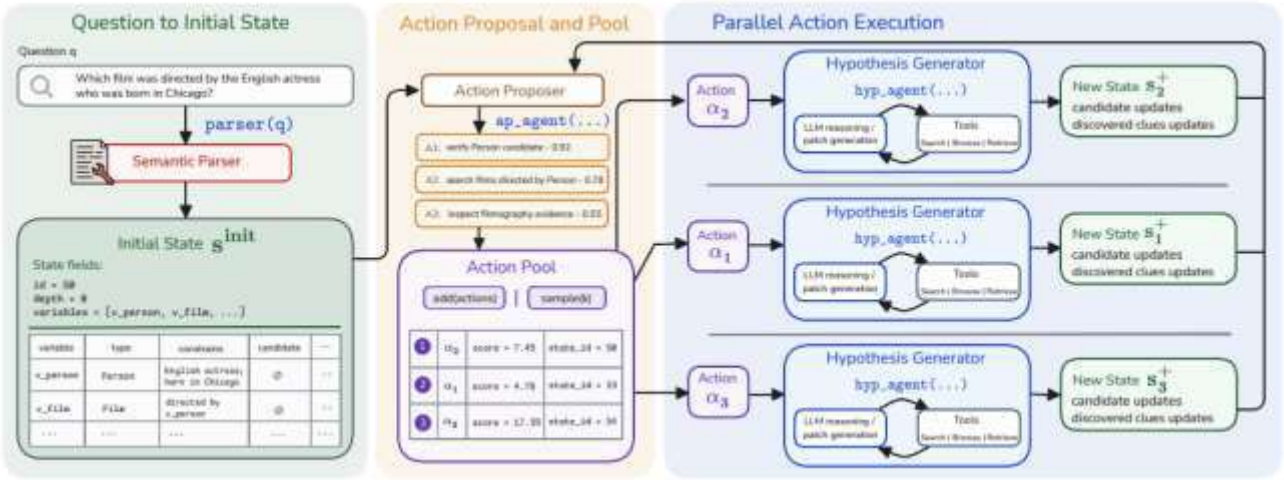
1,136	66	8
Issue-resolution tasks	Repositories	Programming languages

