From Foundation Knowledge Graph Construction to Foundation Knowledge Models

Yangqiu Song

Department of CSE, HKUST

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

THE DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING 計算機科學及工程學系

KnowComp Group

Understanding the World by Computational Knowledge

Main Contributors

Jiaxin Bai

Hong Ting Tsang

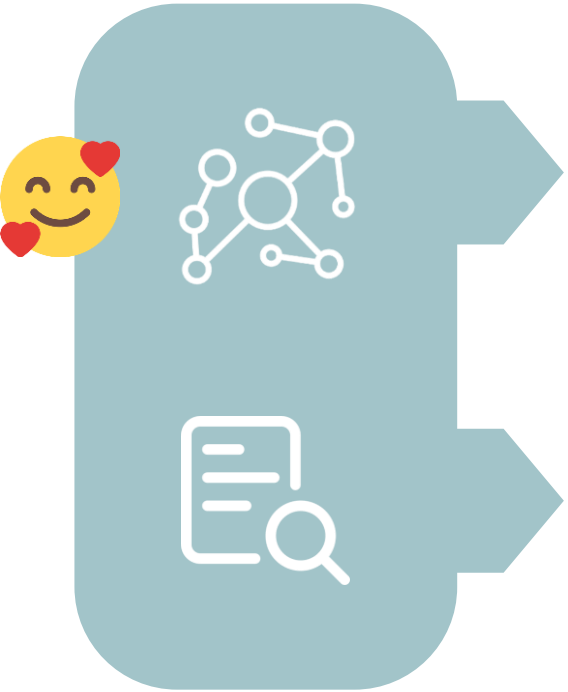
Haoyu Huang

Yufei Li

Zhongwei Xie

Yisen Gao

Why We STILL Need Foundation Knowledge Graphs?

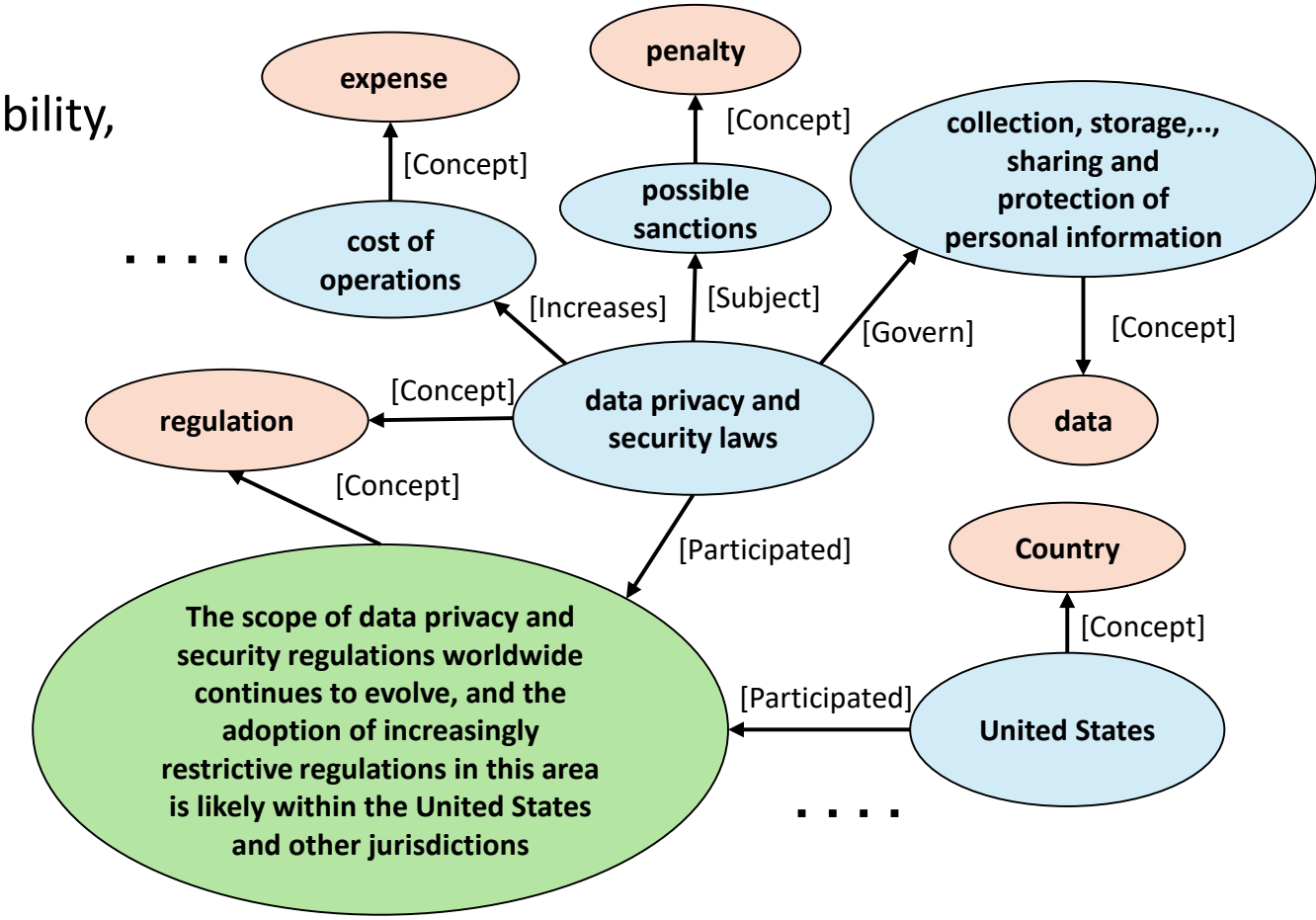


- Deliver **explicit, structured knowledge** through nodes and edges (as a **global structure**). Serve as an **organized, readily navigable** external **memory** that can be injected into LLMs with clear provenance.
- Offers **systematic provenance, semantic layer or schema/ontology**, hampering traceability and increasing the risk of propagating **noisy or contradictory information to LLMs**.

The Inherent Paradox of Traditional KG Construction

Previous KG construction approaches face an inherent paradox: they require predefined schemas created by domain experts, which fundamentally limits their scalability, adaptability, and domain coverage.

- Traditional approaches (Freebase, YAGO) start with a manually designed ontology.
- Even recent LLM-based methods (PiVE, iText2KG, GraphRAG) still rely on fixed prompts or heuristics.
- Traditional ways of KG definitions capture static facts ("Paris is the capital of France"), but miss dynamic knowledge — temporal relationships, causality, and procedural knowledge among events.



AutoSchemaKG Overview

KnowComp

Input Documents

- **Source Selection:** Wikipedia, academic papers, Common Crawl
- **Segmentation:** Divide documents into chunks
- **Batching:** Group chunks into batches

Document 1: Lorem ipsum dolor sit amet...
Document 2: Consectetur adipiscing elit...
Document 3: Sed do eiusmod tempor incididunt...
...

Chunked Batches

Batch1 | Batch2 | Batch3

Knowledge Graph Triples

- **Entity-Entity Extraction:** Using prompt P_{EE}, extract relationships between entities
- **Entity-Event Extraction:** Using prompt P_{EV}, link entities to events
- **Event-Event Extraction:** Using prompt P_{VV}, detect temporal/causal connections
- **Output Processing:** Parse JSON outputs, repair if needed

Triple Extraction Process

Entity — Relation — Entity
Entity — Participates — Event
Event — Causes — Event

Schema Induction

Black Mountain College
↓ Conceptualization
College | School | Institution

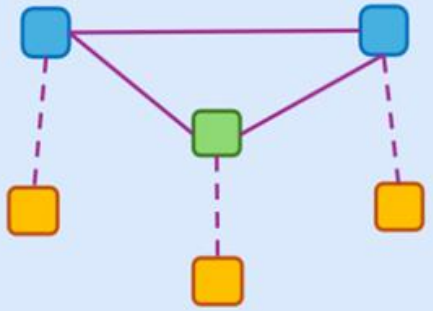
A cat chased a prey
↓ Conceptualization
Hunting | Predation | Pursuit

- **Entity Conceptualization:** Generate abstract entity types using context from neighboring nodes
- **Event Conceptualization:** Abstract events into general event types
- **Relation Conceptualization:** Transform relations into semantic categories

Knowledge Graph Construction

- **Node Creation:** Build entity (V_N), event (V_E), and concept (C) nodes
- **Edge Creation:** Establish connections using extracted relationships (R)
- **Concept Linking:** Connect nodes to corresponding concept nodes
- **Graph Finalization:** Complete knowledge graph $G = (V, E, C, \varphi, \psi)$

ATLAS Knowledge Graph



Entity | Event | Concept | Relation

In short, it **removes the manual bottleneck** and creates **richer**, more **adaptable** knowledge graphs.

AutoSchemaKG: Biggest Open Source GraphRAG SOTA

The ATLAS (Automated Triple Linking And Schema induction) knowledge graph family built from multiple corpora:

Metric	Total
Text Chunks	52.557M
Entities	387.022M
Events	954.548M
Concepts	45.056M
Total Nodes	**1.355B**
Total Edges	**8.600B**

1B+
Total Nodes

8.6B+
Total Edges

78,400
GPU Hours

https://github.com/HKUST-KnowComp/AutoSchemaKG

Multi-hop QA Performance

Dataset	Method	EM	F1
HotpotQA	No Retriever	37.0	47.3
	GraphRAG [1]	55.2	68.6
	HippoRAG2 [2]	**62.7**	75.5
	AutoSchemaKG	61.8	**78.3**
2Wiki	No Retriever	36.5	42.8
	GraphRAG	51.4	58.6
	HippoRAG2	65.0	71.0
	AutoSchemaKG	**65.3**	**73.9**

Key Finding: 12-18% improvement over state-of-the-art baselines on multi-hop QA tasks

Computational Requirements: Built using 80GB GPUs with 1,513 TFLOPS of FP16 compute, running Llama-3-8B-instruct with Flash Attention.

