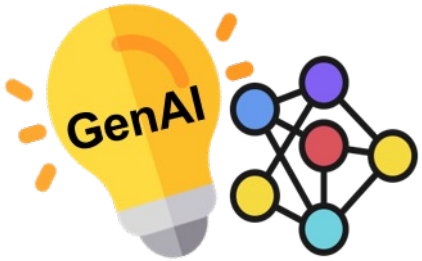
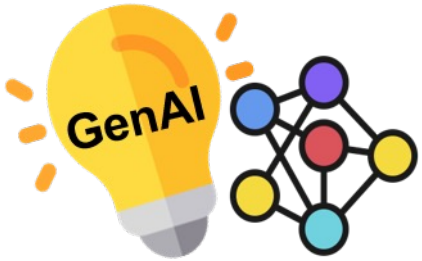




# *Tutorial 4*
# LLMs+Graphs: Toward Graph-Native, Synergistic AI Systems

Arijit Khan, Longxu Sun, Xin Huang

Bowling Green State University, USA
Hong Kong Baptist University, Hong Kong

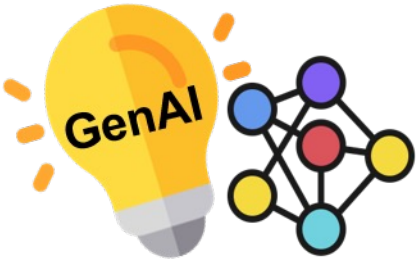

# Outlines




1. Introduction (Xin)
2. LLMs for Graphs (Xin)
3. Graphs for LLMs (Longxu)
4. KGs for LLMs (Arijit)
5. LLMs for KGs (Longxu)
6. Graphs for AI Agents (Arijit)
7. AI Agents for Graphs (Longxu)
8. Future Directions (Arijit)

***Disclaimer:*** *Some slides contain adapted material, and our discussion of related work is selective rather than comprehensive.*

# 1. Introduction




- LLMs
- AI Agents
- Graph Foundation Models
- Retrieval-Augmented Generation
- Knowledge Graphs

Presented by Xin Huang

# Large Language Models

- Large Language Models are transformer-based neural networks trained on massive text corpora to learn patterns, structure, and meaning in language.
- Large language models (LLMs) showcase strong generalization abilities to handle various NLP tasks, such as question answering and machine translation.

OpenAI

GPT - 4

deepseek

**Key Capabilities**

**Text Generation**

**Code Generation**

**Question Answering**

**Translation**

**Reasoning**

**Summarization**


**Scale matters:** modern LLMs often contain billions of parameters and are trained on trillions of tokens.

**Notable Examples**

Meta AI

Gemini

Copilot

ChatGPT

https://blog.dataiku.com/large-language-model-chatgpt

# The LLM Family Tree

From encoders to aligned multimodal systems — the progression reflects increasing scale, adaptability, and task generality.

1

**2018**

**BERT**

Bidirectional encoder, masked language modeling

2

**2019–2020**

**GPT-2/3**

Autoregressive generation at scale

3

**2022**

**InstructGPT / ChatGPT**

RLHF alignment

4

**2025–2026**

**GPT-4o / o3, Gemini 2.5, Claude 3.7, LLaMA 4, DeepSeek-V3**

Reasoning-first models, long-context, agentic capabilities, and open-weight frontier models

# The Power & Limits of LLMs

### Strengths

Semantic understanding, zero-shot generalization, natural language interfaces for data-intensive applications *(Fernandez et al., VLDB 2023; Halevy et al., VLDB 2023)*

---

### Core Challenge

Limitations in **structured reasoning** and **multi-hop logic**

### Hallucination Problem

Lack of grounding in verifiable facts or relational context — LLMs are mainly designed to process pure text, yet many real-world scenarios involve rich graph-structured information *(Jin et al., TKDE 2024)*

> "While LLMs have shown pure text-based reasoning ability, it is underexplored whether such ability can be generalized to graphs" *(Jin et al., TKDE 2024)*

# AI Agents

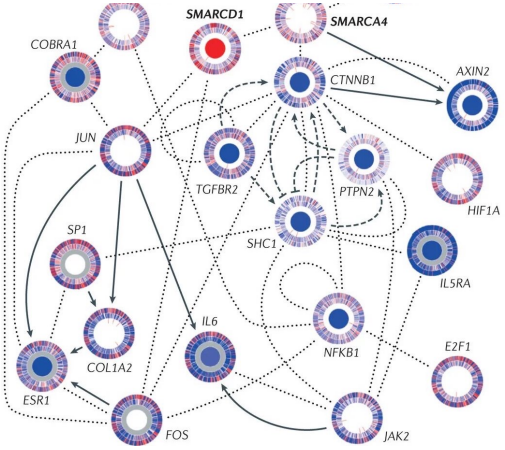
- **AI Agent** is an autonomous software entity that perceives its environment, makes decisions, and acts to achieve specific goals.

- AI agents can interact with users, other agents, or systems.

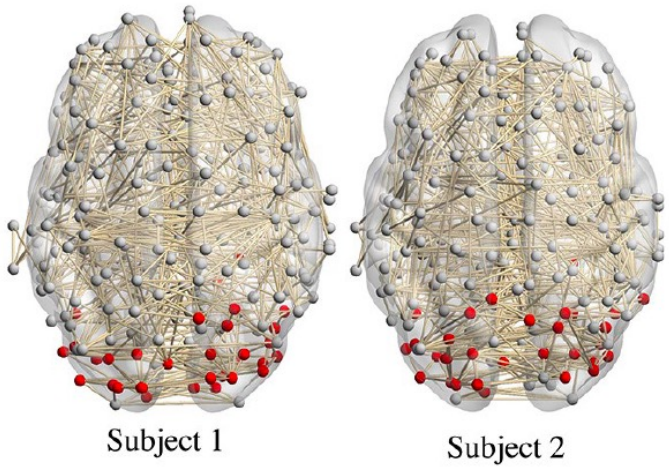
- Examples:
  - Virtual Personal Assistants: Siri, Alexa, Google Assistant, Open Claw
  - Autonomous Vehicles: Waymo, Tesla's Autopilot
  - Game Non-Player Characters (NPCs): Chess AI (Stockfish)

# Graph

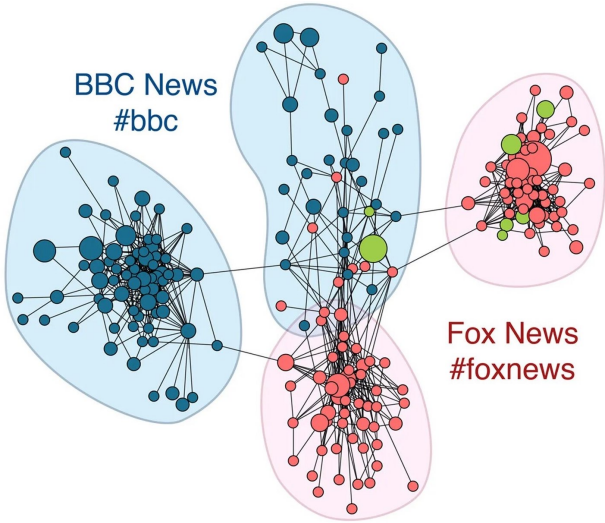
- **Graph** G(V, E) is a mathematical model to describe various entities V and their relations E.
- Graph can be explained as a general language to describe entities' connections and interactions in nature and society.




Protein-Protein Network
[Nature 2013]




Brain Network
[Luo et al. KDD 2020]



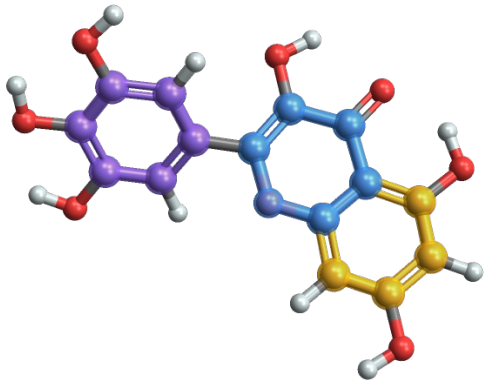

Follower Network
[Nature 2013]

# Text-attributed Graphs

- Text-attributed graphs with raw and detailed text attributes, capturing rich and comprehensive semantics.


Citation networks

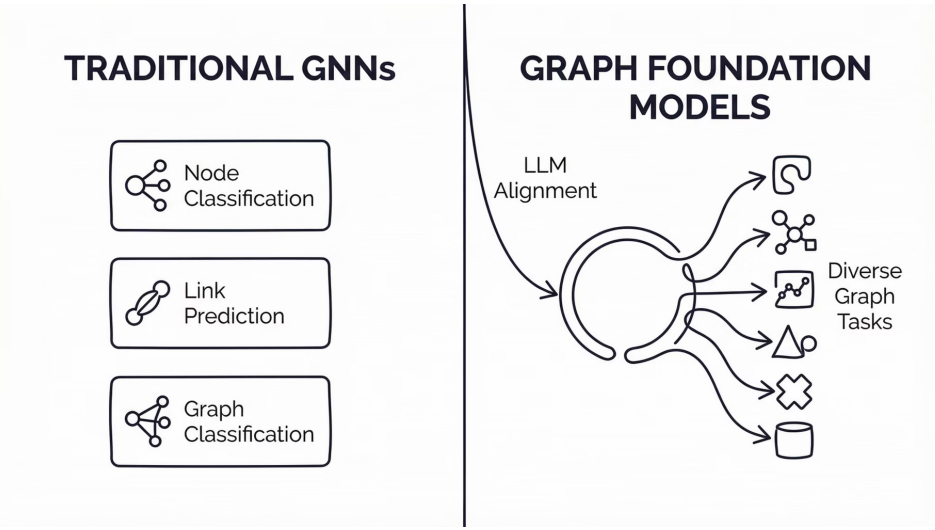
Social networks

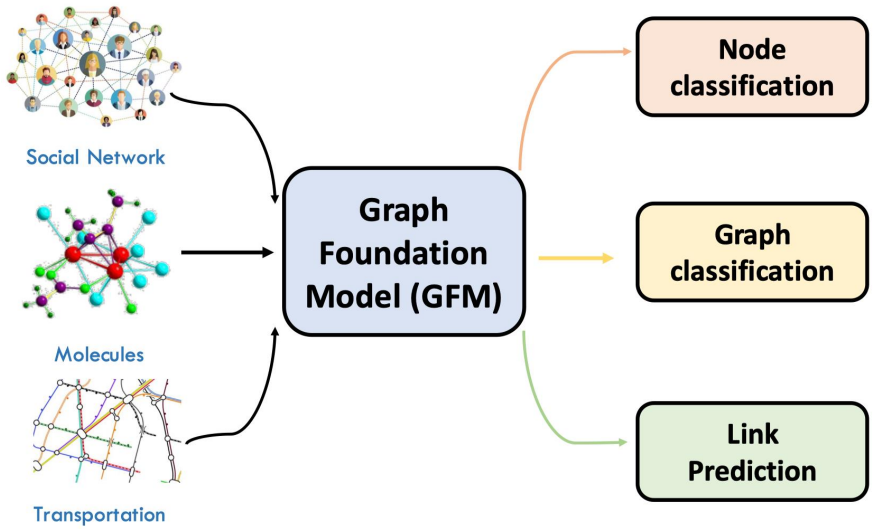


Molecular graphs

# Graph Foundation Models

- A foundation model is a model trained on broad data that can be adapted to a wide range of downstream tasks

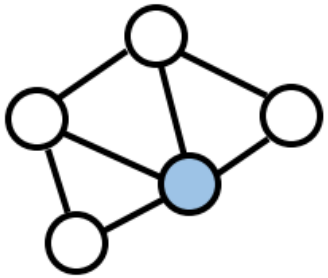



Mao, H., et al. "Position: Graph Foundation Models Are Already Here." ICML. (2024)

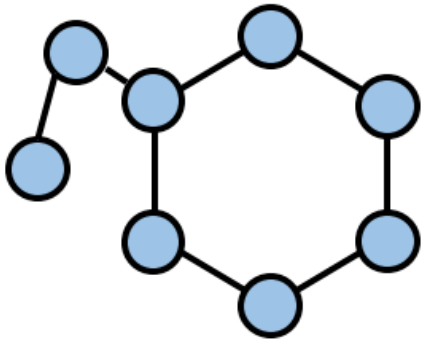
# Goal: Train one graph foundation model for all classification tasks

Node Classification



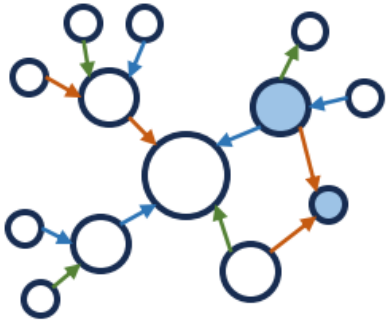
What's the category of this **node**?
1. … 2. … 3. …

Graph Classification



Does this **molecular graph** inhibit HIV?
Yes or No

Link Prediction

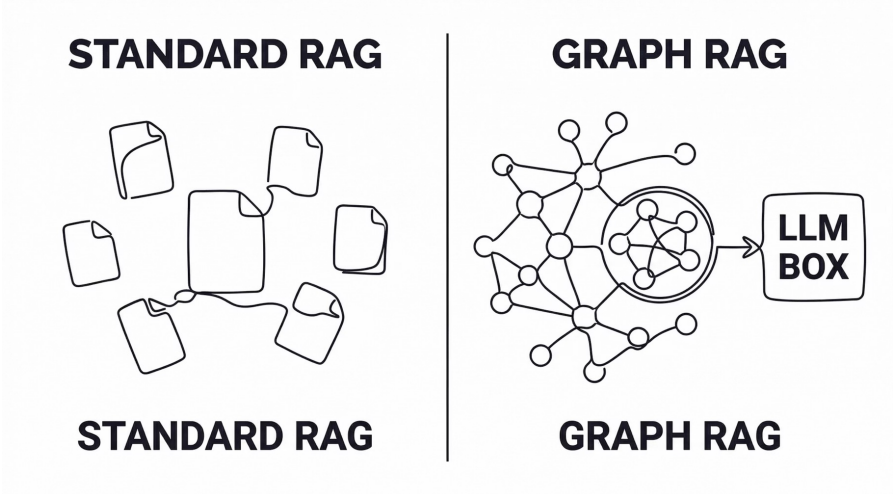


What's the relationship between **these two nodes?**
1. … 2. … 3. …

# Retrieval-Augmented Generation

- RAGs can enhance the capabilities of LLMs by integrating them with search or retrieval systems.
  - Access up-to-date knowledge and external data source

- Examples:
  - Microsoft Bing Chat
  - GitHub Copliot
  - Medical Question Answering



# Microsoft GraphRAG

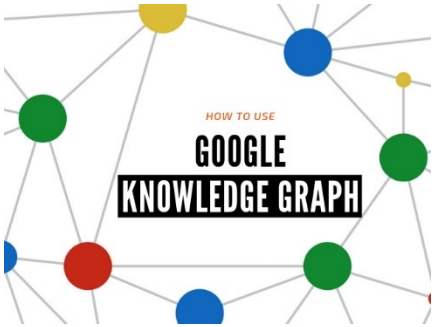
- Key idea:  convert documents into a knowledge graph and then retrieve the relevant document  communities
- Phases:
  - Builds entity-relation graph from documents
  - Detects communities via Leiden algorithm
  - Generates community-level summaries for global QA

# Knowledge Graphs

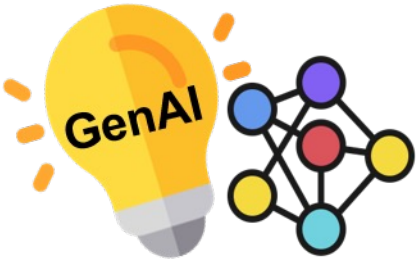
- A knowledge graph keeps a structured network of entities (such as people, places, or things) and their fact/relationships.

- Knowledge graphs enable LLMs, AI Agents, and machines to understand and utilize the complex interconnected information.

- Examples:
  - Google Knowledge Graph
  - Wikidata
  - Amazon Product Graph
  - Biomedical Graphs for Drug Discovery



# 2. LLMs for Graphs



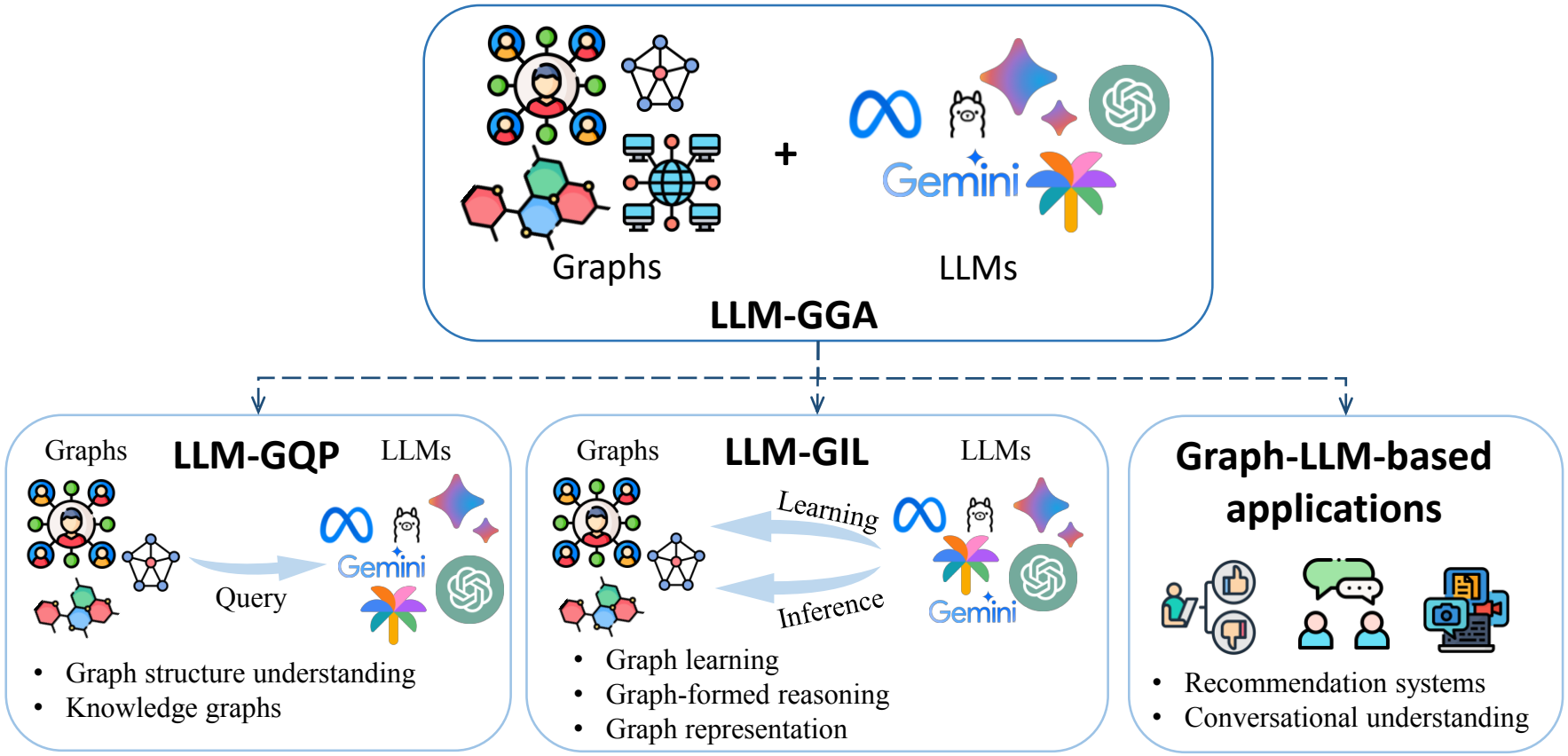

A. Graph Understanding
   - Graph Computation

B. Graph Learning
   - LLMs-as-generators
   - Unsupervised Graph Representation
   - Graph Diffusion Prediction

C. Graph Foundational Models

Presented by Xin Huang

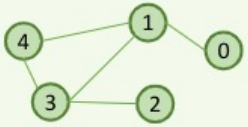
# A Survey on LLM-based Generative Graph Analytics




Wenbo Shang, Xin Huang. A survey of large language models on generative graph analytics: Query, learning, and applications. IEEE TKDE 2025.

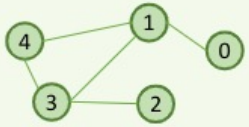
# Graph Structure Understanding

**(a) Graph Size Calculation**



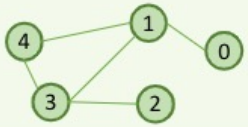

Given <graph>, what is the number of nodes and edges in this graph? Please answer with the number of nodes: X, number of edges: X.

**(b) Degree Calculation**



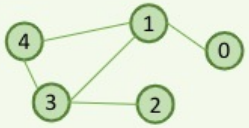

Given <graph>, what is the degree of node 4? Or, like, find the node degree of node [given node] in the given graph.

**(c) Clustering coefficient computing**



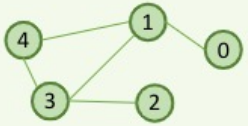

Given <graph>, what is the clustering coefficient of the node 1?

**(d) Path Search**




Given <graph>. **Simple path:** Find a single path from node 0 to node 4 connected by edges in the given graph. **Shortest path:** Give the shortest path from node 0 to node 4.

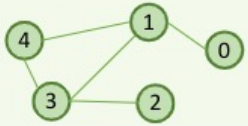
**(e) Attribute Retrieval**




**Abstract:** Text in curve orientation, despite being one of the common... **Title:** Total Text A Comprehensive Dataset For Scene Text Detection And Recognition.

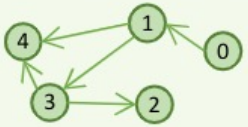
Given <graph>, what is the title of node 0?

**(f) Graph Diameter**



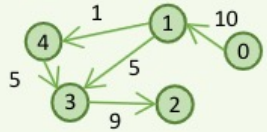

Given <graph>, what is the diameter of the given graph?

**(j) Topological Sorting**



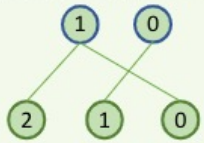

In a directed graph with 5 nodes numbered from 0 to 4: node 0 should be visited before node 1, ... Q: Can all the nodes be visited? Give the solution.

**(h) Maximum Flow**




In a directed graph with 5 nodes numbered from 0 to 4, and the edges are: an edge from node 0 to node 1 with capacity 10... Q: What is the maximum flow from node 0 to node 3?

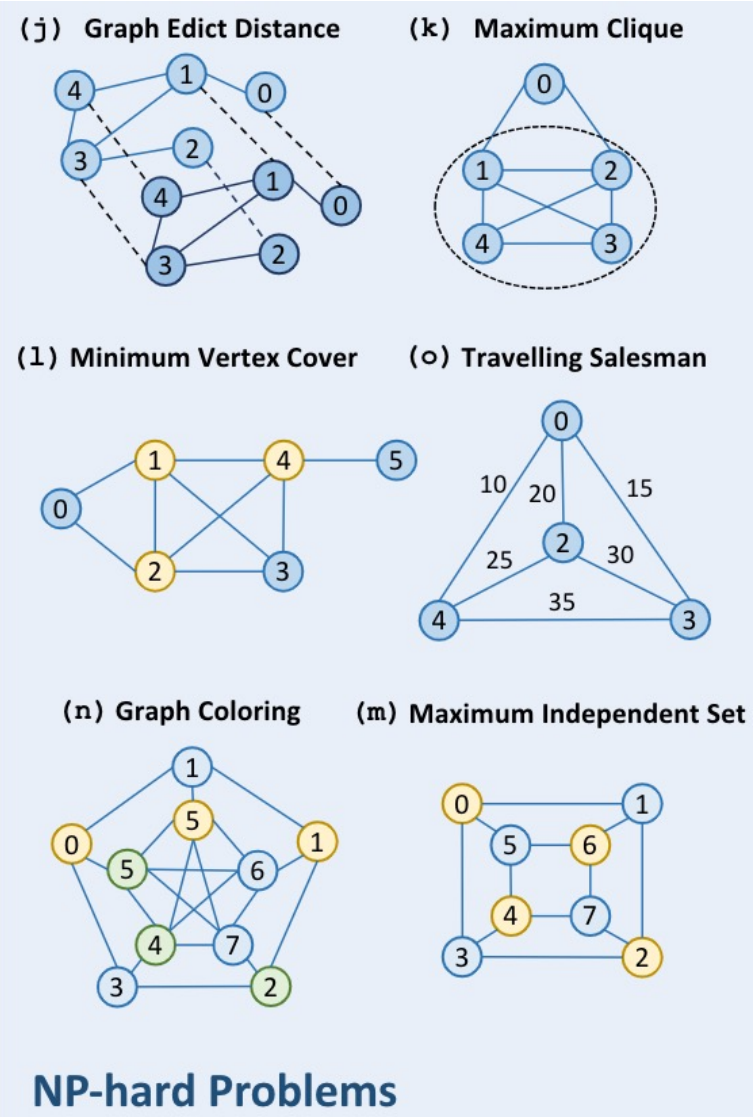
**(i) Bipartite Graph Matching**



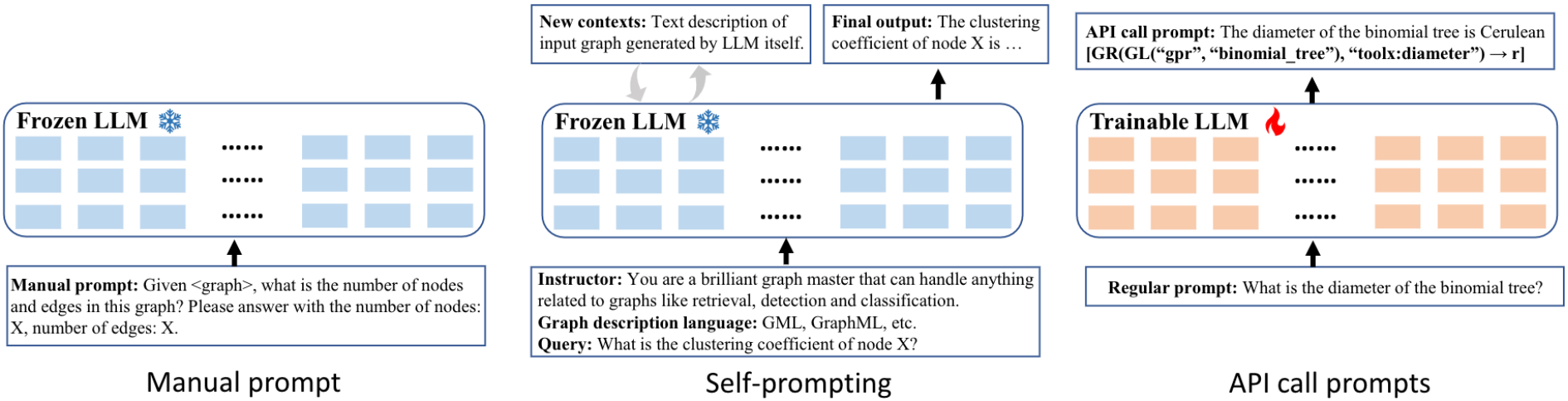

There are 2 job applicants numbered from 0 to 1, and 3 jobs numbered from 0 to 2. Each applicant is interested in some of the jobs. Each job can only accept one applicant and a job applicant can be appointed for only one job. Applicant 0 is interested in job 1, ... Q: Find an assignment of jobs to applicants in such that the maximum number of applicants find the job they are interested in.

## Polynomial-time Problems

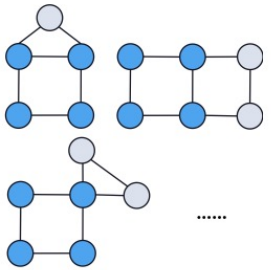
# Graph Structure Understanding

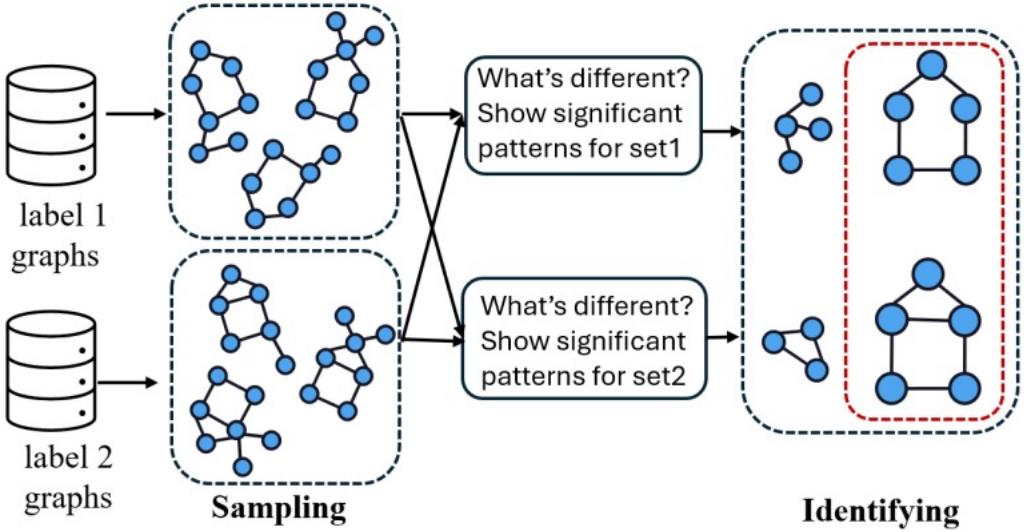


# Graph Structure Understanding Methods



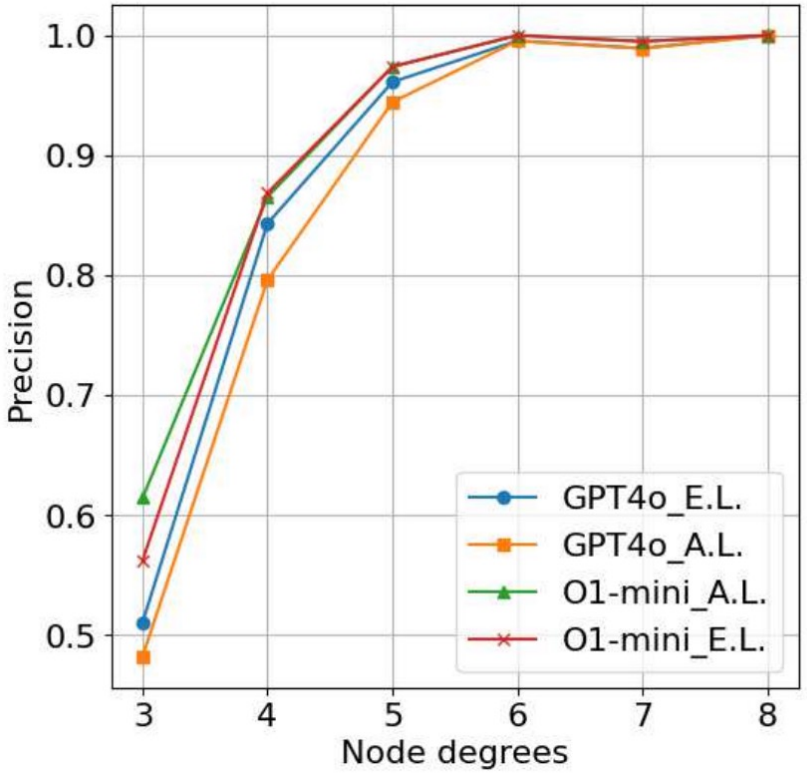
# Graph Pattern Comprehension

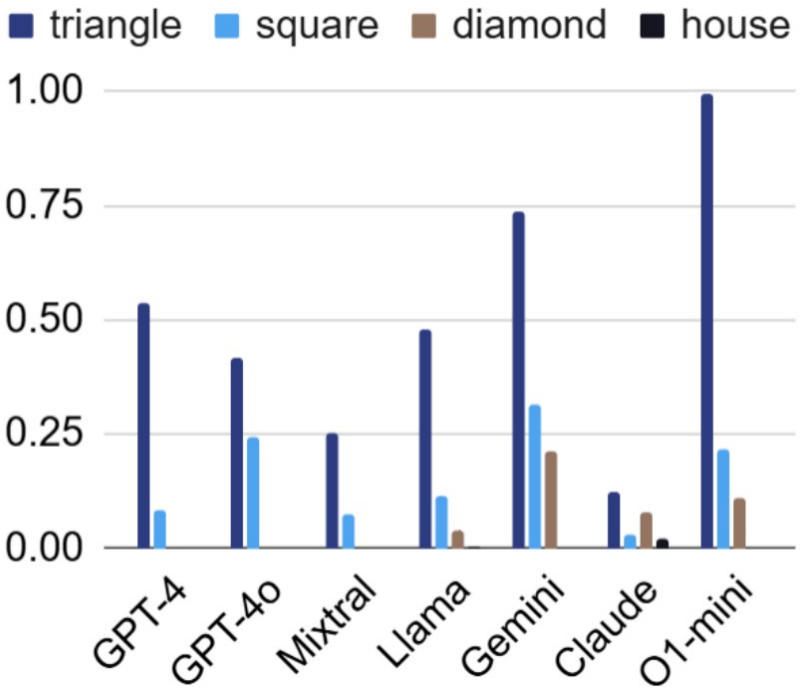
| Terminology-based | Topology-based | Data-driven |
|---|---|---|
| The pattern, named Square, is a 4-node cycle with each node connected to exactly two others. | The pattern, defined as G, is an undirected graph with four Node 0, 1, 2, 3. Node 0 is connected to Nodes 1 and 2. Node 1 is connected to Nodes 2 and 3. |  |




[1] How Do Large Language Models Understand Graph Patterns? A Benchmark for Graph Pattern Comprehension [ICLR'25]