Main Idea: A Code Graph as Structured Repository Knowledge

Agent-system design can improve retrieval with the same backbone model



Gnosis: Scaling DeepSearch with openJiuwen

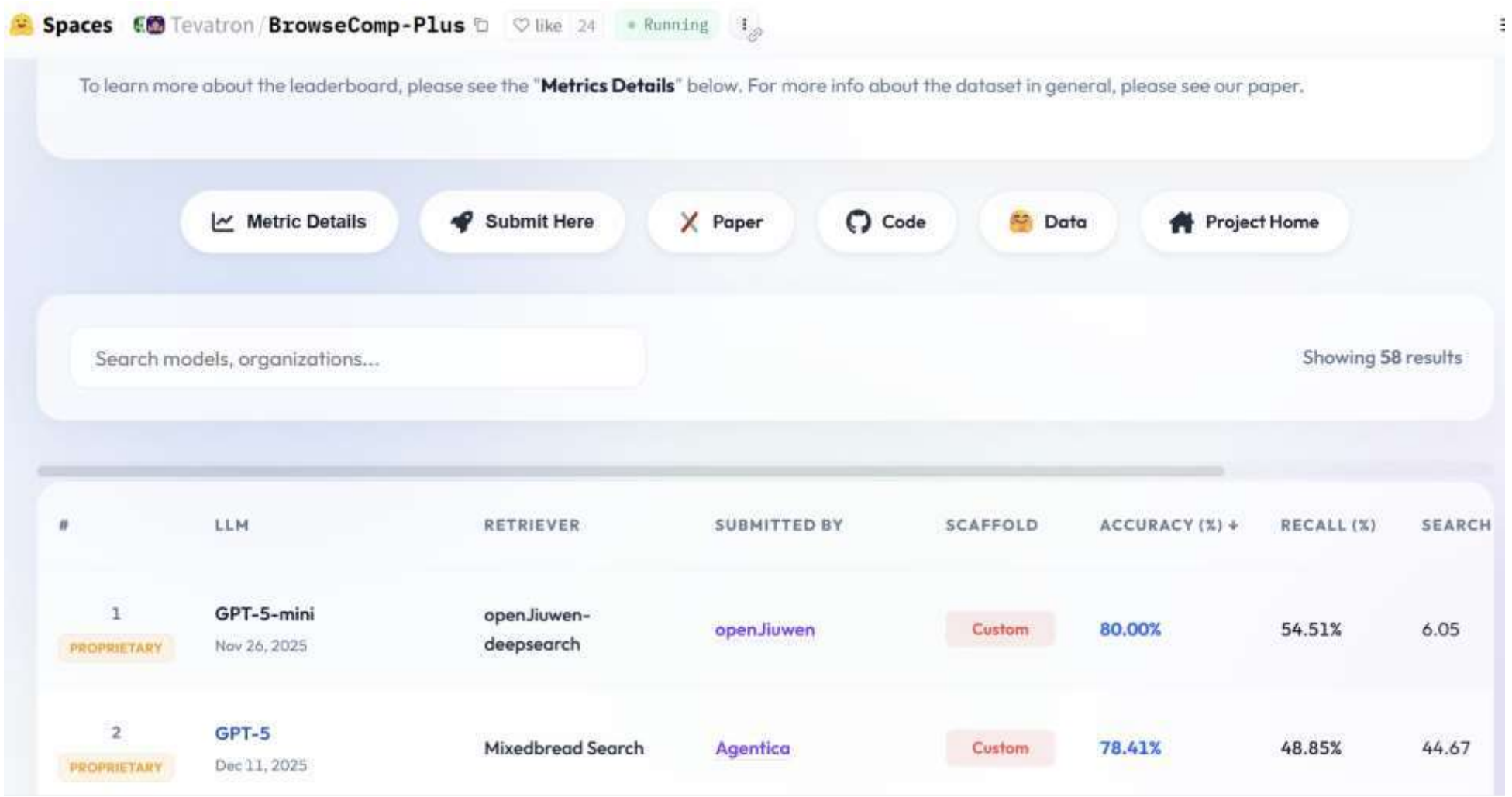


Dimension	ReAct	Folded Summary	ToT / LATS
History storage	Full linear transcript	Compressed summaries	Free-form reasoning nodes
Explicit variables	×	×	×
Explicit constraints	×	×	×
Branchable	× single path	× single path	✓ per node
Merge / comparison	×	×	Heuristic voting

- **Problem**: systematically exploring the search space without losing track of what has already been discovered
- **Core idea**: A **schema-guided planning framework** that maintains a structured task state across the trajectory.
- **Evidence**: against FoldAgent (context-folding baseline), Gnosis wins in every matched setting — 44.0 vs 30.0 (gpt-oss-20b) and 73.0 vs 58.0 (DeepSeek-V4-Flash) on BrowseComp-Plus.
- **How it scales exploration**: Branches state into parallel search paths.
- **Retrieval contribution**: Selects key snippets from long documents.
- **Broad applicability**: Benefits both Gnosis and ReAct agents

Gnosis Deepsearch: Structured Search Space Navigation for Complex QA. EMNLP, 2026.

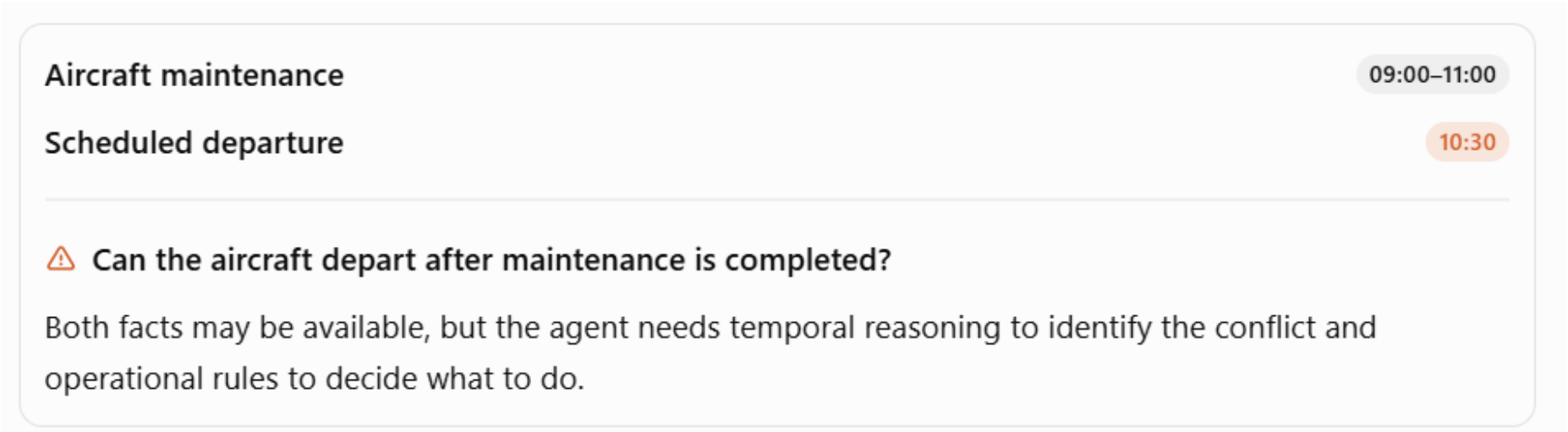
BrowseComp Plus Leaderboard



https://huggingface.co/spaces/Tevatron/BrowseComp-Plus

From Finding Facts to Making Valid Decisions

- **Deep Search can retrieve information, but an operational agent must also determine:**
  - Which facts and relationships are relevant to its current decision?
  - What follows logically from those facts?
  - Which temporal constraints must hold?
  - Which actions are valid under the current state and organizational rules?



Ontology System [data,logic,action,security] x [language,engine,toolchain]

- Traditional knowledge-graph applications typically emphasize representing and querying information
- **Palantir's architecture emphasizes the additional operational loop: representing the state of an organization, applying logic, executing governed actions, incorporating resulting changes, and governing permissions.**

Ontology Language	Defines objects, relationships, actions, and logic
Ontology Engine	Supports queries, subscriptions, state updates, and transactions
Ontology Toolchain	Enables applications, SDKs, and development/governance workflows