Automation ≠ Good Enough for Downstream Tasks

- Automatic Reconstruction
  - AutoGraph-R1
- Automatic Refinement
  - DeepRefine

Building "Good" Graphs for Downstream Tasks

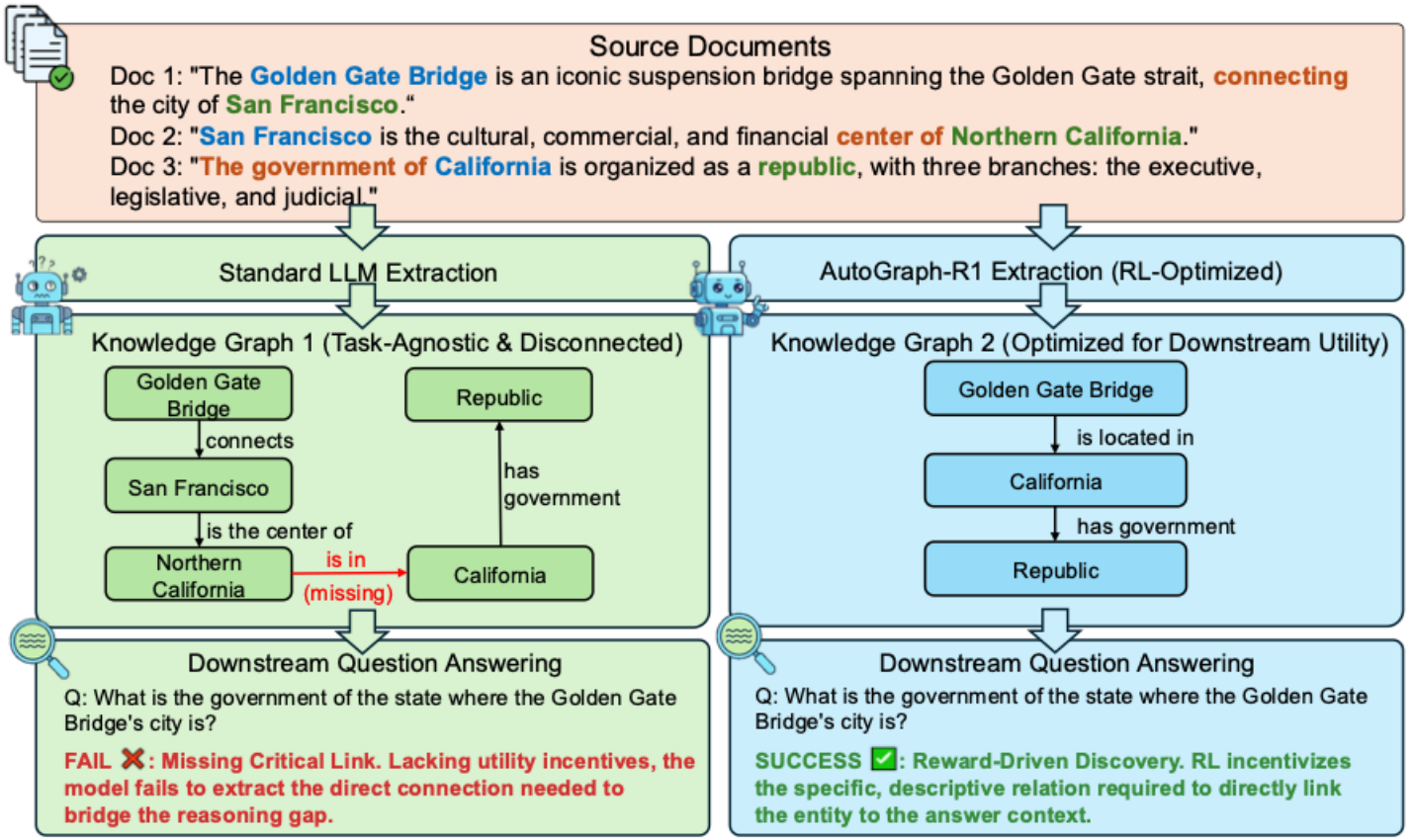
Key Insight

The bottleneck is not graph quality — it's task alignment.

A "good" graph by intrinsic standards may be:

- Too noisy for a text-indexing retriever
- Too sparse for a multi-hop reasoning retriever

Our question: Can we train the KG constructor to build graphs that are **directly useful** for downstream QA?



Different retrievers demand fundamentally different graph structures. This is the core motivation for our reward design.

AutoGraph-R1: Closing the Loop with Reinforcement Learning

KG Constructor (Trainable)

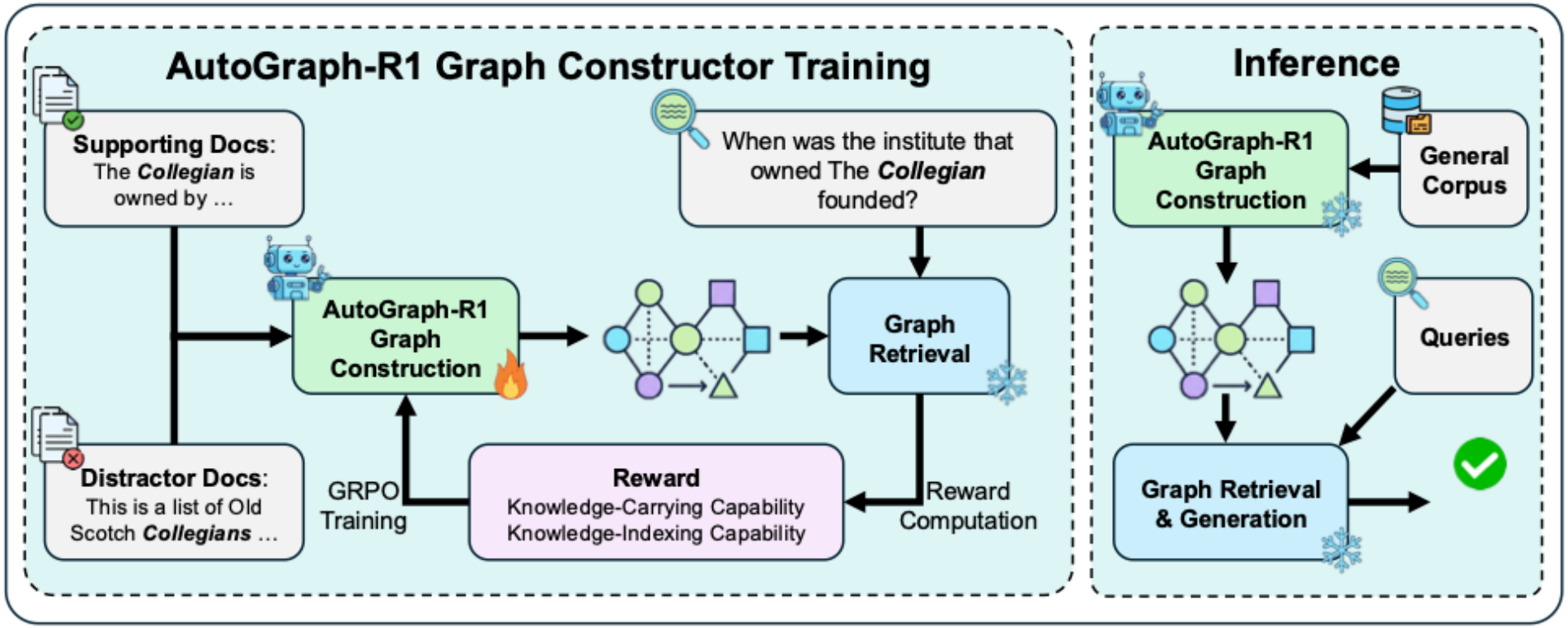
π_θ^KG: LLM maps documents → graph G
Only component that is updated

Frozen RAG Server

Retriever + Answer generator
Never updated during training
Uses G to produce answer ŷ

Task-Specific Reward

R(q, ŷ, y, G): measures graph's functional utility in the RAG pipeline



Core innovation: The reward signal reflects the graph's FUNCTIONAL UTILITY — not intrinsic quality. This closes the loop between construction and application.

Two Retrieval Paradigms, Two Reward Functions

Paradigm 1: Graph as Knowledge Carrier

- Triples serve as the primary reasoning evidence.
- The retriever extracts subgraphs via n-hop expansion.
- **Challenge:** Missing edges break multi-hop chains.
- **Retrievers:** Subgraph, Triples, ToG

$$R_C(q, y, G) = \mathbb{1}[\text{deducible}(q, y \mid G)]$$

Binary judge: "Can the answer be deduced from these triples?" → Encourages dense connectivity

Paradigm 2: Graph as Knowledge Index

- Graph structure guides Personalized PageRank to rank and retrieve relevant text passages.
- **Challenge:** Noisy edges dilute relevance signals.
- **Retrievers:** HippoRAG, HippoRAG2

$$R_I(q, D_{gold}, G) = |\text{Top-k} \cap D_{gold}| \,/\, |D_{gold}|$$

Passage recall — simple set intersection, no model inference → Encourages clean, precise edges

⚠ *Naive F1 score as reward leads to unstable training. Our structured rewards achieve stable convergence (Appendix B).*

Performance Summary of AutoGraph-R1

- **Knowledge Carrier**
  - +4.39 avg F1 (Qwen-3B)
  - Gains on all 5 benchmarks
  - Transfers across 3 retrievers
- **Knowledge Index**
  - +4.35 avg Recall@5
  - Gains on all 5 benchmarks
  - Transfers: HippoRAG & RAG2
- **Key Insight**
  - Both paradigms show consistent improvements. The RL-trained constructor generalizes across all retrievers within each paradigm.

GRPO — Group Relative Policy Optimization

1. Sample a batch of (query, documents, answer) tuples
2. Generate G candidate graphs from the current policy
3. For each graph: run the frozen RAG pipeline → compute R_C or R_I
4. Normalize advantages within the group: $\hat{A}_i = (R_i - \mu) / \sigma$
5. Update the policy via GRPO with clipping

Hong Ting Tsang, Jiaxin Bai, Haoyu Huang, Qiao Xiao, Tianshi Zheng, Baixuan Xu, Shujie Liu, Yangqiu Song: AutoGraph-R1: End-to-End Reinforcement Learning for Knowledge Graph Construction. ACL (1) 2026

DeepRefine: Motivation

In many real deployments, the KB/KG has already been constructed, so re-training the constructor or re-compiling the whole KB/KG is expensive and impractical at scale, motivating our post-construction agent-compiled knowledge refinement paradigm, which is more agent-native.

Table 2: The average time consumptions (s) ↓ of the knowledge refinement process of DeepRefine and the reconstruction process of AutoGraph-R1.

Methods	Simple QA	Multi-hop QA	Conversation QA
AutoGraph-R1	6,782.8	9,780.4	2,826.5
DeepRefine	**3,201.7**	**3,357.5**	**1,115.8**

Huang H, Bai J, Liu S, et al. DeepRefine: Agent-Compiled Knowledge Refinement via Reinforcement Learning[J]. arXiv preprint arXiv:2605.10488, 2026.

KGs/KBs are also Changing

Past RAG Paradigm

- LLMs grounded in external knowledge bases via RAG.
- Graph-based RAG: organizes content into KGs.
- Static KBs are stateless: do not accumulate structured lessons.