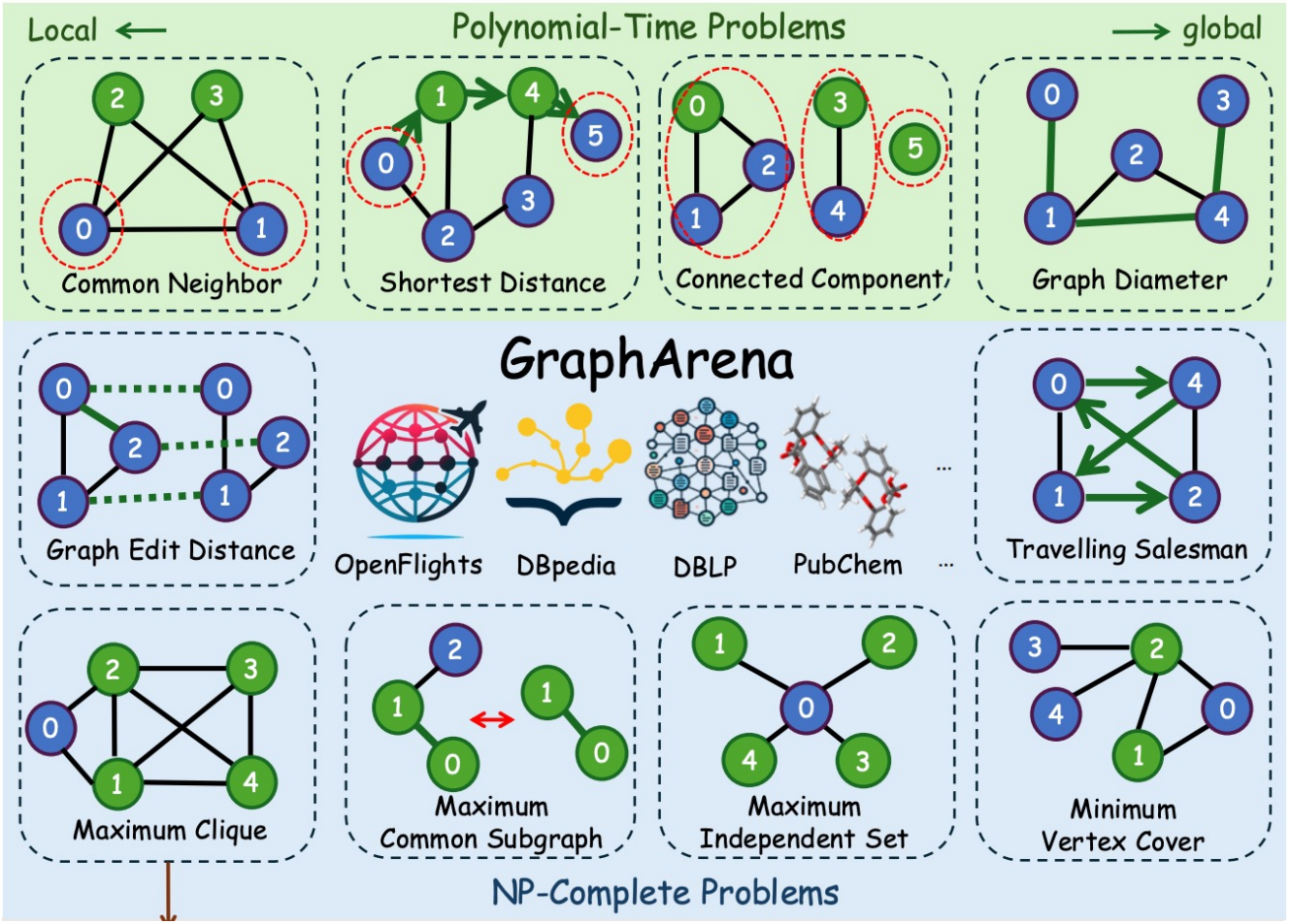
# Graph Pattern Comprehension



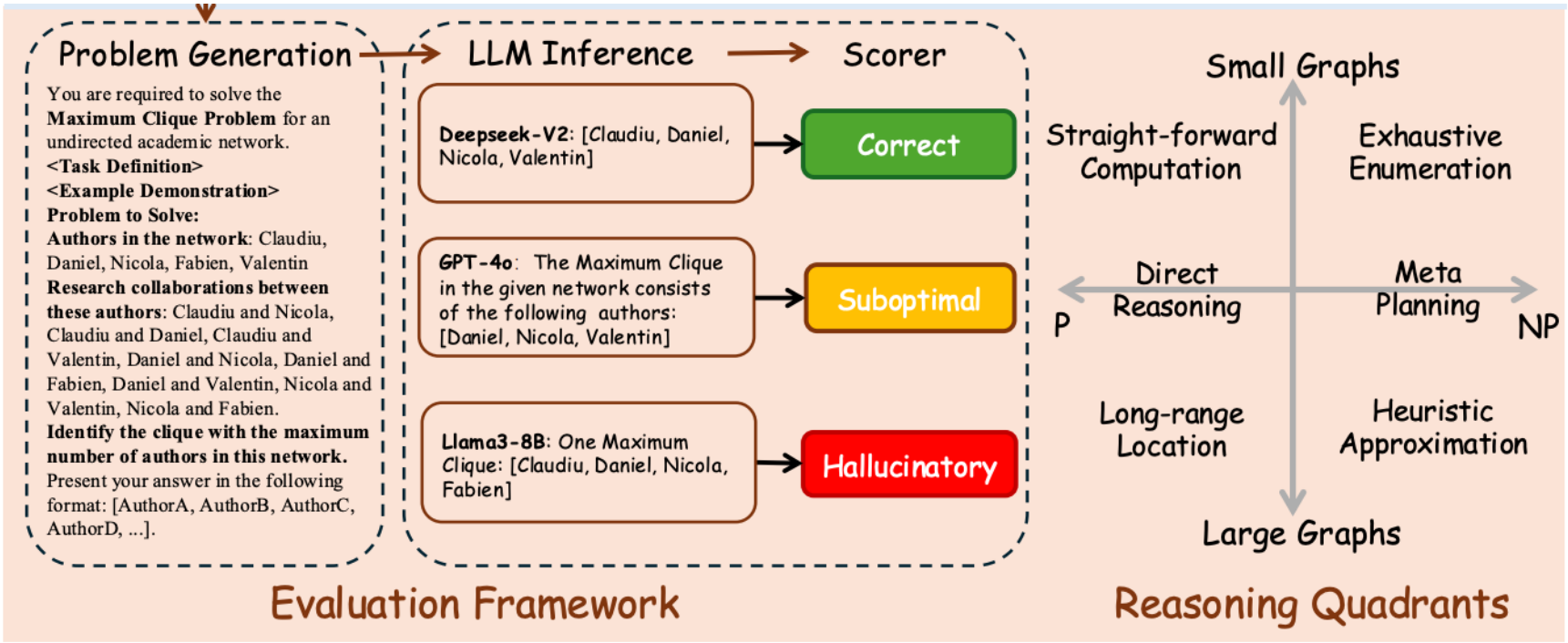

[1] How Do Large Language Models Understand Graph Patterns? A Benchmark for Graph Pattern Comprehension [ICLR'25]

# LLM on Graph Computational Problems



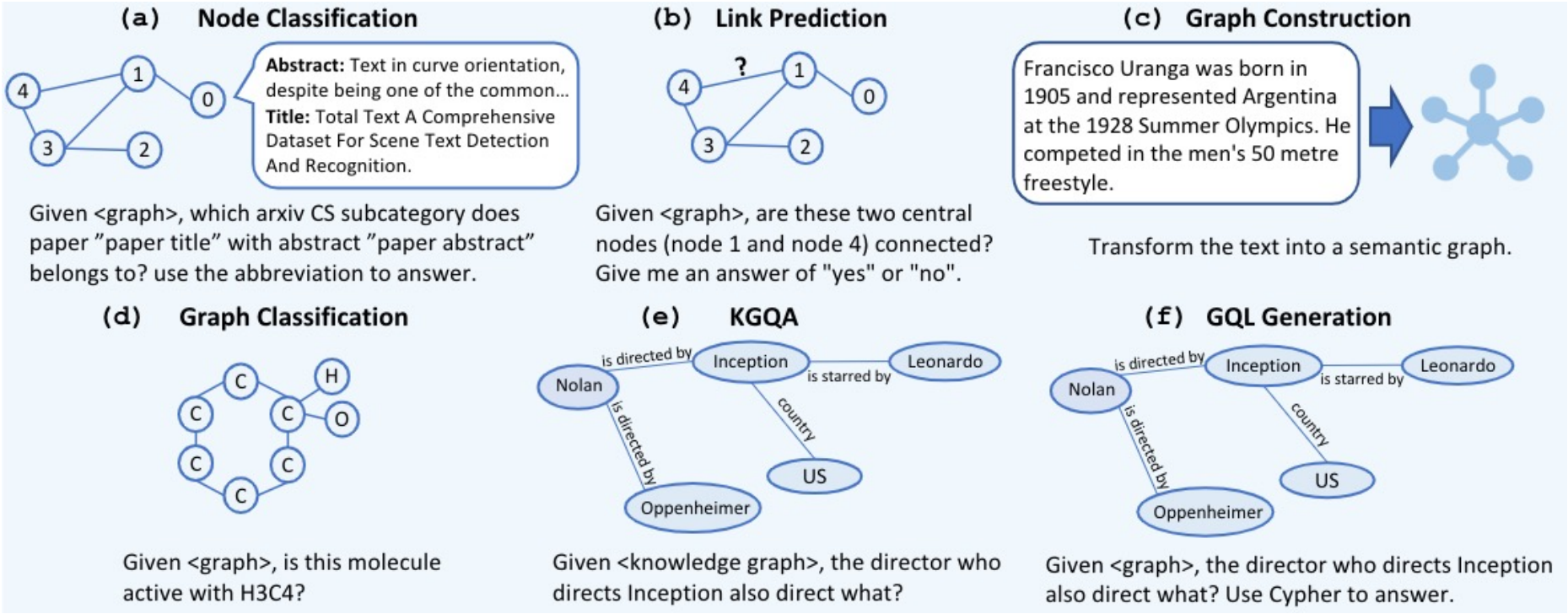

[1] GraphArena: Evaluating and Exploring Large Language Models on Graph Computation [ICLR'25]

# LLM on Graph Computational Problems



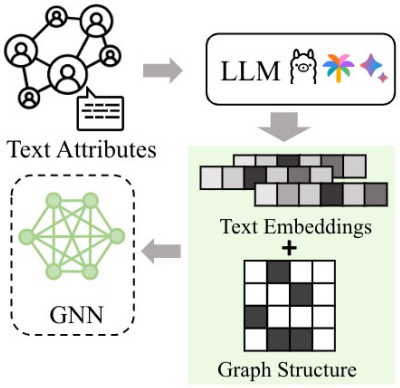

[1] GraphArena: Evaluating and Exploring Large Language Models on Graph Computation [ICLR'25]

# Graph Learning

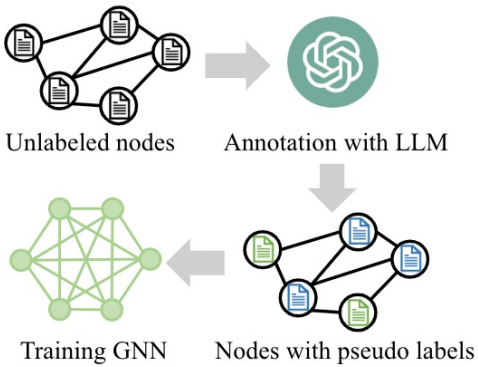


# Graph Learning

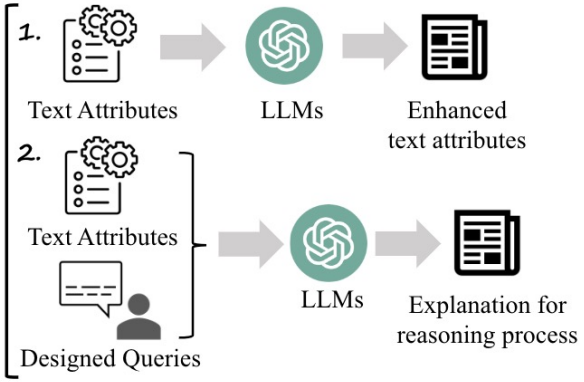
- **LLMs-as-enhancers**




(a) encoding text attributes into embeddings

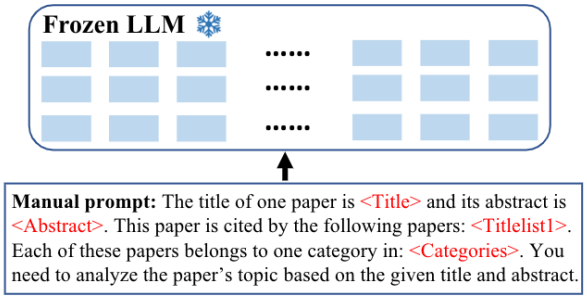



(b) generating graph pseudo labels

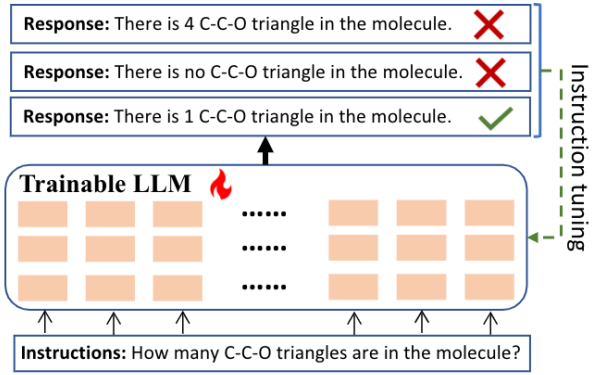



(c) providing external knowledge/explanations

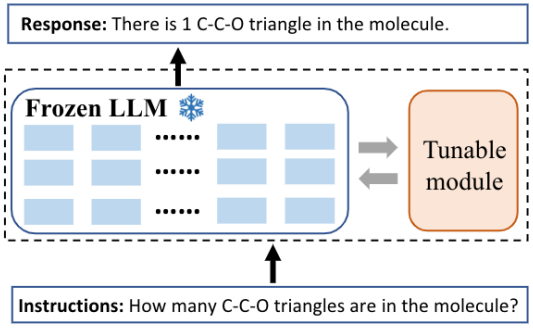
- **LLMs-as-predictors**




(a) Prompting LLMs



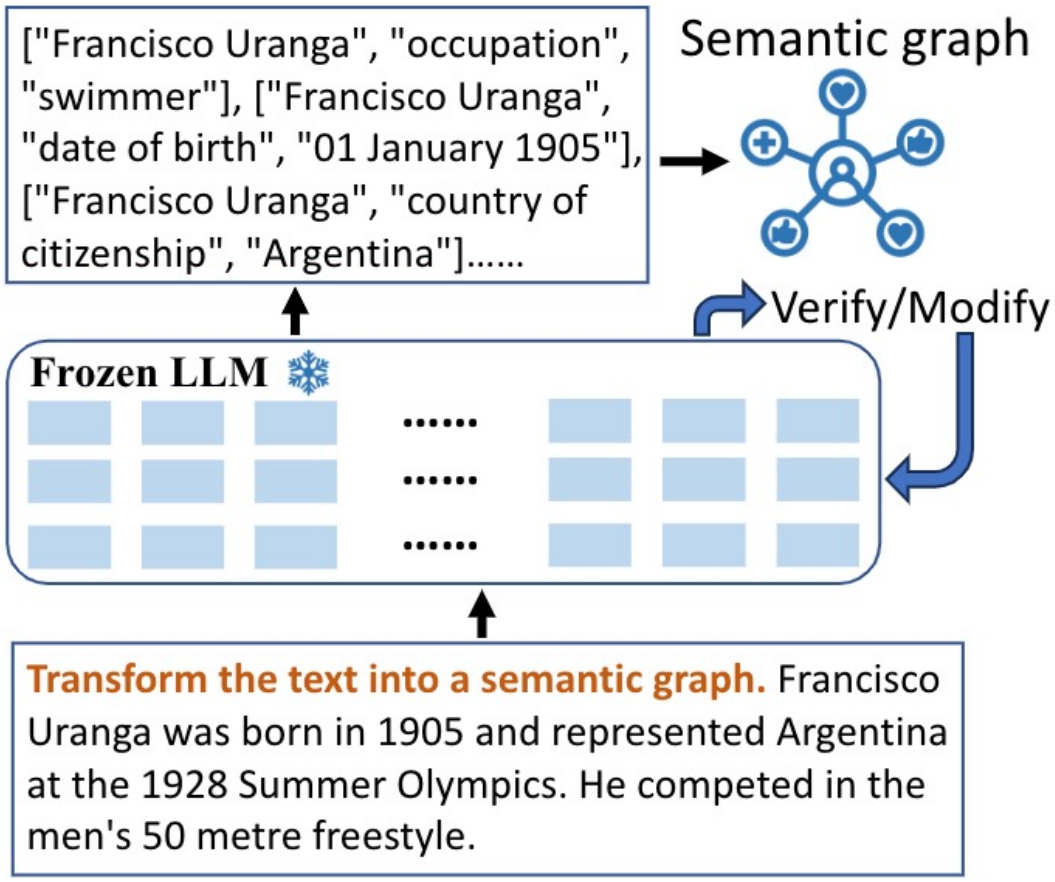

(b) Supervised fine-tuning (SFT) LLMs.



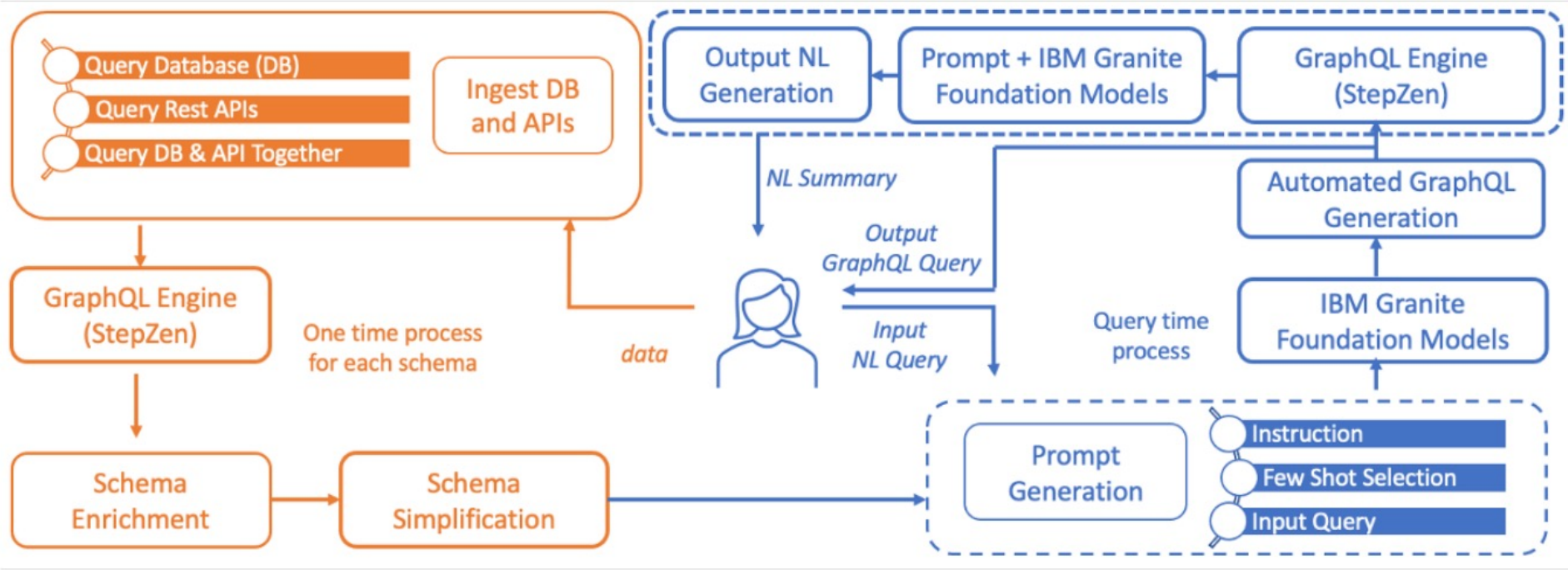

(c) Efficient tunable modules

# Graph Learning

- **LLMs-as-generators**

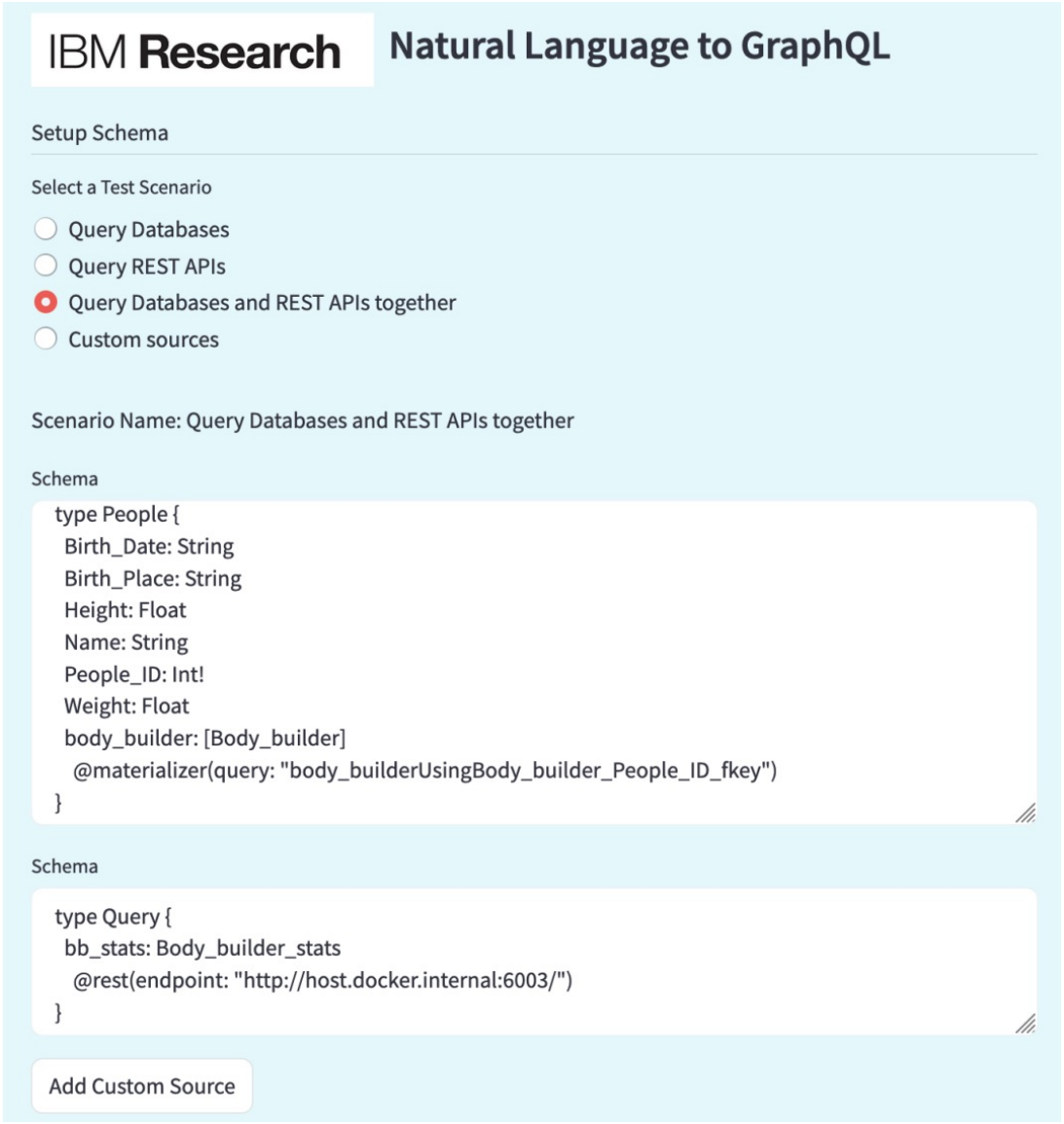


# LLM-powered GraphQL Generator



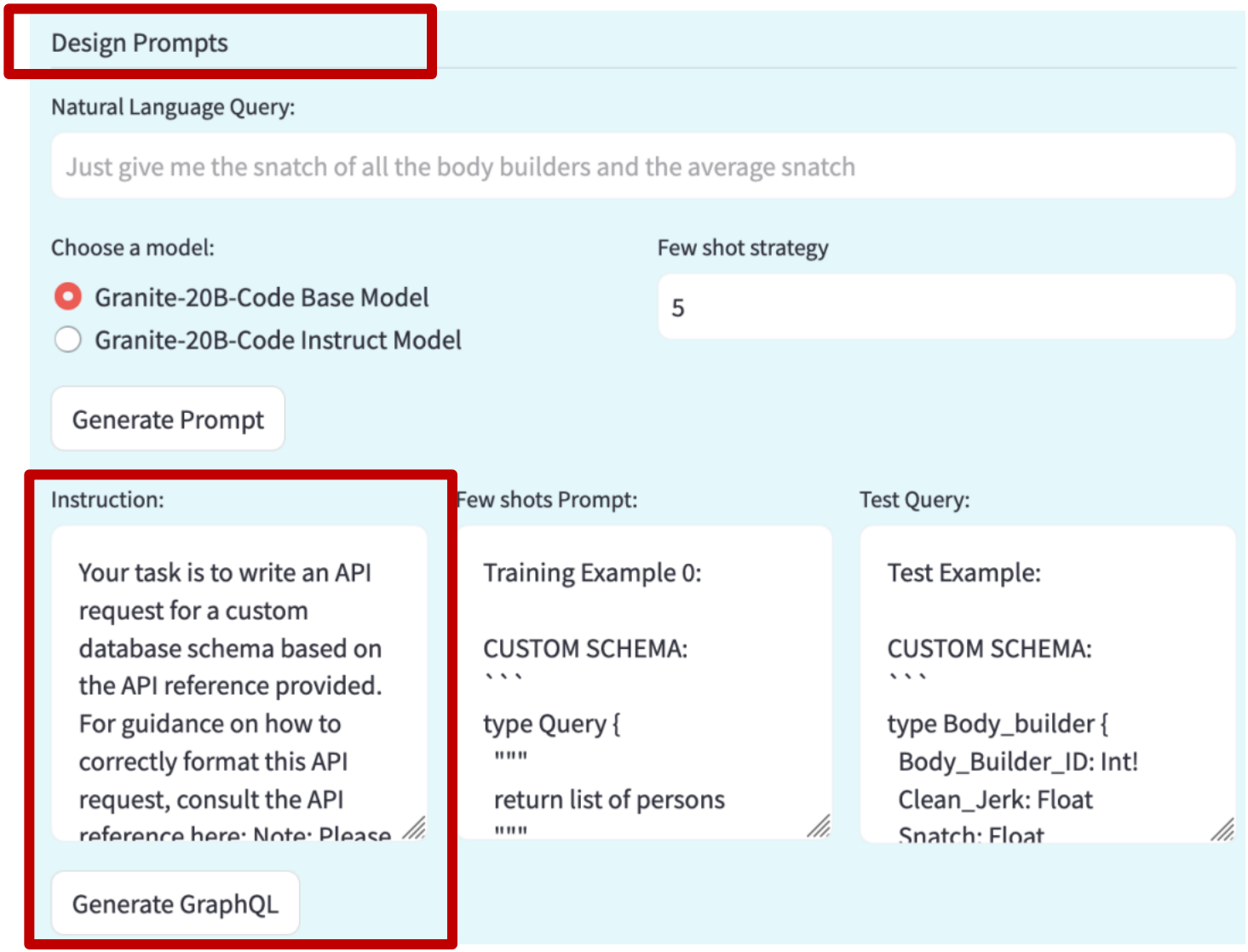

[1] LLM-powered GraphQL Generator for Data Retrieval [IJCAI'24]

# LLM-powered GraphQL Generator




[1] LLM-powered GraphQL Generator for Data Retrieval [IJCAI'24]

# LLM-powered GraphQL Generator




[1] LLM-powered GraphQL Generator for Data Retrieval [IJCAI'24]

# LLM-powered GraphQL Generator

**Natural Language Query:**

Generated GraphQL:

```
graphql {
operationName = ""
query =
 query {
  body_builder(Body_Builder_ID: 1) {
   Body_Builder_ID
```

Execute GraphQL

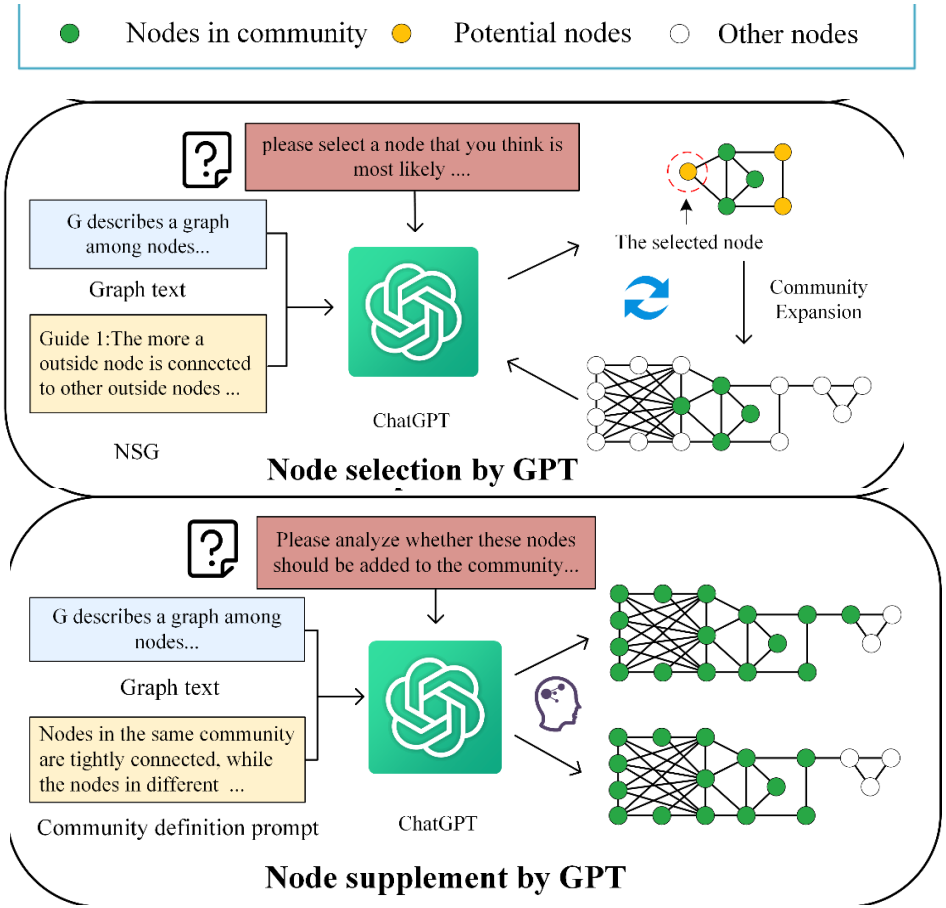
{...}

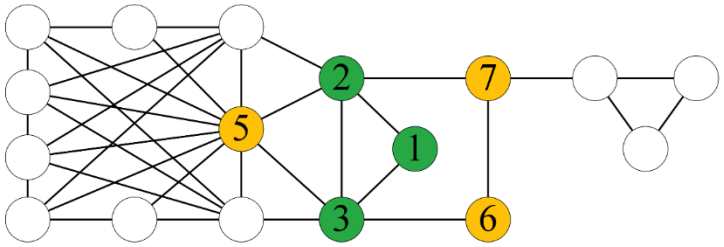
Generate NL Summary

NL Summary: The name of the body builder is Jack Campbell. The height of the body builder is 182 cm. The weight of the body builder is 80 kg. The total time of the body builder is 317.5 seconds. The clean jerk of the body builder is 175.0 seconds. The snatch of the body builder is 142.5 seconds. The birth date of the body builder is January 1, 1992. The birth place of the body builder is Port Huron, Michigan.

[1] LLM-powered GraphQL Generator for Data Retrieval [IJCAI'24]

# ComGPT: Detecting Local Community Structure with LLMs




1 Graph topology

G describes a graph among nodes:1,2,3,5,6,7.
Node 1 is connected to nodes 2,3.
Node 2 is connected to nodes 1,3,5,7.
Node 3 is connected to nodes 1,2,5,6.
Node 5 is connected to nodes 2,3.
Node 6 is connected to nodes 3,7.
Node 7 is connected to nodes 2,6.

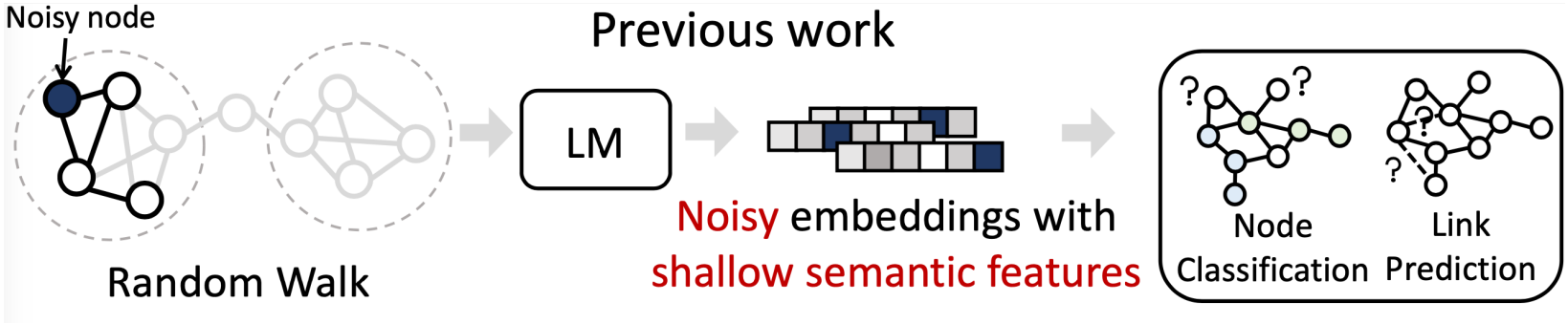
2 Supplementary knowledge

Nodes in the current community:[1,2,3]
The outside nodes contains:[5,6,7]
Node 5 is connected to nodes in community:[1,3]
Node 5 is connected to nodes outside community:null
Node 6 is connected to nodes in community:[3]
Node 6 is connected to nodes outside community:[7]
Node 7 is connected to nodes in community:[2]
Node 7 is connected to nodes outside community:[6]

[1] ComGPT: Detecting Local Community Structure with Large Language Models [TCSS'26]

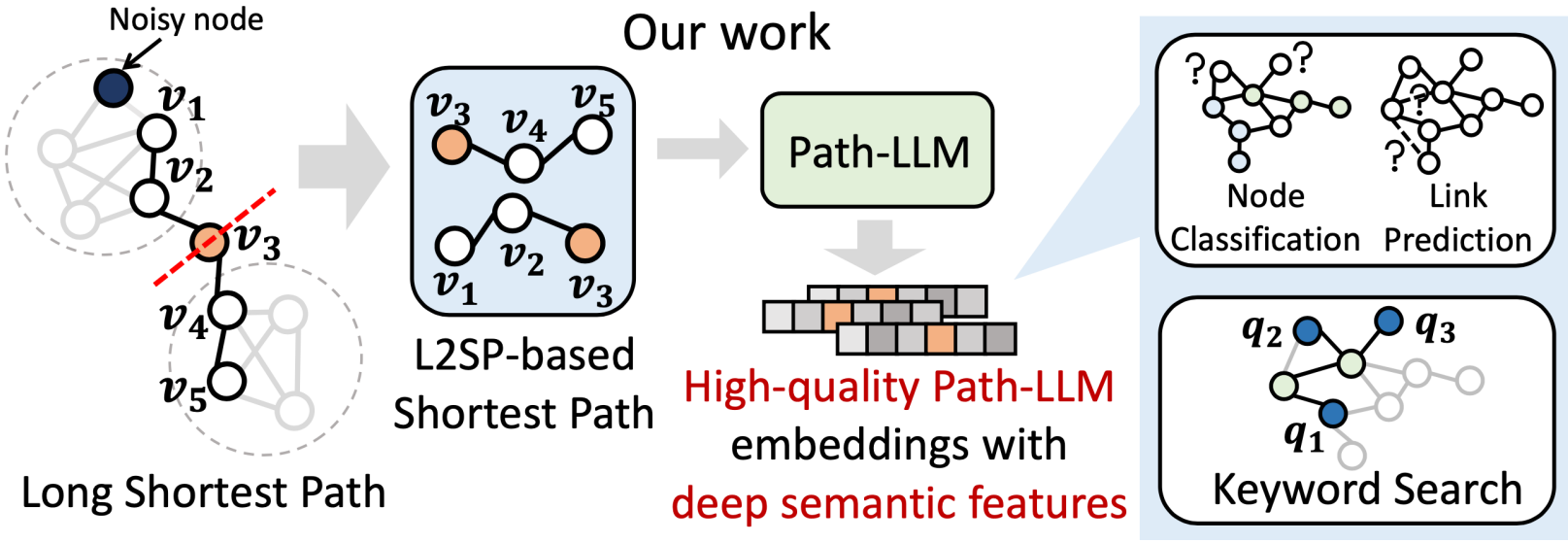
# WalkLM

1) Random walks can hardly cover the bridge edges between different dense groups.
2) Random walks can easily involve noisy nodes to damage the embedding quality. Meanwhile, the node texts of irregular paths cannot match the linguistic rules, bringing more difficulty for learning models.
3) The transferability of WalkLM to LLMs with causal language modeling is limited, as it is primarily designed for relatively small LMs with masked language modeling



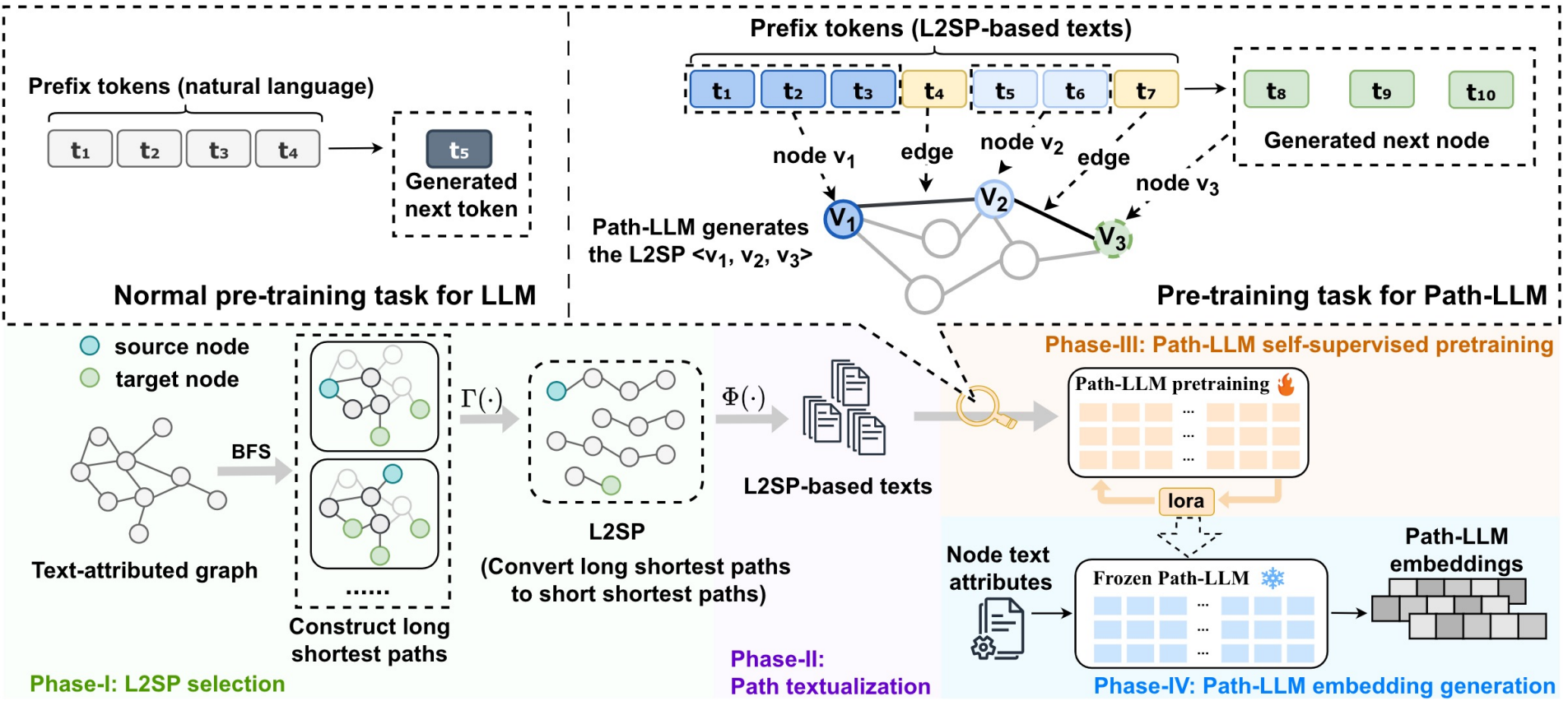
# Path-LLM

- Leverage graph structures, graph information, and representation capabilities of a large language model to derive unified graph embeddings for graph learning tasks and one NP-hard task, keyword search.