Data

Objects, properties, and links representing the operational world

Logic

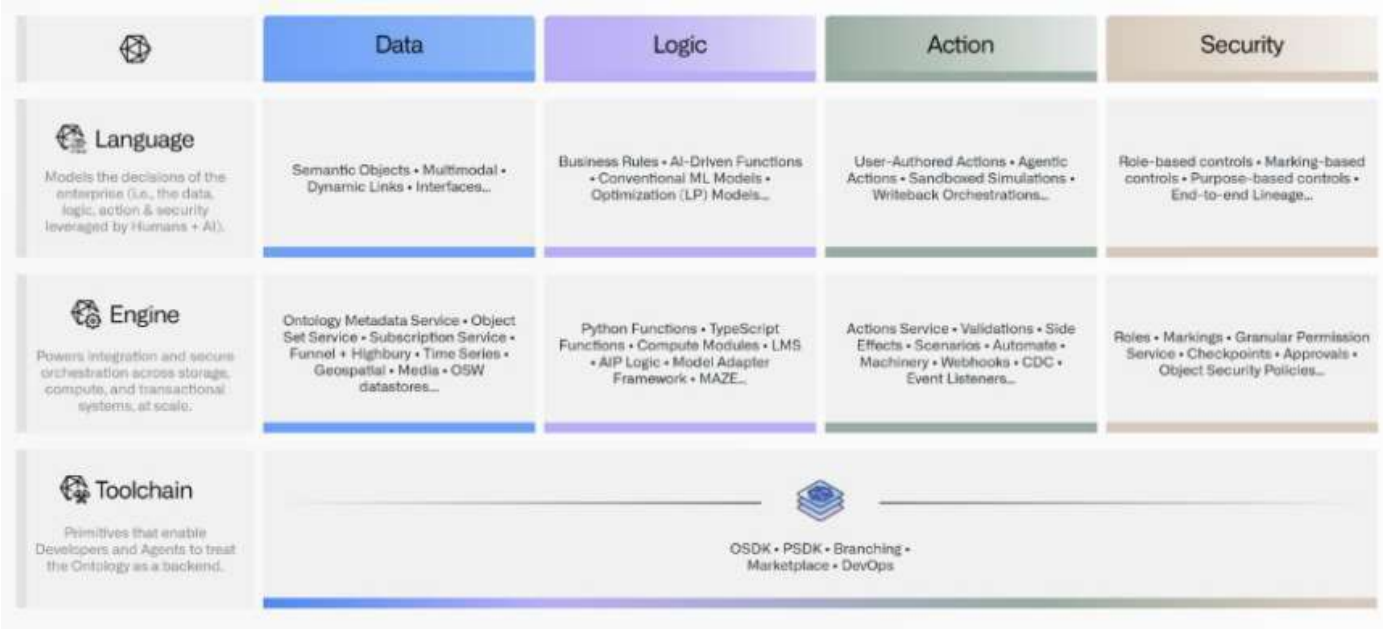
Business rules, optimizers, ML models, LLM functions, and orchestration

Action

Operational transactions, workflows, and write-back into enterprise systems

Security

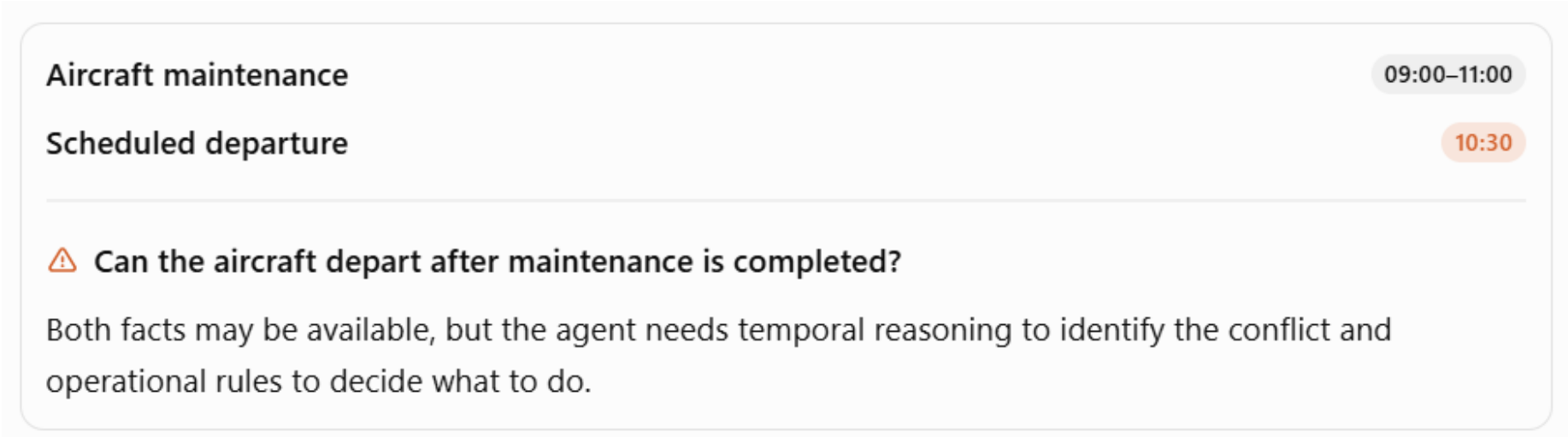
Governed permissions for humans, agents, data, and actions



Ontology System [data,logic,action,security] x [language,engine,toolchain]