Agent-Compiled Knowledge Bases

- LLM agents ingest sources, synthesize articles.
- Maintain cross-references over time (e.g., **LLM-Wiki**).
- Make KBs evolvable rather than frozen at test time.
- Re-constructing is expensive and operationally impractical.

Three Systematic Quality Defects:

- Incompleteness: Missing evidence chains, cross-document links.
- Incorrectness: Low-confidence or imprecise assertions.
- Redundancy: Ambiguous or coreference resolution issues.

Huang H, Bai J, Liu S, et al. DeepRefine: Agent-Compiled Knowledge Refinement via Reinforcement Learning[J]. arXiv preprint arXiv:2605.10488, 2026.

DeepRefine: Agentic Harness Design

Task of knowledge refinement: generating sequences of refinement actions that mitigates issues in the KB.

$$\arg\max_{S_{\mathcal{A}_q} \in S} p_\theta(S \mid \mathcal{G}_f, \mathcal{Q}) = \{\mathcal{A}_q^1, \ldots, \mathcal{A}_q^{|\mathcal{Q}|}\}$$

DeepRefine's agentic harness as a three-step reasoning process, including

- Answerability Judgement Loop,
- Error Abduction, and
- Refinement Actions Generation.

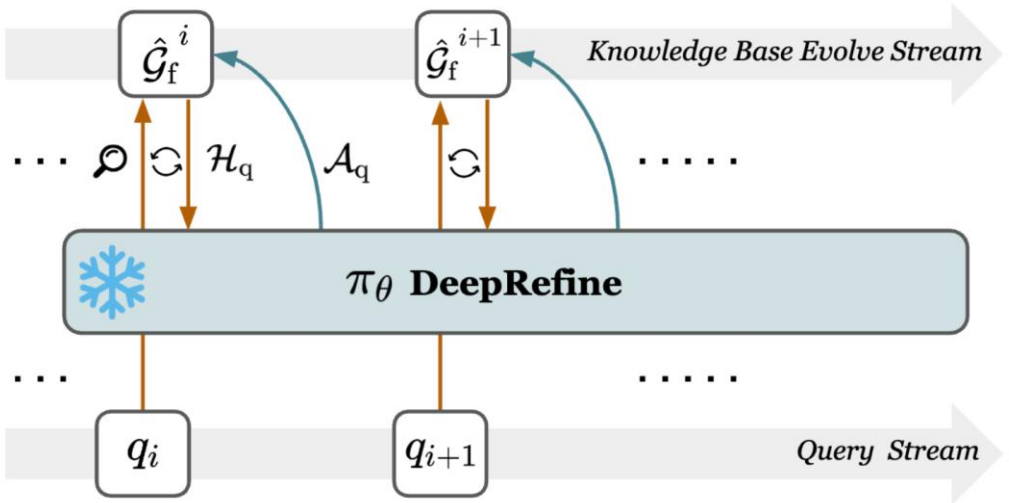


Figure 3: An overview of the inference process of DeepRefine.

Huang H, Bai J, Liu S, et al. DeepRefine: Agent-Compiled Knowledge Refinement via Reinforcement Learning[J]. arXiv preprint arXiv:2605.10488, 2026.

DeepRefine: Agentic Harness Design

Answerability Judgement

- Conduct a simple dense retrieval to obtain the top-N related triples to the query.
- Judge the answerability, if not, continue to expand the retrieved subgraph.

Error Abduction

Prompt Template for Error Abduction.

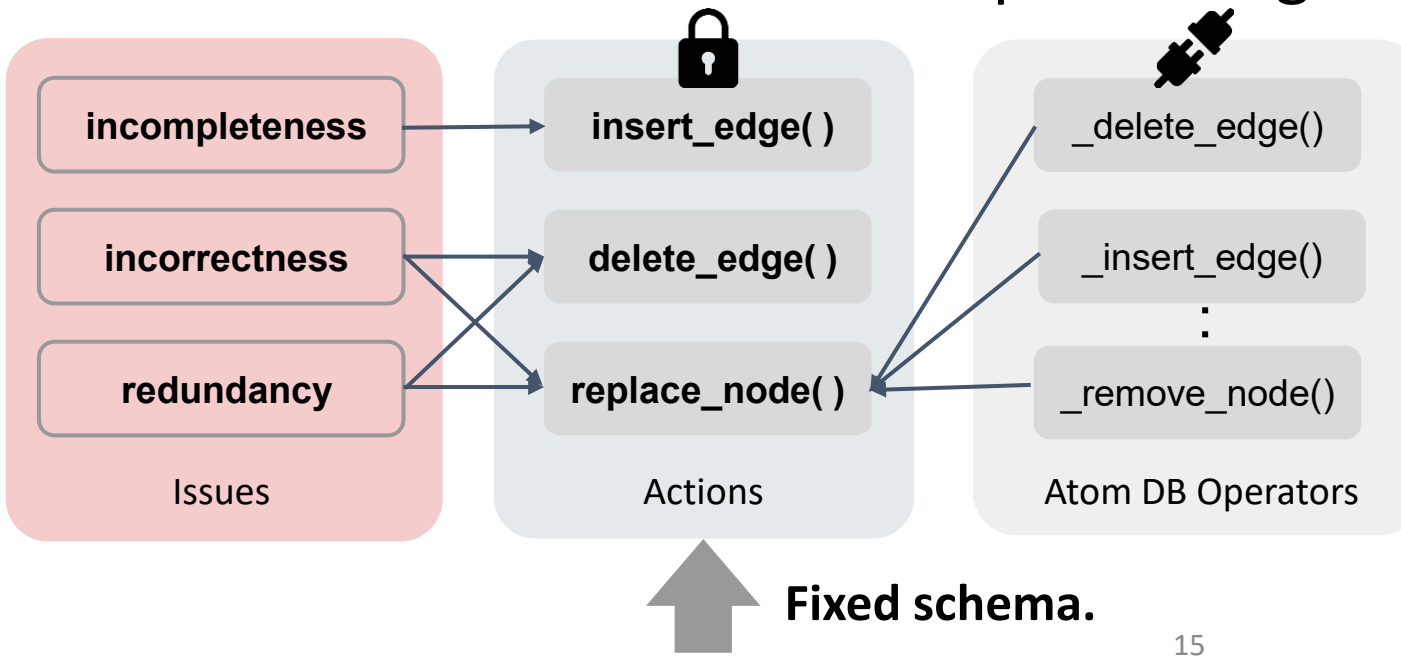
System:
As an advanced error abduction assistant, your task is to analyze the error reasons based on the given interaction history.
Analyze the reasons of the unanswerable questions based on the given interaction history.
Output your analysis in the following format:
<abduction>...</abduction>
Important: You must think carefully about the interaction history before making your analysis. And output your analysis result directly in the specified format.

User:
Interaction history: **{interaction_history}**

Output:

```
<abduction>...</abduction>
```

Refinement Actions Generation space design:



DeepRefine: Performance

OOD experimental settings.

DeepRefine brings **consistent downstream gains** over various knowledge base constructors and knowledge retrievers.

DeepRefine can sometimes introduce **larger or competitive downstream gains** than the reconstruction method, e.g., AutoGraph-R1(AR1).

Constructor	Methods	Simple QA		Multi-hop QA		Conversation	Overall
		NQ*	PopQA*	2WikiQA	Musique	LOCOMO	Avg.
Naive	Subgraph	26.43	54.48	33.53	**13.82**	25.86	30.82
	Subgraph + DeepRefine-4B	**27.50**	56.00	34.64	12.75	26.53	31.48
	Subgraph + DeepRefine-8B	26.65	**56.29**	**35.62**	12.86	**27.49**	**31.72**
	ToG	25.55	54.92	43.74	18.21	22.64	33.01
	ToG + DeepRefine-4B	25.60	**56.10**	42.88	16.91	23.81	33.06
	ToG + DeepRefine-8B	**25.72**	55.66	**44.02**	**19.32**	**24.05**	**33.75**
	HippoRAG	35.54	63.56	49.77	26.54	33.25	41.73
	HippoRAG + DeepRefine-4B	**35.85**	**63.73**	**50.23**	26.27	**33.73**	41.96
	HippoRAG + DeepRefine-8B	35.69	62.96	49.84	**28.27**	33.56	**42.06**
	HippoRAG2	35.91	63.08	51.47	25.33	**33.33**	41.82
	HippoRAG2 + DeepRefine-4B	35.93	62.81	51.26	25.31	32.52	41.57
	HippoRAG2 + DeepRefine-8B	**36.03**	**63.40**	**53.02**	**26.33**	32.42	**42.24**
AR1	Subgraph	29.27	58.40	35.60	14.42	26.65	32.87
	Subgraph + DeepRefine-4B	28.10	**59.07**	37.21	15.58	27.71	33.53
	Subgraph + DeepRefine-8B	**29.41**	58.99	**37.76**	**15.86**	**27.79**	**33.96**
	ToG	29.88	60.34	48.56	19.06	23.64	36.30
	ToG + DeepRefine-4B	29.47	60.50	50.08	18.17	**27.84**	37.21
	ToG + DeepRefine-8B	**29.91**	**61.20**	**50.13**	**20.14**	26.16	**37.51**
	HippoRAG	39.84	64.43	50.34	26.21	35.02	43.17
	HippoRAG + DeepRefine-4B	39.75	63.87	51.89	26.44	**36.13**	43.62
	HippoRAG + DeepRefine-8B	**40.09**	**64.63**	**52.75**	**27.44**	35.64	**44.11**
	HippoRAG2	37.25	64.69	52.61	27.96	35.75	43.65
	HippoRAG2 + DeepRefine-4B	37.71	63.91	53.06	27.65	36.52	44.17
	HippoRAG2 + DeepRefine-8B	**38.60**	**65.24**	**55.45**	**30.30**	**36.75**	**45.27**
Graphify (LLM-Wiki)	Subgraph	20.42	11.38	25.37	11.69	5.24	14.82
	Subgraph + DeepRefine-4B	20.97	**15.86**	25.85	11.36	5.75	15.96
	Subgraph + DeepRefine-8B	**21.26**	15.78	**26.34**	**12.43**	**5.95**	**16.35**
	ToG	16.69	8.97	21.46	11.73	7.30	13.23
	ToG + DeepRefine-4B	17.90	**10.88**	21.28	11.98	7.08	13.82
	ToG + DeepRefine-8B	**18.22**	10.61	**21.55**	**12.33**	**7.39**	**14.02**
	HippoRAG	13.23	5.57	29.03	8.67	3.44	11.99
	HippoRAG + DeepRefine-4B	12.90	6.02	29.84	**8.77**	**4.50**	12.41
	HippoRAG + DeepRefine-8B	**13.46**	**6.42**	**31.61**	8.76	4.33	**12.92**
	HippoRAG2	13.62	6.19	27.98	8.10	3.57	11.89
	HippoRAG2 + DeepRefine-4B	**13.90**	6.60	28.35	8.08	3.69	12.12
	HippoRAG2 + DeepRefine-8B	13.77	**6.69**	**32.91**	**8.45**	**4.09**	**13.18**

Huang H, Bai J, Liu S, et al. DeepRefine: Agent-Compiled Knowledge Refinement via Reinforcement Learning[J]. arXiv preprint arXiv:2605.10488, 2026.