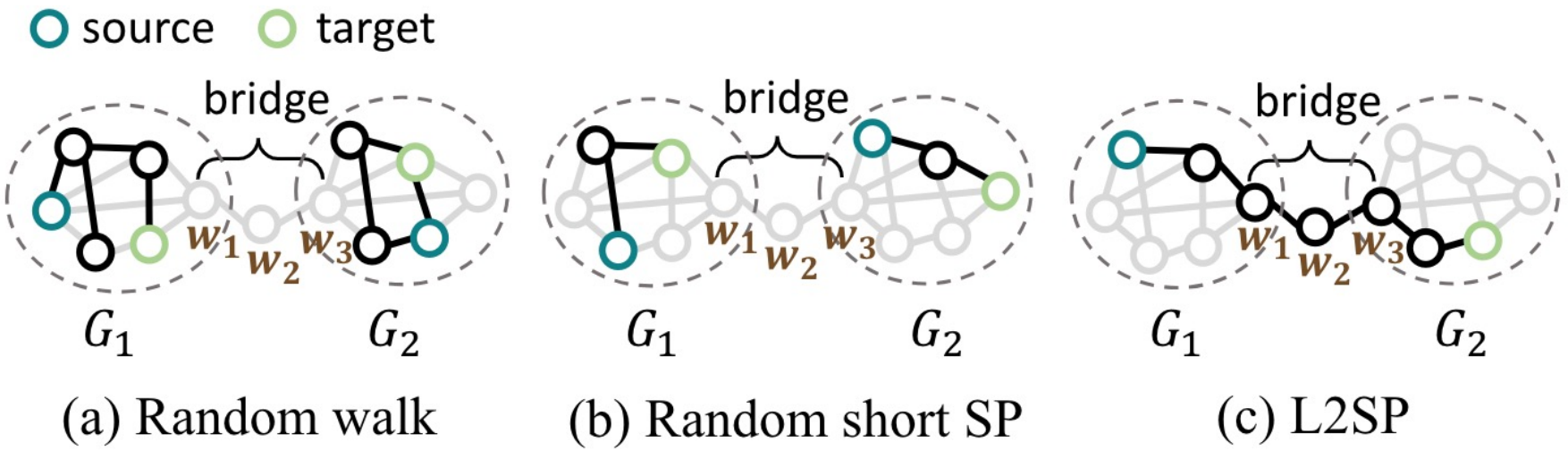

Shang et al.. 2024. Path-LLM: A Shortest-Path-based LLM Learning for Unified Graph Representation

# Path-LLM Framework



# L2SP Selection

- Three path sampling for a comparison



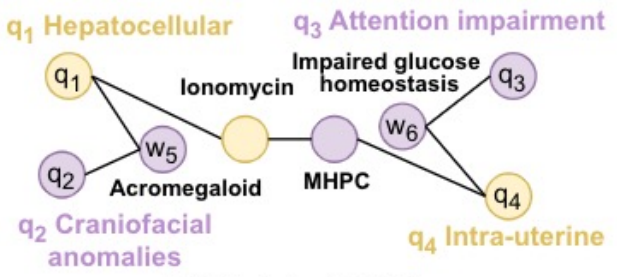
# Experiments

| | Statistics of graphs | | | The number of training paths | | |
|---|---|---|---|---|---|---|
| Dataset | #Nodes | #Edges | Graph type | #RW-paths [86] | #L2SP-paths | The ratio of saving paths |
| PubMed [87] | 63,109 | 244,986 | *heterogeneous* | 295,512 | 19,670 | **93.35%** |
| Cora [60] | 2,708 | 5,429 | *homogeneous* | 100,000 | 12,932 | **87.07%** |
| Citeseer [27] | 3,312 | 8,554 | *homogeneous* | 100,000 | 11,504 | **88.50%** |
| OGB-ARXIV [21] | 169,343 | 1,166,243 | *homogeneous* | 350,000 | 20,523 | **94.14%** |

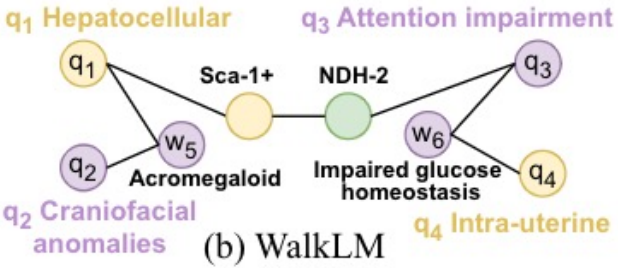
| Datasets | PubMed | | Cora | | ARXIV | | Citeseer | |
|---|---|---|---|---|---|---|---|---|
| Metrics | Macro-F1 | Micro-F1 | Macro-F1 | Micro-F1 | Macro-F1 | Micro-F1 | Macro-F1 | Micro-F1 |
| GCN [43] | 0.2593 | 0.3570 | 0.3628 | 0.4040 | 0.1681 | 0.5199 | 0.4627 | 0.4903 |
| GraphSage [31] | 0.2140 | 0.2430 | 0.3684 | 0.4140 | 0.1962 | 0.5581 | 0.4903 | 0.5240 |
| GATv2 [5] | 0.2501 | 0.2870 | 0.3706 | 0.4740 | 0.1199 | 0.4443 | 0.4816 | 0.5530 |
| WalkLM [86] | 0.2721 | 0.3699 | 0.4031 | 0.5336 | 0.0872 | 0.3823 | 0.5761 | 0.6616 |
| Llama 2 [18] | 0.7167 | 0.7246 | 0.6608 | 0.6946 | 0.3484 | 0.5748 | 0.6739 | 0.7281 |
| GraphGPT [74] | 0.7088 | 0.7202 | 0.6766 | 0.7097 | 0.3610 | 0.5811 | 0.6785 | 0.7335 |
| **Path-LLM (Ours)** | **0.7471** | **0.7555** | **0.7524** | **0.7773** | **0.4529** | **0.6616** | **0.6807** | **0.7411** |

| Datasets | PubMed | | Cora | | ARXIV | | Citeseer | |
|---|---|---|---|---|---|---|---|---|
| Metrics | AUC | Accuracy | AUC | Accuracy | AUC | Accuracy | AUC | Accuracy |
| GCN [43] | 0.5155 | 0.5426 | 0.6680 | 0.7018 | 0.5216 | 0.4415 | 0.6392 | 0.6052 |
| GraphSage [31] | 0.5133 | 0.5211 | 0.7445 | 0.4052 | 0.5110 | 0.3120 | 0.7822 | 0.5227 |
| GATv2 [5] | 0.5204 | 0.5011 | 0.5143 | 0.4488 | 0.2550 | 0.6120 | 0.7488 | 0.5986 |
| WalkLM [86] | 0.5962 | 0.5684 | 0.8581 | 0.7746 | 0.8799 | 0.7923 | 0.9149 | 0.8424 |
| Llama 2 [18] | 0.7144 | 0.6665 | 0.8568 | 0.7809 | 0.9157 | 0.8379 | 0.9290 | 0.8550 |
| GraphGPT [74] | 0.7134 | 0.6631 | 0.8679 | 0.7857 | 0.9221 | 0.8430 | 0.9329 | 0.8570 |
| **Path-LLM (Ours)** | **0.7497** | **0.7111** | **0.9244** | **0.8476** | **0.9655** | **0.9060** | **0.9499** | **0.8825** |

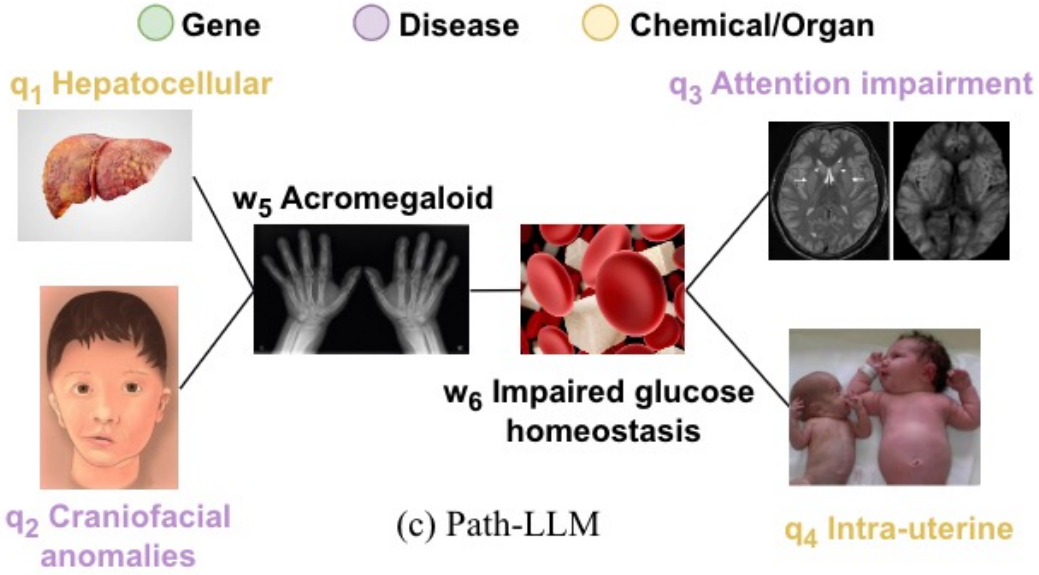
# Case Study: Keyword Search



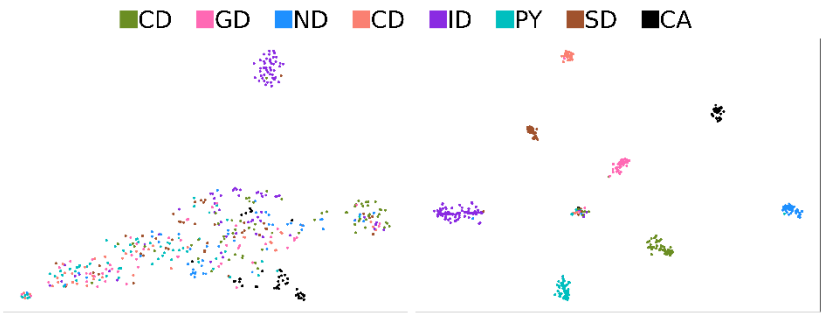

(a) Original TAG



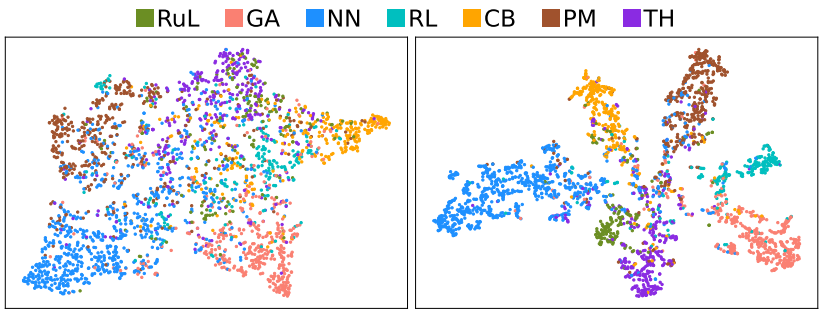

(b) WalkLM




(c) Path-LLM



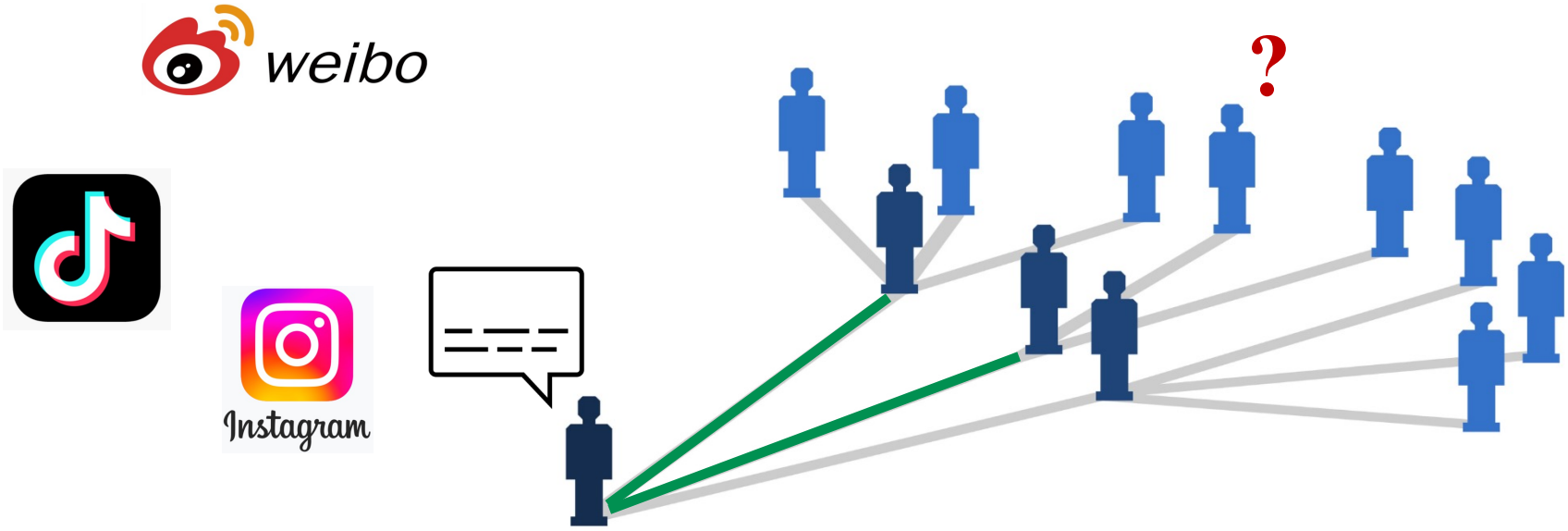

(a) WalkLM (b) Path-LLM




(a) WalkLM (b) Path-LLM

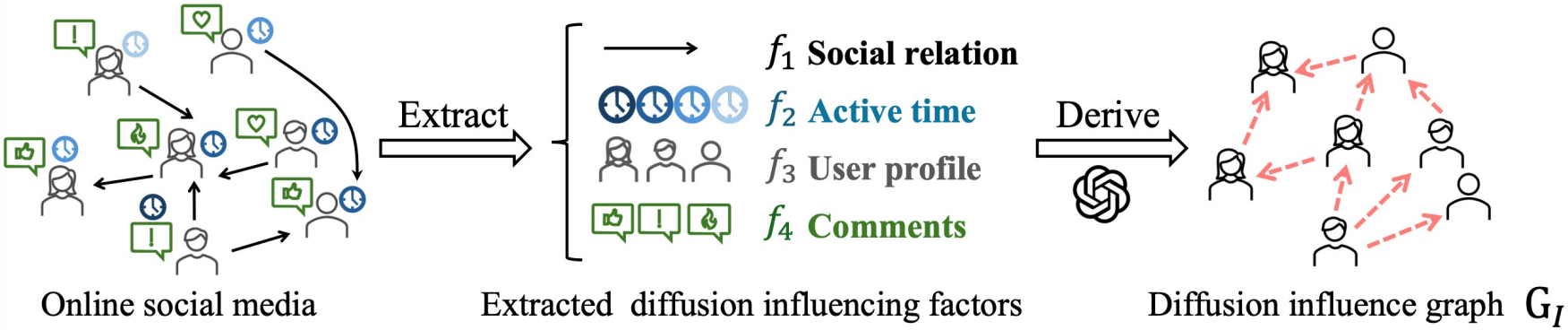
# Information Diffusion Cascade

- Many social platforms allow users to post various content and communicate on topics of mutual interest, facilitating the fast diffusion of information and form **information cascades**.
- **Diffusion prediction** aims to predict the potential users who will be infected based on the observed diffusion cascades.



# Motivation

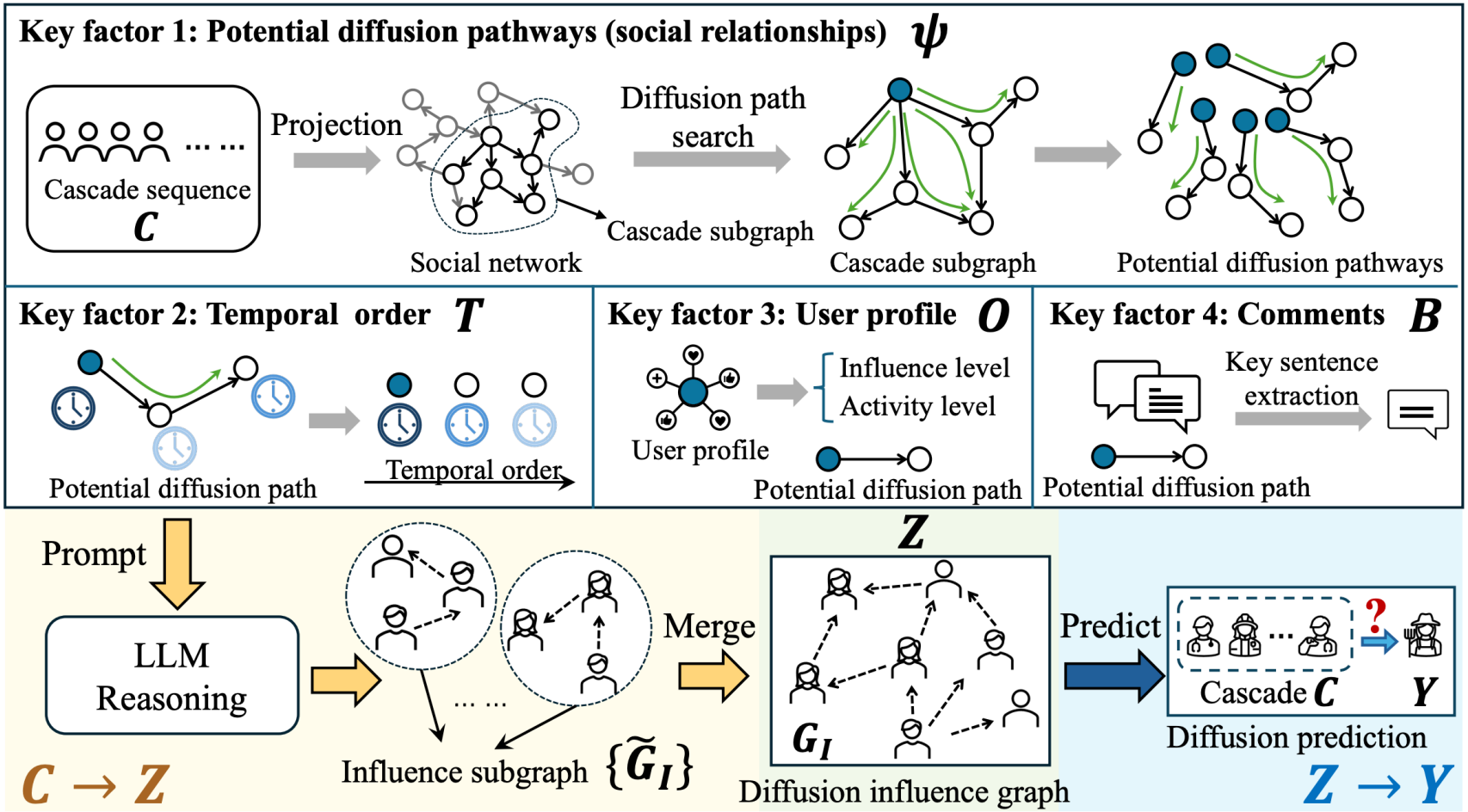
- As the authentic diffusion process is *unknown, unexplainable,* and *non-transparent*, constructing Z becomes challenging.





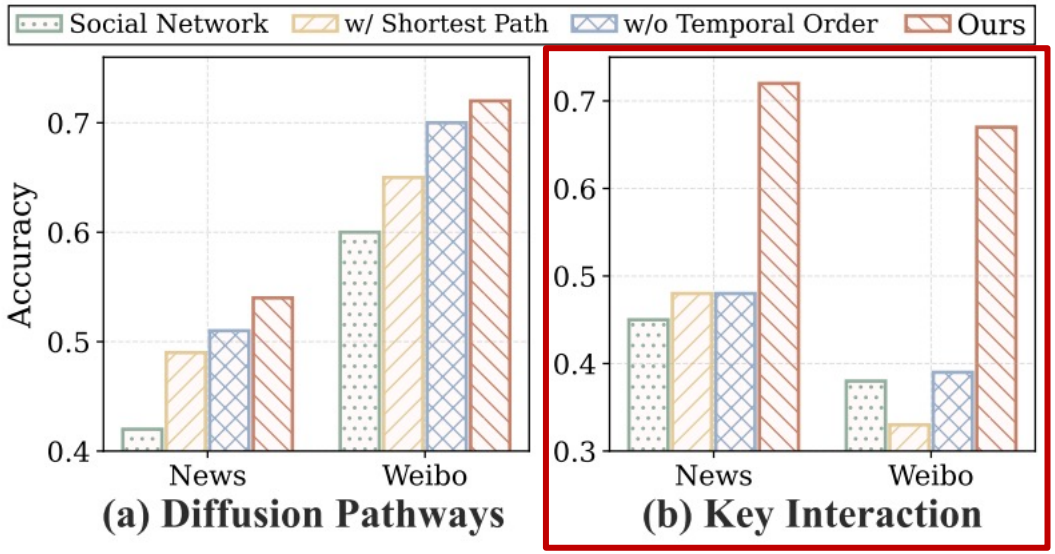

Shang et al. Make Information Diffusion Explainable: LLM-based Causal Framework for Diffusion Prediction. NeurIPS 2025.

# Method


Key factor 1: Potential diffusion pathways (social relationships) $\psi$
Cascade sequence $C$
Projection
Social network
Diffusion path search
Cascade subgraph
Cascade subgraph
Potential diffusion pathways
Key factor 2: Temporal order $T$
Potential diffusion path
Temporal order
Key factor 3: User profile $O$
User profile
Influence level
Activity level
Potential diffusion path
Key factor 4: Comments $B$
Key sentence extraction
Potential diffusion path
Prompt
LLM Reasoning
$C \rightarrow Z$
Influence subgraph $\{\tilde{G}_I\}$
Merge
$Z$
$G_I$
Diffusion influence graph
Predict
Cascade $C$
$Y$
Diffusion prediction
$Z \rightarrow Y$

# Key Interaction Evaluation

- We assess whether $G_I$ can observe the high-frequency interactions among participants.
- To avoid information leakage, we split the dataset by the cascade initiation time in an 8:2 ratio. We assess whether our method can observe key interactions in the 20% of newly occurred cascades based on the interactions in the 80% of historical cascades.
- $G_I$ more effectively captures strong ties and key interactions among participants in new cascades.



# Experiments

- Datasets: News and Weibo, including *the social network structure*, *detailed user profiles*, and *information diffusion cascades.*
- Competitors: 8 state-of-the-art diffusion prediction methods
- MILD significantly outperforms eight state-of-the-art baselines on two real-world information diffusion prediction datasets, achieving average improvements of 6.3% on Hits@K and 7.4% on MAP@K.

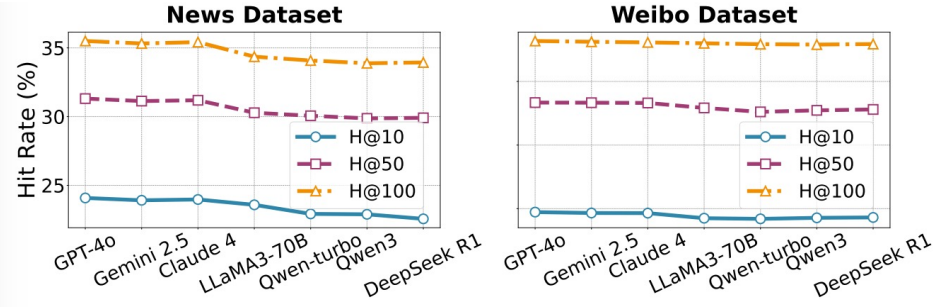
| Datasets | News | | | | | | Weibo | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metrics | H@10 | H@50 | H@100 | M@10 | M@50 | M@100 | H@10 | H@50 | H@100 | M@10 | M@50 | M@100 |
| DyHGCN [37] | 18.94 | 24.50 | 27.31 | 10.50 | 10.76 | 10.80 | 10.51 | 15.39 | 18.50 | 6.01 | 6.23 | 6.27 |
| MSHGAT [28] | 20.10 | 25.82 | 28.85 | 11.68 | 11.95 | 11.99 | 11.41 | 18.34 | 21.77 | 6.16 | 6.48 | 6.53 |
| DisenIDP [3] | 20.47 | 26.33 | 29.08 | 12.03 | 12.29 | 12.36 | 12.04 | 18.83 | 22.61 | 6.73 | 7.01 | 7.06 |
| RotDiff [23] | 20.95 | 27.02 | 29.80 | 12.52 | 12.80 | 12.84 | 12.99 | 20.46 | 24.70 | 7.76 | 8.12 | 8.18 |
| MINDS [14] | 20.04 | 25.66 | 28.70 | 11.72 | 11.93 | 11.98 | 11.69 | 18.07 | 21.66 | 6.24 | 6.51 | 6.57 |
| MGCL [5] | 21.26 | 28.11 | 31.93 | 12.90 | 13.18 | 13.24 | 13.04 | 20.18 | 24.75 | 7.60 | 7.99 | 8.03 |
| GODEN [29] | 22.01 | 29.14 | 33.17 | 13.34 | 13.67 | 13.73 | 14.08 | 21.95 | 26.71 | 8.09 | 8.28 | 8.52 |
| CARE [41] | 22.47 | 28.71 | 32.01 | 14.53 | 14.81 | 14.85 | 13.35 | 21.96 | 26.62 | 7.65 | 8.05 | 8.11 |
| **MILD (Ours)** | **24.07** | **31.33** | **35.52** | **15.50** | **15.85** | **15.91** | **14.71** | **23.37** | **28.17** | **8.71** | **9.10** | **9.17** |
| *Improvement (%)* | +7.12 | +7.52 | +7.08 | +6.67 | +5.19 | +7.14 | +4.47 | +6.42 | +5.47 | +7.66 | +9.89 | +7.63 |

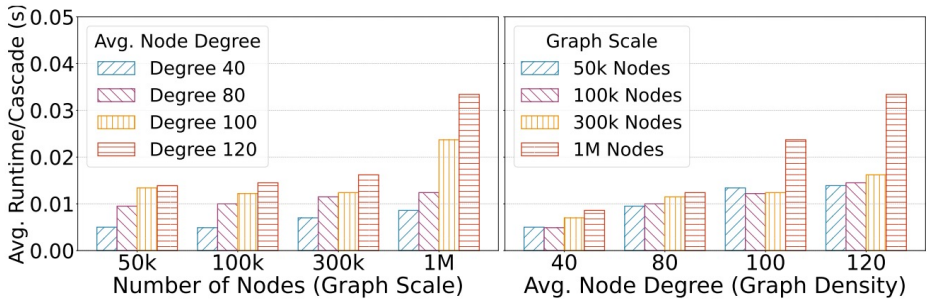
# Experiments

- Ablation study on Four Key Factors.
- Ablation study on *GI*.

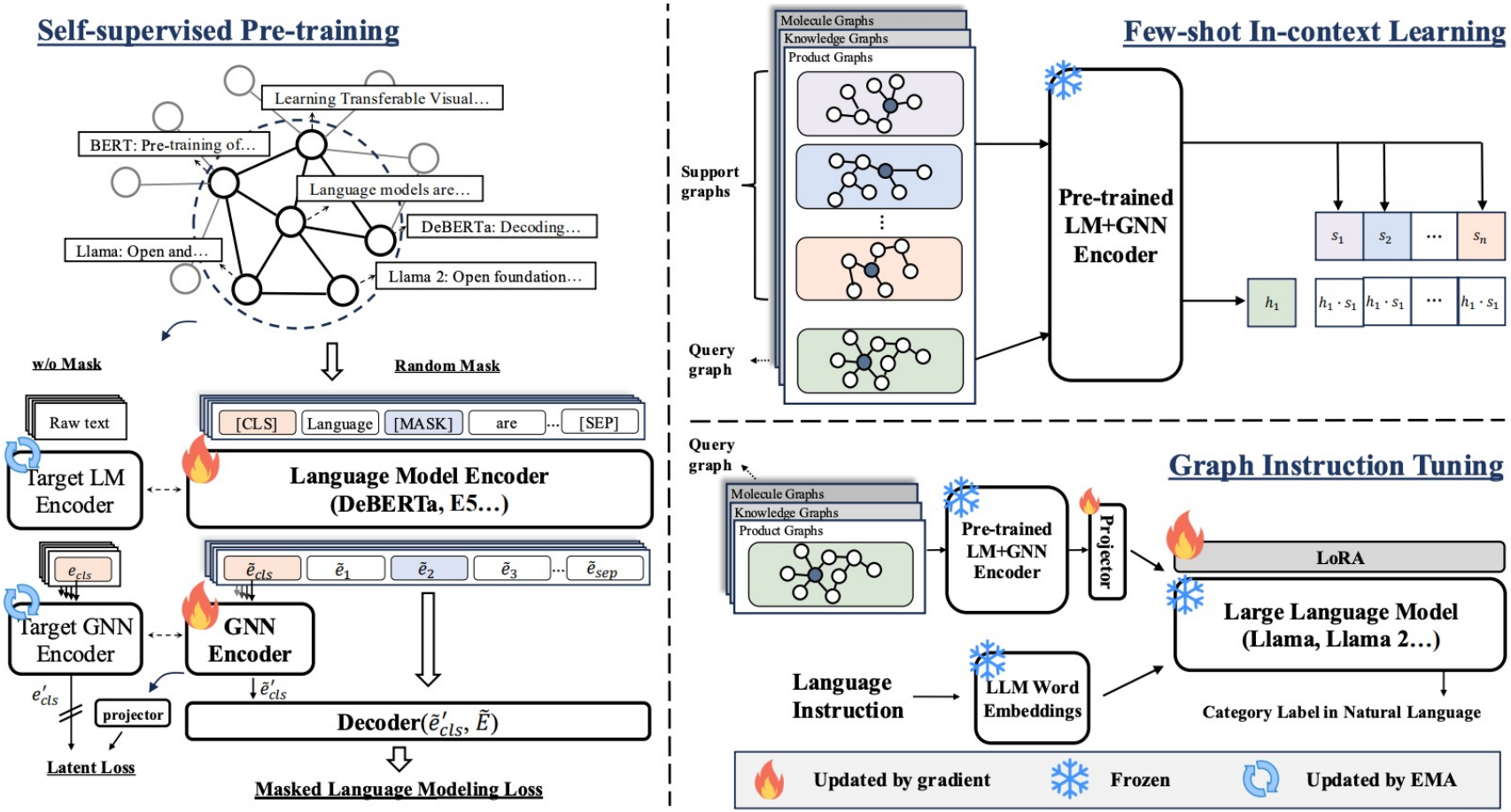
| Datasets / Metrics | News H@10 | News H@50 | News H@100 | News M@10 | News M@50 | News M@100 | Weibo H@10 | Weibo H@50 | Weibo H@100 | Weibo M@10 | Weibo M@50 | Weibo M@100 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **w/o** Path Selection | 23.14 | 29.20 | 33.72 | 14.53 | 14.76 | 14.82 | 14.33 | 21.96 | 27.15 | 8.44 | 8.82 | 8.87 |
| **w/o** Post Content | 23.72 | 29.98 | 33.65 | 14.90 | 15.21 | 15.24 | 14.58 | 22.87 | 27.64 | 8.51 | 8.96 | 9.02 |
| **w/o** Time Information | 23.16 | 30.08 | 33.97 | 14.77 | 14.83 | 14.89 | 14.36 | 22.15 | 26.83 | 8.49 | 8.84 | 8.89 |
| **w/o** User Profile | 23.77 | 30.73 | 34.62 | 14.96 | 15.24 | 15.28 | 14.52 | 22.81 | 27.46 | 8.52 | 8.90 | 8.93 |
| **w** Social bias | 22.92 | 27.64 | 31.35 | 14.68 | 14.75 | 14.90 | 14.33 | 22.12 | 26.64 | 8.45 | 8.77 | 8.96 |
| **w/o** Influence bias | 21.81 | 27.52 | 30.62 | 14.03 | 14.31 | 14.35 | 14.08 | 21.98 | 26.21 | 8.26 | 8.62 | 8.68 |
| **Full Model** | **24.07** | **31.33** | **35.52** | **15.50** | **15.85** | **15.91** | **14.71** | **23.37** | **28.17** | **8.71** | **9.10** | **9.17** |

- MILD across Different LLMs



# UniGraph: Graph Foundation Model for Text-Attributed Graphs




[1] UniGraph: Learning a Unified Cross-Domain Foundation Model for Text-Attributed Graphs [KDD'25]

# UniGraph: Graph Foundation Model for Text-Attributed Graphs

UniGraph consists of three main components:

- **Cascaded LM-GNN Backbone**
  - Language Model (LM) encoder for processing text attributes
  - Graph Neural Network (GNN) encoder for capturing graph structure
  - Fusion layer to combine LM and GNN outputs
- **Graph Siamese Masked Autoencoders**
  - Masked Language Modeling (MLM) for text understanding
  - Latent space regularization for better representation learning
  - Target networks with exponential moving average updates
- **Instruction Tuning**
  - Zero-shot transfer capabilities
  - Task-specific instruction processing
  - LoRA-based parameter-efficient fine-tuning

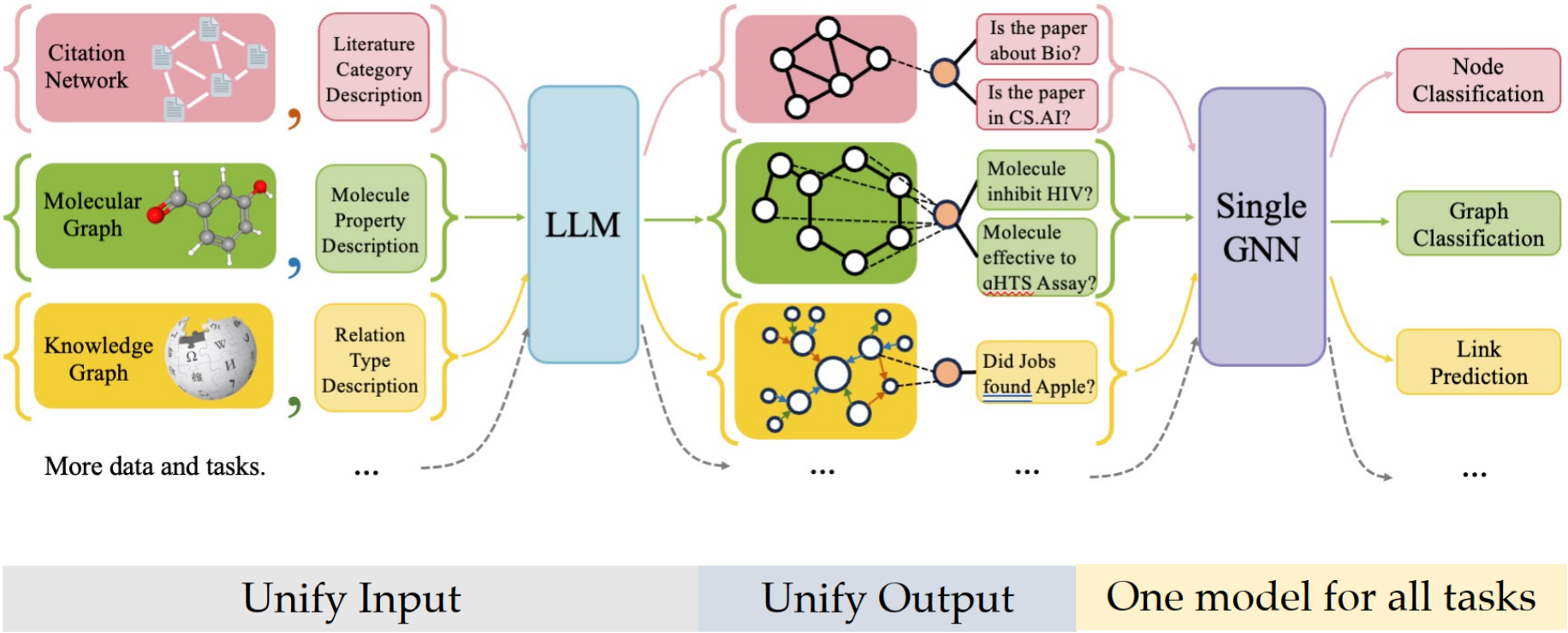
[1] UniGraph: Learning a Unified Cross-Domain Foundation Model for Text-Attributed Graphs [KDD'25]