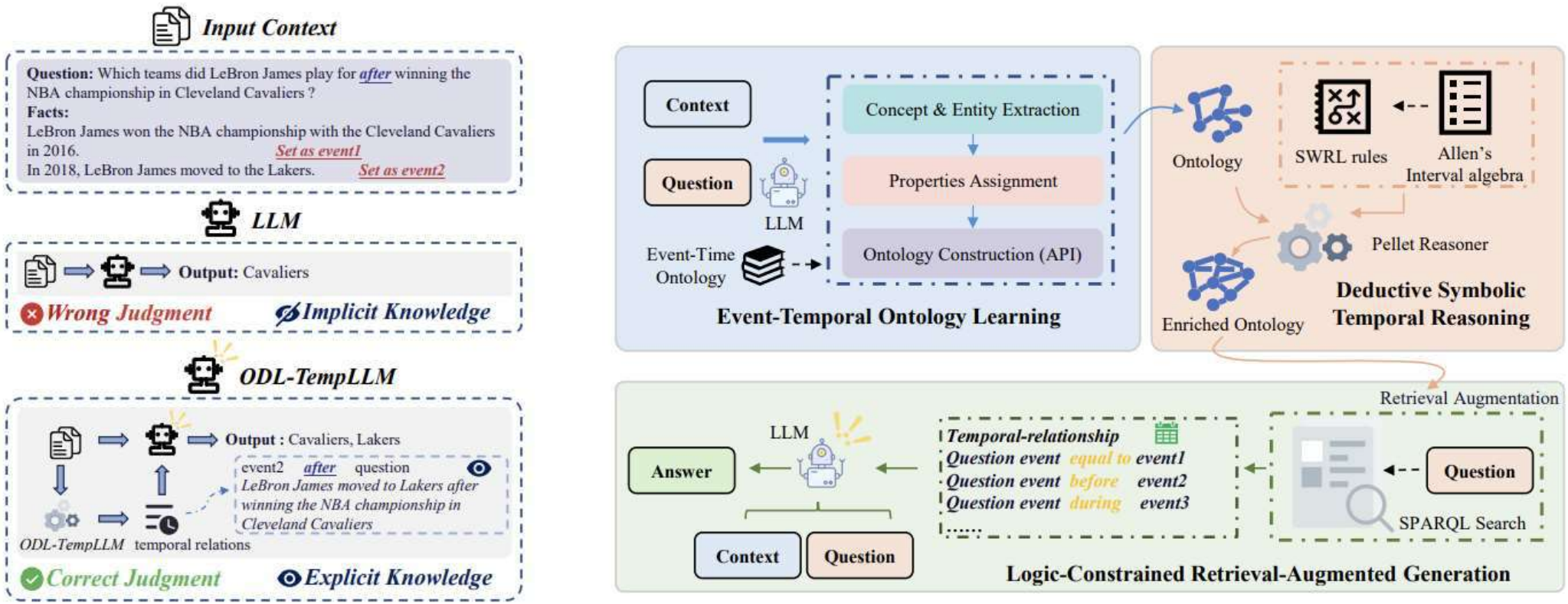
- **Research question behind this vision: even if an agent has access to the operational ontology, how can it reason reliably about its contents?**
- **Challenges:**
  - C1. Implicit temporal structure: Important relations between events are buried in natural-language records
  - C2. Unreliable inference: LLMs may miss or incorrectly derive before, after, during, or overlapping relationships
  - C3. Excessive information: Even a correctly constructed temporal knowledge base can overwhelm an LLM with irrelevant inferred facts

Return to the airline example. Real decisions depend on relationships between events, intervals, and constraints.



From Implicit LLM Reasoning to Explicit Ontology-Guided Reasoning

- **Problem**: As LLMs move into agentic workflows, temporal errors can affect operational decisions rather than merely produce incorrect answers
- **Idea**: externalize temporal structure and deduction, then return the relevant inferred facts to LLM.



ODL-TempLLM: Ontology-Guided and Description Logic-Reasoned Temporal Reasoning with LLMs. ACL 2026.

From Implicit LLM Reasoning to Explicit Ontology-Guided Reasoning

- **Evidence 1**: Improved temporal reasoning across benchmarks
- **Evidence 2**: Deductive reasoning recovers relations missed by LLMs

Token-level F1 using GPT-3.5-Turbo as the common backbone. Higher is better.

Benchmark	Zero-shot	Temp-CoT	ODL-TempLLM
CoTempQA (Mix)	46.10	79.20	**84.17**
MentaQA (All Factors)	27.99	71.67	**75.72**
TOT (Semantic)	27.79	68.16	**70.32**