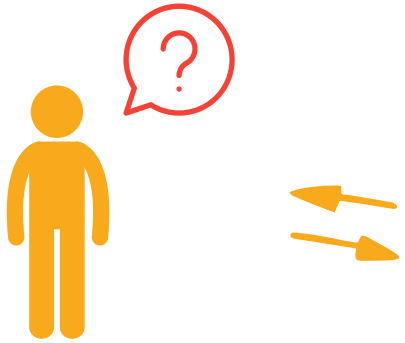
From Foundation KG Construction to Foundation Knowledge Models

KnowComp
Understanding the World by Computational Knowledge

1. Read/write: relevancy and scalability.

2. Augmentation: LLM native (embeddings) or text native (natural language).

3. Interaction: Proactive or reactive.



Agent-Native: retrieve text information and perform context engineer

Large Language Models (LLMs)

AI-Native: retrieve vectors and perform learning/inference

Knowledge Bases

Embedding 1010 1010

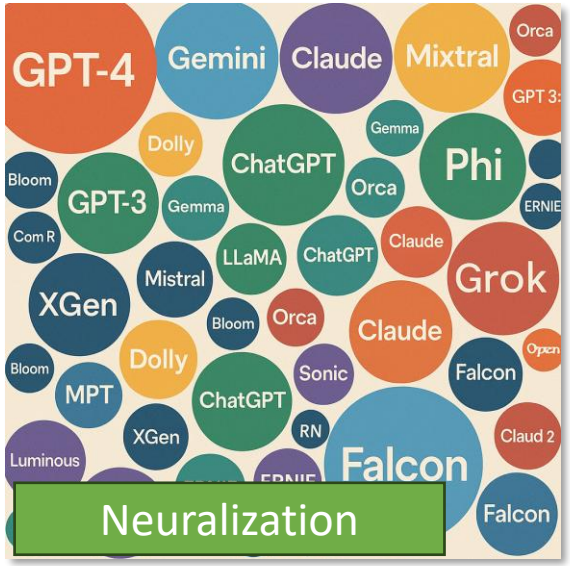
Vector Databases

1010 1010 1010 1010
1010 1010 1010 1010
1010 1010 1010 1010
1010 1010 1010 1010

Reactive: Enabled by LLMs to have a better fuzzy semantic search

Proactive: Database/knowledge base as a smart agent: understanding LLMs/users' intention and making predictions for more structured data and NP-complete problems: self-evolving, reasoning, in-context learning, etc.

Generalization from Structured Data?

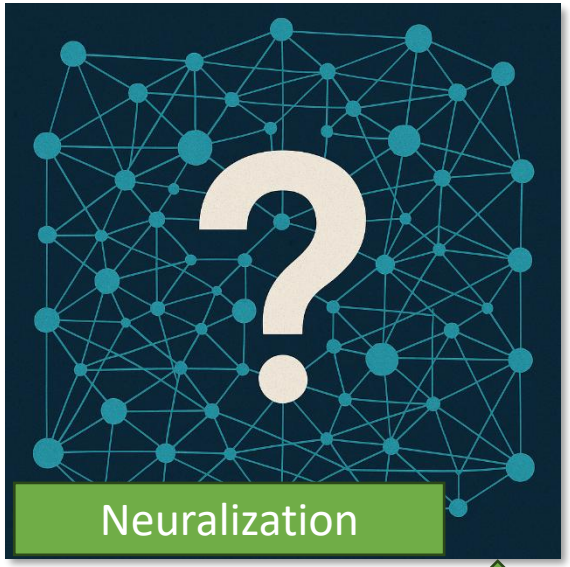


Large Language Model

Ilya Sutskever [1]

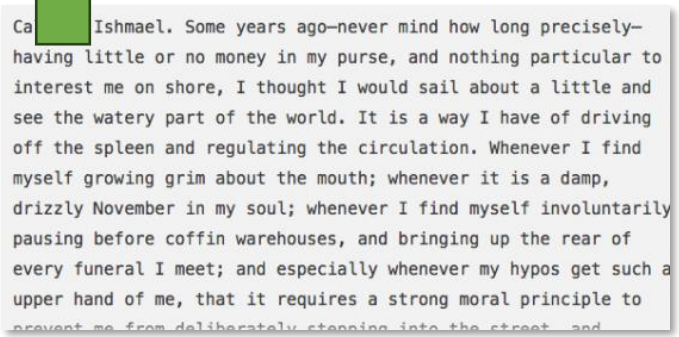
LLM Pre-training scaling will stop because we only have one internet!

Neural Graph Database (NGDB) show great potential for Database Memorization and Generalization!



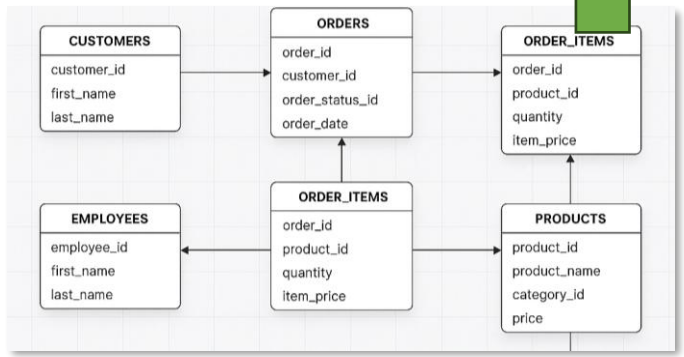
Neural (Graph) Databases

Text Data



Ca Ishmael. Some years ago—never mind how long precisely—
having little or no money in my purse, and nothing particular to
interest me on shore, I thought I would sail about a little and
see the watery part of the world. It is a way I have of driving
off the spleen and regulating the circulation. Whenever I find
myself growing grim about the mouth; whenever it is a damp,
drizzly November in my soul; whenever I find myself involuntarily
pausing before coffin warehouses, and bringing up the rear of
every funeral I meet; and especially whenever my hypos get such a
upper hand of me, that it requires a strong moral principle to

Database Data

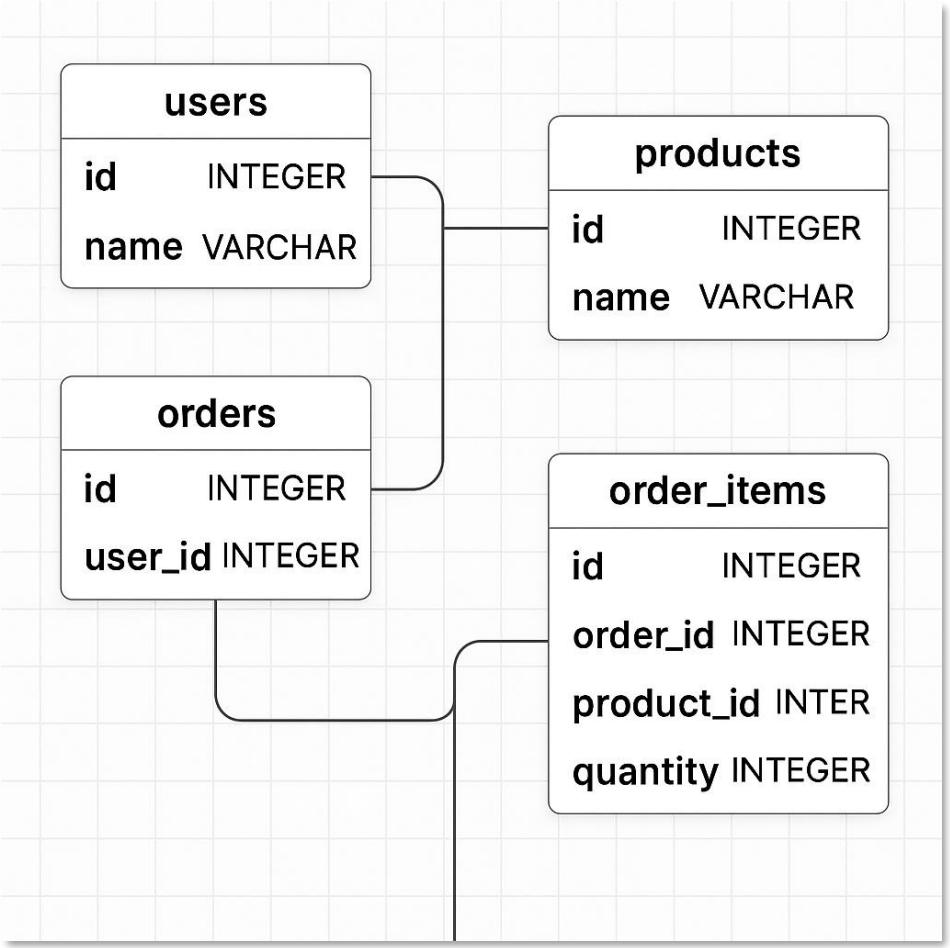


- Data within databases **continues to grow**;
- At **very early stage** where universities can do research [2];
- Combines **general applicability** to real-world applications, positioning it to make industry impact;
- Many **private domain** databases.

[1] https://youtu.be/1yvBqasHLZs

[2] Wang, Y., Wang, X., Gan, Q., Wang, M., Yang, Q., Wipf, D., & Zhang, M. (2025). Griffin: Towards a Graph-Centric Relational Database Foundation Model. arXiv preprint arXiv:2505.05568.