# UniGraph: Graph Foundation Model for Text-Attributed Graphs

| | Node Classification | | | | | Edge Classification | | Graph Classification | |
|---|---|---|---|---|---|---|---|---|---|
| | Cora | PubMed | Arxiv | Products | Wiki-CS | FB15K237 | WN18RR | HIV | PCBA |
| **Use word2vec to encode raw text as input features.** | | | | | | | | | |
| Linear | 50.12±0.12 | 61.99±0.21 | 50.11±0.17 | 66.29±0.21 | 66.23±0.11 | 81.21±0.21 | 69.03±0.32 | 60.99±0.31 | 54.35±1.34 |
| DGI | 51.99±0.45 | 55.76±0.56 | 55.21±0.21 | 64.21±0.32 | 67.11±0.12 | 26.99±0.22 | 52.04±0.22 | 60.12±0.32 | 54.22±1.23 |
| BGRL | 56.73±0.23 | 63.77±0.23 | 62.21±0.21 | 66.22±0.39 | 70.12±0.15 | 64.91±0.22 | 56.44±0.21 | 60.67±0.39 | 54.89±1.11 |
| GraphMAE | 60.12±0.87 | 66.22±0.35 | 65.22±0.22 | 67.19±0.39 | 68.11±0.12 | 61.11±0.12 | 59.76±0.29 | 59.21±0.31 | 52.10±1.24 |
| GraphMAE2 | 61.19±0.45 | 65.99±0.21 | 67.19±0.11 | 67.73±0.12 | 68.84±0.37 | 63.76±0.12 | 60.24±0.23 | 60.23±0.35 | 53.90±0.99 |
| GCN | 71.98±1.33 | 69.86±1.01 | 70.11±0.14 | 79.12±0.12 | 78.12±0.37 | 90.21±0.56 | 74.21±0.63 | 70.11±1.35 | **60.23±0.45** |
| GAT | 72.42±1.21 | 70.45±1.21 | 70.89±0.43 | 79.67±0.34 | 79.09±0.67 | 88.65±0.26 | 74.80±0.64 | 71.12±1.34 | 56.24±1.01 |
| **Use DeBERTa-base to encode raw text as input features.** | | | | | | | | | |
| Linear | 29.34±0.11 | 48.51±0.22 | 43.22±0.25 | 41.29±0.21 | 41.09±0.10 | 78.11±0.32 | 65.03±0.11 | 60.11±0.34 | 53.46±1.02 |
| DGI | 30.36±0.36 | 52.91±0.51 | 49.15±0.21 | 56.18±0.36 | 63.15±0.52 | 29.12±0.13 | 51.98±0.53 | 59.12±0.34 | 53.23±0.47 |
| BGRL | 40.10±0.34 | 52.99±0.41 | 56.19±0.22 | 60.15±0.44 | 66.87±0.32 | 45.69±0.25 | 46.15±0.39 | 61.33±0.62 | 54.22±1.04 |
| GraphMAE | 43.11±0.51 | 54.14±0.32 | 57.11±0.64 | 65.22±0.43 | 69.01±0.33 | 56.21±0.21 | 53.22±0.39 | 62.01±0.65 | 51.45±1.01 |
| GraphMAE2 | 42.87±0.43 | 53.98±0.31 | 59.39±0.49 | 67.91±0.48 | 70.47±0.13 | 55.82±0.28 | 51.78±0.24 | 61.42±0.61 | 52.35±0.35 |
| GCN | 48.42±1.33 | 60.33±1.98 | 60.76±2.42 | 66.98±2.32 | 77.43±0.43 | 85.23±0.65 | 72.04±0.32 | 66.24±1.31 | 58.21±1.04 |
| GAT | 47.99±1.89 | 61.01±1.18 | 63.11±2.24 | 67.02±2.11 | 78.10±0.34 | 83.01±1.01 | 73.98±0.23 | 67.12±1.23 | 56.45±0.45 |
| **Use raw text as input features.** | | | | | | | | | |
| GIANT-XRT | 70.23±0.87 | 64.35±0.43 | 70.87±0.11 | 66.93±0.32 | 70.13±0.88 | 89.65±0.85 | 72.78±0.66 | 65.14±0.32 | 51.34±1.98 |
| +GraphMAE2 | 80.11±0.35 | 69.43±0.45 | 72.01±0.24 | 75.23±0.34 | 76.58±0.21 | 76.12±1.03 | 57.32±0.66 | 67.23±0.98 | 52.01±0.45 |
| NoPretrain | 40.98±0.32 | 53.01±0.35 | 62.22±0.20 | 67.12±0.21 | 73.21±0.15 | 23.19±0.21 | 51.03±0.29 | 58.01±0.21 | 51.01±0.43 |
| UniGraph | **81.43±0.55** | **74.33±0.23** | **72.91±0.42** | **80.11±0.23** | **79.98±1.21** | **94.81±1.32** | **85.45±0.34** | **71.23±1.93** | 57.67±0.85 |

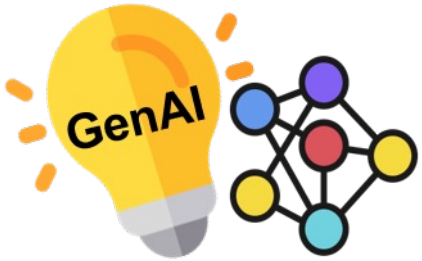
[1] UniGraph: Learning a Unified Cross-Domain Foundation Model for Text-Attributed Graphs [KDD'25]

# Graph Foundational Model for All Classification Tasks




[1] One for All: Towards Training One Graph Model for All Classification Tasks [ICLR'24]

# 3. Graphs for LLMs




A. Question-Answering: GraphRAG
   - From RAG to GraphRAG
   - RAG vs GraphRAG

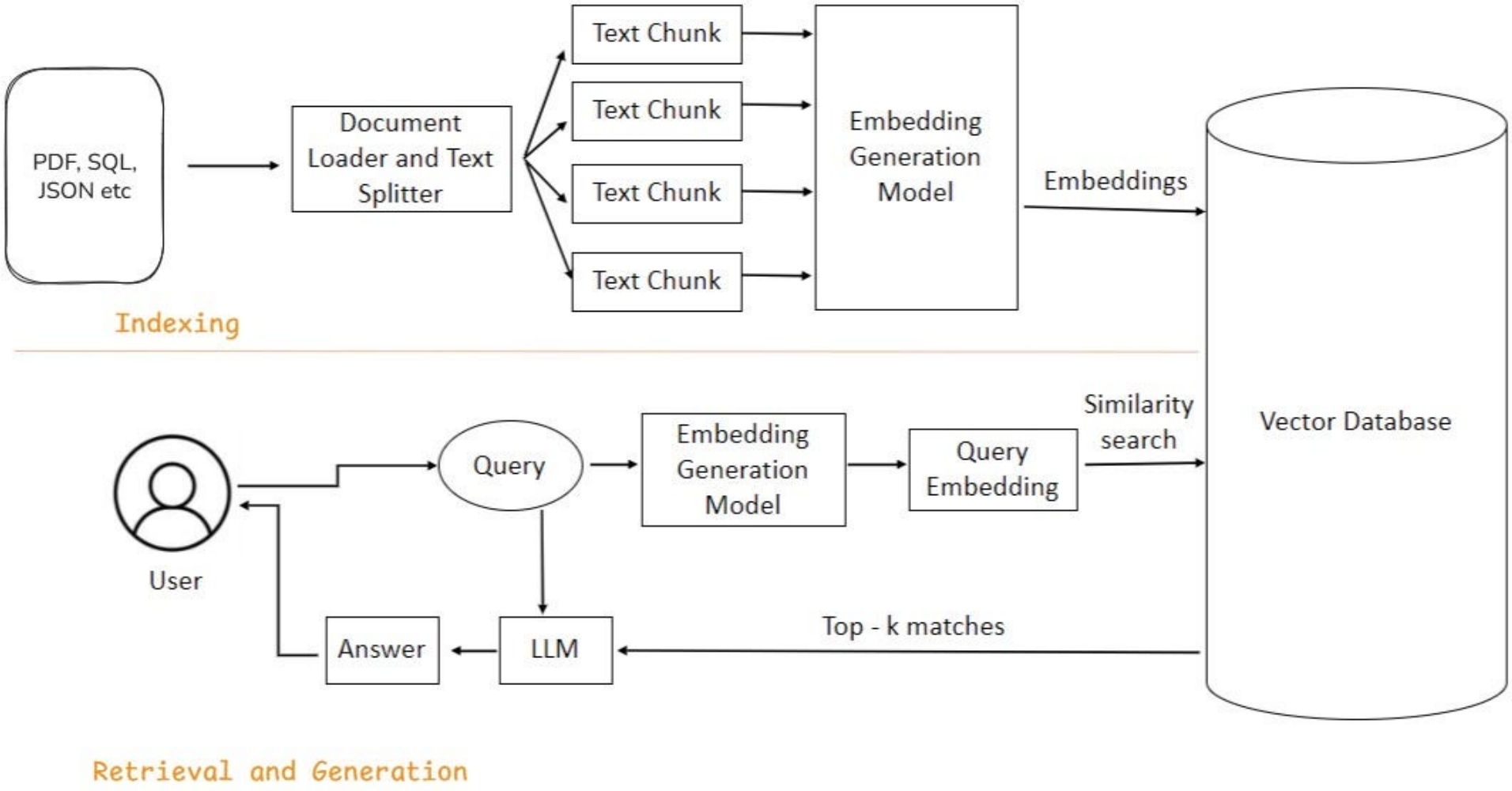
B. Agent Memory: Memory Graph

C. LLM Planning: WorkFlow Graph

D. LLM Selection: GraphRouter

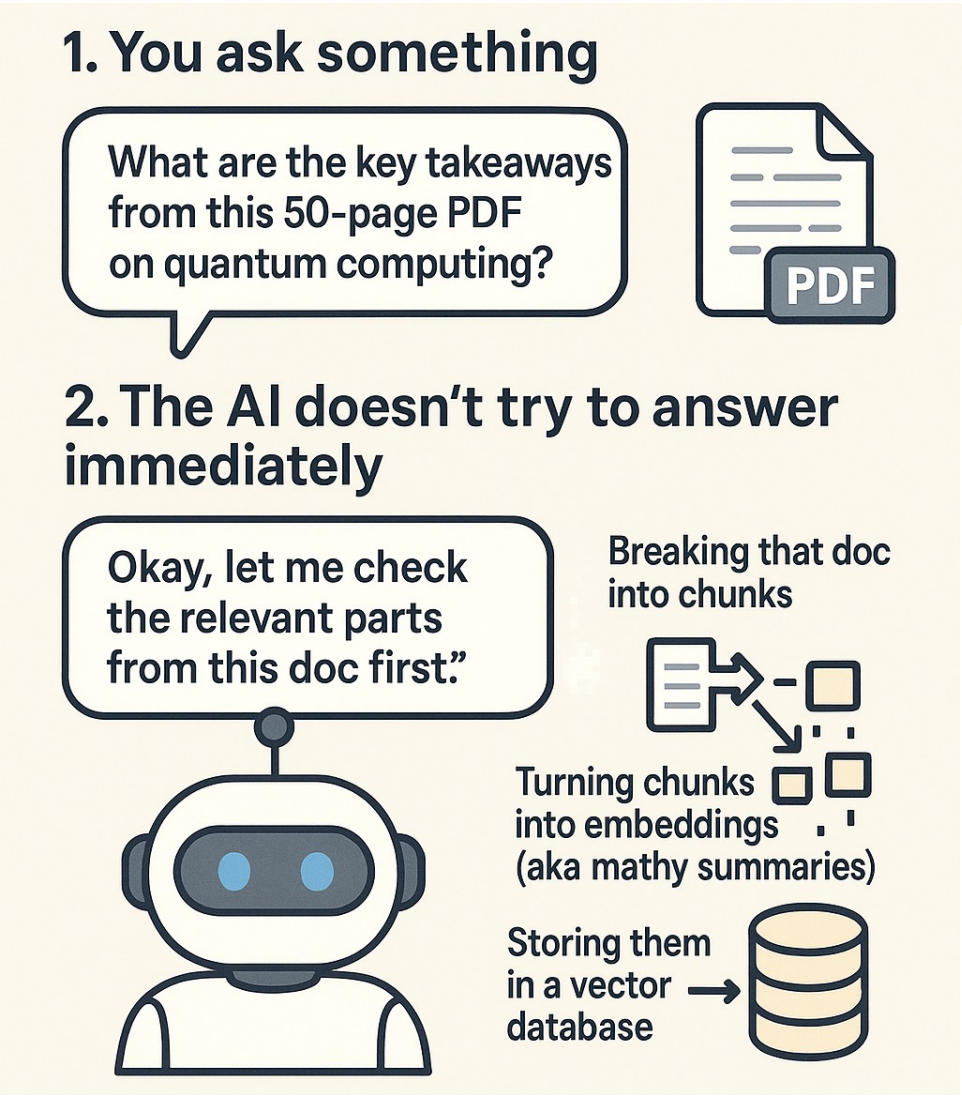
Presented by Longxu Sun

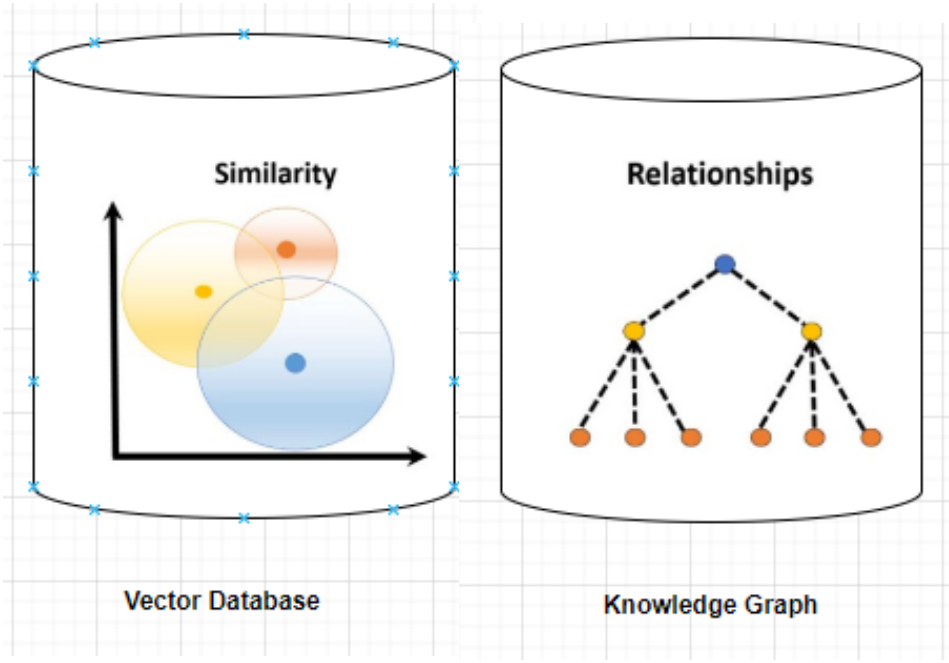
# From RAG to GraphRAG




Standard RAG retrieves flat text chunks by semantic similarity.

# From RAG to GraphRAG




GraphRAG enables structure-aware retrieval and generation.

# GraphRAG Surveys

Most GraphRAG systems can be viewed as a three-stage pipeline:

**1. Graph-based indexing**
Build graphs, communities, or hierarchical indexes from documents/KGs.

**2. Graph-guided retrieval**
Retrieve nodes, paths, subgraphs, neighborhoods, or community summaries.

**3. Graph-enhanced generation**
Generate answers using structured and connected evidence.

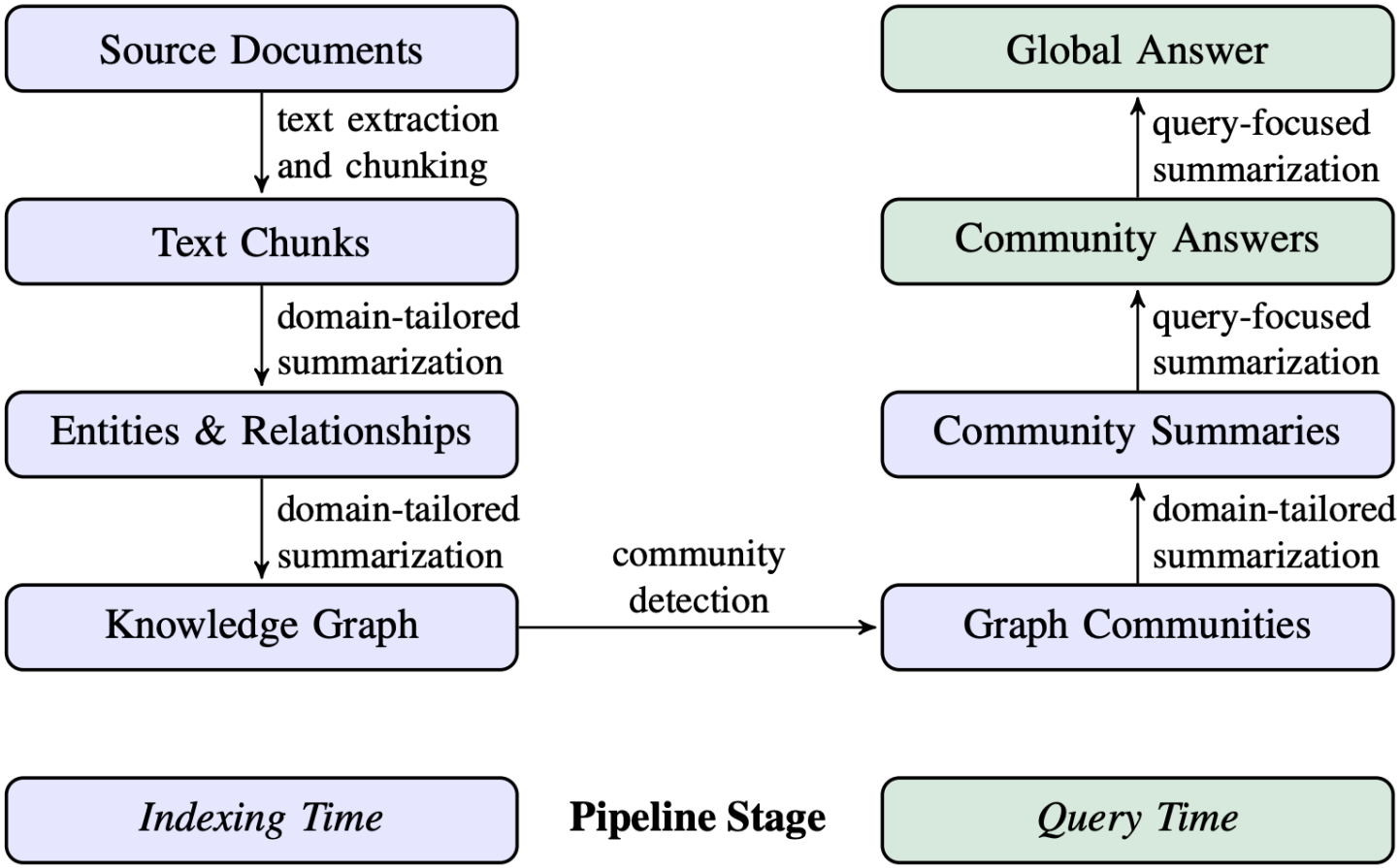
**Representative directions:**

- Community-based global summarization
- Attributed hierarchical retrieval
- Unified analysis and benchmarking: VLDB'25 unified framework

[1] https://arxiv.org/abs/2501.13958
[2] https://arxiv.org/abs/2408.08921
[3] https://papers.ssrn.com/sol3/papers.cfm?abstract_id=6713979

# From Local to Global: A GraphRAG Approach to Query-Focused Summarization

Microsoft

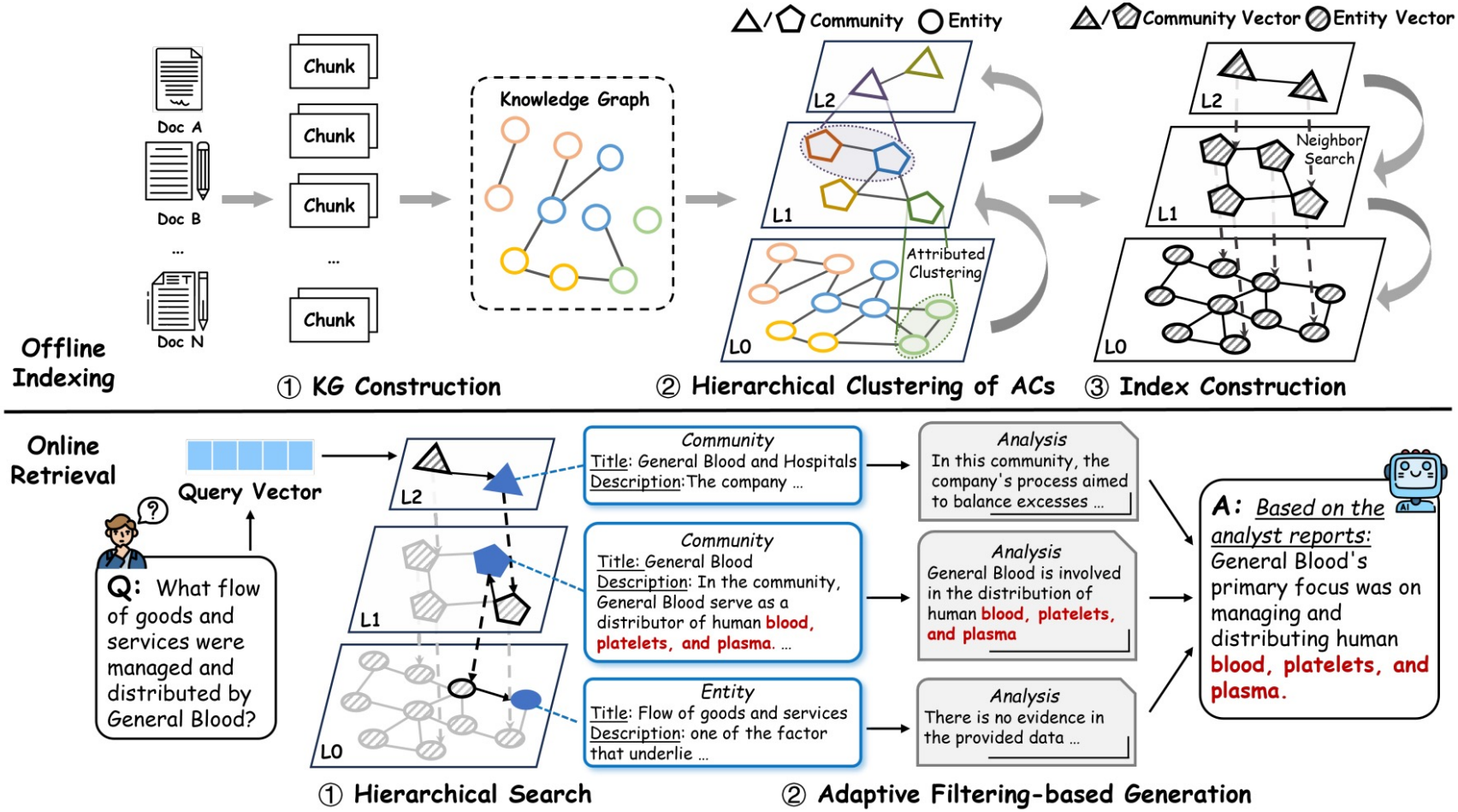



[1] https://arxiv.org/html/2404.16130v2

# ArchRAG: Attributed Community-based Hierarchical Retrieval-Augmented Generation [AAAI'26]

| Issue | Description |
|---|---|
| Low Community Quality | Leiden algorithm relies solely on graph structure, ignoring semantic info. Produces mixed-topic communities with poor summaries, degrading performance. |
| Limited Compatibility | Global and local search operate only at a single granularity. Cannot handle abstract and specific queries simultaneously, limiting real-world applicability. |
| High Generation Cost | Abstract queries perform well, but analyzing many communities is costly. On Multihop-RAG, 2,984 communities detected; 100 questions required ~$650 and 106M tokens. |
| Inaccurate Entity Recognition | Methods like HippoRAG show recognition issues: 5.0% of HotpotQA questions missed entities; 15.8% of recognized entities were low quality, causing irrelevant retrieval. |
| Inaccurate Retrieval Process | Graph RAG often returns irrelevant content during search (e.g., GraphRAG local search inaccuracies), reducing overall performance. |

# ArchRAG: Attributed Community-based Hierarchical Retrieval-Augmented Generation [AAAI'26]




[1] https://arxiv.org/pdf/2502.09891v2

# ArchRAG: Attributed Community-based Hierarchical Retrieval-Augmented Generation [AAAI'26]

**Responses of Different Methods**

**Zero-shot**

**Jesse Schell**.
He is the CTO of Calm and was the lead designer on the "Star Wars" theme park ride at Disney.

---

**Vanilla RAG**

It seems like there are multiple articles and topics discussed in this text. Here's a brief summary of each section:
1.**Robotics predictions.** The article discusses the future of robotics, with predictions made by Brian Heater. </>
2.**TechCrunch's Week in Review.** This section summarizes various tech industry happenings. </>
3.**Bumble gets a new CEO.** Bumble announces a change in leadership. </>...

---

**HippoRAG**

There is **no question** in the provided text. It appears to be a collection of news articles and updates related to OpenAI's ChatGPT. </>

---

**GraphRAG-Global**

*Key Points and Implications*
The individual associated with generative AI technology who was reportedly ousted is **Andrew Ng**. </>
According to multiple analysts, **Andrew Ng** [Data: Reports (5, 6)] </>
*Implications of the New Venture* </>

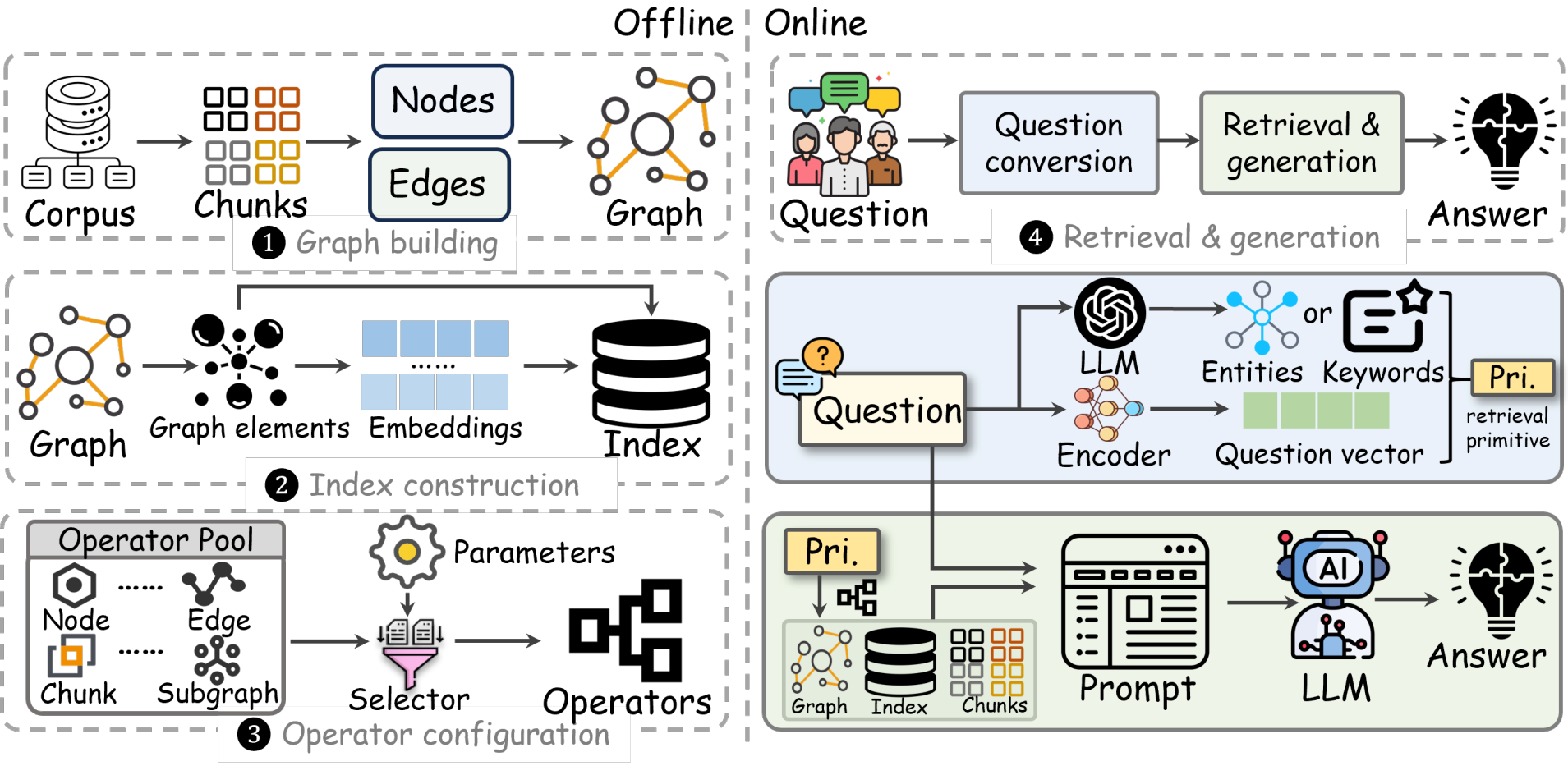
---

**ArchRAG**

**Sam Altman**.
This information is supported by multiple analysts' reports, which highlight **Sam Altman's** involvement in various AI-related projects.
The direct answer is simply **"Sam Altman"**. </>
Note: I have not included any information from Analyst 2, 3, or 7 as their reports are not relevant. </>

[1] https://arxiv.org/pdf/2502.09891v2

# In-depth Analysis of Graph-based RAG in a Unified Framework [VLDB'25]




[1] https://arxiv.org/pdf/2503.04338

# RAG vs GraphRAG: When Does Graph Help?

**RAG works well when:**

- Questions ask for local factual details.
- Relevant evidence is contained in a few text chunks.
- Simple and efficient retrieval is preferred.

**GraphRAG helps when:**

- Questions require multi-hop reasoning.
- Evidence is distributed across the corpus.
- Corpus-level themes, communities, or relationships matter.
- Structured aggregation improves answer coverage and diversity.

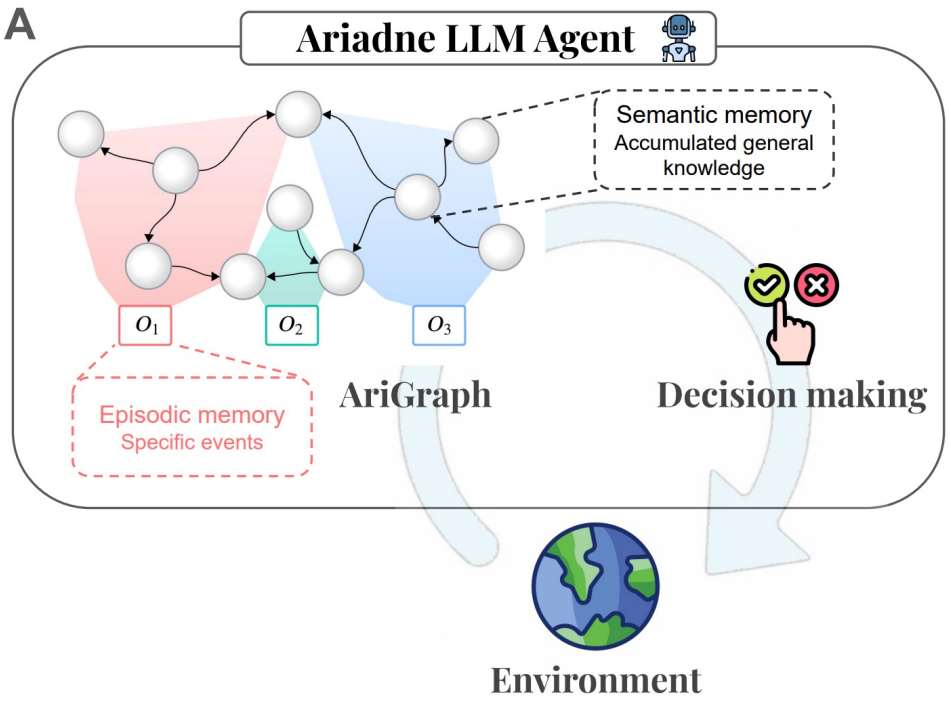
**Design implication:**

GraphRAG should be selected and configured based on task type, graph quality, retrieval granularity, and computational cost.

[1] https://arxiv.org/abs/2502.11371, [2] https://arxiv.org/abs/2509.16780
[3] https://arxiv.org/abs/2411.05844

# AriGraph: Learning Knowledge Graph World Models with Episodic Memory for LLM Agents [IJCAI'25]

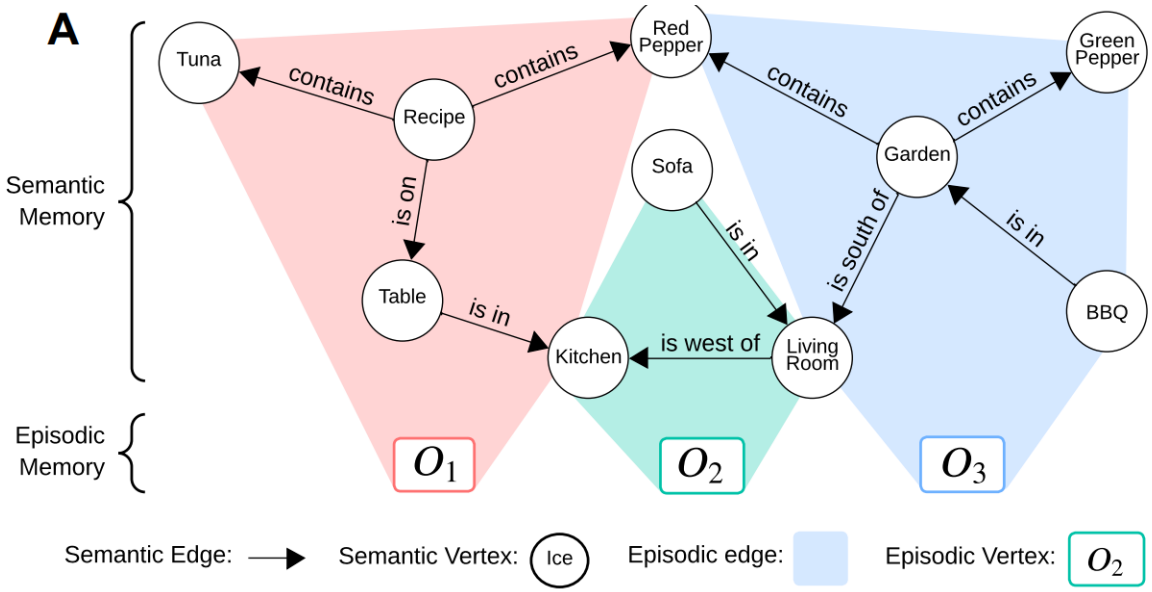
**Motivation:**
LLM agents need long-term memory to accumulate and reuse knowledge from past interactions.