ODL-TempLLM

Over 80%

Temporal-relation extraction accuracy on TOT

Compared approaches

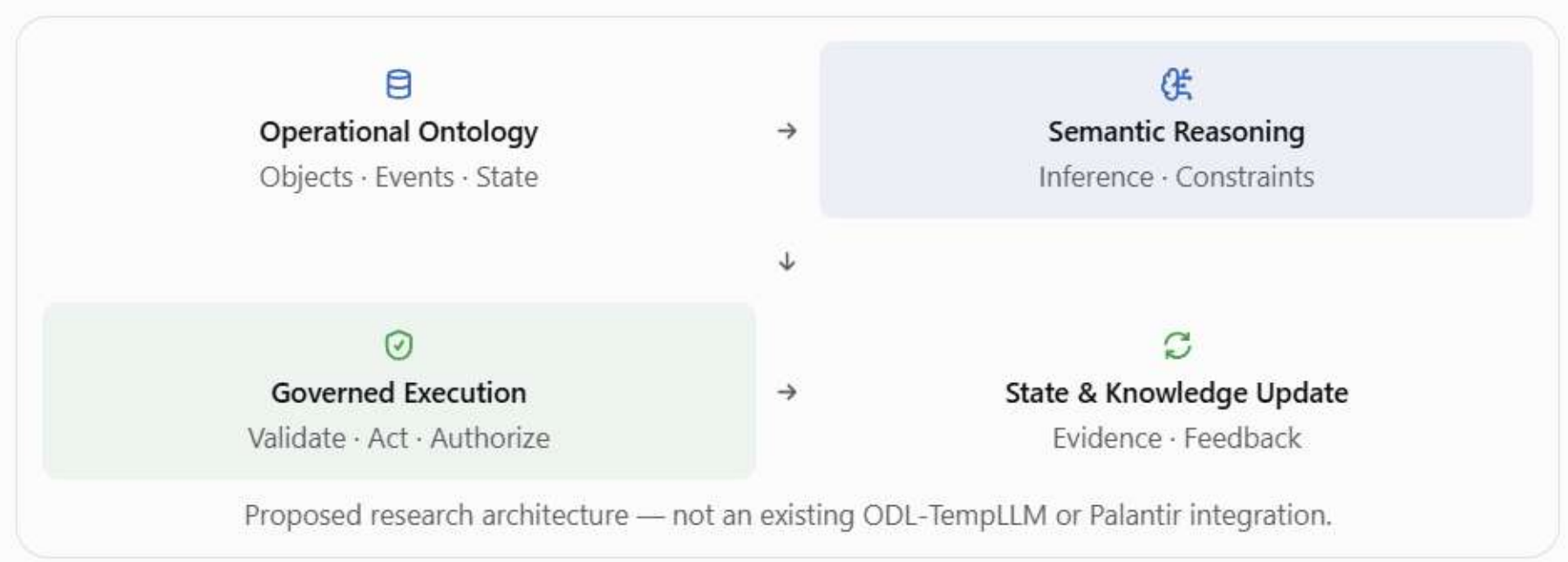
Below 40%

LLM-Context and LLM-Ontology variants

ODL-TempLLM: Ontology-Guided and Description Logic-Reasoned Temporal Reasoning with LLMs. ACL 2026.

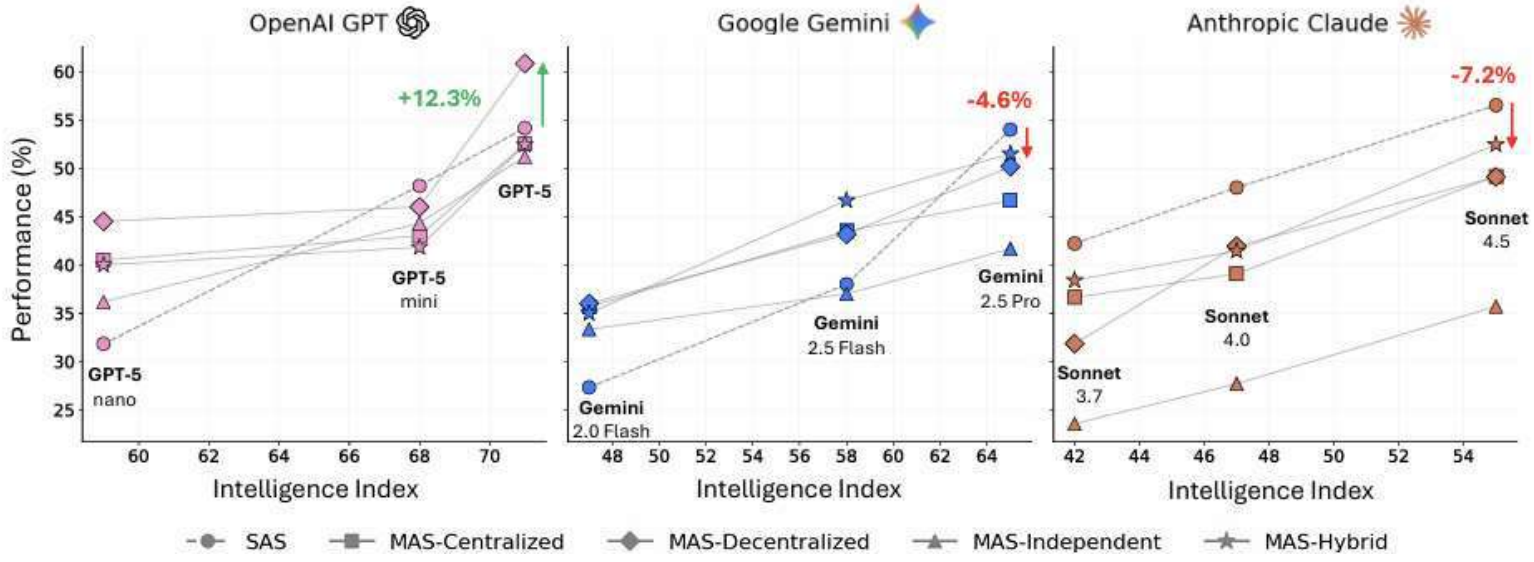
From Temporal Reasoning to Ontology-Governed Agent Behaviour

- **State-aware reasoning**: determine whether facts, remain valid as the operational world changes.
- **Semantic action control**: represent action preconditions, effects, temporal invariants, and authorization requirements so that agent proposals can be checked before execution.
- **Incremental and efficient reasoning**: update relevant parts without expensive global recomputation.