Recent Related Efforts: Relational Deep Learning

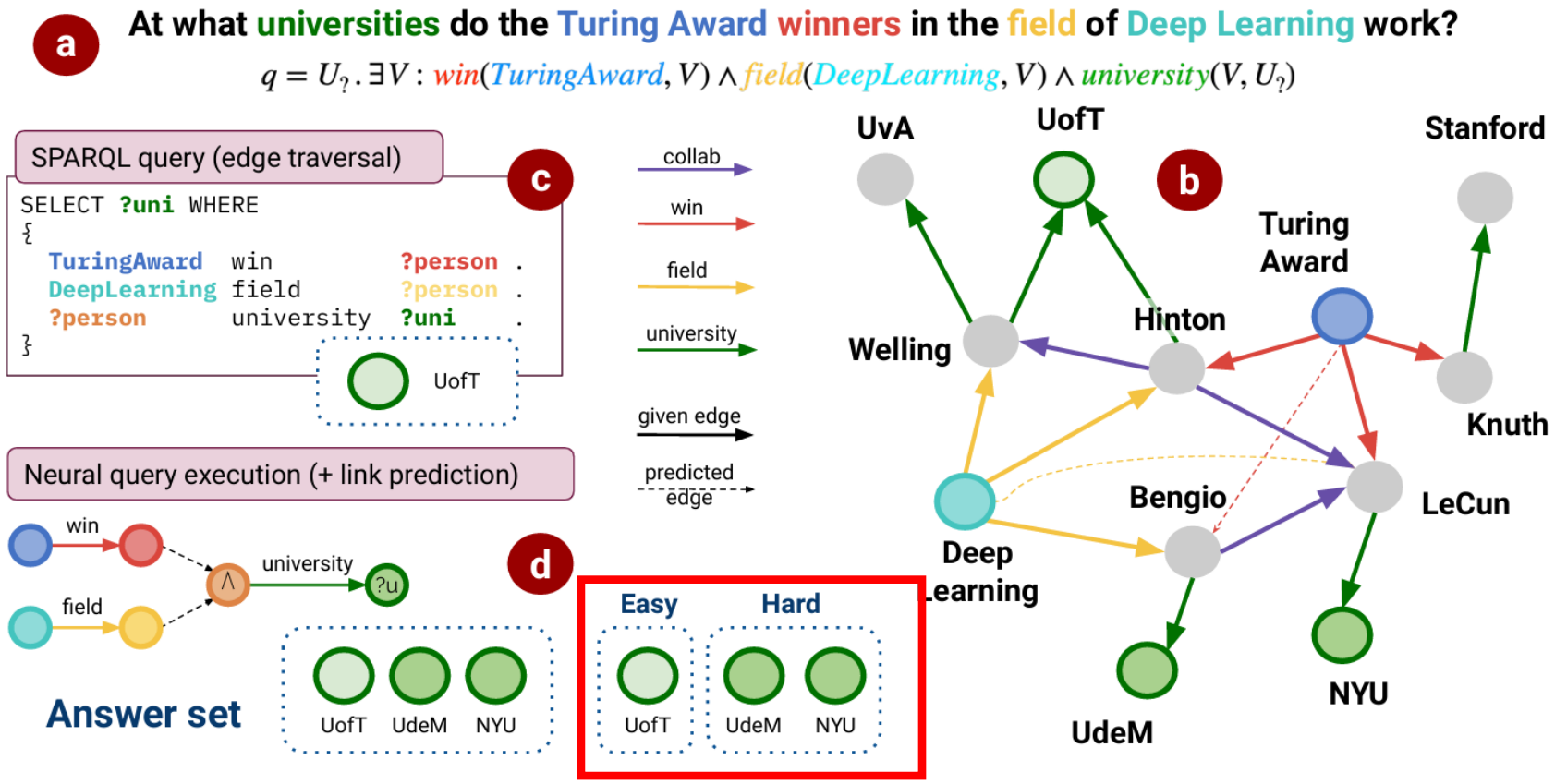


- **Data Is Stored in Databases**
  - **Commerce**
  - **Finance**
  - **Social Media**
  - **Medical**
- **Core Concept**: Applying deep learning directly to relational databases tables by treating them as graphs [1]
- **Statistical Relational Learning** (SRL) has been a long standing research topic.

Jure Leskovek
(Stanford)

[1] Fey, M., Hu, W., Huang, K., Lenssen, J. E., Ranjan, R., Robinson, J., ... & Leskovec, J. (2024, July). Position: Relational deep learning-graph representation learning on relational databases. In *Forty-first International Conference on Machine Learning*.

Graph Database Query

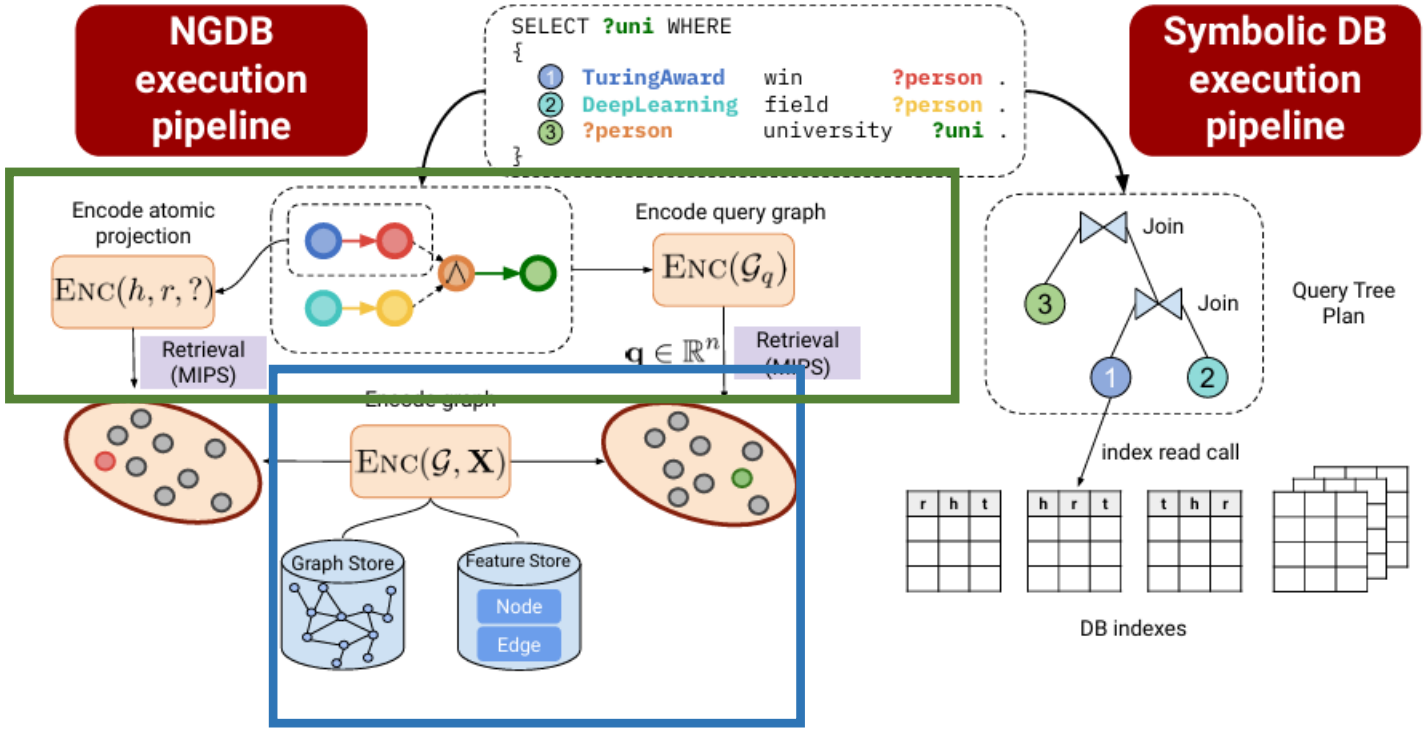


Limitation: Missing knowledge results in incomplete answer set.

Complex Graph Queries (Figure taken from Ren et al)

Ren H, Galkin M, Cochez M, Zhu Z, Leskovec J. Neural graph reasoning: Complex logical query answering meets graph databases. arXiv preprint arXiv:2303.14617. 2023 Mar 26.

Neural Graph Databases (NGDBs)



Neural Graph Databases (Figure taken from Ren et al)

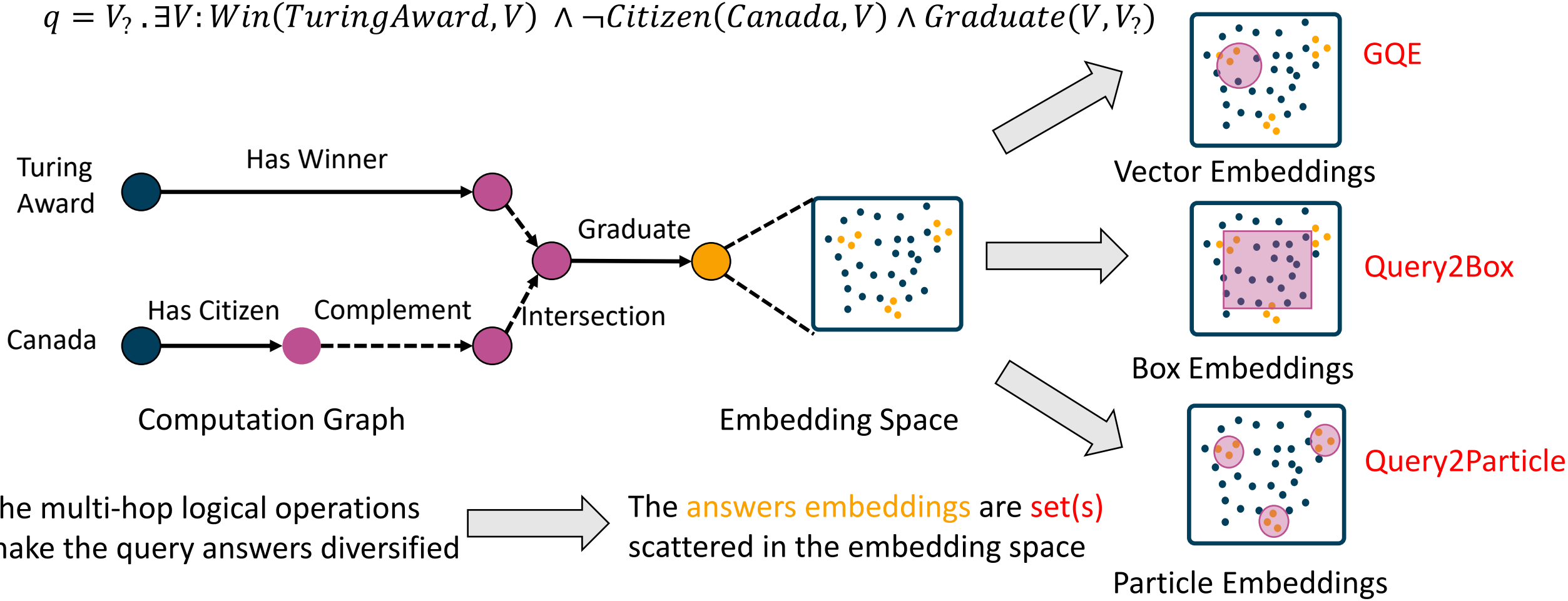
Neural Graph Storage: employ graph store and feature store to obtain latent representations in the embedding store.