**Key Idea:** AriGraph represents agent memory as a graph world model that combines:

- Semantic memory: accumulated general knowledge
- Episodic memory: specific past observations and events
- Graph-based retrieval for planning and decision-making

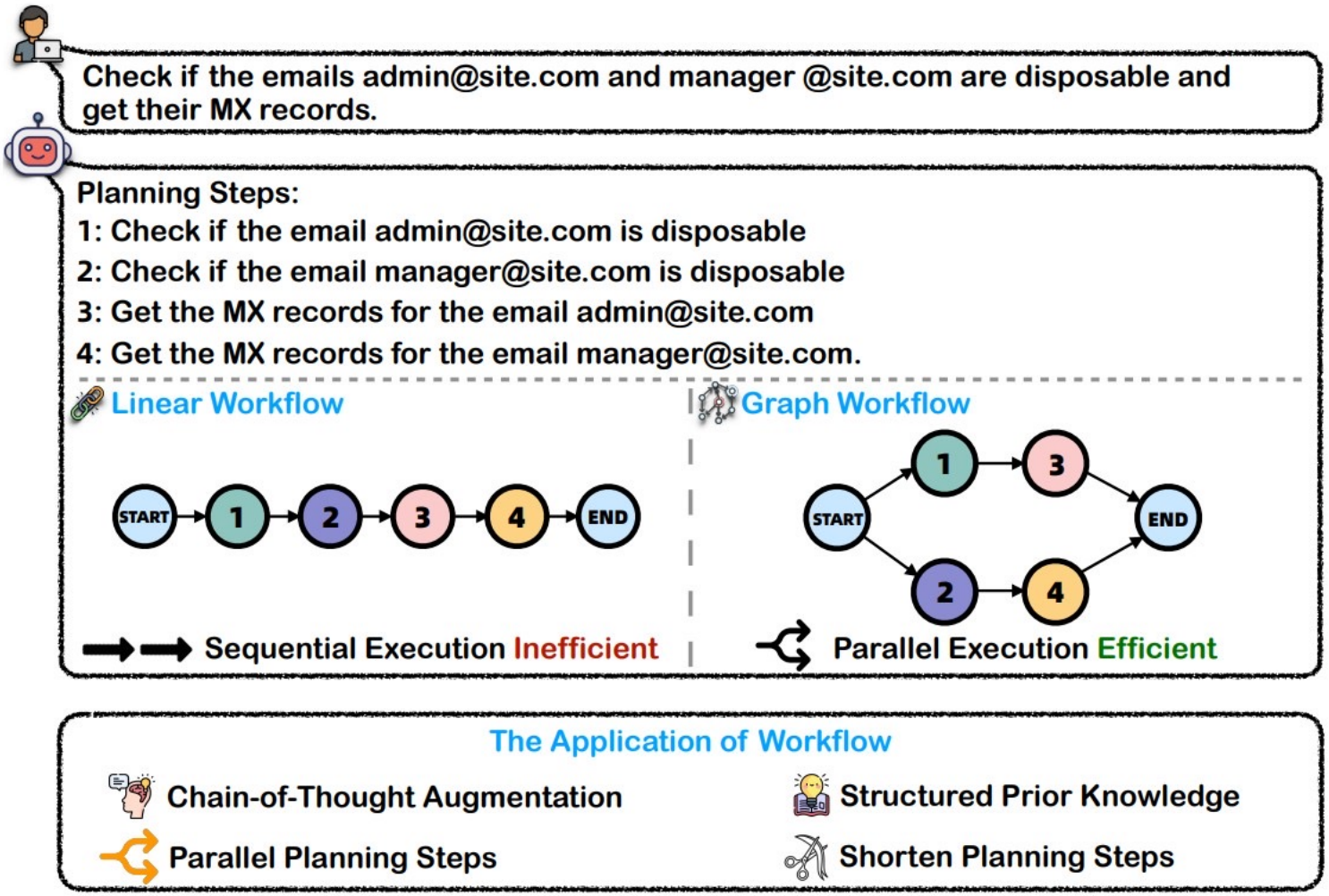
# AriGraph: Learning Knowledge Graph World Models with Episodic Memory for LLM Agents [IJCAI'25]




At each step:

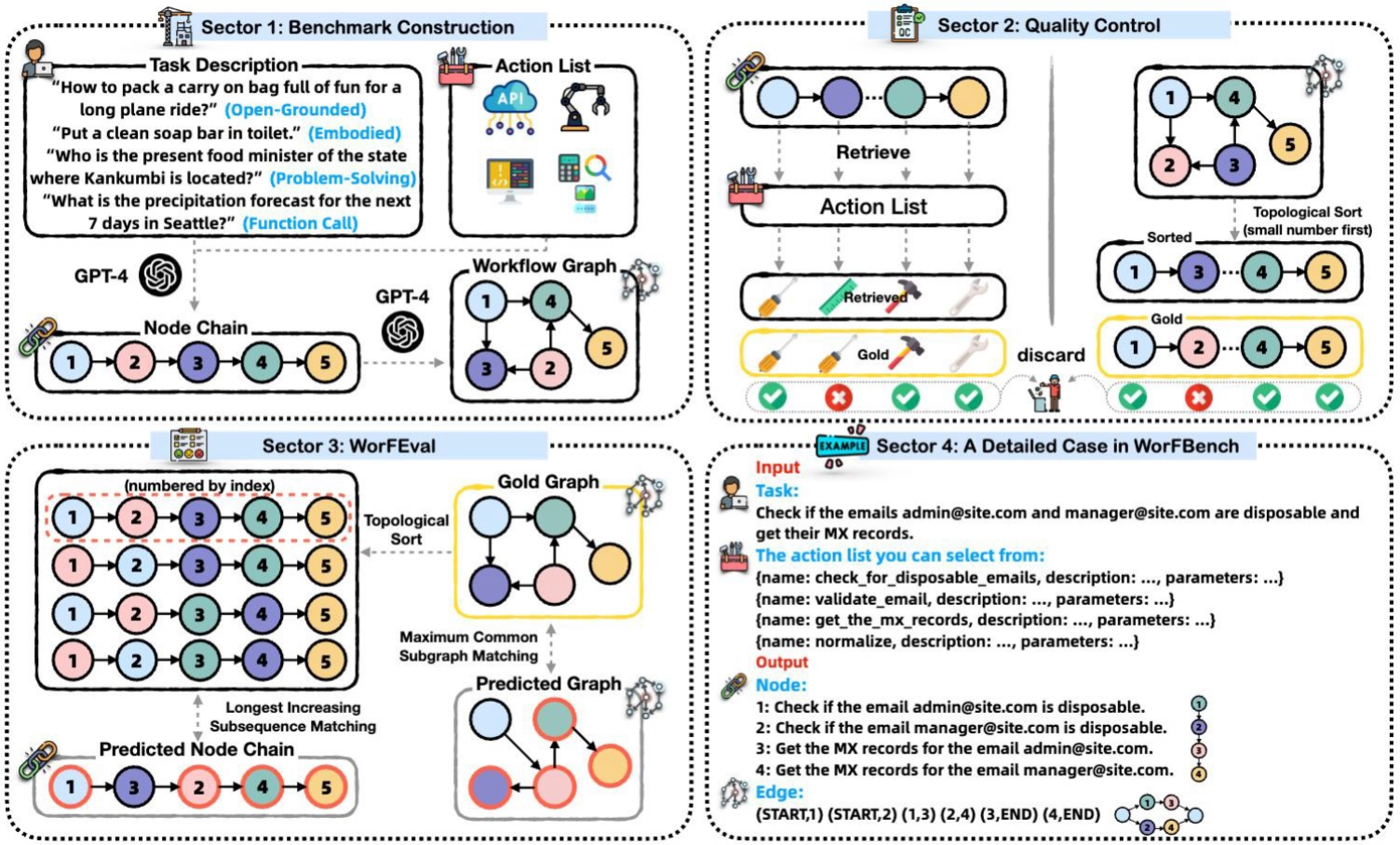
1. The agent receives a textual observation.
2. LLM extracts semantic triplets: (object1, relation, object2)
3. Triplets update the semantic memory graph.
4. The original observation is stored as an episodic vertex.
5. Episodic edges link the observation to the extracted triplets.

# Benchmarking Agentic Workflow Generation [ICLR 2025]




[1] https://arxiv.org/pdf/2410.07869

# Benchmarking Agentic Workflow Generation [ICLR 2025]




- Multi-faceted scenarios.
- Complex workflow structures.
- Strict quality control and data filtering.
- Accurate quantitative evaluation (WorfEval).

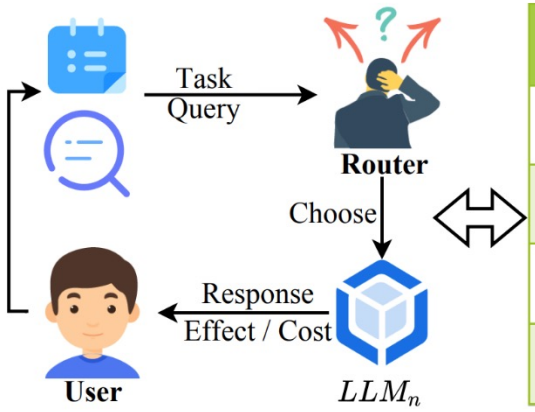
[1] https://arxiv.org/pdf/2410.07869

# Benchmarking Agentic Workflow Generation [ICLR 2025]

| Model | Function Call | | Problem-Solving | | Embodied | | Open-Grounded | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $f1_{\text{chain}}$ | $f1_{\text{graph}}$ | $f1_{\text{chain}}$ | $f1_{\text{graph}}$ | $f1_{\text{chain}}$ | $f1_{\text{graph}}$ | $f1_{\text{chain}}$ | $f1_{\text{graph}}$ | $f1_{\text{chain}}$ | $f1_{\text{graph}}$ |
| ***Closed-Sourced*** | | | | | | | | | | |
| Claude-3-5-sonnet-0620 | 65.58 | 51.93 | 62.44 | 45.45 | 65.04 | 48.47 | **61.73** | 41.49 | 63.70 | 46.84 |
| Claude-3-5-sonnet-1022 | 66.44 | 55.06 | 67.28 | 55.50 | **71.74** | **56.71** | 61.33 | **42.88** | 66.70 | **52.53** |
| GPT-3.5 | 73.36 | 60.32 | 69.86 | 54.50 | 64.57 | 49.17 | 47.67 | 28.10 | 63.86 | 48.02 |
| GPT-4 | **74.87** | **62.11** | 67.18 | 55.24 | 70.94 | 56.17 | 56.30 | 36.36 | **67.32** | 52.47 |
| O1-preview | 70.68 | 57.11 | **72.76** | **59.25** | 69.90 | 54.19 | 53.47 | 35.97 | 66.70 | 51.63 |
| ***Open-Sourced*** | | | | | | | | | | |
| GLM-4-9B | 59.27 | 36.34 | 58.91 | 40.15 | 53.17 | 36.15 | 44.04 | 22.56 | 53.85 | 33.80 |
| Phi-3-small | 57.66 | 40.71 | 55.76 | 39.75 | 54.77 | 37.52 | **44.65** | 22.66 | 53.21 | 35.16 |
| Llama-3.1-8B | 63.30 | 43.62 | 64.49 | 46.79 | 56.23 | 36.40 | 44.58 | **25.48** | 57.15 | 38.08 |
| Mistral-7B | 67.30 | 51.67 | 61.27 | 45.35 | 64.59 | **48.83** | 40.97 | 21.48 | 58.53 | 41.83 |
| Qwen-2-7B | **70.79** | **55.50** | 68.65 | 52.13 | 62.83 | 46.25 | 39.29 | 20.89 | 60.39 | 43.69 |
| InternLM-2.5-7B | 68.43 | 52.99 | **72.92** | **57.80** | **65.77** | 48.09 | 40.84 | 21.27 | **61.99** | **45.03** |
| Llama-2-13B | 53.32 | 34.33 | 53.74 | 38.69 | 44.27 | 30.55 | 37.17 | 23.14 | 47.12 | 31.68 |
| WizardLM-13B | 55.78 | 36.94 | 65.42 | 49.71 | 55.41 | 37.34 | 37.23 | 21.66 | 53.46 | 36.41 |
| Vicuna-13B | 53.75 | 37.66 | 64.58 | 50.25 | 57.99 | 42.61 | 38.93 | 23.11 | 53.81 | 38.41 |
| Qwen-1.5-14B | 65.73 | 46.86 | 58.80 | 43.89 | 60.55 | 44.14 | 41.73 | 21.44 | 56.70 | 39.08 |
| Phi-3-medium | **67.71** | **47.26** | **71.15** | **54.85** | **65.11** | **49.99** | **42.73** | **23.77** | **61.68** | **43.97** |
| WizardLM-70B | 63.47 | 45.46 | 63.92 | 47.93 | 59.15 | 42.87 | 45.27 | 26.89 | 57.95 | 40.79 |
| Mixtral-8×7B | 66.13 | 48.83 | **71.89** | 57.58 | 72.08 | 54.94 | 42.96 | 23.21 | 63.26 | 46.14 |
| Llama-3.1-70B | 64.41 | **52.72** | 70.37 | 57.05 | 69.98 | 55.52 | **53.64** | **33.06** | 64.60 | 49.59 |
| Qwen-2-72B | **71.67** | 52.31 | 70.63 | **58.13** | **73.24** | **58.49** | 53.43 | 32.89 | **67.24** | **50.46** |

[1] https://arxiv.org/pdf/2410.07869

# GraphRouter: A Graph-based Router for LLM Selections [ICLR 2025]




| Task | Query | LLM | Response | Performance | Cost |
|---|---|---|---|---|---|
| Alpaca | Give three tips for staying healthy? | LLaMA-3 (7b) | Nutrition plays... | 0.19 | 103.8 |
| Alpaca | What are the three primary colors? | LLaMA-3 (70b) | Red, blue, and yellow. | 0.21 | 467.1 |
| GSM8K | How many musicians total...? | LLaMA-3 (7b) | There are 55... | 0.00 | 230.1 |
| GSM8K | **How many seats are there...?** | **?** | ... | ? | ? |

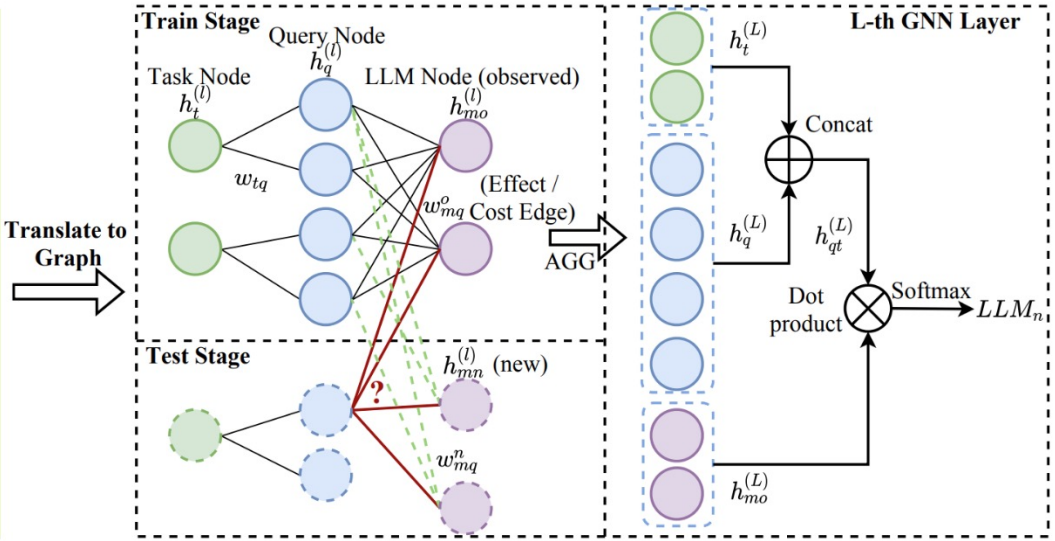
**LLM Selection Process:**

a) User input associated with a specific task
b) Router analyzes the input and selects the appropriate LLM
c) Selected LLM generates the final output
d) Evaluation of the output's effectiveness and cost
e) Feedback training loop for continuous improvement

[1] https://arxiv.org/pdf/2503.04338

# GraphRouter: A Graph-based Router for LLM Selections [ICLR 2025]

| Task | Query | LLM | Performance | Cost |
|---|---|---|---|---|
| Alpaca | $Q_1$ | LLaMA-3 (7b) | 0.19 | 103.8 |
| Alpaca | $Q_2$ | LLaMA-3 (70b) | 0.21 | 467.1 |
| GSM8K | $Q_3$ | LLaMA-3 (7b) | 0.00 | 230.1 |
| GSM8K | $Q_4$ | ? | ? | ? |

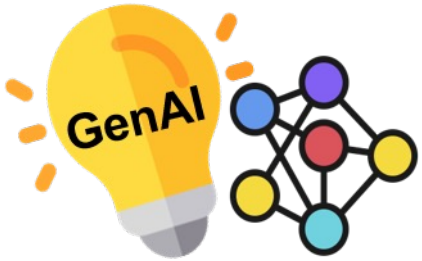



## Overall Framework:

GraphRouter converts the interaction data among tasks, queries, and LLMs into a graph by representing each as nodes and their relationships as edge features. Then, it uses a GNN to generate the probability distribution for selecting the appropriate LLM.

[1] https://arxiv.org/pdf/2503.04338

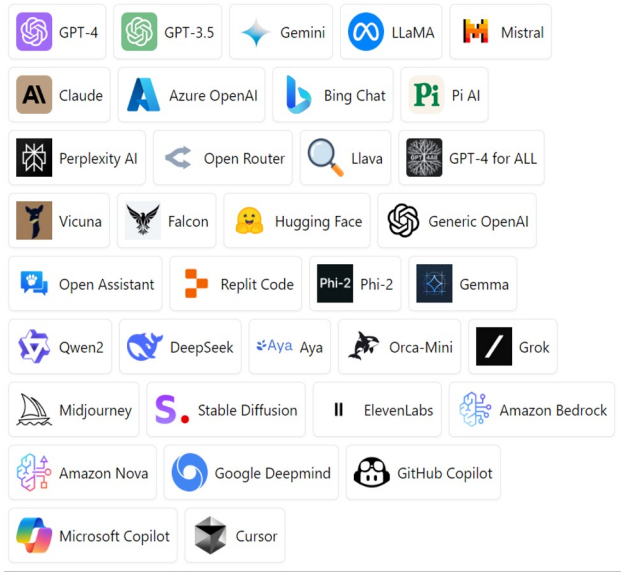
# 4. KGs for LLMs




- Large Language Models
- Knowledge Graphs
- KGs as background knowledge for LLMs
- KGs as reasoning guidelines for LLMs
- KGs as refiners and validators for LLMs
- LLM+KG for domain-specific applications (e.g., data integration)

Presented by Arijit Khan

# LLMs - Introduction



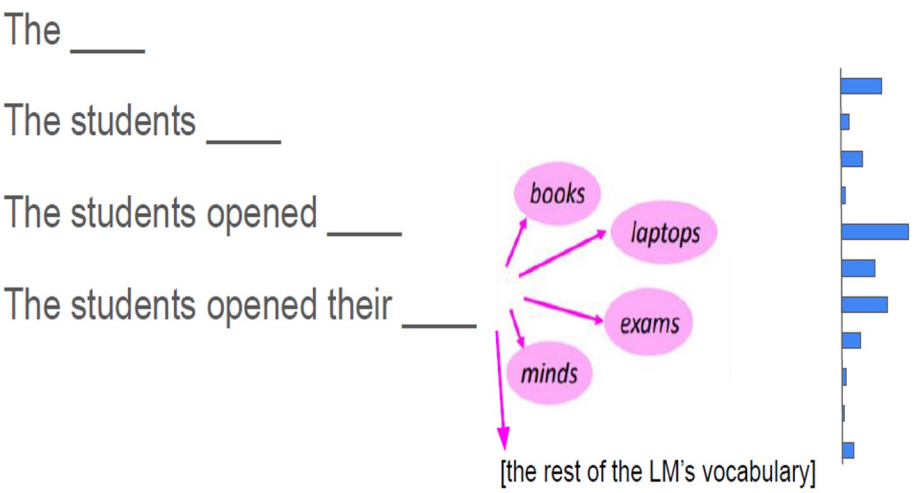

**LLM models**

- **Generative AI (genAI):** AI system whose primary task is to generate content, e.g., GANs, VAEs, RNNs, LLMs, VLMs, VALL-E.
- **Large Language Models (LLMs):** Generative AI systems primarily designed for natural language processing tasks.
- **Foundation Models (FMs):** AI systems serving as the basis for a wide range of AI applications - can be adapted to a range of different, more specific purposes. E.g., LLMs, VLMs, speech FMs.

- often used interchangeably.

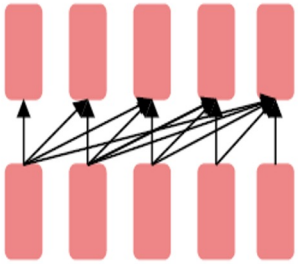
- Models the probability of the next word, given the history (context) of preceding words.

$$p(w) = p(w_1) \times p(w_2|w_1) \times p(w_3|w_1, w_2) \times p(w_l|w_1, \ldots, w_{l-1})$$
$$= \prod_{t=1}^{|w|} p(w_t|w_1, \ldots, w_{t-1})$$



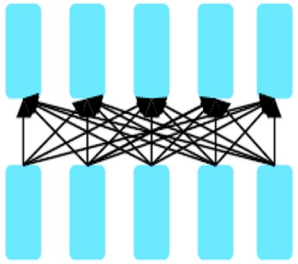

**Language models text generation**

# PLMs and LLMs Architecture



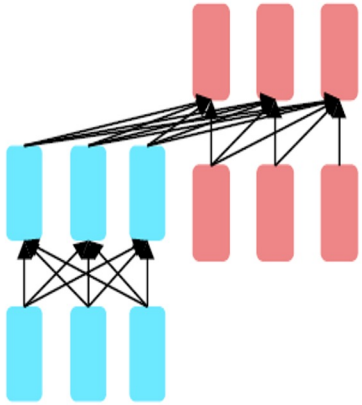
## Decoders

- GPT, Claude, Llama, ..
- Text generation
- Emergent properties (text classification, summarization, translation, question answering, and diverse tasks)
- New tasks without updating model parameters via prompt-based in-context learning and retrieval augmented generation (RAG)



## Encoders

- BERT, RoBERTa, ..
- Text comprehension (sentiment analysis, text classification, question-answering, and named entity recognition)



## Encoder-decoders

- BART, T5 , ..
- Both text comprehension and generation (machine translation, summarization, and question answering)

# LLMs – Benefits


**Applications of LLMs**

Emerging abilities; generalizing to unseen tasks; task descriptions provided as text.

Scaling the models, compute, and data leads in increase in performance.

Perform new and creative tasks using prompt-based Interaction and retrieval-augmented generation (RAG) without updating model parameters.

✓ LLM pipelines remove task-specific supervision and need for labeled data – easy to use, less expensive, and fast to iterate.

✓ LLMs act as knowledge bases - can be probed for QA and querying tasks.

# LLMs – Challenges

- **Alignment Problem:** LLMs may produce harmful, unsafe, toxic, or undesirable outputs – inappropriate language, misinformation, bias, and discrimination.

- **Hallucination:** Parametric knowledge, lack consistent representations of knowledge, fail to understand a question due to lack of context, knowledge gap (lack up-to-date and domain-specific knowledge), cannot recall facts (about not so popular or long-tail entities), output unreliable and incoherent responses, hallucinate by generating factually incorrect statements.

- **Lack of Consistency:** Generate logically contradicting outputs, low semantic similarity of LLM outputs due to paraphrased versions of a question (meaning-preserving text alternations), violate important relational properties such as negation, symmetry, and transitivity; Adversarial LLM Jailbreaks.

- **Privacy Concern:** Data privacy, personally identifiable information, data retention policy, IP leakage, security vulnerabilities, legal compliance.

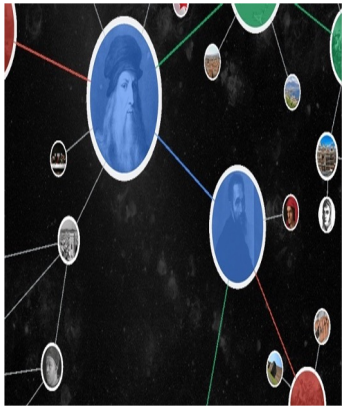
- **Black-box Model:** Many LLMs are proprietary and little information is released about them. Difficult to explain LLM predictions with billions of parameters. Knowledge in LLMs is hard to interpret, update, and is prone to bias. Challenging to deploy LLMs in decision-critical applications.

- **Environmental Concern:** High cost, energy consumption, carbon emissions, and water usage.

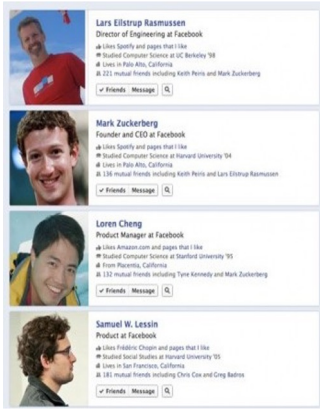
- **Societal Impacts:** Job loss, disparities, phishing, fraud, manipulation, plagiarism, cheating, fake news, big tech monopolies, societal unrest, …

# KGs – Introduction

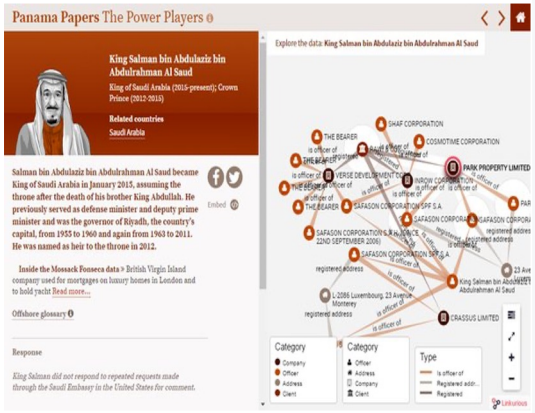
- Integrating knowledge + data at large scale ⇒ Knowledge graph



**Google Knowledge Graph** (2012)

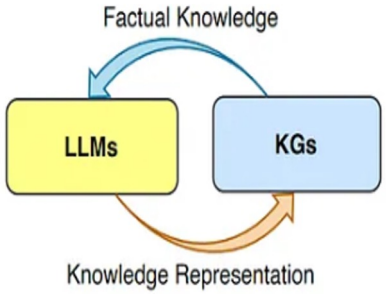



**"People who like things I like" Facebook graph search** (2013)



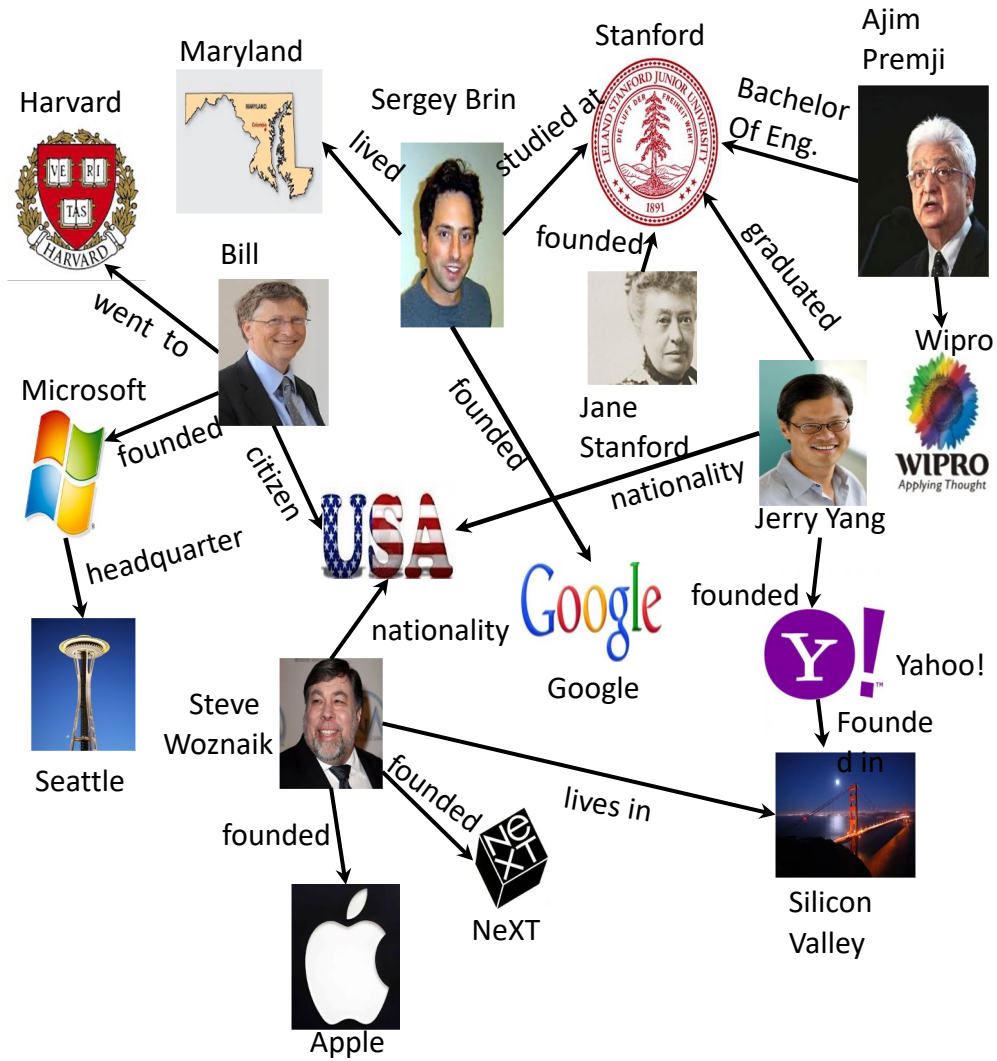

**Panama Papers investigation, led by ICIJ, exposed highly connected networks of offshore tax structures used by world's richest elites** (2016)


**In 2020, Gartner put Knowledge Graphs at the peak of its AI hype cycle** (2020)




**LLM + KG** (2024 - present)

# KGs – Data Sources and Categories




Freebase

ConceptNet 5

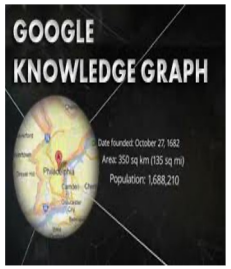
Microsoft Academic

DBpedia

WebChild

Product Graph

Event2Mind




Commonsense KG

ClaimsKG

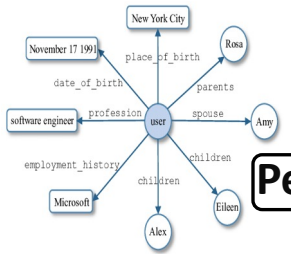
Domain-specific KG

WIKIDATA


Industrial KG

yago
select knowledge

Factual KG




Personal KG (PKG)

# Knowledge Graphs (KGs) – Components, Representation, and Usage

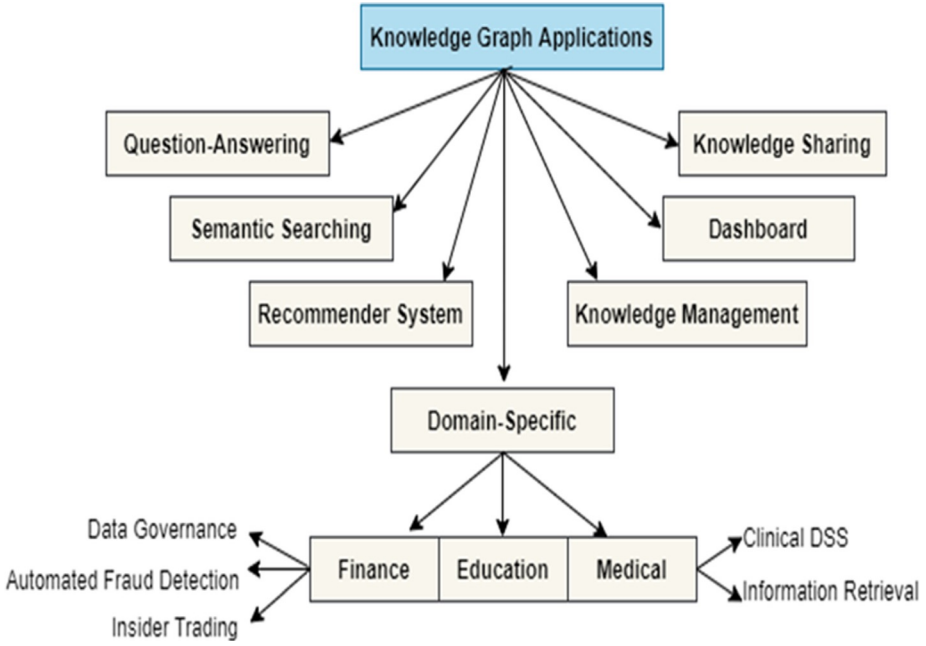
## KG Components

- **Nodes:** entities, concepts, or instances within a domain, e.g., people, places, organizations, concepts, events, etc.
- **Edges:** relationships and connections between nodes, e.g., Is-a relationship, Part-of relationship, Related-to relationship, etc.
- **Properties:** additional descriptive information and metadata associated with nodes or edges, e.g., attributes, features, labels, Qualifiers, metadata, etc.
- **Ontology:** schema and vocabulary used within the KG, providing a structured framework for representing domain knowledge.

## KG Representations

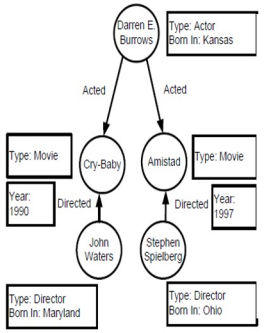
- **RDF Triples:** collection of <subject, predicate, object> triples.
- **Property Graph:** graph model having nodes and edges with arbitrary number of properties, where a node (a subject or an object) denotes an entity and a directed edge (a predicate) is a relationship between two entities.
- **KG Embedding:** Vector representation of KG nodes and edges in low-dimensional space, such that the original structures and relations in the KG are preserved in these learned semantic vectors.

## KG Applications

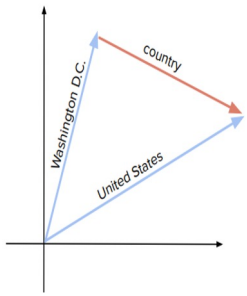



```
Person1 isNamed "John Waters"
Person2 isNamed "Stephen Spielberg"
Person3 isNamed "Darren E. Burrows"
Movie1 hasTitle "Cry-Baby"
Movie1 hasActor Person3
Movie1 hasDirector Person1
Movie2 hasTitle "Amistad"
Movie2 hasActor Person3
Movie2 hasDirector Person2
```

**RDF Triples**




**Property Graph**




**KG Embedding**

# KGs – Benefits and Challenges

**KG Benefits**

structured, highly curated, and reliable representation of knowledge via explicit relationships.

support symbolic reasoning and inference, with answer validation and explainability.

✓ schema-flexible: updated dynamically with new knowledge via addition or deletion of triples/ nodes & edges.

✓ offer accurate explicit knowledge in many downstream applications, e.g., web search, QA, semantic search, personal assistants, fact-checking, and recommendation.

**KG Challenges**

- **Difficult to construct**.
- **Difficult to query** due to incompleteness, schema-flexibility, heterogeneity, and massive-scale.
- **User Barriers in SPARQL/Cypher-Style Querying:** non-professional users find it challenging to write an accurate query, e.g., via SPARQL, Cypher, Gremlin, GSQL, etc., since users must have full knowledge of the query language, schema, and the vocabulary used in a KG. Current KG querying approaches generally lack language understanding, are inadequate to deal with unseen entities and new facts, and often ignore multi-modal information in KGs.
- **Interoperability issue**: existing methods are tailored for specific KGs or downstream tasks.

# LLMs+ KGs: Synergy

**KG for LLM**

KGs offer external knowledge (up-to-date, domain specific, and symbolic knowledge) for enhancing the accuracy, consistency, transparency, and the overall capabilities of LLMs.

KG-enhanced pre-training, fine-tuning, and inference (KG-RAG).

KG-enhanced validation (LLM guardrail) and explainability.

**LLM for KG**

LLMs augment KGs via knowledge extraction, auto-completion, and incorporating multi-modal information, enhancing usability and performance of downstream tasks with natural language understanding and generalization capabilities.

LLM-enhanced KG creation and completion.

LLM-enhanced KG embedding.