Team Scaling Works Only with Task–Architecture Fit

- **Performance change relative to single-agent baselines ranges from +80.8% on decomposable financial reasoning to −70.0% on sequential planning**
- Architectures without centralized verification tend to propagate errors more; centralized coordination reduces trace-level error amplification from 17.2× to 4.4× memorisation





Towards a Science of Scaling Agent System, Arxiv 2026. (Google Research)

JiuwenSwarm Starts with Mode Selection, Not Agent Count

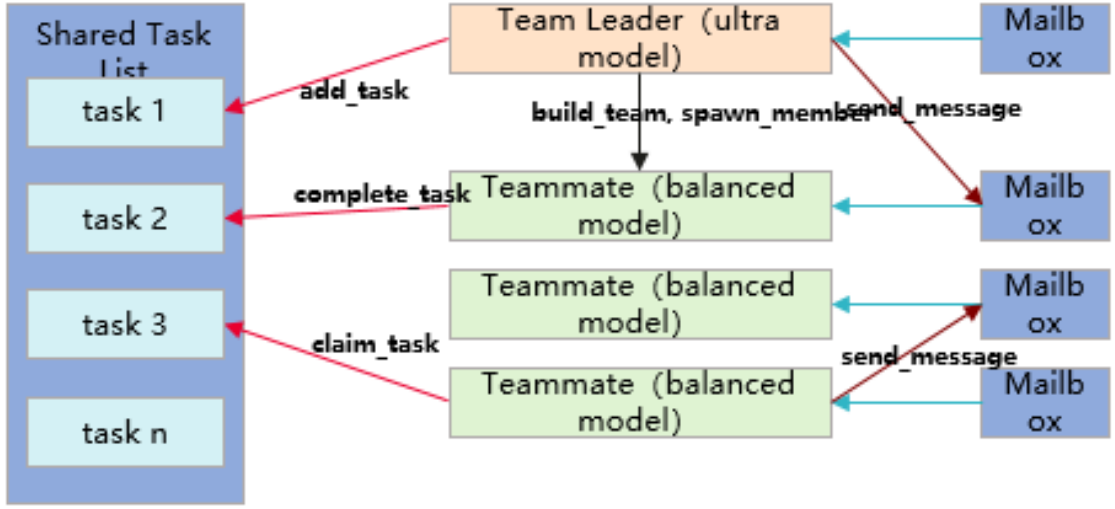
- The first design question is not “how many agents,” but “what mode does this task need?”
- JiuwenSwarm presents three execution modes, mapping naturally to the L1–L4 ladder:
  - Performance: lightweight execution
  - Plan: structured workflow
  - Swarm: multi-agent collaboration

Mode	Description	Best For
Plan Mode	Decomposes requirements into concrete steps, executes step by step	Complex tasks, need to confirm each step
Performance Mode	Flexible handling, supports parallel tasks	Simple tasks, fast response
Swarm Mode	Leader orchestrates multiple specialized Agents in coordinated collaboration	Large complex tasks, multi-role collaboration

JiuwenSwarm: Coordination Becomes a Reusable System Capability

- **From ad-hoc teamwork to coordination assets:** Successful collaboration becomes a portable Swarm Skill
- **Encode the coordination contract:** Roles, workflow, dependencies, bounds, and quality gates
- **Learn from coordination friction**: Bad handoffs become explicit Evolution Records





Swarm Skills: A Portable, Self-Evolving Multi-Agent System Specification for Coordination Engineering, arxiv 2026.

A Self-Evolution Mechanism for JiuwenSwarm

1. **CREATE**: Distil collaboration into a reusable Swarm Skill
2. **USE**: Reuse the right team, progressively
3. **PATCH**: Learn from coordination friction
4. **GOVERN**: Improve without accumulating chaos