Neural Query Engine: derive the computation graph of the query and execute in the latent space.

Ren H, Galkin M, Cochez M, Zhu Z, Leskovec J. Neural graph reasoning: Complex logical query answering meets graph databases. arXiv preprint arXiv:2303.14617. 2023 Mar 26.

Embedding Space and Set Representations

$$q = V_? . \exists V : Win(TuringAward, V) \wedge \neg Citizen(Canada, V) \wedge Graduate(V, V_?)$$



The multi-hop logical operations make the query answers diversified → The answers embeddings are set(s) scattered in the embedding space

William L. Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, Jure Leskovec. Embedding Logical Queries on Knowledge Graphs. NeurIPS 2018.
Hongyu Ren, Weihua Hu, Jure Leskovec. Query2box: Reasoning over Knowledge Graphs in Vector Space using Box Embeddings. ICLR 2020.
Example from: Jiaxin Bai, Zihao Wang, Hongming Zhang, Yangqiu Song: Query2Particles: Knowledge Graph Reasoning with Particle Embeddings. NAACL-HLT (Findings) 2022.

Scaling Predictive Complex Graph Queries

KnowComp
Understanding the World by Computational Knowledge

- Existing predictive graph queries are not enough
  - NGDBench
- Graph databases are usually very large; training is still not scaling
  - NGDB-ZOO
- Training free; inference stage scaling: from foundation models to in-context learning (ICL)
  - KGPFN

NGDBench: Data Type Scaling

Yufei Li, Yisen Gao, Jiaxin Bai, Jiaxuan Xiong, Haoyu Huang, Zhongwei Xie, Hong Ting Tsang, Yangqiu Song: Towards Neural Graph Data Management. CoRR abs/2603.05529 (2026)

KnowComp
Understanding the World by Computational Knowledge

- Supports full Cypher queries and dynamic data management operations

Benchmark / Dataset	Data Format	Operators	Multi-variable Query	Aggregation Support	Dynamic Updates	General Graph Query (e.g., Cypher)
Q2B (Ren et al., 2020)	Id-based Triples	EPFO (AND, OR, ∃)	No	No	No	No
LitCQD (Demir et al., 2023)	Id-based Triples & Numbers	EPFO + Numeric Operators	No	Yes	No	No
NRN (Bai et al., 2023)	Id-based Tr	Numeric Operators	No	No	No	No
EFO_1-CQA (Wang et al.) 2021	I	AND, OR, NOT ∃)	No	No	No	No
EFO_k-CQA (Yin et al., 2025)	Id-based Tr	AND, OR, NOT ∃)	Yes	No	No	No
NGDBench (Ours)	**Ids, Numbers, and Strings**	**Cypher Operators**	**Yes**	**Yes**	**Yes**	**Yes**

From KnowComp

Category		NGD-BI	NGD-Fin	NGD-Prime	NGD-MCP	NGD-Econ	Summary
Graph Structures (Groundtruth)	#Nodes	2,997,352	10,865	129,312	181,689	33,164	–
	#Edges	17,196,776	57,818	8,100,498	351,263	63,757	–
Graph Structures (Perturbed)	#Nodes	2,997,352	10,865	129,312	181,689	33,164	–
	#Edges	17,971,365	60,476	8,465,124	351,263	63,757	–
Query Categories	#Analytical (No Agg)	17,880	4,264	14,043	8,336	1,707	58,321
	#Analytical (Agg.)	9,580	4,412	6,494	4,167	1,050	25,703
	#Management	2,503	1,400	2,207	–	–	6,110

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings, ICLR 2020
Caglar Demir, Michel Wiebesiek, Renzhong Lu, Axel- Cyrille Ngonga Ngomo, and Stefan Heindorf. 2023. LitCQD: Multi-hop reasoning in incomplete knowledge graphs with numeric literals. ECML PKDD 2023
Jiaxin Bai, Chen Luo, Zheng Li, Qingyu Yin, Bing Yin, and Yangqiu Song. 2023. Knowledge graph reasoning over entities and numerical values, KDD 2023
ZihaoWang, Hang Yin, and Yangqiu Song. Benchmarking the combinatorial generalizability of complex query answering on knowledge graphs, NeurIPS Benchmarks Track (Round 2), 2021
Hang Yin, Zihao Wang, Weizhi Fei, and Yangqiu Song. EfOk-CQA: Towards knowledge graph complex query answering beyond set operation. KDD 2025.

NGDBs Training Scaling

Low training efficiency

Expressivity are constrained by structure-exclusive embeddings

NGDB-Zoo: Towards Efficient and Scalable Neural Graph Databases Training

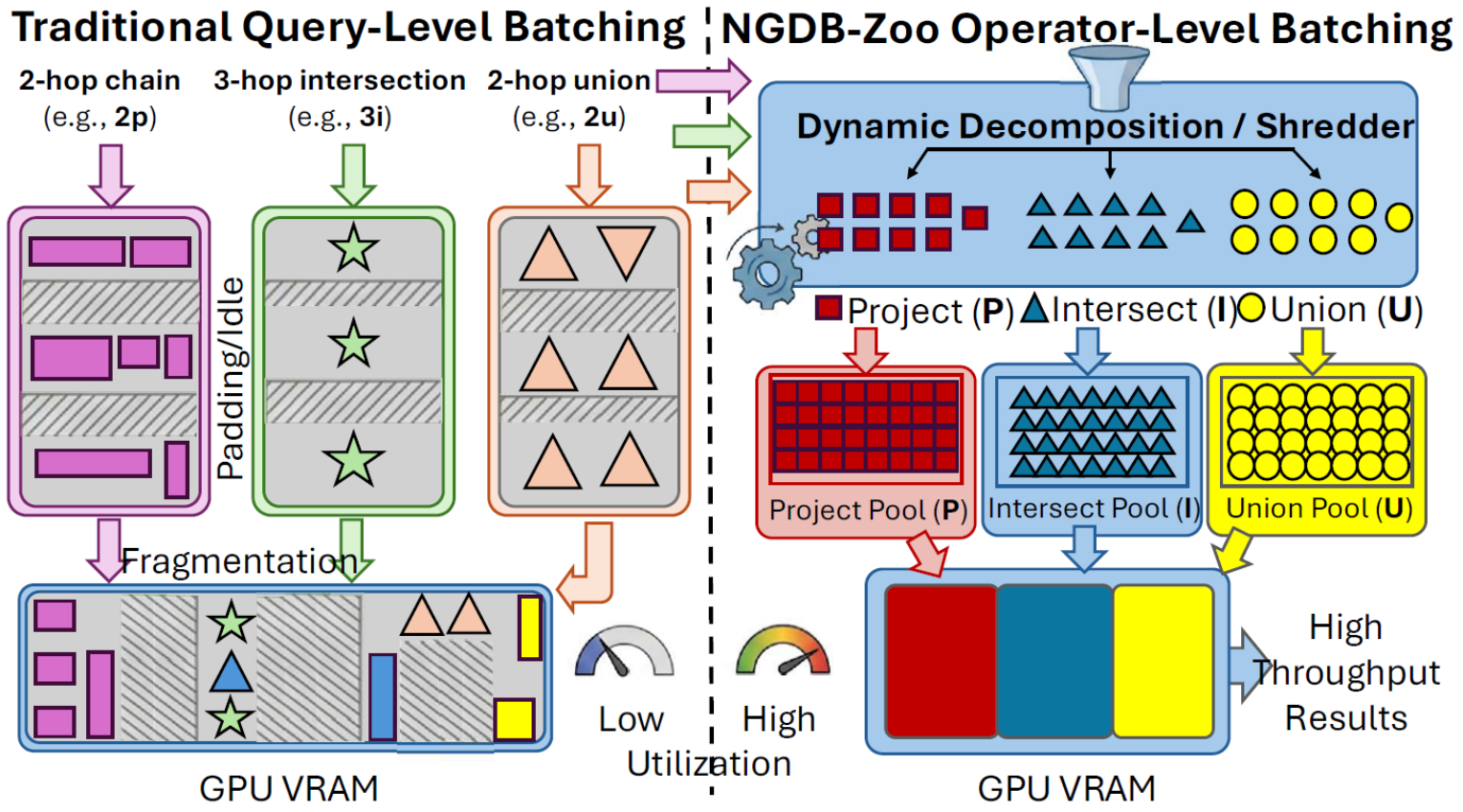
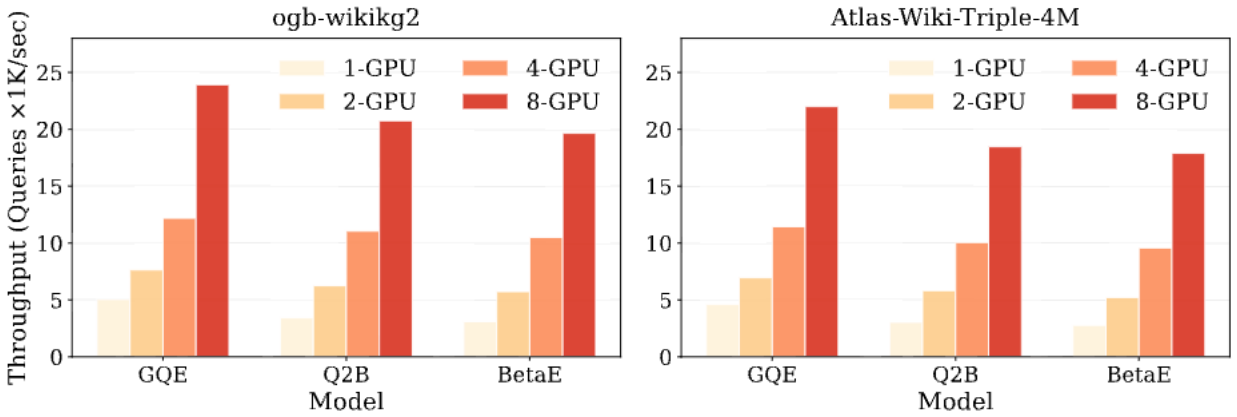


Figure 3. **Overcoming Topological Rigidity. (Left):** Query-level batching incurs high fragmentation overheads by segregating distinct query structures. **(Right):** NGDB-Zoo aggregates atomic operators into homogeneous Operator Pools for unified execution, eliminating structural constraints and saturating GPU cores.

Scalability and Predictive Performance on Massive Knowledge Graphs



The unit of throughput is 10^3 queries/sec.