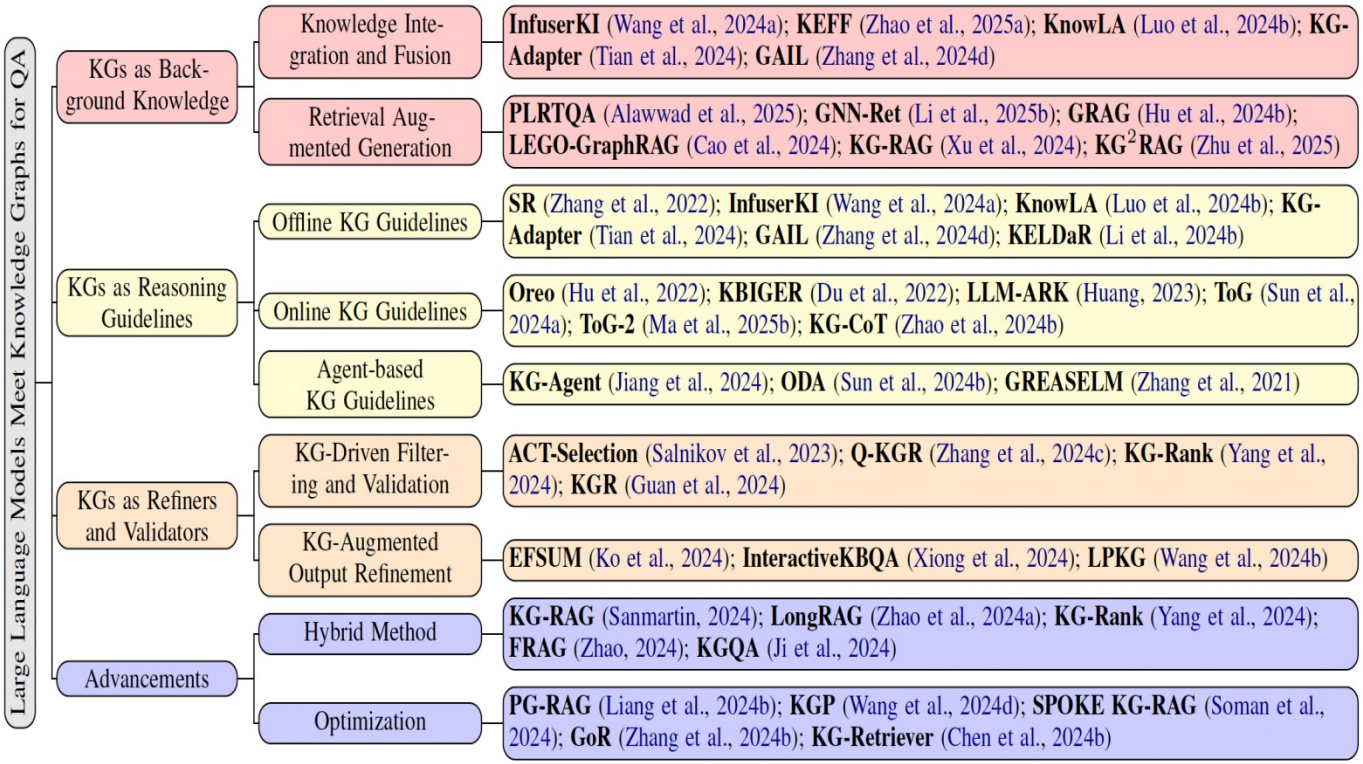
✓ LLM-enhanced KG querying, analytics, and domain-specific applications.

**LLM+KG**

Downstream applications benefit from the complementarity of LLMs and KGs – LLMs and KGs offer parametric vs. explicit knowledge, respectively.

# Role of KGs in Synthesizing with LLMs

- Our Recent Survey
  - LLMs Meet Knowledge Graphs for Question Answering: Synthesis and Opportunities [EMNLP Main 2025] https://github.com/machuangtao/LLM-KG4QA




Chuangtao Ma, Yongrui Chen, Tianxing Wu, **Arijit Khan**, and Haofen Wang, "Large Language Models Meet Knowledge Graphs for Question Answering: Synthesis and Opportunities", in Conference on Empirical Methods in Natural Language Processing 2025 (Main).

# Question Answering (QA): Introduction

- **QA:** QA deals with answering unstructured natural language questions (NLQs) – it also includes a natural language understanding task.

**QA Categories**

- **Simple vs. Complex Questions:**
  - ✓ Simple question → a single triple and a single relation, e.g., "*where was Albert Einstein born?*" can be answered based on the relation 'born': <Albert Einstein, born, ?place>.
  - ✓ Complex question → multiple KG relations (multi-hop) and/or additional operations (e.g., aggregate, order, temporal), e.g., *"what was the first movie of James Cameron that own an Oscar?"*

- **Multi-document QA**
- **Multi-lingual QA**
- **Multi-modal QA**
- **Multi-run and conversational QA**
- **Temporal QA**
- **Factoid QA**
- **Explainable QA**

**QA Applications**

- Text generation, chatbots, dialog generation, web search, entity linking, natural language query, fact-checking, …
- Open-domain QA, domain-specific QA
- AI, NLP, information retrieval, and data management

# LLM+KG for QA: Motivation and Challenges

- PLMs & LLMs for QA based on their pre-trained knowledge and natural language understanding capabilities [35]

**Challenges of PLMs and LLMs in QA**

- Limited reasoning ability for complex QA
- Lack of up-to-date and domain-specific knowledge
- Hallucination and inconsistency

**Motivation of KGs+LLMs in QA**

KGs can offer external, precise, up-to-date, and domain-specific knowledge to LLMs via pre-training, fine-tuning, and RAG (Graph RAG, KG-RAG)

✓ Improve LLM's accuracy and consistency

✓ Support answer validation (LLM guardrail) and explainability.

**Challenges of KGs+LLMs in QA**

- Knowledge conflict
- Poor relevance and quality of retrieved data, limited context size of LLMs
- Large-scale and dynamic KGs
- Lack of iterative and multi-hop reasoning:

# LLM+KG for QA: Roles of KG in Complex QA

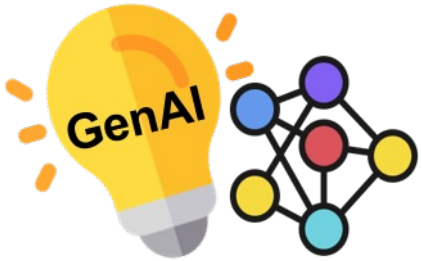
| Approach | Strength | Limitation | KG Requirement |
|---|---|---|---|
| KG as Background Knowledge | Broad Coverage | Static Knowledge | High Domain Coverage |
| KG as Reasoning Guidelines | Multi-hop Capabilities | Computational Overhead | Rich Relational Paths |
| KG as Refiners and Validator | Hallucination Reduction | Validation Latency | High Accuracy & Recency |

**Comparison of Approaches with Different Roles of KG**

**Alignment of Approaches to Complex QA with Different Roles of KG**

| Approach | Multi-doc QA | Multi-modal QA | Multi-hop QA | Multi-run QA | XQA |
|---|---|---|---|---|---|
| KG as Background Knowledge | ✔ | ✔ | ✔ | ✔ | X |
| KG as Reasoning Guidelines | ✔ | ✔ | ✔ | X | ✔ |
| KG as Refiners and Validator | X | X | ✔ | ✔ | X |
| Hybrid Methods | ✔ | ✔ | ✔ | ✔ | ✔ |

# Relevant Tutorials




## QA, LLMs, KG


- Danqi Chen and Wen tau Yih. 2020. Open-domain question answering. In ACL. 34–37.

- Lihui Liu, Zihao Wang, Jiaxin Bai, Yangqiu Song, and Hanghang Tong. 2024. New frontiers of knowledge graph reasoning: Recent advances and future trends. In WWW. 1294–1297.

- Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2024. Large language models for recommendation: Progresses and future directions. In WWW Companion (2024). 1268–1271.


## LLMs+KGs/Graphs, RAG


- Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. In SIGKDD. 6491–6501.

- Chao Huang, Xubin Ren, Jiabin Tang, Dawei Yin, and Nitesh Chawla. 2024. Large language models for graphs: Progresses and directions. In WWW. 1284-1287.

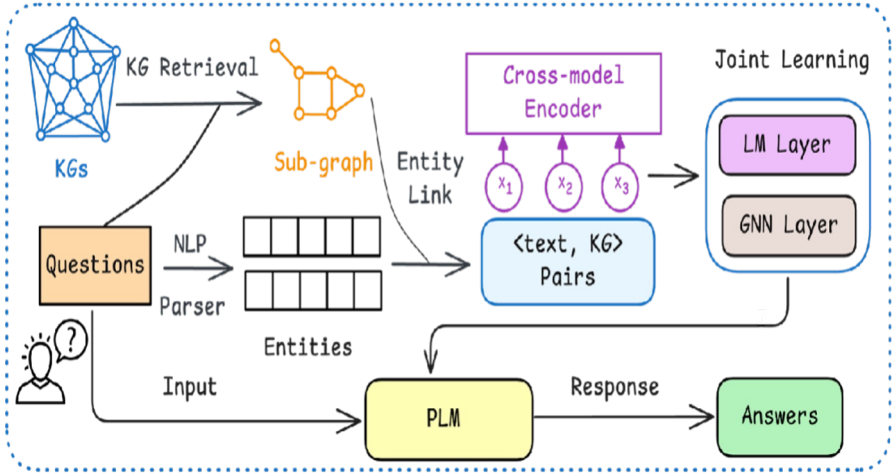
- Qiang Zhang, Jiaoyan Chen, Zaiqiao Meng. 2024. Integrating Knowledge Graphs and Large Language Models for Advancing Scientific Research. Learning on Graph Conference (LoG).

- Chuangtao Ma, Yongrui Chen, Tianxing Wu, Arijit Khan, and Haofen Wang, "Unifying Large Language Models and Knowledge Graphs for Question Answering: Recent Advances and Opportunities ", in EDBT 2025.

# KGs as Background Knowledge

- KGs and Text Alignment: **Joint Learning** with PLMs
  - Where are KGs?
    - KGs and text data are stored separately (Common scenario for QA task)
    - KGs (entities or relations) having the textual description (Text-KG pair)

| Entity Linking |
| --- |
| KG Retrieval |

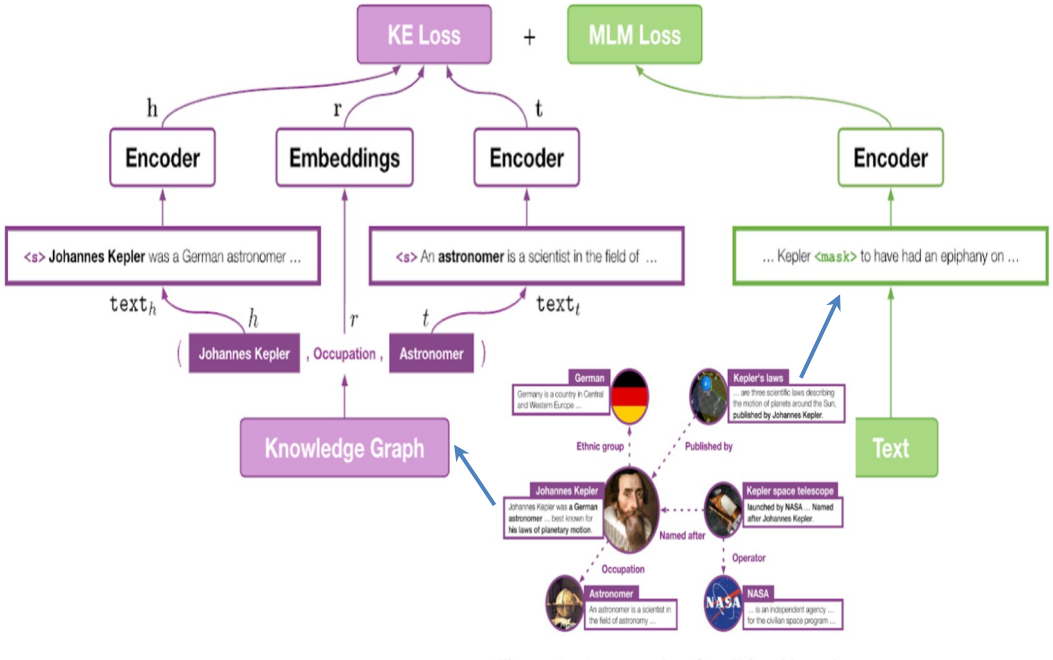



Joint learning

# KGs as Background Knowledge

- Knowledge Integration and Fusion
  - **Joint Learning:** Unified representation for KG and PLM [ACL2021]

    - Encode textual description of entity as entity embeddings and jointly train the KE and MLM on the same PLM



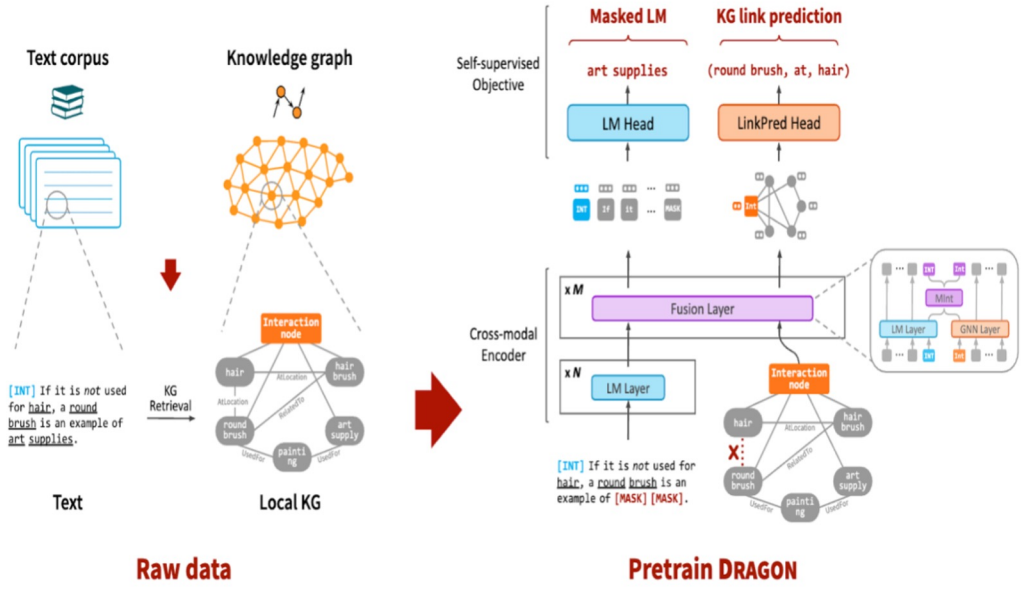

$$\mathcal{L} = \mathcal{L}_{\mathrm{KE}} + \mathcal{L}_{\mathrm{MLM}},$$

Joint loss for knowledge embeddings and masked language model

Wang, X., et al. A unified model for knowledge embedding and pre-trained language representation. ACL, 2021, 9: 176-194.

# KGs as Background Knowledge

- Knowledge Integration and Fusion
  - **Joint Learning:** Bidirectional language and KG pretraining [NeurIPS2022]
    - Retrieving relevant subgraph from KG based on text to create text-KG pair.
    - Leveraging cross-modal encoder that fuses the input **text-KG pair** bidirectionally.
    - Unifying masked LM and KG link prediction for and joint learning reasoning.



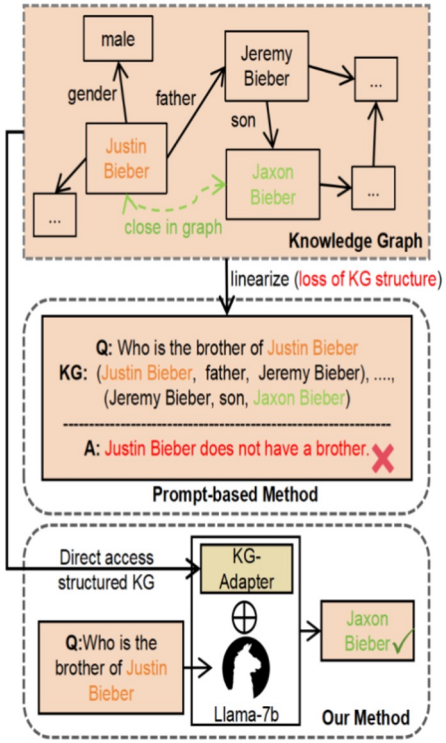

Modality interaction (Mint) with interaction token and node to mix representation for joint learning

$$[\boldsymbol{h}_{int}^{(\ell)}; \boldsymbol{e}_{int}^{(\ell)}] = \text{MInt}([\tilde{\boldsymbol{h}}_{int}^{(\ell)}; \tilde{\boldsymbol{e}}_{int}^{(\ell)}]),$$

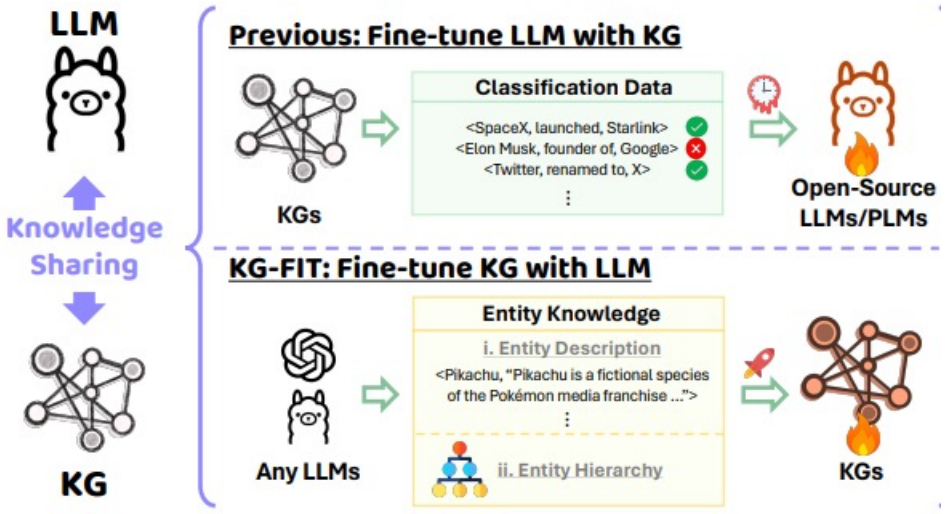
Yasunaga, Michihiro, et al. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems* 35 (2022): 37309-37323.

# KGs as Background Knowledge

- Knowledge Integration and Fusion
  - **Fine-tuning:** Parameter-efficient fine-tuning (KG-Adapter) [ACL 2024]
  - Only ~28M adapter parameters are trained, while the 7B-parameter LLM remains frozen.




- KG-FIT (NeurIPS 2024): fine-tunes the KG itself using open-world knowledge extracted from LLMs.



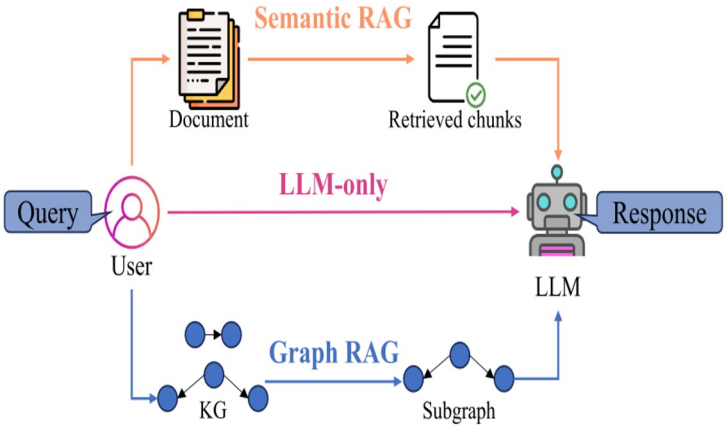


Shiyu Tian et al. KG-Adapter: Enabling Knowledge Graph Integration in Large Language Models through Parameter-Efficient Fine-Tuning. ACL 2024.

Jiang, Pengcheng, et al. KG-FIT: Knowledge graph fine-tuning upon open-world knowledge. *Advances in Neural Information Processing Systems* 37 (2024): 136220-136258.

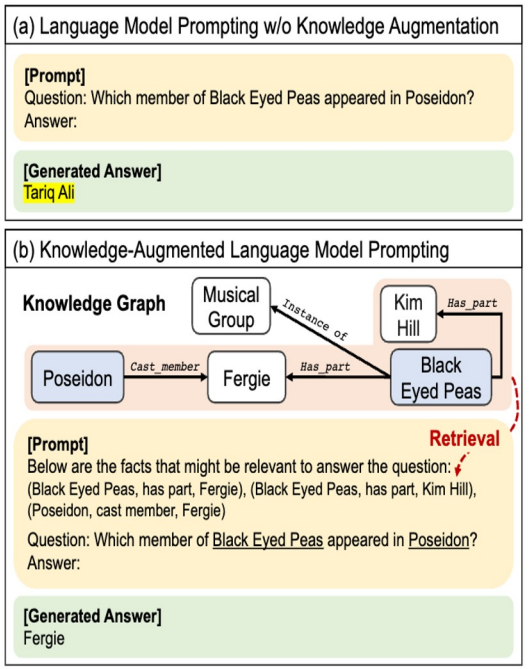
# KGs as Background Knowledge

- **Retrieval Augmented Generation (RAG)**: from PLM to LLM
  - Semantic RAG: retrieve document or chunks with **limited reasoning abilities**
  - KG-RAG: retrieve subgraph (triples) from KGs with factual-based relationships




LLM *vs* RAG *vs* Graph RAG

**How to fuse and align LLM and KG ?**



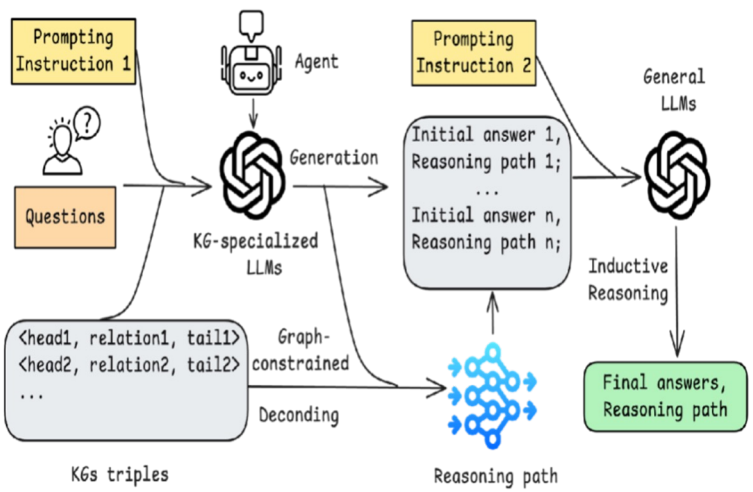

Prompt-based Augmentation

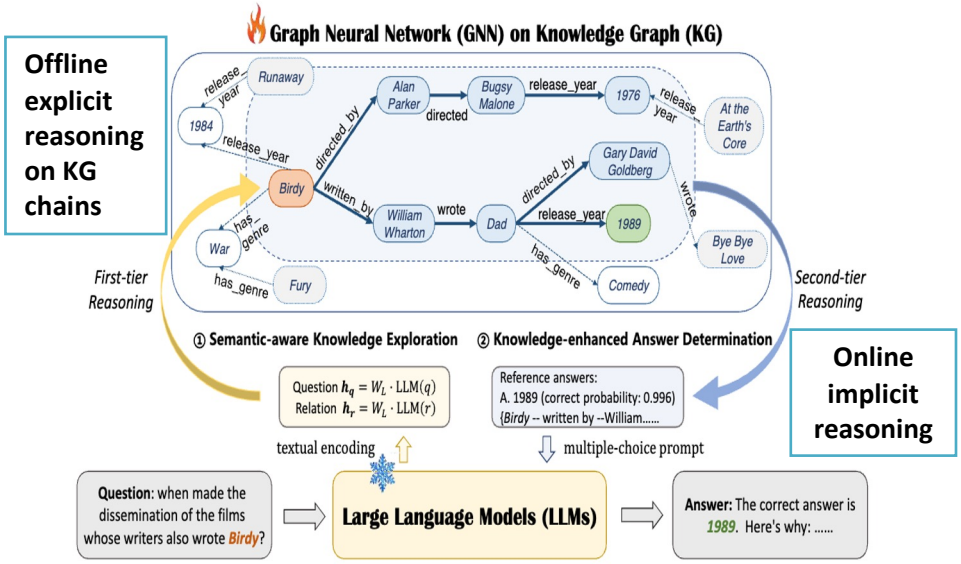

Xiangrong, Zhu, et al. Knowledge Graph-Guided Retrieval Augmented Generation. *arXiv preprint arXiv:2502.068641* (2025).

Baek, Jinheon, et al. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136* (2023).

# KGs as Reasoning Guidelines

- KGs serves as reasoning guidelines to LLMs for QA
  - **Offline KG guidelines:** KGs-based reasoning before LLMs reasoning
  - **Online KG guidelines:** KGs-based reasoning directly involves in LLMs reasoning
  - **Agent-based KG guidelines:** Agent-based autonomous reasoning

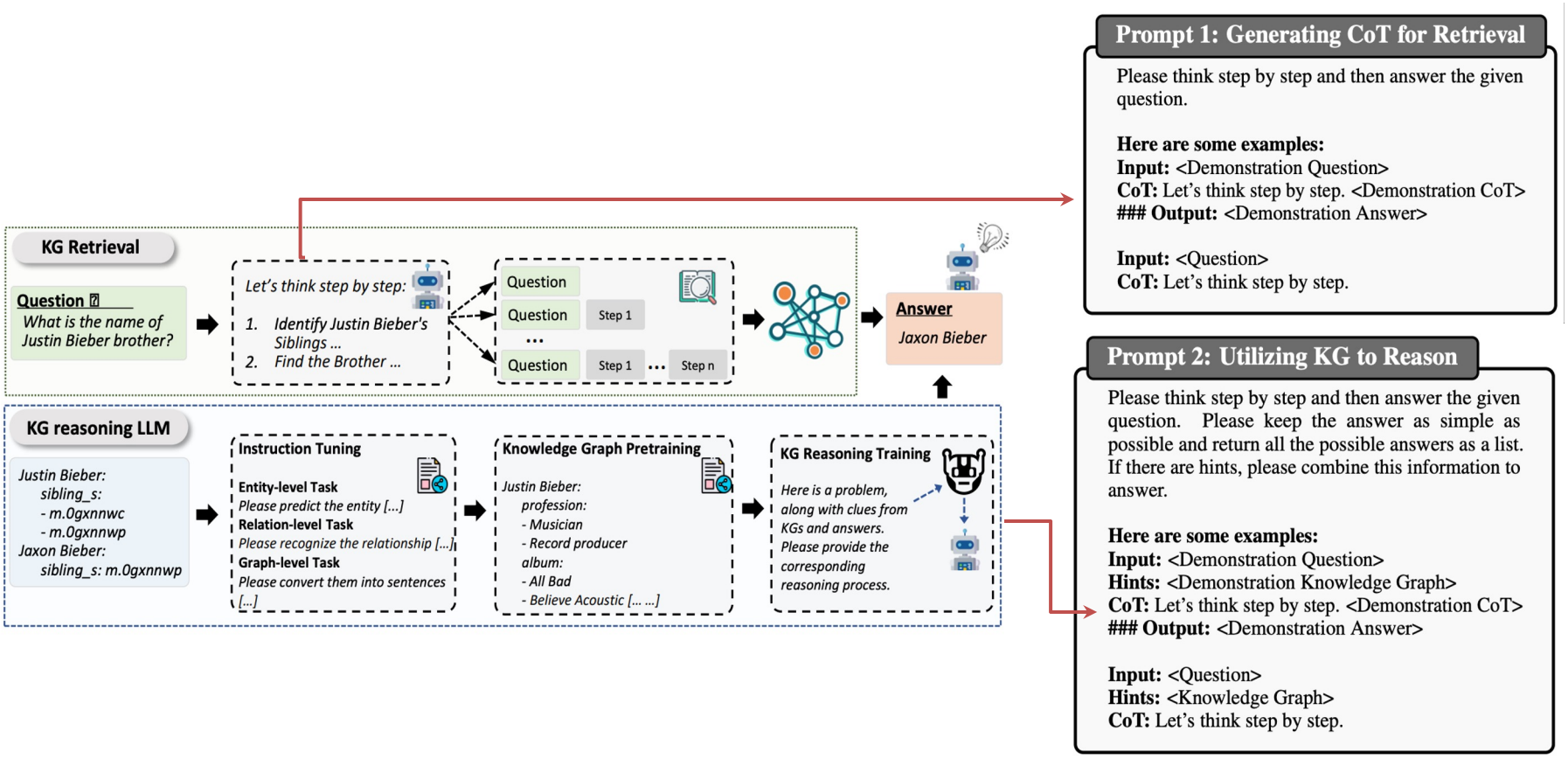



Liu, Guangyi, et al. Dual Reasoning: A GNN-LLM Collaborative Framework for Knowledge Graph Question Answering. CPAL, (2025).

# KGs as Reasoning Guidelines

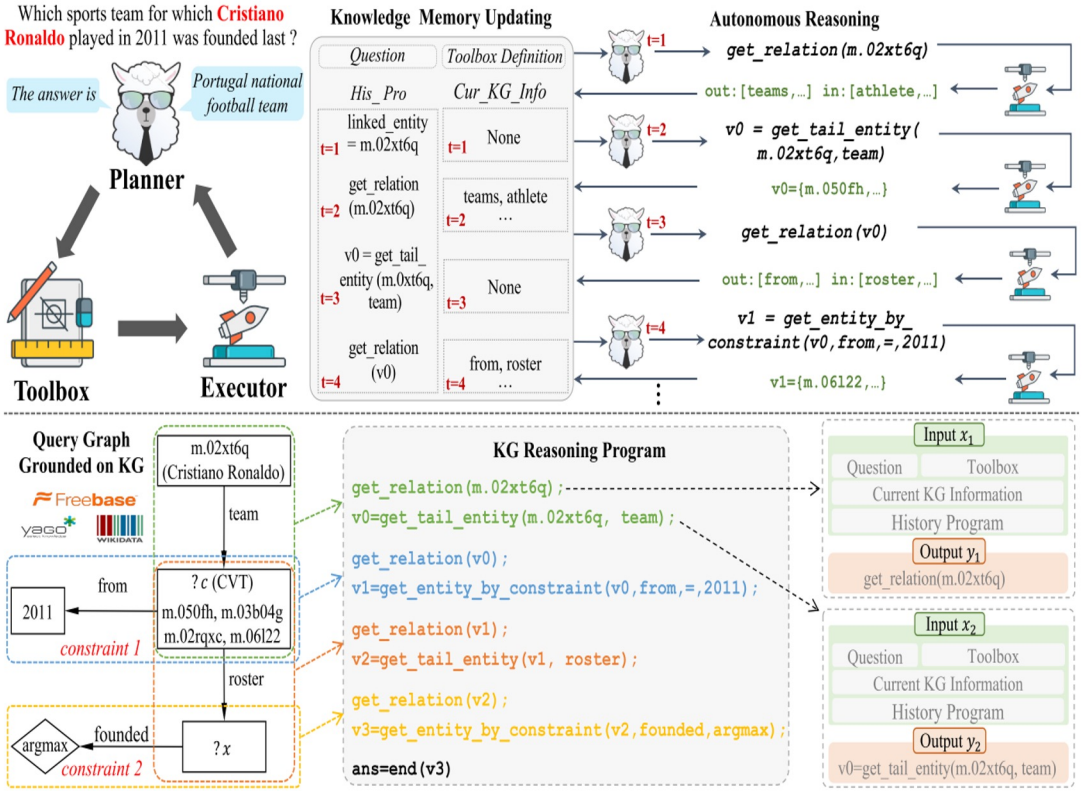
- Online KG Guidelines
  - KG-based CoT Reasoning for KGQA [EMNLP, 2024]
    - Integrate the reasoning process and subgraphs into knowledge retrieval
    - Employ instruction tuning and continual pre-training to learn the KG reasoning




Ji, Yixin, et al. Retrieval and reasoning on KGs: Integrate knowledge graphs into large language models for complex question answering. *EMNLP*. (2024).

# KGs as Reasoning Guidelines

- Agent based Reasoning
  - KG-Agent: Agent-based autonomous reasoning for KGQA [ACL 2025]




**Instruction-tuned LLM for planner**

**KG-based executor for knowledge updating**

**Agent-based autonomous reasoning over KG**

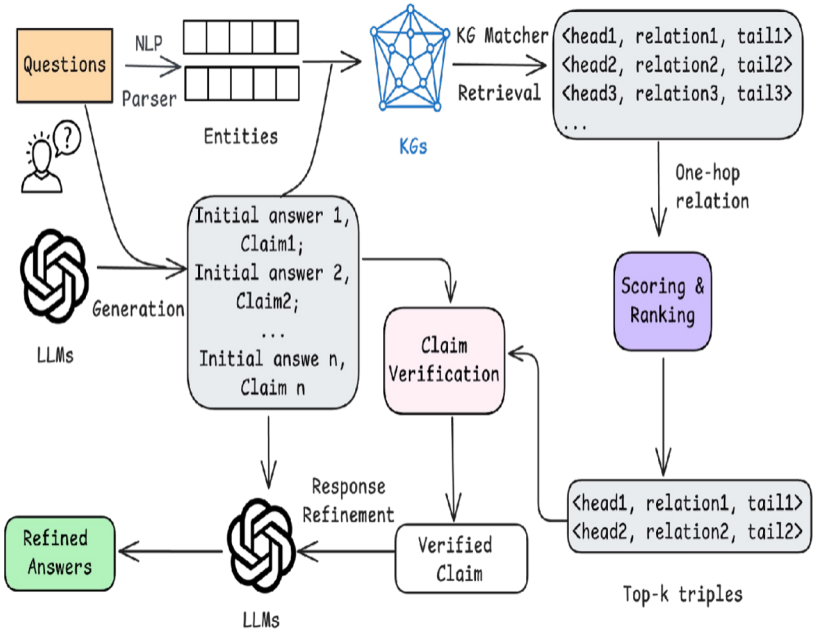
**Agent autonomously iterates the tool selection and knowledge updating for reasoning util reaching the answer entities**

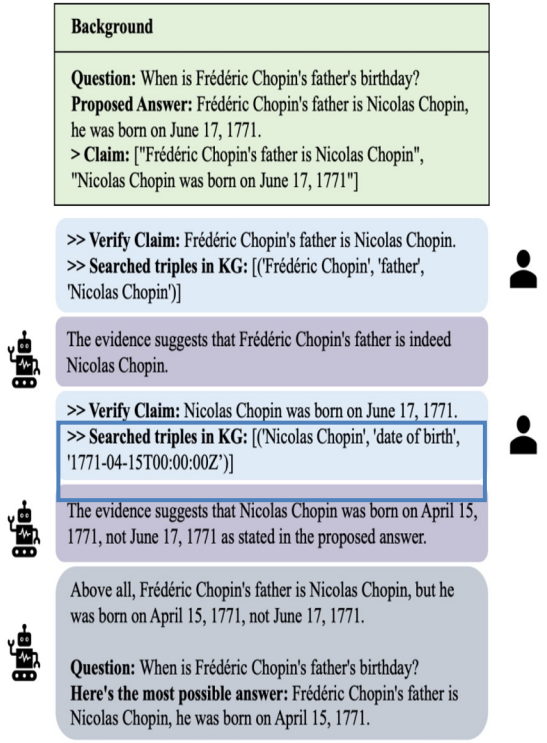
Example of instruction fine-tuning data synthesis and KG reasoning for the input-output pairs

Jiang, Jinhao, et al. KG-Agent: An efficient autonomous agent framework for complex reasoning over knowledge graph. https://aclanthology.org/2025.acl-long.468/ (2025).

# KGs as Refiners and Validators

- Refine and validate the answers for QA
  - **KG-Driven Filtering and Validation:** validate and filter out the incorrect answers
  - **KG-Augmented Output Refinement:** refine intermediate output for final answer




Guan, Xinyan, et al. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. AAAI. (2024).

# Takeaways

- KG-RAG
  - *Vector-based graph retrieval:* vector-based search is **time-consuming and computing-consuming** task for large KGs.
  - *Query-based graph retrieval*: **convert NLQ to GQL is still a challengeable** task as the specific schema structure is agnostic for LLMs.
- KG-guided Reasoning
  - *Complex reasoning*: reasoning over the large-scale KGs is a **time-consuming and computing-consuming** task.
  - *Faithful reasoning:* generate the **reasoning paths from KGs relies on the prompt and tuning LLMs** while the faithful of the KG reasoning need to be addressed.
- LM and KG Alignment
  - *Effective knowledge fusion:* integrating LLMs with KGs with the **prompt-based fusion** is not the optimal one as the **topological information of KGs will be lost** during the conversion.
  - *Knowledge conflicts mitigation:* the **knowledge conflicts** between the **internal knowledge of LLMs** and the **retrieved external knowledge** need to be mitigated.