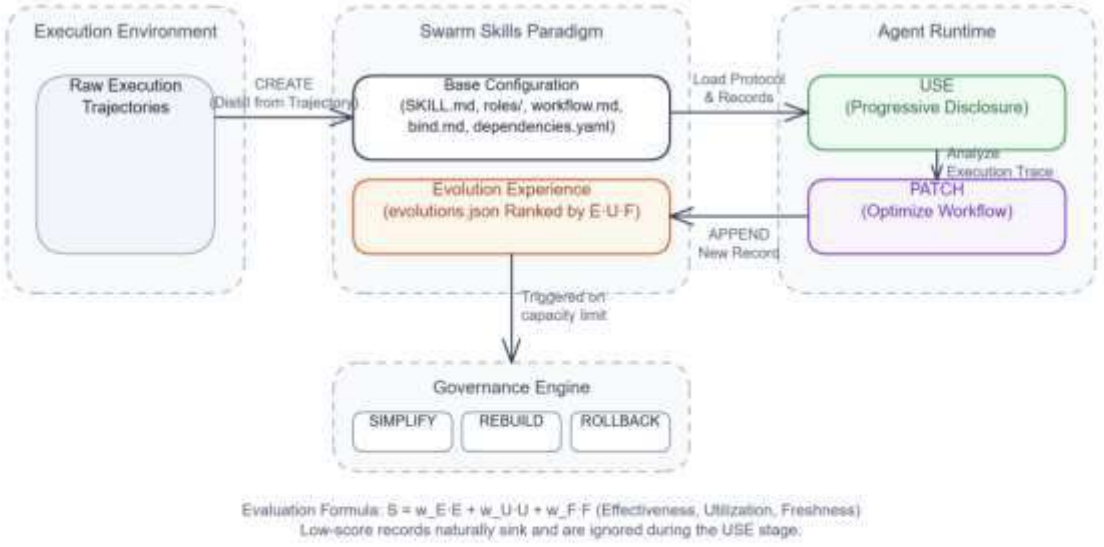
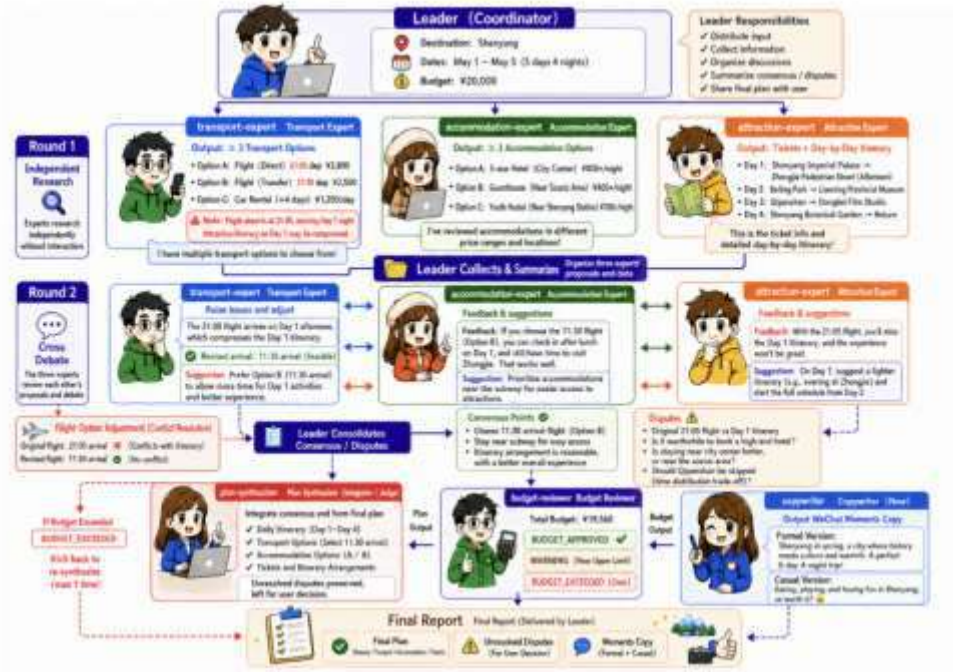


Figure 3: The Self-Evolution Lifecycle. The algorithm orchestrates a continuous loop starting from Trajectory Distillation (**CREATE**). During execution (**USE**), progressive disclosure securely loads protocols and historical records. Post-execution Friction Analysis (**PATCH**) appends new records. A Governance Engine periodically simplifies, rebuilds, or rolls back the asset based on the multidimensional $S = w_E \cdot E + w_U \cdot U + w_F \cdot F$ scoring of accumulated records.



Swarm Skills: A Portable, Self-Evolving Multi-Agent System Specification for Coordination Engineering, arxiv 2026.

Towards Scaling Agent Systems

- There are **other** dimensions of scaling for LLM agents than **retrieval scaling:**
  - **Context scaling**
    - Lost in the middle, Lost in between, …
  - **Memory scaling**
    - Can agents keep useful experience while forgetting outdated, repeated or unreliable information?
  - **State scaling**
    - Long horizon agents need to maintain query-able and updatable state objects
  - **Evolvability scaling**
    - cf. recent work of the red queen Gödel machine
  - **Organisational scaling**
    - Scale the collaboration between human and agent



Large Language Models and Knowledge Graphs: Opportunities and Challenges

Jeff Z. Pan
The University of Edinburgh, United Kingdom
Jan-Christoph Kalo
University of Amsterdam,
The Netherlands
Jiaoyan Chen
The University of Manchester, United Kingdom
University of Oxford, United Kingdom
Hajira Jabeen
GESIS – Leibniz Institute for the Social Sciences,
Cologne, Germany
Wen Zhang
Zhejiang University, China
Russa Biswas
Hasso-Plattner Institute, Potsdam, Germany
Angela Bonifati
Lyon 1 University, CNRS, IUF, France
Mauro Dragoni
Fondazione Bruno Kessler, Trento, Italy
Simon Razniewski
Bosch Center for AI, Renningen, Germany
Sneha Singhania
Max Planck Institute for Informatics,
Saarbrücken, Germany
Stefan Dietze
GESIS – Leibniz Institute for the Social Sciences,
Cologne, Germany
Heinrich-Heine-Universität Düsseldorf, Germany
Janna Omeliyanenko
University of Würzburg, Germany
Matteo Lissandrini
Aalborg University, Denmark
Gerard de Melo
Hasso-Plattner Institute, Potsdam, Germany
Edlira Vakaj
Birmingham City University, United Kingdom
Damien Graux
Edinburgh Research Centre, CSI,
Huawei Technologies UK, United Kingdom