Dataset	Model	MRR (%)	TPut (↑)	Mem (GB ↓)
FB400k	GQE	35.84	**24.68**	**7.5**
	Q2B	**52.33**	21.55	11.0
	BetaE	50.40	19.97	14.0
ogbl-wikikg2	GQE	32.88	**23.92**	**8.0**
	Q2B	42.01	20.75	11.0
	BetaE	**44.54**	19.65	14.0
ATLAS-Wiki -Triple-4M	GQE	7.31	**22.00**	**10.0**
	Q2B	**9.22**	18.47	12.0
	BetaE	9.01	17.89	15.0

Dataset	Entities	Relations	Train Edges	Valid Edges	Test Edges	Total Edges
FB15k	14,951	1,345	483,142	50,000	59,071	592,213
FB15k-237	14,505	237	272,115	17,526	20,438	310,079
NELL995	63,361	200	114,213	14,324	14,267	142,804
FB400k	409,829	918	1,075,837	537,917	537,917	2,151,671
ogbl-wikikg2	2,500,604	535	16,109,182	429,456	598,543	17,137,181
ATLAS-Wiki-Triple-4M	4,035,238	512,064	23,040,868	2,880,108	2,880,110	28,801,086

KG statistics with number of entities, relations, training, validation and test edges.

NGDB-Zoo enables high throughput and robust reasoning across diverse paradigms on production-scale graphs with millions of entities.

Revisit Knowledge Graph Foundation Model

- Knowledge Graph Foundation Model (**KGFM**) need to be able to generalize on new graphs where the entities and relationships are completely different.

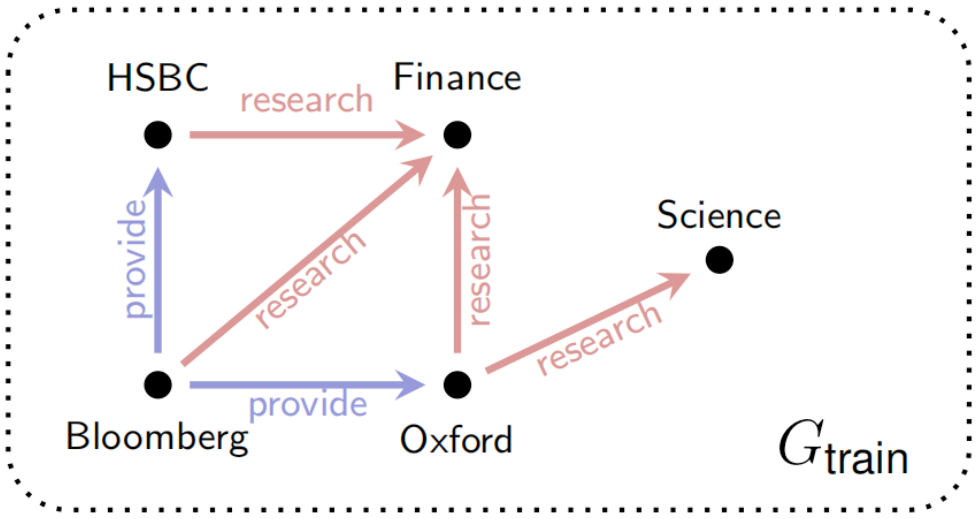


Figure1. Train graph

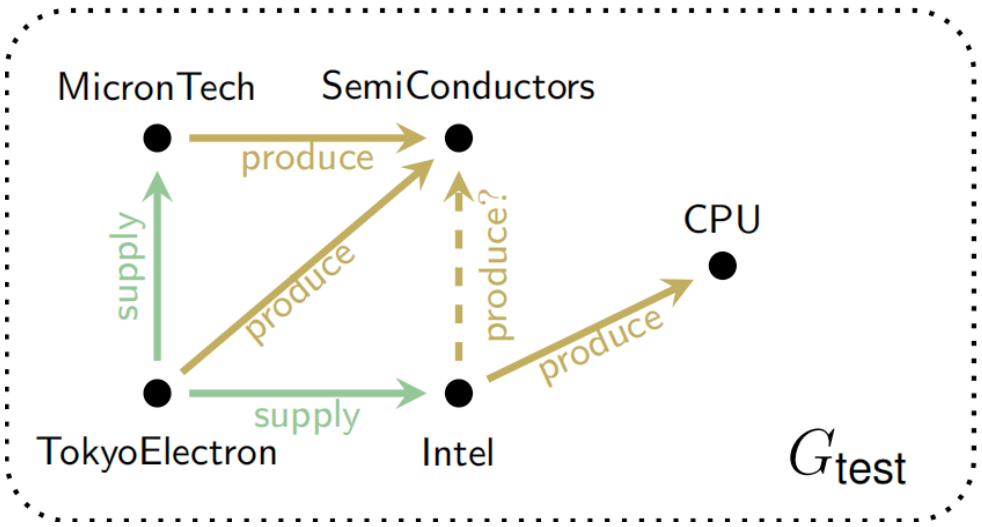


Figure2. Test graph

Figure Ref: ICML 2025

Towards Foundation Models For Knowledge Graph Reasoning. ICLR 2024
How Expressive are Knowledge Graph Foundation Models? ICML 2025

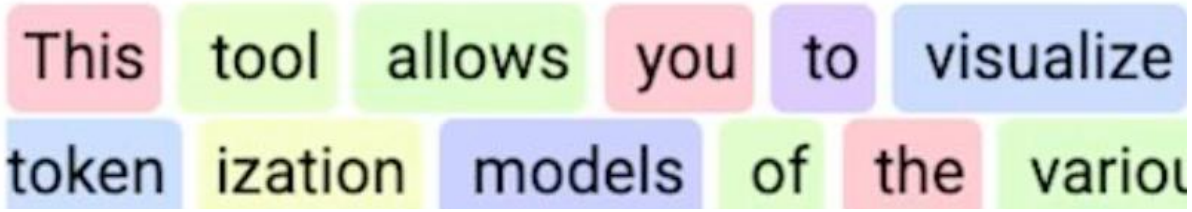
Invariance of the relational structure facilitates the transition to new relations——capturing the 'relation' of relation.

Rethink the Foundation Model

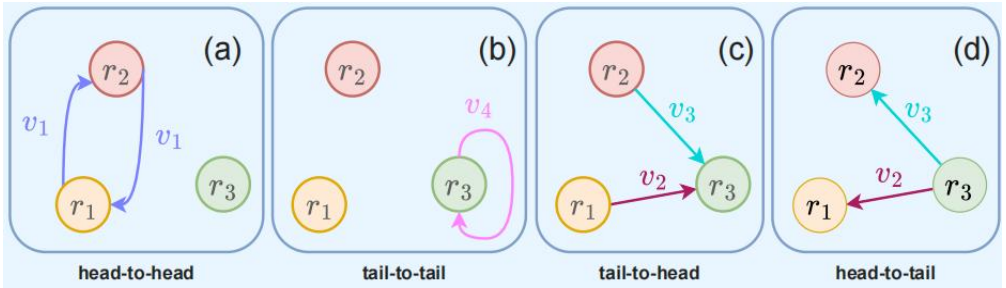
LLM vs KGFM

General invariant basic unit

Token

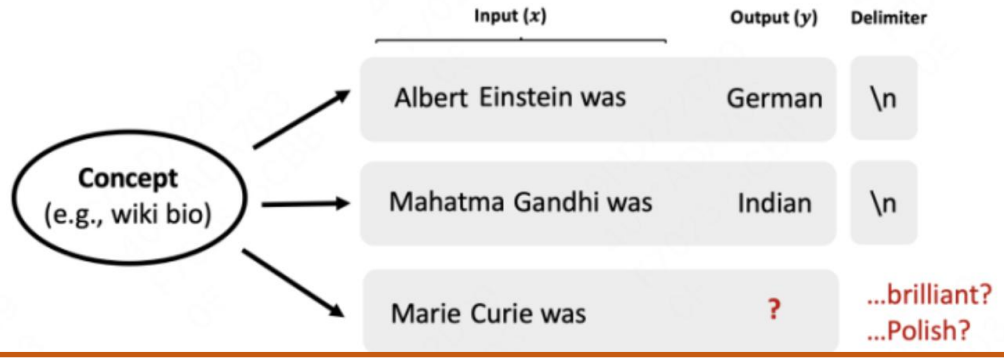


Relation Motif



Incontext Learning

Prompt



Missing

?

In-context Learning from a Bayesian View

- ICL in LLM: Extract the shared concept from the prompt and conduct Bayesian inference.

- Posterior predictive distribution: a prior specifies a hypothesis space Φ that maps inputs x to labels y in sampled dataset D.

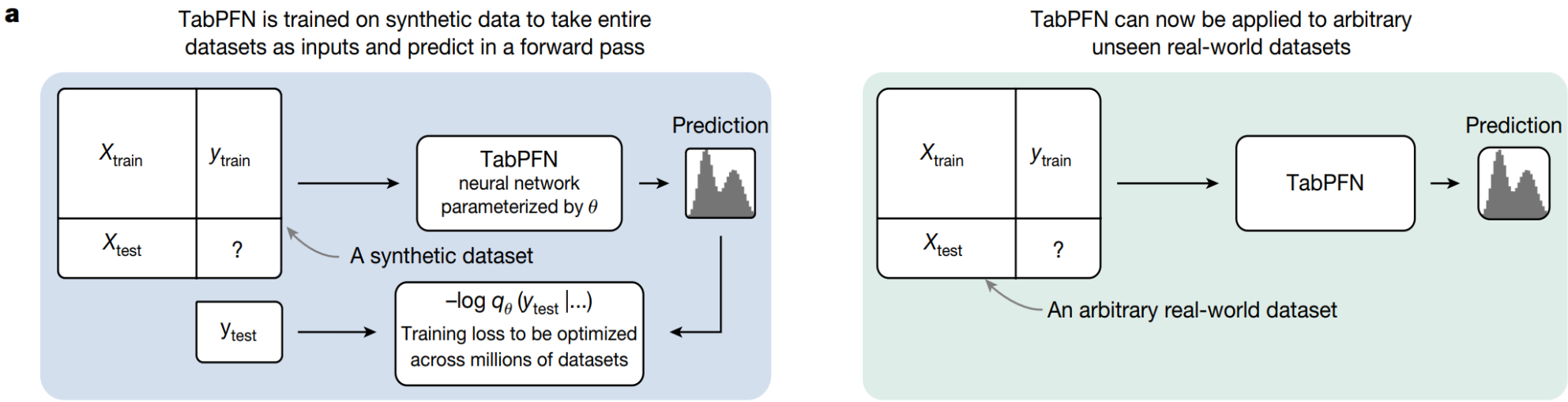
Prior-data Fitted Networks (PFNs)