# KG for LLM: Domain-specific Applications


- Bishwamittra Ghosh, Sarah Hasan, Naheed Anjum Arafat, and **Arijit Khan**, "Logical Consistency of Large Language Models in Fact-checking", in ICLR 2025.
- Feiyang Li, Peng Fang, Zhan Shi, **Arijit Khan**, Fang Wang, Weihao Wang, Xin Zhang, and Yongjian Cui, "CoT-RAG: Integrating Chain of Thought and Retrieval-Augmented Generation to Enhance Reasoning in Large Language Models", in EMNLP 2025 (Findings)
- Chuangtao Ma, Sriom Chakrabarti, **Arijit Khan**, and Balint Molnar, "Knowledge Graph-based Retrieval-Augmented Generation for Schema Matching, under submission. https://arxiv.org/abs/2501.08686
- Chuangtao Ma, Zeyu Zhang, **Arijit Khan**, Sebastian Schelter, and Paul Groth, "Cost-Efficient RAG for Entity Matching with LLMs: A Blocking-based Exploration“, under submission. https://arxiv.org/abs/2602.05708

# Logical Consistency of Large Language Models in Fact-checking

[ICLR 2025]

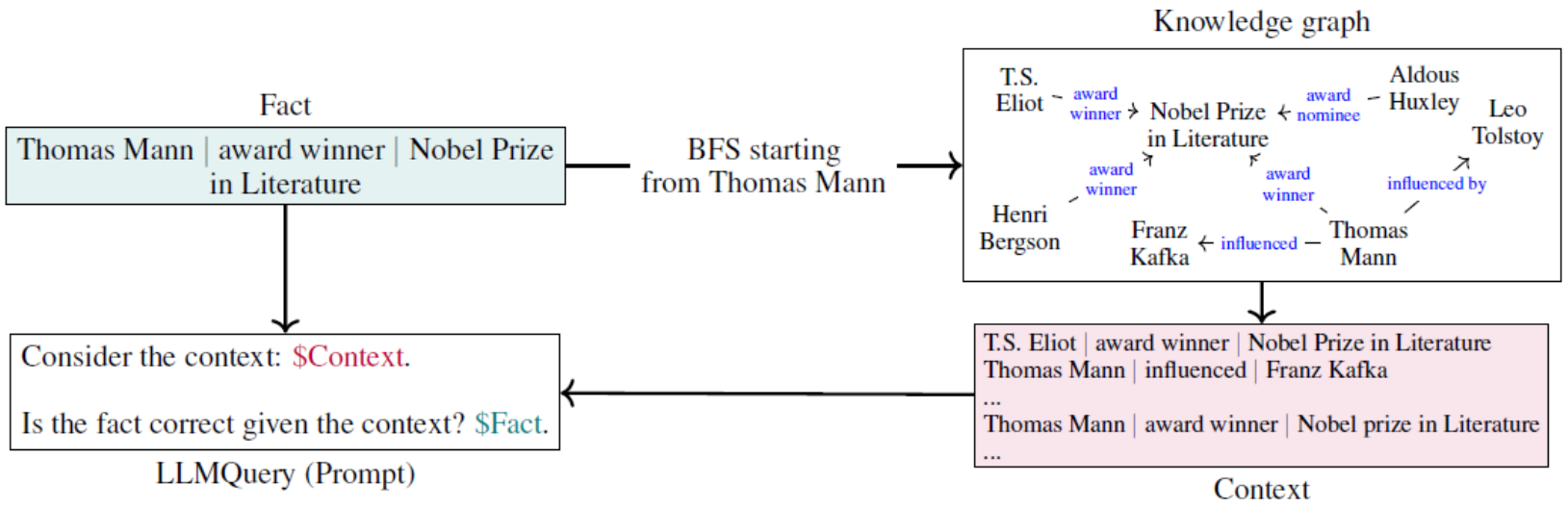
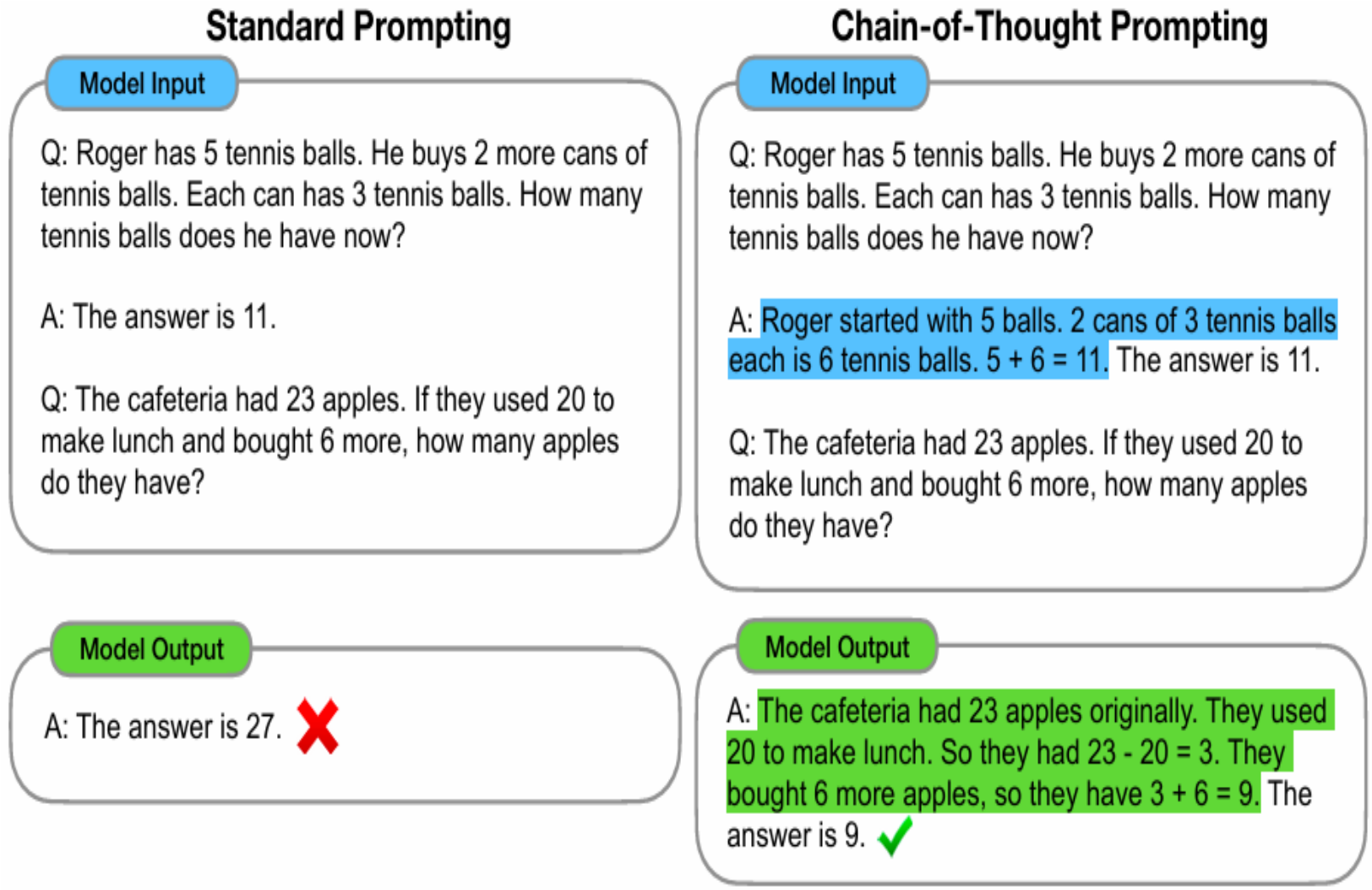



Figure 1: Our LLM-based fact-checking framework with the KG context. The LLMQuery shown is an example of zero-shot prompt, that is, it does not contain examples or demonstrations.

- What is their consistency over logically equivalent changes in the input query?
- How to improve an LLM's consistency under logically equivalent perturbation in input query?

| Datasets | Fact types and Rules | Training | Evaluation | Test |
|---|---|---|---|---|
| FreebaseLFC | $p, \neg p, p \wedge q, p \vee q$ | 1K $(\times 4)$ | 5K $(\times 4)$ | 5K $(\times 4)$ |
| | $p \wedge (q \vee r), p \vee (q \wedge r), p \vee q \leftrightarrow q \vee p, p \wedge q \leftrightarrow q \wedge p$ | — | 5K $(\times 4)$ | 5K $(\times 4)$ |
| NELLLFC | All above fact types and rules | — | 5K $(\times 8)$ | 5K $(\times 8)$ |
| WikiLFC | $p, \neg p$ | — | 5K $(\times 2)$ | 5K $(\times 2)$ |

# COT-RAG (EMNLP 2025): Background




*《Chain-of-Thought Prompting Elicits Reasoning in Large Language Models》 by Google in NeurIPS 2022*

# Motivation 1 ： The low reliability of relying solely on LLMs to generate reasoning chains

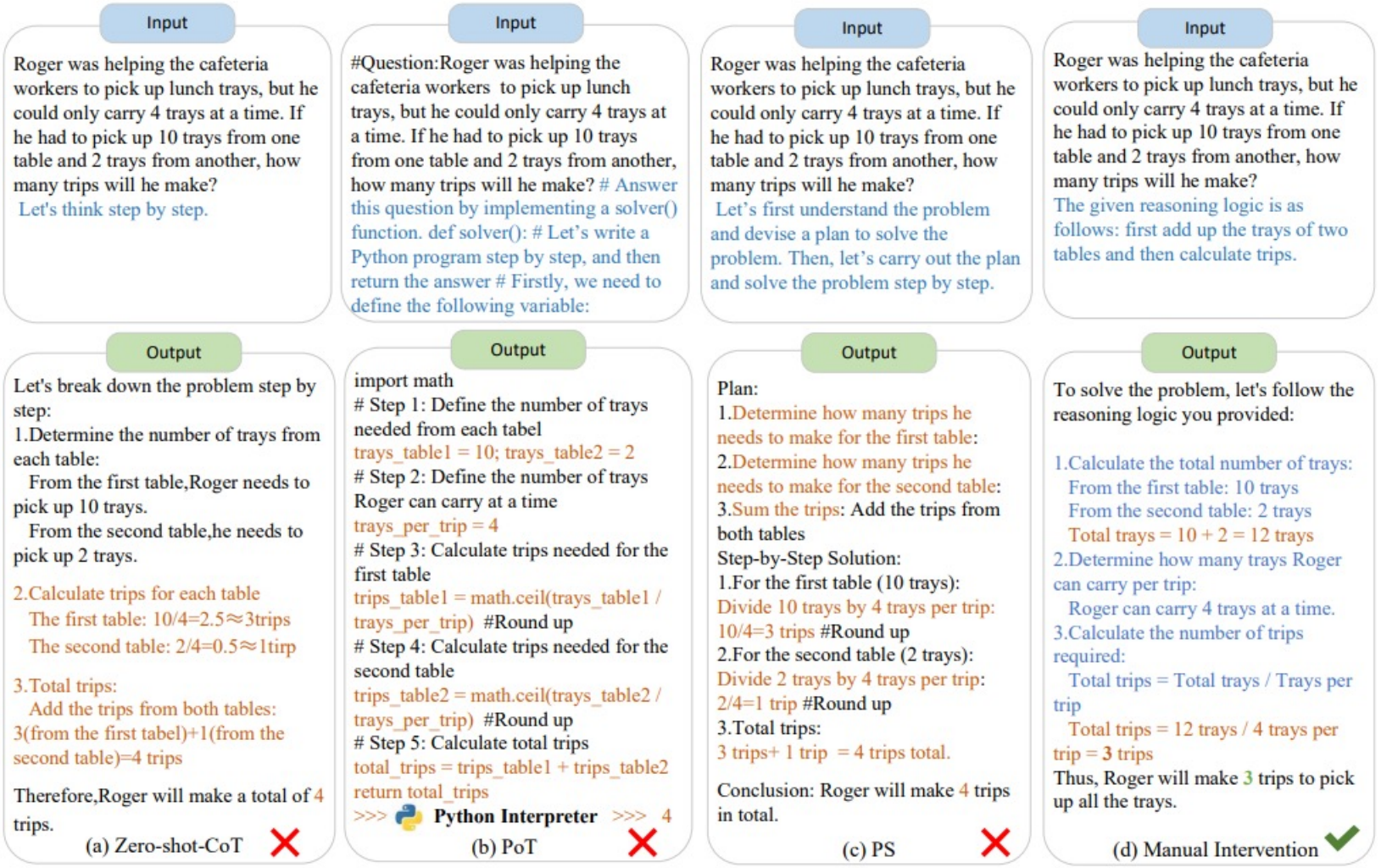



Figure 1: Example inputs and outputs of GPT-4o mini with (a) Zero-shot-CoT (Kojima et al., 2022), (b) PoT (Chen et al., 2023), (c) PS (Wang et al., 2023a) and (d) Manual Intervention on MultiArith (Roy and Roth, 2015).

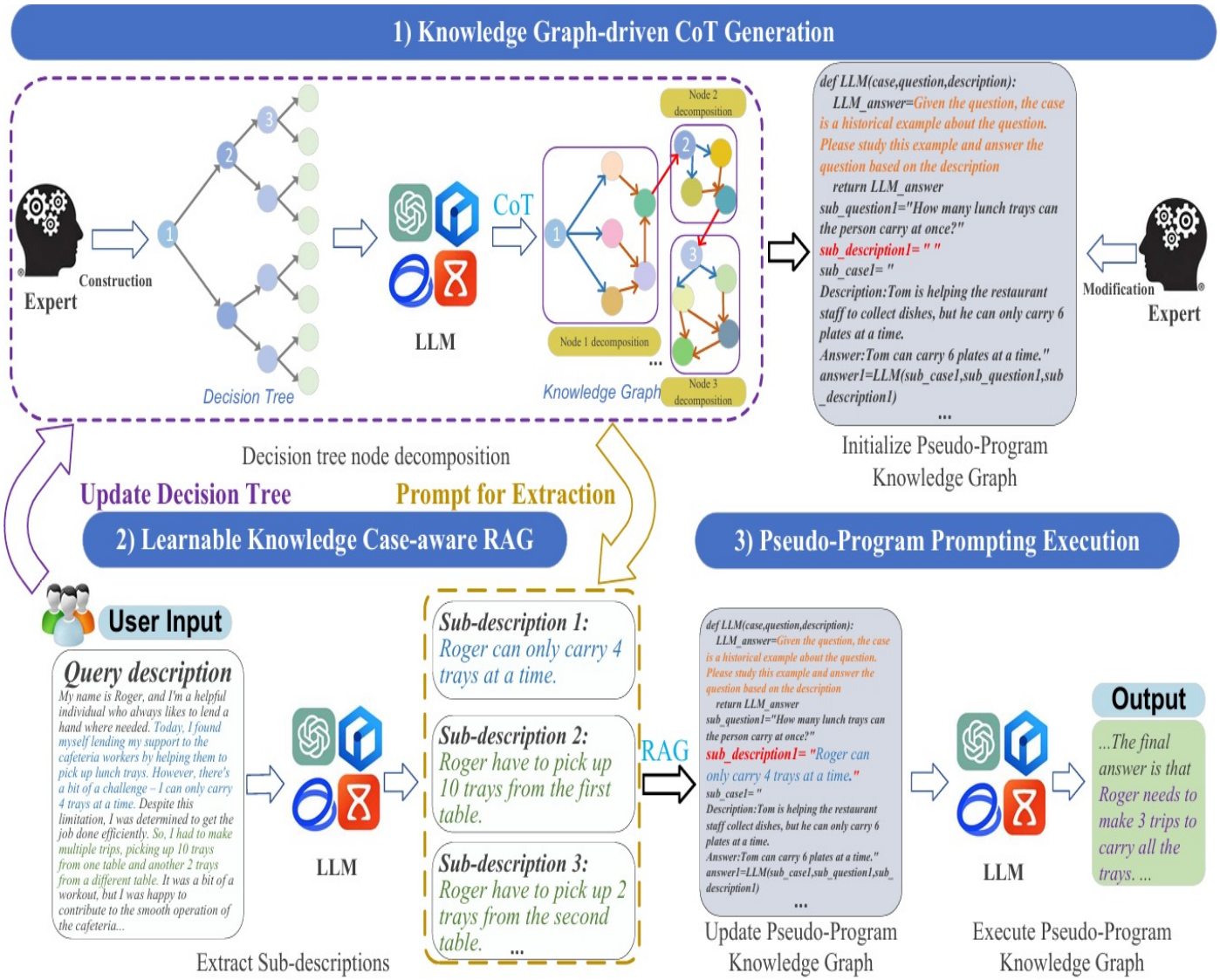
# Motivation 2： The poorer reasoning performance from natural language prompts compared with code prompts

| Method | Math Word Problems | | | | | Planning | Multi-hop QA | | | Relation |
|---|---|---|---|---|---|---|---|---|---|---|
| | GSM8K | SVAMP | MultiArith | ASDiv | AQuA | SayCan | StrategyQA | Date | Sport | CLUTRR |
| Greedy Decoding | | | | | | | | | | |
| Standard | 19.6 | 69.5 | 43.8 | 72.1 | 31.5 | 82.5 | 63.9 | 51.3 | 71.9 | 42.0 |
| CoT | 63.3 | 77.3 | 96.5 | 80.0 | 42.1 | 86.4 | **72.5** | 59.9 | 98.6 | 48.5 |
| LtM | 38.3 | 80.3 | 74.0 | 76.5 | 40.6 | 77.7 | 72.2 | 76.6 | **99.5** | 47.2 |
| Faithful CoT (ours) | **72.3** | **83.4** | **98.8** | **80.2** | **47.2** | **89.3** | 63.0 | **81.6** | 99.1 | **58.9** |
| Self-Consistency Decoding | | | | | | | | | | |
| CoT | 78.0 | 86.8 | **100.0** | 84.2 | 52.0 | 89.3 | **79.8** | 63.8 | 98.0 | 45.7 |
| LtM | 38.8 | 80.5 | 74.0 | 76.3 | 44.9 | 76.7 | 71.9 | 77.2 | **99.4** | 50.9 |
| Faithful CoT (ours) | **80.0** | **88.8** | 99.2 | **84.4** | **61.4** | **94.2** | 65.2 | **85.5** | 99.0 | **71.9** |

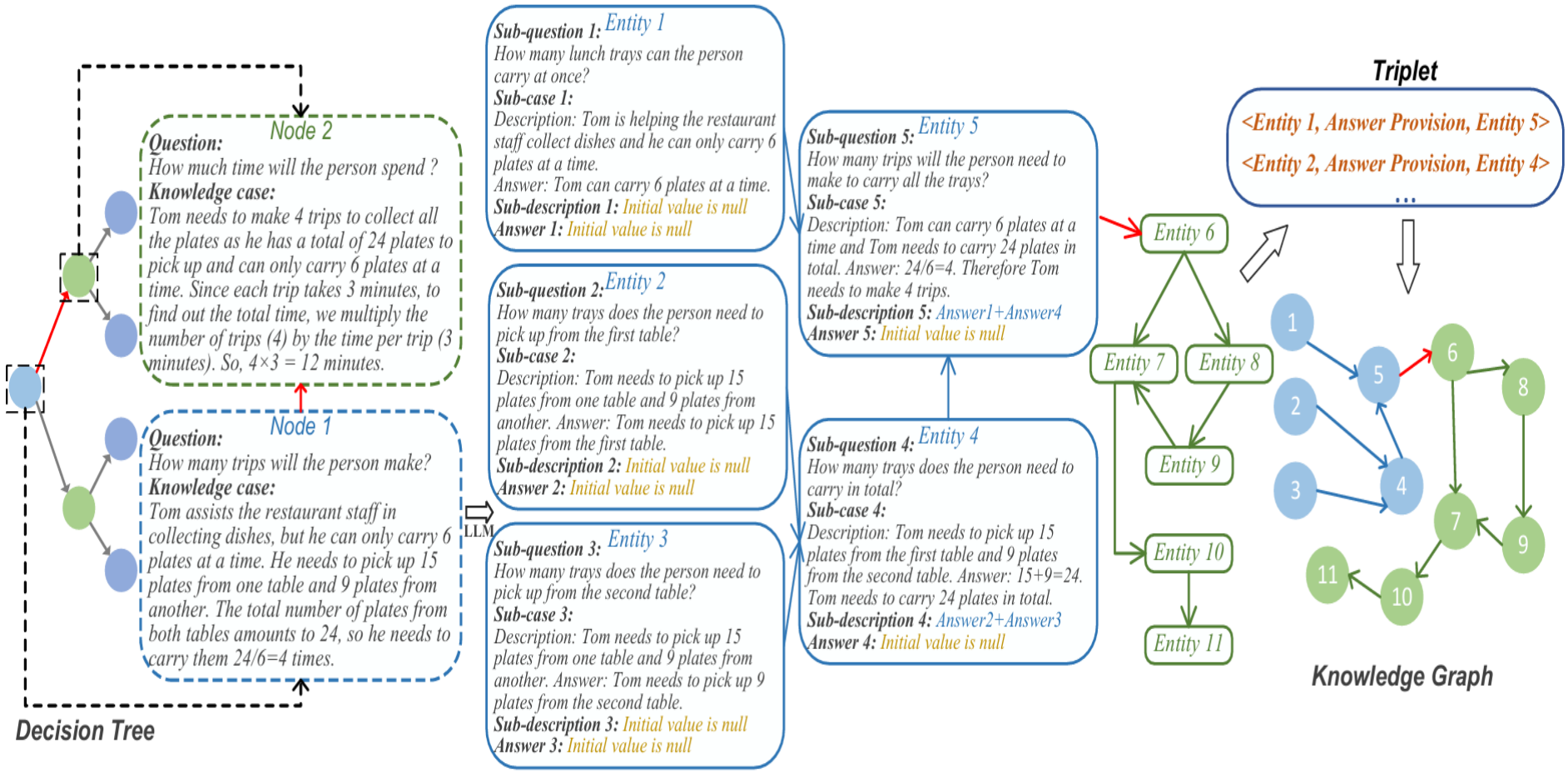
- ✓ The above Table demonstrates the poor performance of natural-language reasoning methods (CoT, LtM)
- ✓ Prior work[2,3] establishes the superior performance of code over natural language prompts.


[1] Faithful Chain-of-Thought Reasoning. ***IJCNLP-AACL 2023***
[2] PAL: Program-aided Language Models. ***ICML 2023***
[3] AdaPlanner: Adaptive Planning from Feedback with Language Models. ***NeurIPS 2023***

# CoT-RAG: A reasoning framework integrating Chain-of-Thought (CoT) and Retrieval-Augmented Generation (RAG)



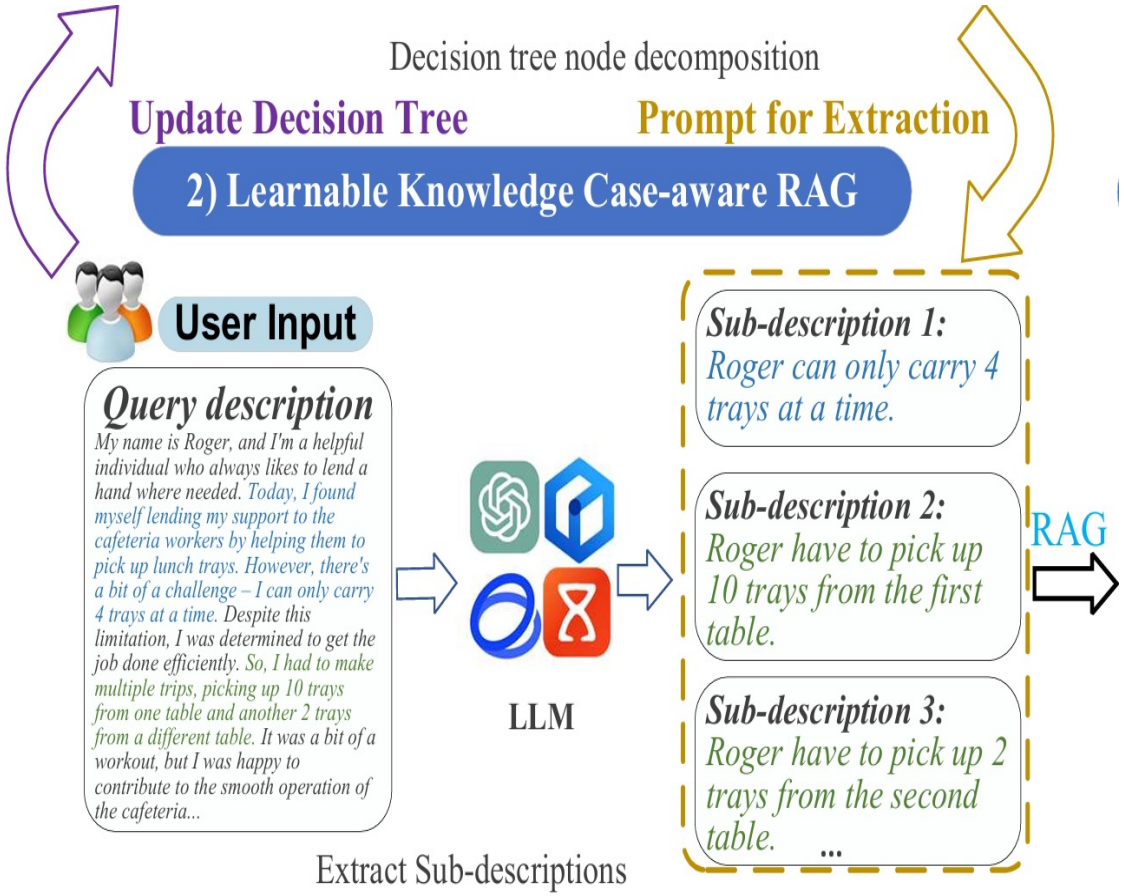
# Knowledge Graph-driven CoT Generation




**Key Innovations**:

- Structuring LLM reasoning with expert-defined, coarse-grained decision trees
- Decomposing trees into fine-grained knowledge graphs to model complex relationships

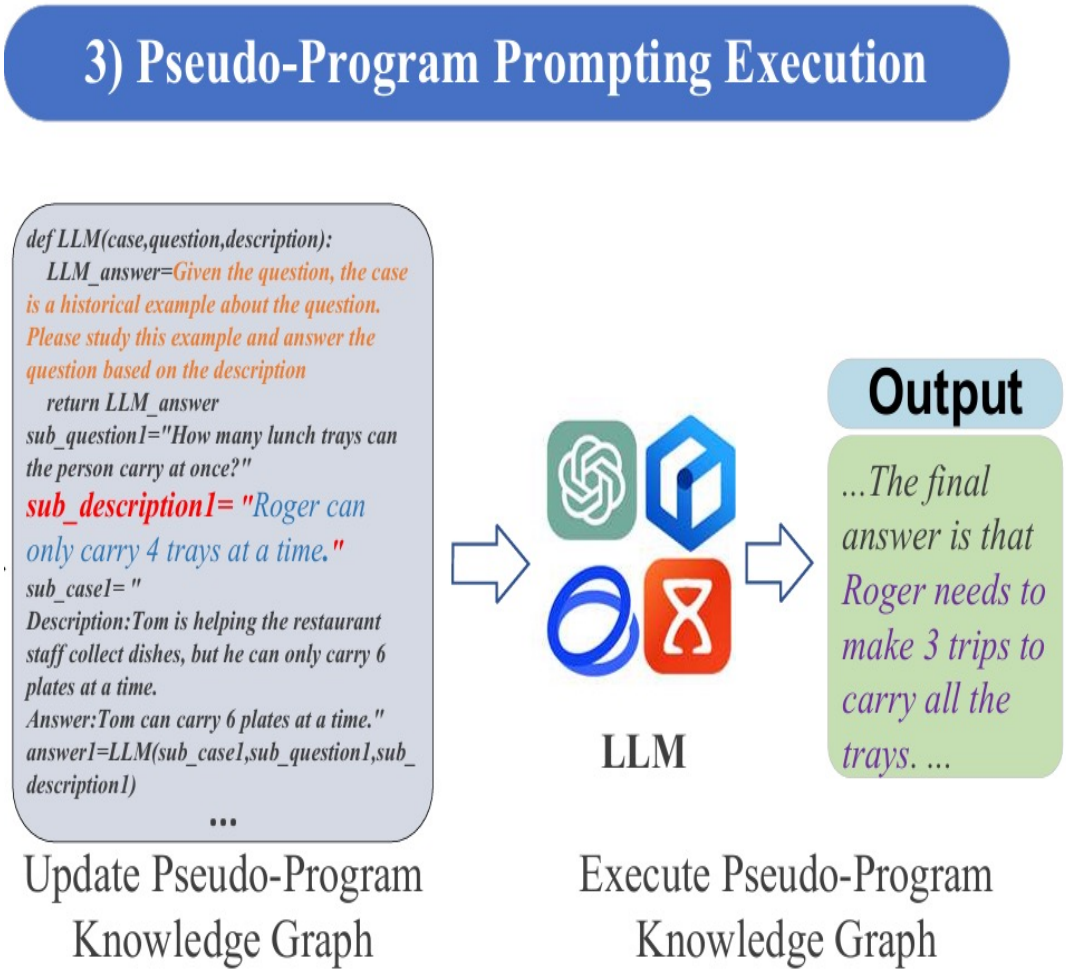
# Learnable Knowledge Case-aware RAG




**Key Innovations**:

- Extracting sub-descriptions from user queries by retrieving relevant sub-cases from the knowledge graph
- Updating the decision tree dynamically based on user input to optimize the knowledge graph

# Pseudo-Program Prompting Execution




**Key Innovations**:

- Replaces natural language with structured pseudo-code prompts to boost the logicality of LLM-generated reasoning chains
- Leverages the clear structure and broad applicability of pseudo-program

# Experimental Results

| Method | AQuA | GSM8K | MultiArith | SingEq | HotpotQA | CSQA | SIQA | Letter | Coin | Average |
|---|---|---|---|---|---|---|---|---|---|---|
| *Without external knowledge and exemplars* | | | | | | | | | | |
| Zero-shot | 42.6 | 77.8 | 95.9 | 86.8 | 80.1 | 74.5 | 77.3 | 35.8 | 76.7 | 71.9 |
| Zero-shot-CoT | 43.4 | 78.3 | 96.7 | 88.5 | 81.4 | 75.6 | 78.0 | 34.5 | 75.6 | 72.4 |
| PS | 50.1 | 82.8 | 96.9 | 89.4 | 83.0 | 74.2 | 76.3 | 44.7 | 79.5 | 75.2 |
| QDMRPS | 47.3 | 83.8 | 95.2 | 90.7 | 86.7 | 76.6 | 77.8 | 36.5 | 76.3 | 74.5 |
| *With exemplars* | | | | | | | | | | |
| Manual-CoT | 54.3 | 85.8 | 97.2 | 92.3 | 85.7 | 79.6 | 82.4 | 39.6 | 79.2 | 77.3 |
| Auto-CoT | 47.8 | 82.4 | 97.5 | 91.6 | 86.1 | 76.4 | 80.6 | 41.0 | 81.2 | 76.1 |
| Complex-CoT | 51.7 | 83.5 | 96.6 | 92.8 | 82.8 | 76.9 | 78.5 | 37.7 | 81.9 | 75.8 |
| Iter-CoT | 51.6 | 80.0 | 97.8 | 93.4 | 64.8 | 76.9 | 77.3 | 41.8 | 77.5 | 73.5 |
| ZEUS | 51.9 | 88.4 | 97.3 | 92.8 | 84.9 | 77.4 | 81.7 | 42.8 | 82.5 | 77.7 |
| Pattern-CoT | 52.8 | 85.3 | 97.7 | 91.3 | 82.5 | 76.3 | 78.9 | 41.4 | 83.7 | 76.7 |
| *With external knowledge* | | | | | | | | | | |
| KD-CoT | 22.3 | 68.4 | 76.0 | 62.3 | 79.9 | 85.6 | 90.8 | 24.6 | 58.3 | 63.1 |
| IRCoT | 20.7 | 65.6 | 78.3 | 65.1 | 87.5 | 82.8 | 87.9 | 28.2 | 54.5 | 63.4 |
| KG-CoT | 12.3 | 66.8 | 78.6 | 61.5 | 73.5 | 88.9 | 92.1 | 26.7 | 53.2 | 61.5 |
| *With both external knowledge and exemplars* | | | | | | | | | | |
| **CoT-RAG (ours)** | **65.7** | **94.7** | **98.5** | **98.7** | **98.4** | **97.9** | **98.7** | **54.6** | **94.7** | **89.1** |

CoT-RAG significantly outperforms the existing **CoT** methods on 9 datasets in the fields of arithmetic, common sense, and symbols, with an accuracy improvement of 4.0% - 44.3%

# Experimental Results

| Method | LaB | LeB | CFB | AGI | Average |
|---|---|---|---|---|---|
| *Graph-form LLM-based RAG methods* | | | | | |
| GraphRAG | 94.8 | 97.5 | 73.1 | 54.6 | 80.0 |
| KG-CoT | 89.3 | 93.6 | 72.8 | 32.6 | 72.1 |
| ToG | 86.7 | 90.2 | 68.3 | 64.2 | 77.4 |
| PoG | 93.8 | 91.7 | 89.5 | 45.3 | 80.1 |
| RRKG | 91.3 | 92.4 | 74.7 | 27.4 | 71.5 |
| RoG | 90.4 | 88.1 | 88.7 | 67.5 | 83.7 |
| Graph-CoT | 54.7 | 68.2 | 63.1 | 24.7 | 52.7 |
| ToG-2 | 92.5 | 93.7 | 76.6 | 70.8 | 83.4 |
| AtomR | 83.6 | 82.5 | 77.3 | 37.5 | 70.2 |
| *Variants of CoT-RAG* | | | | | |
| CoT-RAG (IndexFlatL2) | 92.7 | 93.7 | 85.2 | 67.8 | 84.9 |
| CoT-RAG (IndexFlatIP) | 91.5 | 91.9 | 87.1 | 72.4 | 85.7 |
| CoT-RAG (IndexIVFFlat) | 93.6 | 92.1 | 86.8 | 75.3 | 87.0 |
| CoT-RAG (IndexLSH) | 90.8 | 92.5 | 88.1 | 74.6 | 86.5 |
| CoT-RAG (IndexPQ) | 93.1 | 92.9 | 87.4 | 73.9 | 86.8 |
| CoT-RAG (IndexIVFPQ) | 93.6 | 93.1 | 87.8 | 76.2 | 87.7 |
| CoT-RAG (Zero-expert) | 93.6 | 94.7 | 86.3 | 74.8 | 87.4 |
| **CoT-RAG (ours)** | **99.3** | **98.6** | **94.7** | **88.3** | **95.2** |

CoT-RAG significantly outperforms existing **Graph form LLM based RAG** methods on four datasets across three vertical domains of law, finance, and logic, with an accuracy improvement of 8.9% -40.6%

# Summary

**Motivation:**

① The low reliability of relying solely on LLMs to generate reasoning chains

② The poorer reasoning performance from natural language prompts compared with code prompts

**Method:**

A reasoning framework integrating Chain-of-Thought (CoT) and Retrieval-Augmented Generation (RAG)

- ✓ **Key Technique 1**: Knowledge Graph-driven CoT Generation
- ✓ **Key Technique 2**: Learnable Knowledge Case-aware RAG
- ✓ **Key Technique 3**: Pseudo-Program Prompting Execution

**Experimental Results:**

Compared with CoT methods, CoT-RAG has increased accuracy by 4.0% -44.3%, and compared with Graph form LLM based RAG methods, accuracy has increased by 8.9% -40.6%

**Key Achievement:**

✓ CoT-RAG: Integrating Chain of Thought and Retrieval-Augmented Generation to Enhance Reasoning in Large Language Models. **EMNLP Findings 2025**

# KG-RAG4SM

- **KG+LLM for domain-specific schema matching** https://arxiv.org/abs/2501.08686
- **Challenges:** LLMs without sufficient background knowledge tend to hallucinate results for schema matching and retrieve the relevant subgraphs from the large-scale external KG is expensive.
- **Motivation**: We introduce a KG-based RAG for schema matching that integrates multiple retrieval mechanisms to extract the relevant subgraphs from KGs for augmenting LLMs in schema matching.

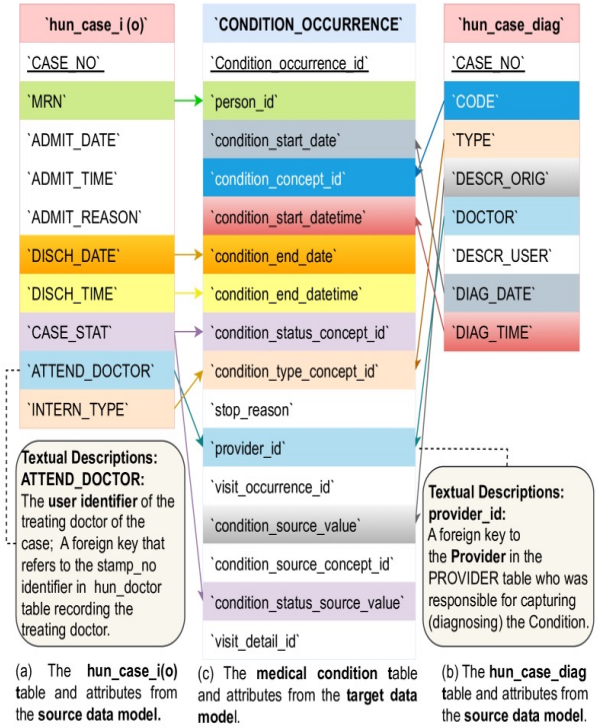



Figure: Example of schema matching in the EHR data model.