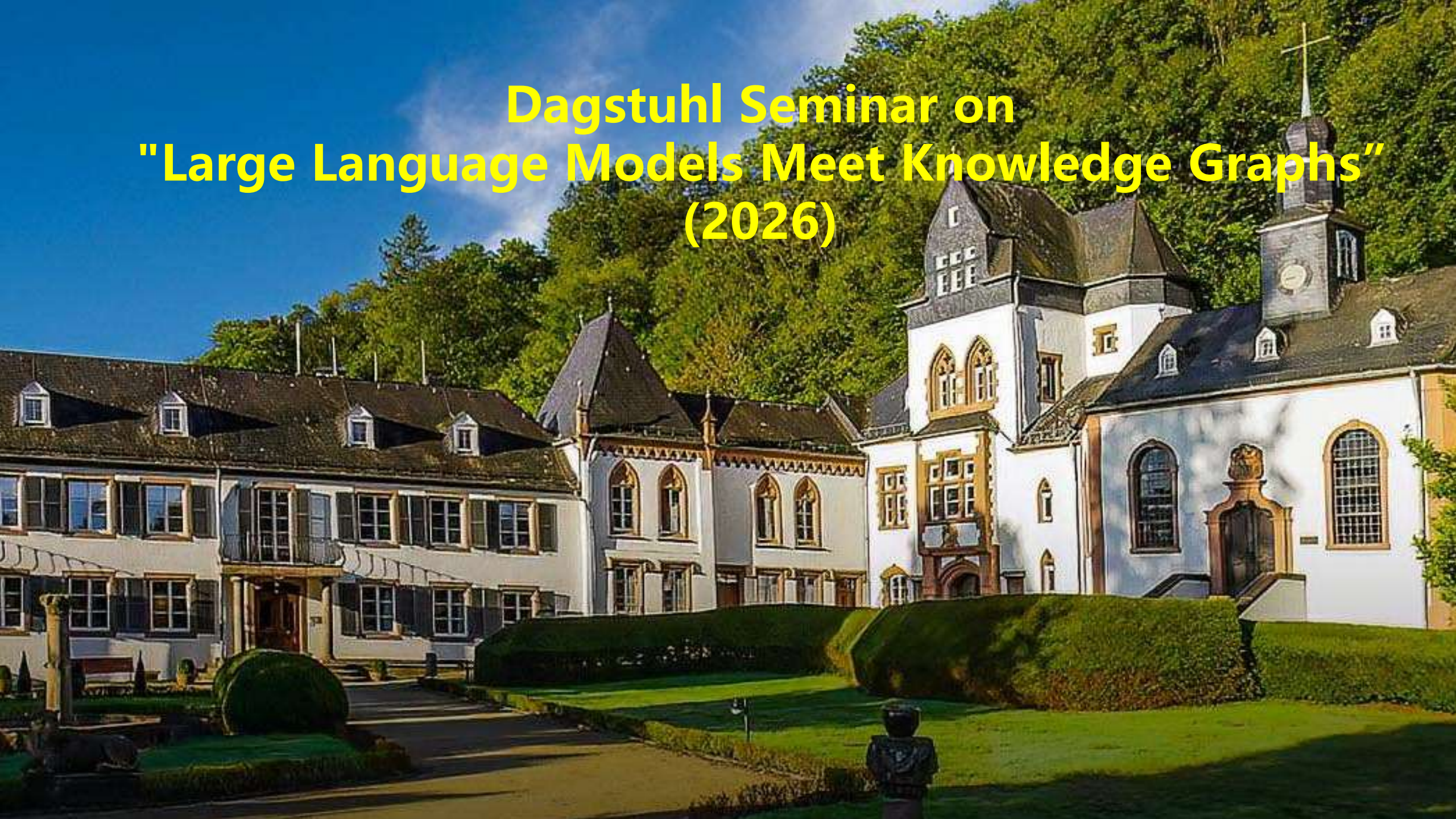
Abstract

Large Language Models (LLMs) have taken Knowledge Representation – and the world – by storm. This inflection point marks a shift from explicit knowledge representation to a renewed focus on the hybrid representation of both explicit knowledge and parametric knowledge. In this position paper, we will discuss some of the common debate points within the community on LLMs (parametric knowledge) and Knowledge Graphs (explicit knowledge) and speculate on opportunities and visions that the renewed focus brings, as well as related research topics and challenges.

2012 ACM Subject Classification General and reference → Surveys and overviews; Computing methodologies → Knowledge representation and reasoning; Computing methodologies → Natural language processing
Keywords and phrases Large Language Models, Pre-trained Language Models, Knowledge Graphs, Ontology, Retrieval Augmented Language Models
Digital Object Identifier 10.4230/TGDK.1.1.2
Category Position
Related Version *Full Version*: https://arxiv.org/abs/2308.06374
Acknowledgements We would like thank Xiaoqi Han's helpful discussion and support when finalising the camera ready version of the paper.
Received 2023-06-26 Accepted 2023-08-31 Published 2023-12-19
Editors Aidan Hogan, Ian Horrocks, Andreas Hotho, and Lalana Kagal
Special Issue Trends in Graph Data and Knowledge

© Jeff Z. Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, Russa Biswas, Gerard de Melo, Angela Bonifati, Edlira Vakaj, Mauro Dragoni, and Damien Graux;
licensed under Creative Commons License CC-BY 4.0
Transactions on Graph Data and Knowledge, Vol. 1, Issue 1, Article No. 2, pp. 2:1–2:38
Transactions on Graph Data and Knowledge
TGDK Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Large Language Models and Knowledge Graphs: Opportunities and Challenges. TGDK, 2023.


Dagstuhl Seminar on
"Large Language Models Meet Knowledge Graphs"
(2026)