- Prior-data fitted networks(PFN) : cast in-context learning as amortized Bayesian inference:

  a single parametric model is trained to output the posterior predictive distribution conditioned on a set of labeled context examples.

$$p(y|x, \mathcal{D}) = \int_t p(y|x, t)p(t|\mathcal{D}).$$

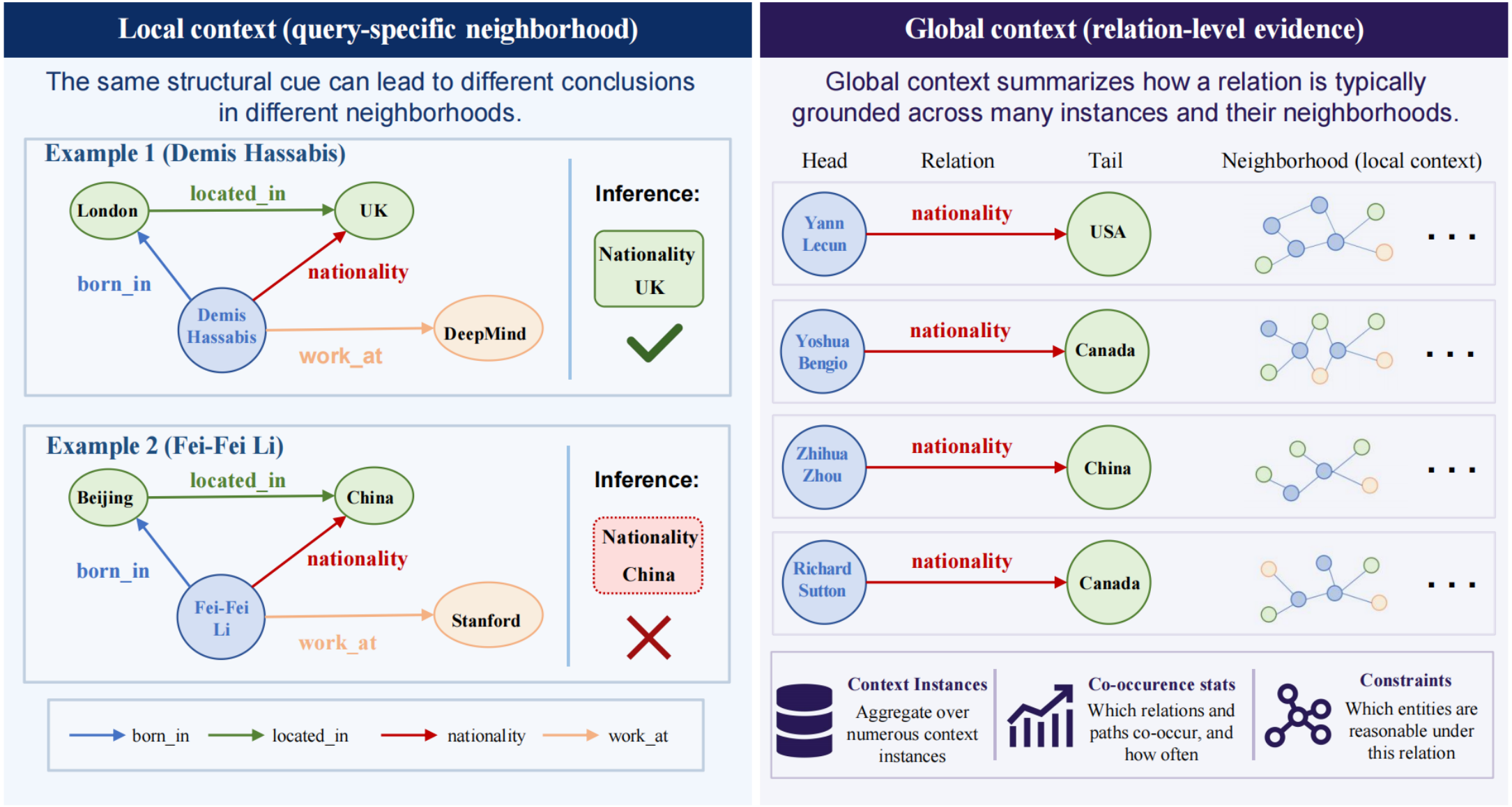
- PFN has become the cornerstone of structured data incontext learning, including tables, relational databases and time series data.



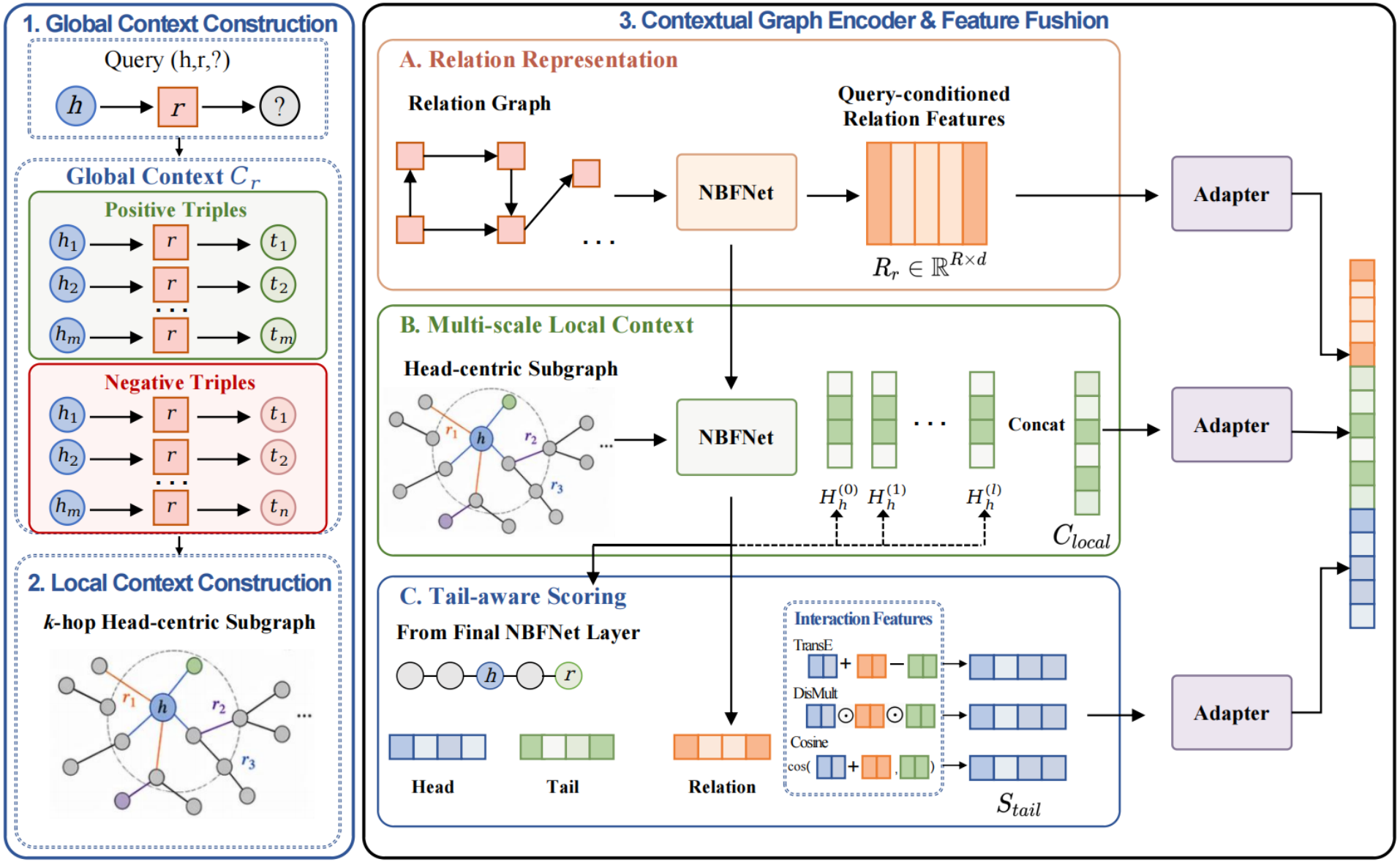
Accurate predictions on small data with a tabular foundation model. Nature, 2024

ICL for Knowledge Graph Foundation Model

- Context in KG: local context and global context



KGPFN: In Context Learning KGFM+PFN



Experiment: Test/Inference Time Scaling

KnowComp
Understanding the World by Computational Knowledge

- Link Prediction Performance on 57 KGs

KGPFN can defeat the fine-tuned KGFMs with in-context learning.

Table 1: Average MRR and Hits@10 over 57 KGs. (**Bold**: best; Underline: runner-up.)

Setting	Model	Inductive e, r (23 graphs)		Inductive e (18 graphs)		Transductive (16 graphs)		Total Avg (57 graphs)		Pretraining (3 graphs)	
		MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10
Zero-shot	ULTRA	0.345	0.513	0.431	0.566	0.329	0.479	0.367	0.520	–	–
	KGICL	0.362	0.544	0.440	0.584	0.346	0.493	0.382	0.542	–	–
	Motif	0.337	0.509	0.431	0.570	0.335	0.492	0.370	0.527	–	–
	TRIX	0.368	0.540	0.455	0.592	0.342	0.522	0.388	0.551	–	–
Fine-tuned	ULTRA	0.397	0.556	0.442	0.582	0.384	0.548	0.407	0.562	0.407	0.568
	KGICL	0.426	0.616	0.464	0.630	**0.397**	0.554	0.430	0.603	–	–
	Motif	0.401	0.558	0.455	0.594	0.390	0.549	0.415	0.569	0.415	0.565
	TRIX	0.401	0.556	0.459	0.594	0.390	0.558	0.418	0.569	0.415	0.563
In-context	KGPFN	**0.433**	**0.639**	**0.465**	**0.638**	0.393	**0.600**	**0.432**	**0.628**	**0.423**	**0.623**

Yisen Gao, Jiaxin Bai, Haoyu Huang, Zhongwei Xie, Yufei Li, Hong Ting Tsang, Sirui Han, Yangqiu Song: KGPFN: Unlocking the Potential of Knowledge Graph Foundation Model via In-Context Learning. (2026)

Conclusions

- We build foundation knowledge graphs automatically extracted or refined from Web-scale documents
  - All entities, events, relations, and schemas are automatic
- We build neuralized knowledge foundation model that can efficiently trained over large-scale graph databases
- We extend the foundation model into in-context learning, which is one more big step towards foundation model real use

Thank you for your attention ☺