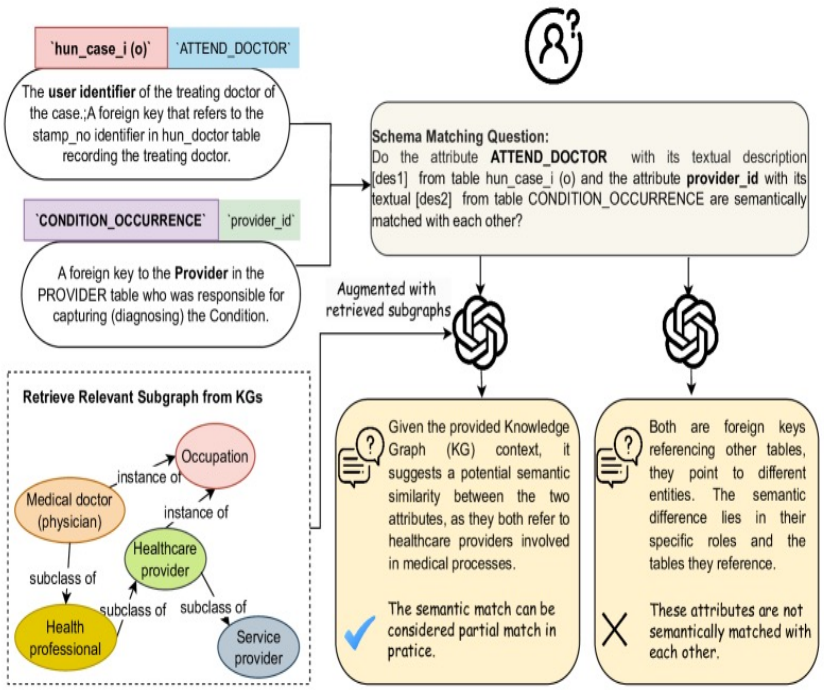



Figure :The Role of KG context in augmenting LLMs for schema matching.

# KG-RAG4SM

- **Method**: We introduce a novel KG-RAG4SM that identifies the most relevant subgraphs from external large KGs to augment LLMs for schema matching.

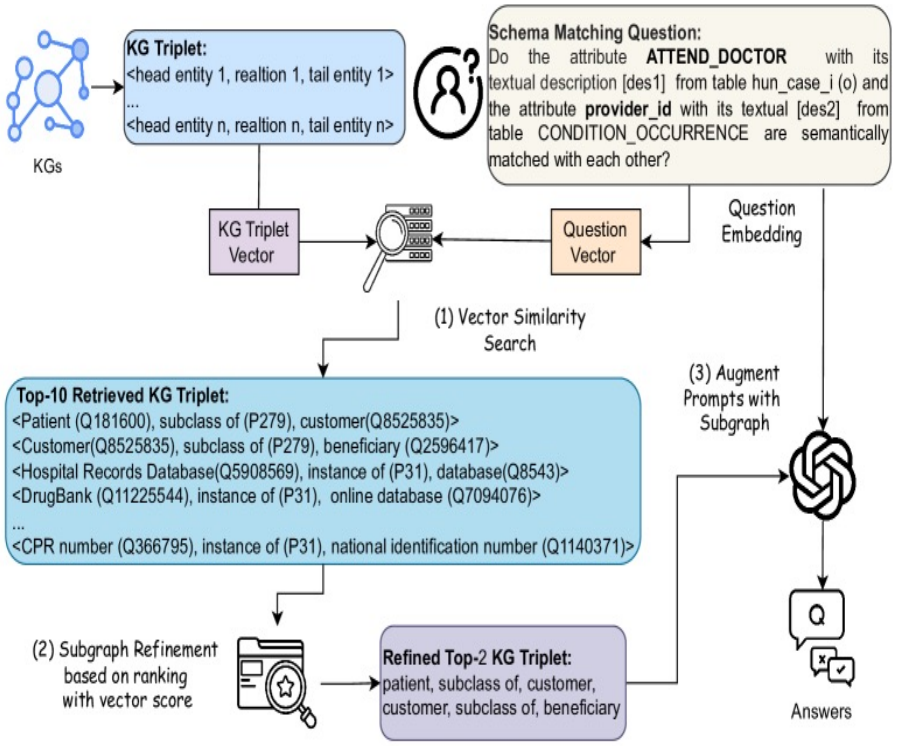



Figure: Overview of our proposed **KG-RAG4SM** method

- Retrieve the relevant KG triplets based on **vector similarity search** between questions embeddings and KG triplet embeddings
- Prune the retrieved relevant KG triplets with **vector similarity-based ranking**.
- Augment prompts with the **retrieved and refined subgraphs from large-size KG** and generate the final answer for the given schema matching questions.

# KG-RAG4SM

- **Results**: The results on 6 benchmark across domains demonstrated that our KG-RAG4SM outperforms PLMs and LLM-based SOTA methods in terms of effectiveness and efficiency.

| Dataset | Metric (%) | KG-RAG4SM | | LLM | |
|---|---|---|---|---|---|
| | | GPT-4o-mini | Jellyfish-8B | GPT-4o-mini | Jellyfish-8B |
| MIMIC | P | 6.25 | **9.52** (↑35.89%) | 6.25 | 6.81 |
| | R | 33.33 | 66.66(↓33.34%) | 33.33 | **100** |
| | F1 | 10.52 | **16.66**(↑30.56%) | 10.52 | 12.76 |
| Synthea | P | 20.00(↑90.11%) | **26.31**(↑150%) | 15.00 | 10.52 |
| | R | 36.36(↑33.33%) | **45.45**(↑25.00%) | 27.27 | 36.36 |
| | F1 | 25.80(↑58.21%) | **33.33**(↑104.2%) | 19.35 | 16.32 |
| CMS | P | **52.38**(↑17.86%) | 23.33(↓22.23%) | 44.44 | 30.00 |
| | R | 44.00(↓8.33%) | 28.00(↓22.22%) | **48.00** | 36.00 |
| | F1 | **47.82**(↑3.61%) | 25.45(↓22.21%) | 46.15 | 32.72 |
| EMED | P | **3.03**(↑1.67%) | 0 | 2.98 | 0 |
| | R | **50.00** | 0 | **50.00** | 0 |
| | F1 | **5.71**(↑1.42%) | 0 | 5.63 | 0 |
| Bank | P | **66.67**(↑200.05%) | 28.57(↑28.58%) | 22.22 | 22.22 |
| | R | 100 | 100 | 100 | 100 |
| | F1 | **80.00**(↑80.02%) | 44.44(↑22.22%) | 36.36 | 36.36 |
| IMSA | P | **100**(↑200.03%) | **100**(↑300%) | 33.33 | 25.00 |
| | R | 16.67 | 16.67 | 16.67 | 16.67 |
| | F1 | **28.57**(↑29.86%) | **28.57**(↑29.99%) | 22.22 | 20.00 |

Figure : Effectiveness of KG-RAG4SM against LLM baseline

| Dataset | Metric (%) | KG-RAG4SM | Unicorn | | SMAT |
|---|---|---|---|---|---|
| | | | Trained | Fine-tuned | Trained |
| MIMIC | P | **6.25** | 0.04 | 0 | 0.04 |
| | R | 33.33 | 99.99 | 0 | **100** |
| | F1 | **10.52** | 0.09 | 0 | 0.09 |
| Synthea | P | **20.00** | 0.37 | 0 | 11.82 |
| | R | 36.36 | 99.99 | 0 | **100** |
| | F1 | **25.80** | 0.74 | 0 | 21.15 |
| CMS | P | 52.38 | 0.97 | **59.99** | 31.57 |
| | R | 44.00 | **99.99** | 35.99 | 48.00 |
| | F1 | **47.82** | 1.93 | 44.99 | 38.09 |
| EMED | P | **3.03** | 0.04 | 0.98 | 1.25 |
| | R | 50.00 | 99.99 | 24.99 | **100** |
| | F1 | **5.71** | 0.09 | 1.88 | 2.46 |
| Bank | P | **66.67** | 1.37 | 39.99 | 1.36 |
| | R | **100** | 99.99 | 99.99 | **100** |
| | F1 | **80.00** | 2.70 | 57.14 | 2.70 |
| IMSA | P | **100** | 1.70 | 99.99 | 1.70 |
| | R | 16.67 | 99.99 | 33.33 | **100** |
| | F1 | 28.57 | 3.32 | **49.99** | 3.32 |

Figure : Effectiveness of KG-RAG4SM against PLM baseline

# KG-RAG4SM

- **Results**: The results on 6 benchmark across domains demonstrated that our KG-RAG4SM outperforms PLMs and LLM-based SOTA methods in terms of effectiveness and efficiency.

| Dataset | KG-RAG | | | LLMs | | PLMs | | |
|---|---|---|---|---|---|---|---|---|
| | KG-RAG4SM (GPT-4o-mini) | | | Jellyfish-8B | GPT-4o-mini | Unicorn | | SMAT |
| | Question Embedding Time (Sec) | Retrieval Time (Sec) | Generation Time (Sec) | Generation Time (Sec) | Generation Time (Sec) | Fine-tuning Time (Sec) | Training Time (Sec) | Training Time (Sec) |
| MIMIC | 0.01 | 2.60 | 0.64 | 0.26 | 0.77 | 3389.90 | 1627.45 | 3064.62 |
| Synthea | 0.04 | 2.59 | 0.51 | 0.26 | 0.60 | 2039.59 | 752.77 | 1404.00 |
| CMS | 0.04 | 2.89 | 0.58 | 0.25 | 0.59 | 1766.67 | 651.01 | 1399.92 |
| EMED | 0.05 | 2.55 | 0.61 | 0.21 | 0.67 | 3526.78 | 2062.03 | 7890.48 |
| Bank | 0.04 | 6.25 | 0.60 | 0.26 | 0.69 | 62.83 | 21.55 | 196.20 |
| IMSA | 0.04 | 3.89 | 0.53 | 0.25 | 0.64 | 146.70 | 52.82 | 391.20 |

Efficiency of KG-RAG4SM against LLM and PLM baselines

# CE-RAG4EM

- **KG+LLM for domain-specific entity matching** https://arxiv.org/abs/2602.05708
- **Challenges:** The knowledge grounding gap in LLM-based entity matching and costly retrieval and inference overhead in directly adapting vanilla RAG to entity matching in large-scale data integration.
- **Motivation**: We propose a blocking-based batch preprocessing strategy that groups similar records into the same batch, reducing redundant retrieval and inference for efficient RAG-based entity matching.

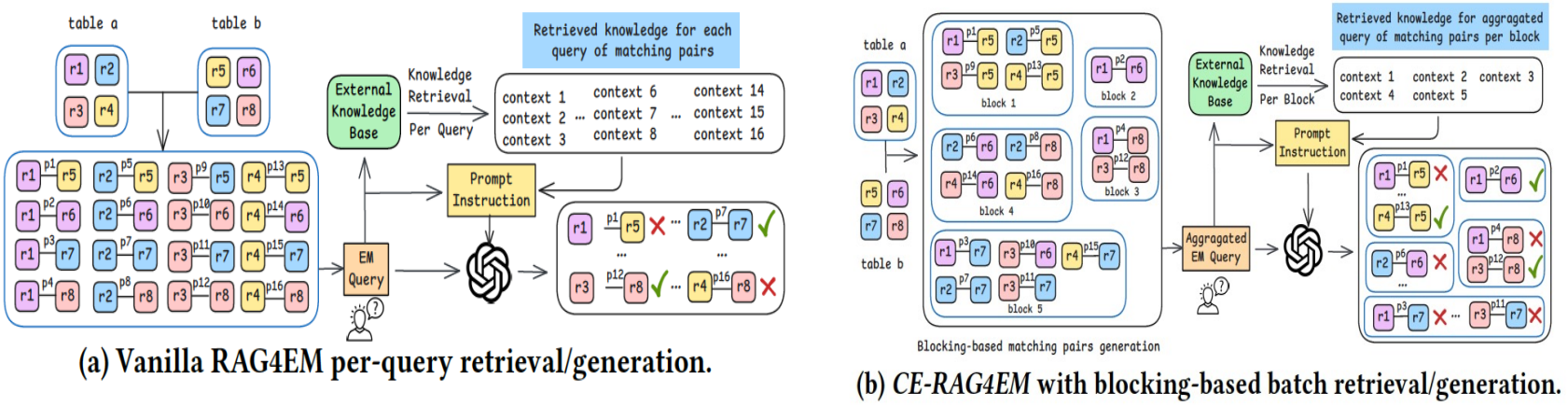



Figure : Vanilla RAG4EM vs. CE-RAG4EM. Matched records share the same color.

# CE-RAG4EM

- **Method**: We introduce a cost-efficient RAG framework for zero-shot entity matching that reduces computation through blocking-based batch retrieval and generation.

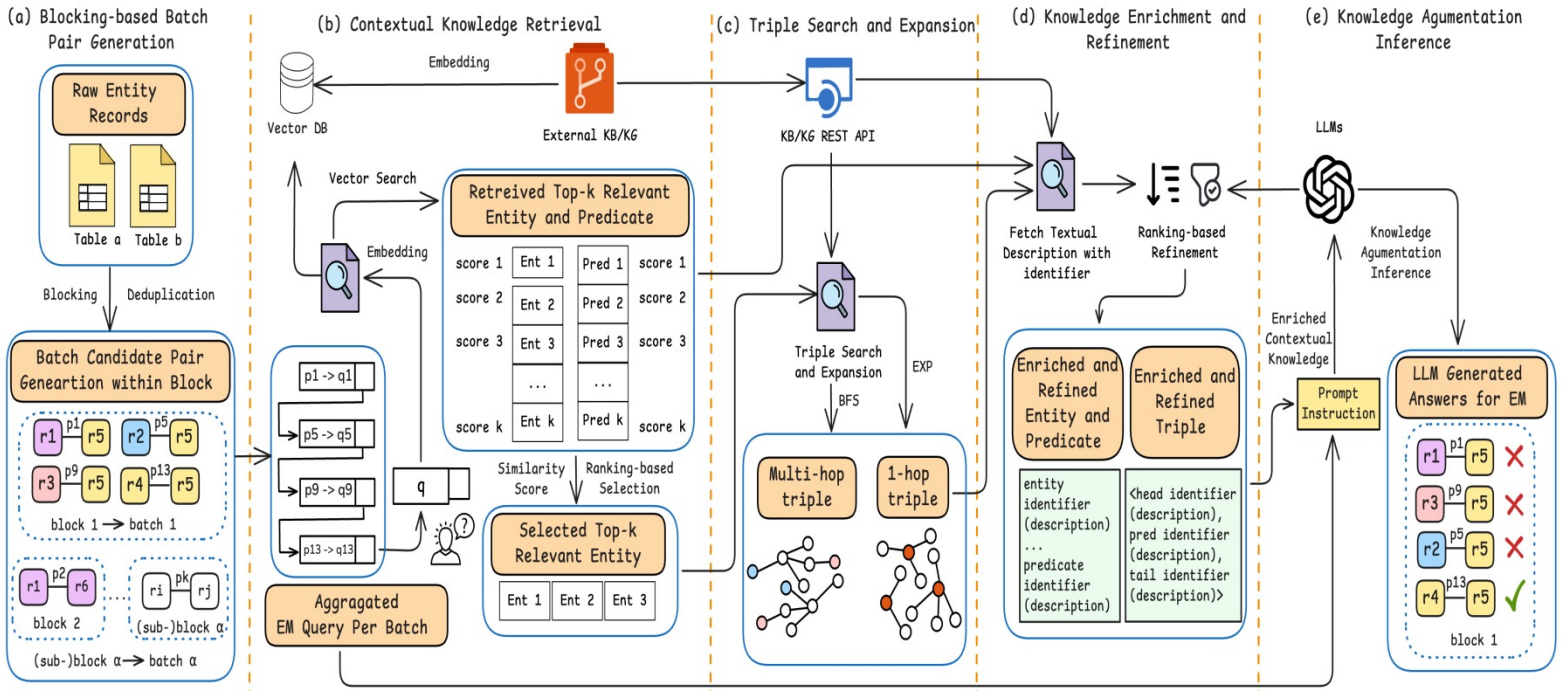



Figure: Overview of CE-RAG4EM. The framework consists of five phases (a)–(e). A check mark indicates that the LLM predicts Yes for the corresponding EM query, whereas a cross mark indicates No.

# CE-RAG4EM

- **Method**: We design a space exploration of CE-RAG4EM is structured around two core dimensions: blocking-based optimization and retrieval granularity.

**Algorithm 1** End-to-End *CE-RAG4EM*

**Input:** Table with entity records to be matched $T_s, T_t$, KG $\mathcal{G}$, LLM $\mathcal{F}_{LLM}$, max batch size $max_{bs}$, max triple search depth $D_{\max}$, retrieval budget $k$
**Output:** Answer for the given EM query $\mathcal{A}$
**Initiation:** $\mathcal{R} \leftarrow T_s \cup T_t$, $\mathcal{B}, P_B, \mathbb{B} \leftarrow \emptyset$, $\mathcal{Q}_\mathcal{B}, \mathcal{G}_\mathcal{B} \leftarrow \emptyset$, $\mathcal{A} \leftarrow \emptyset$
$\mathcal{B} \leftarrow$ ApplyBlocking$(T_s, T_t)$
**for each** block $B \in \mathcal{B}$ **do**
  $\{P_B, B_{id}\} | P_B \in \mathcal{R} \leftarrow$ GenerateMatchingPairs$(B, T_s, T_t)$
  $P_B \leftarrow$ Deduplication$(P_B)$
  **if** $|P_B| > max_{bs}$ **then**
    $\mathbb{B}_{sub} \leftarrow$ BatchDecomposer$(P_B, max_{bs})$
  **else**
    $\mathbb{B} \leftarrow \{P_B, B_{id}\}$
  **end if**
  $\mathbb{B} \leftarrow \mathbb{B} \cup \mathbb{B}_{sub}$
**end for**
**for each** batch $b \in \mathbb{B}$ **do**
  $Q_B \leftarrow$ AggregateQueries$(P_B)$
  $V_k, P_k \leftarrow$ BatchRetrieval$(Q_B, \mathcal{G}, k)$
  $\mathcal{G}_B \leftarrow$ TripleSearch$(V_k, \mathcal{G}, D_{\max})$
  $\mathcal{G}_B \leftarrow$ EnrichRefinement$(\mathcal{G}_B, \mathcal{G})$
  $bx \leftarrow$ Serializer$(P, \mathcal{G}_P)$
  $\mathcal{A} \leftarrow \mathcal{F}_{LLM}$.Inference$(bx)$
**end for**
**return** $\mathcal{A}$

| | Retrieval Granularity (RG) | |
|---|---|---|
| (Batch Optimization) BO | *Entity & Predicate* | *Triple* |
| Batch Retrieval (BR) | CE-RAG4EM-BR | CE-KG-RAG4EM-BR |
| Batch Generation (BG) | CE-RAG4EM-BG | CE-KG-RAG4EM-BG |
| BR & BG | CE-RAG4EM-BR-BG | CE-KG-RAG4EM-BR-BG |

Table: Summary of the Design Solutions of CE-RAG4EM.

# CE-RAG4EM

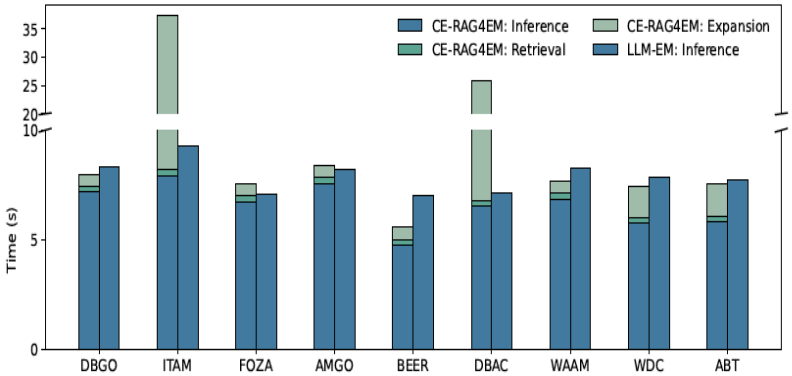
- **Results**: Extensive experiments on nine benchmark datasets show that CE-RAG4EM achieves comparable or improved matching quality while substantially reducing end-to-end overhead.

| Dataset | F1 (%) | | Precision (%) | | Recall (%) | |
|---|---|---|---|---|---|---|
| | *CE-RAG4EM* | LLM-EM | *CE-RAG4EM* | LLM-EM | *CE-RAG4EM* | LLM-EM |
| DBGO | 80.77 (+24.22) | 65.02 | 92.45 (-0.73) | 93.13 | 71.71 (+43.47) | 49.98 |
| ITAM | 72.61 (+12.70) | 64.43 | 97.62 (+0.18) | 97.44 | 58.03 (+20.52) | 48.15 |
| FOZA | 83.11 (+10.34) | 75.32 | 100.00 (0.00) | 100.00 | 71.21 (+17.50) | 60.61 |
| AMGO | 55.47 (+14.02) | 48.65 | 51.40 (-17.60) | 62.37 | 60.26 (+55.00) | 38.89 |
| BEER | 73.49 (+8.73) | 67.59 | 96.67 (+9.32) | 88.43 | 59.52 (+8.70) | 54.76 |
| DBAC | 81.87 (+7.12) | 76.43 | 95.14 (-1.05) | 96.15 | 71.85 (+13.26) | 63.44 |
| WAAM | 74.85 (+5.84) | 70.72 | 84.59 (-1.13) | 85.56 | 67.18 (+14.08) | 58.89 |
| WDC | 73.35 (+5.13) | 69.77 | 81.74 (-2.07) | 83.47 | 66.53 (+11.01) | 59.93 |
| ABT | 78.72 (+3.24) | 76.25 | 91.27 (-2.35) | 93.47 | 69.26 (+7.21) | 64.60 |

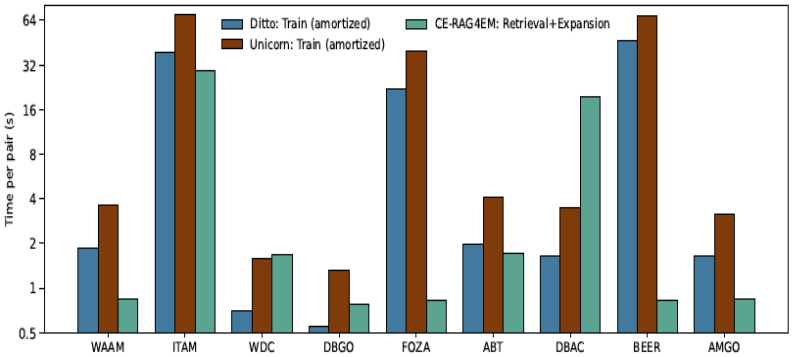
Overall Performance *CE-RAG4EM vs. LLM-EM*

| Dataset | F1 (%) | | | Precision (%) | | | Recall (%) | | |
|---|---|---|---|---|---|---|---|---|---|
| | *CE-RAG4EM* | Ditto | Unicorn | *CE-RAG4EM* | Ditto | Unicorn | *CE-RAG4EM* | Ditto | Unicorn |
| WAAM | 74.85 | 56.50 | 61.47 | 84.59 | 64.32 | 70.99 | 67.18 | 50.37 | 68.01 |
| ITAM | 72.61 | 64.33 | 67.45 | 97.62 | 67.32 | 68.12 | 58.03 | 61.59 | 66.79 |
| WDC | 73.35 | 45.16 | 70.03 | 81.74 | 49.89 | 70.24 | 66.53 | 41.26 | 69.83 |
| DBGO | 80.77 | 77.62 | 78.06 | 92.45 | 81.23 | 81.71 | 71.71 | 74.31 | 74.72 |
| FOZA | 83.11 | 69.64 | 82.61 | 100.00 | 89.16 | 90.13 | 71.21 | 57.13 | 76.25 |
| ABT | 78.72 | 67.50 | 78.72 | 91.27 | 64.71 | 89.41 | 69.26 | 59.43 | 70.32 |
| DBAC | 81.87 | 82.96 | 88.72 | 95.14 | 87.65 | 92.12 | 71.85 | 78.76 | 85.57 |
| BEER | 73.49 | 84.19 | 82.20 | 96.67 | 87.33 | 84.39 | 59.52 | 81.27 | 80.13 |
| AMGO | 55.47 | 53.86 | 68.86 | 51.40 | 56.70 | 67.97 | 60.26 | 51.29 | 69.77 |

Overall Performance *CE-RAG4EM vs. PLM-EM*



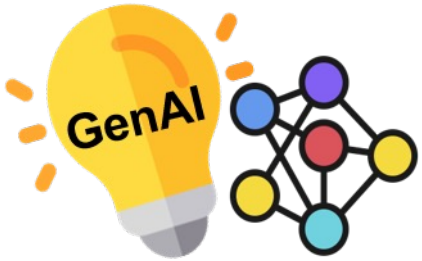

(a) End-to-end matching time per entity pair.




(b) Amortized training time (PLMs) vs. retrieval + expansion time (*CE-RAG4EM*).

Efficiency comparison of CE-RAG4EM against (a) LLM-EM and (b) PLM baselines.

# 4. KGs for LLMs

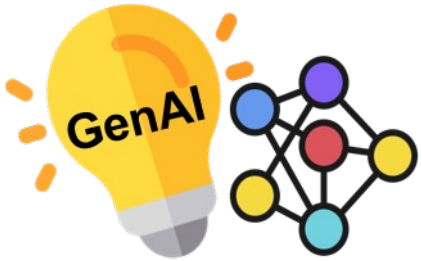



- Large Language Models
- Knowledge Graphs
- KGs as background knowledge for LLMs
- KGs as reasoning guidelines for LLMs
- KGs as refiners and validators for LLMs
- LLM+KG for domain-specific applications (e.g., data integration)

Questions?

Presented by Arijit Khan

# 5. LLMs for Knowledge Graphs




A. Knowledge Graph Link Prediction

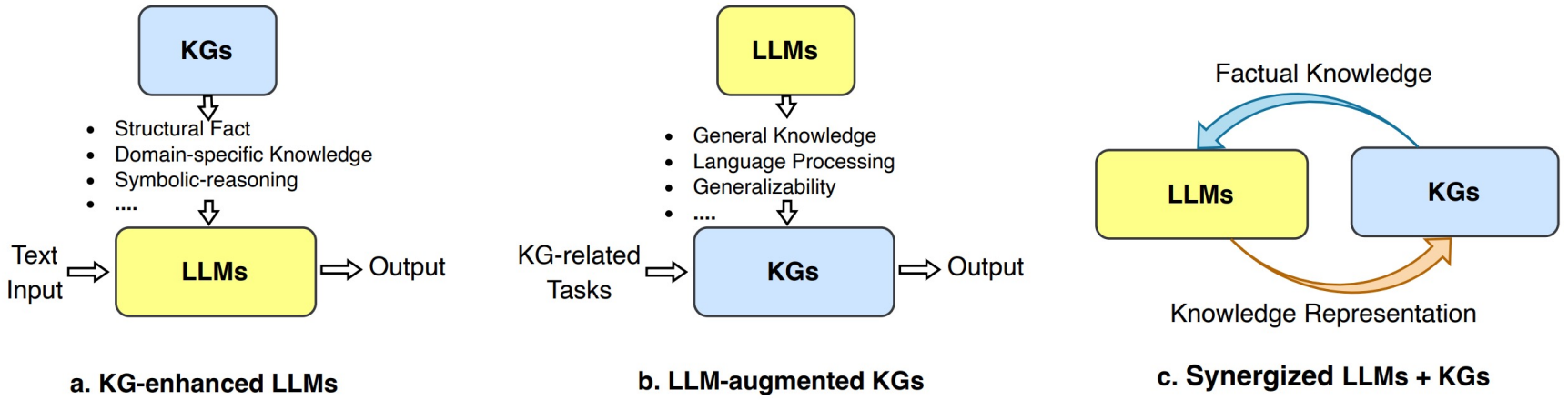
B. Knowledge Graph Embedding

C. Knowledge Graph Construction

D. Knowledge Graph Completion

Presented by Longxu Sun

# Unifying LLMs and Knowledge Graphs




**1. KG-enhanced LLMs**

Use KGs to improve LLM pre-training, inference, or interpretability.

**2. LLM-augmented KGs**

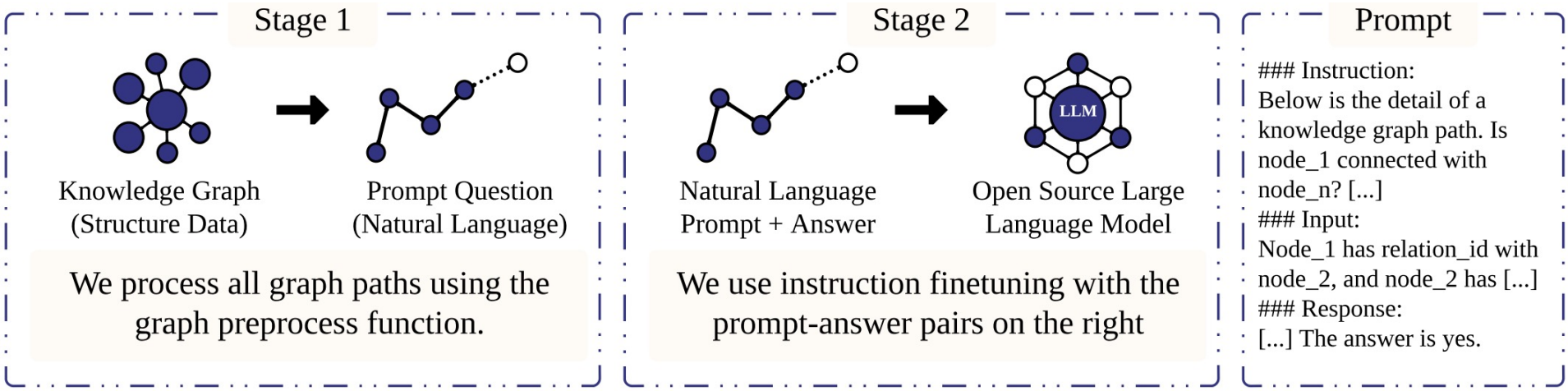
Use LLMs for KG embedding, completion, construction, QA, and graph-to-text generation.

**3. Synergized LLMs + KGs**

Jointly leverage LLMs and KGs for bidirectional reasoning.

[1] https://arxiv.org/pdf/2306.08302

# Knowledge Graph Large Language Model (KG-LLM) for Link Prediction [ACML'24]

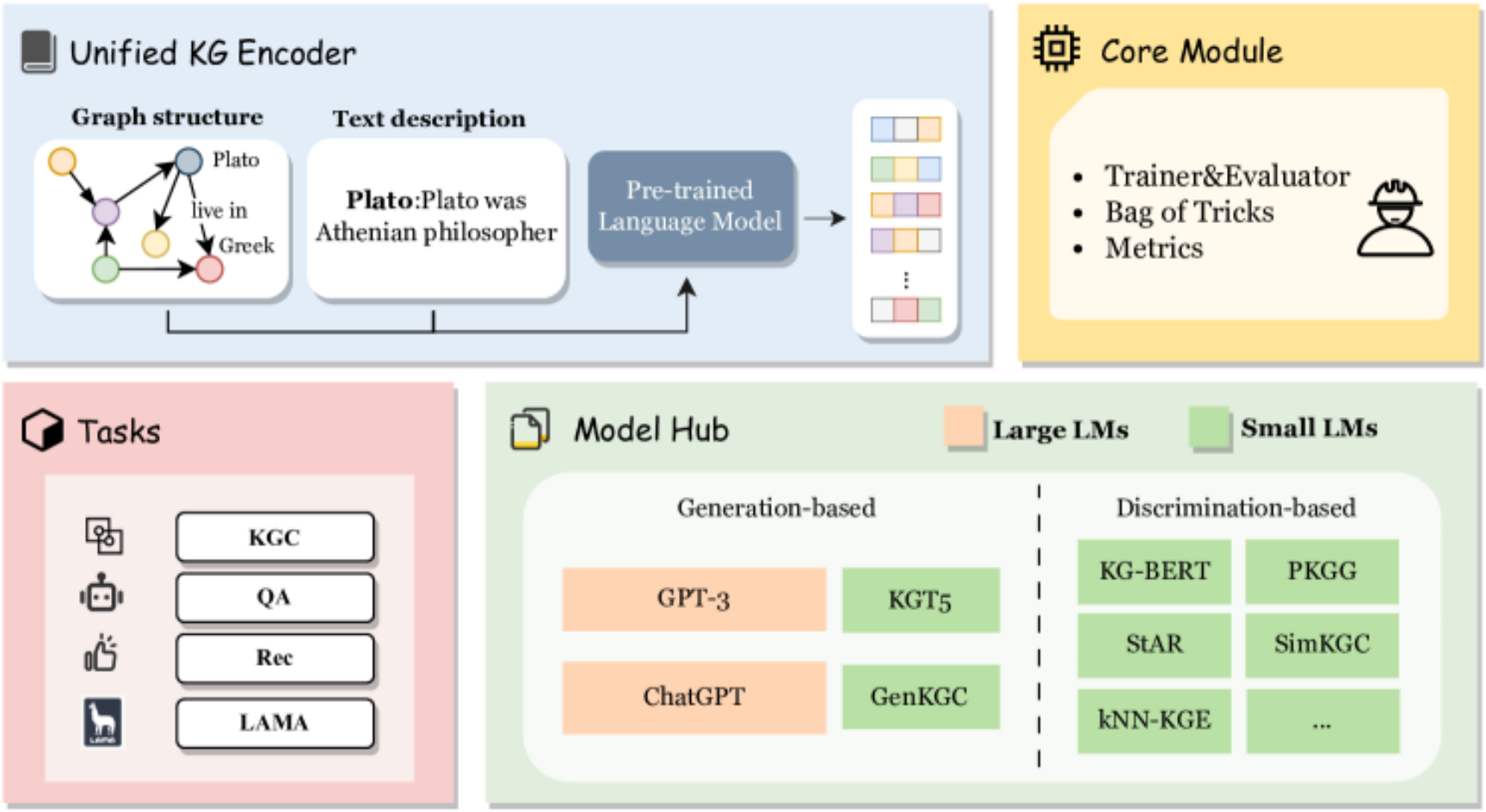



**KG-LLM idea:**

- Convert KG paths into natural-language prompts.
- Use chain-of-thought style reasoning over KG paths.
- Fine-tune LLMs with instruction-following objectives.
- Predict whether two nodes are connected, or what relation holds between them.

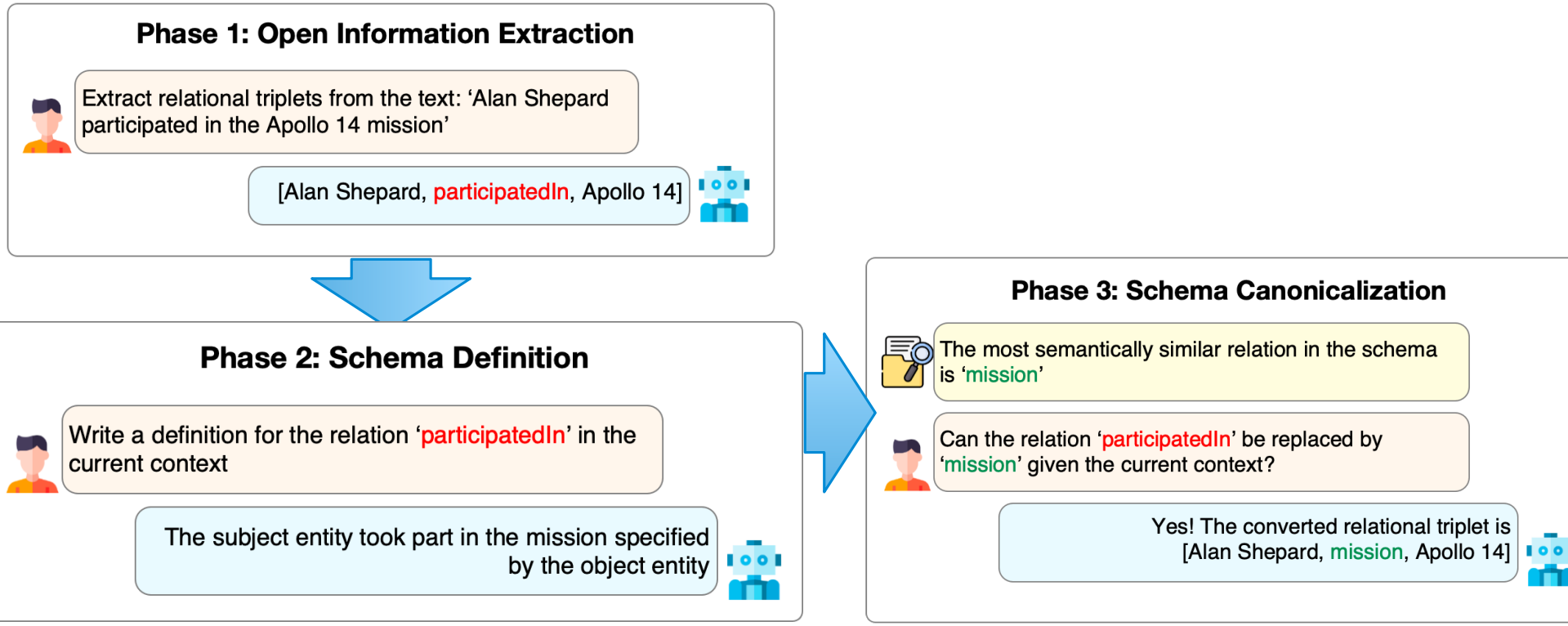
[1] https://arxiv.org/pdf/2403.07311

# LambdaKG: A Library for Pre-trained Language Model-Based Knowledge Graph Embeddings



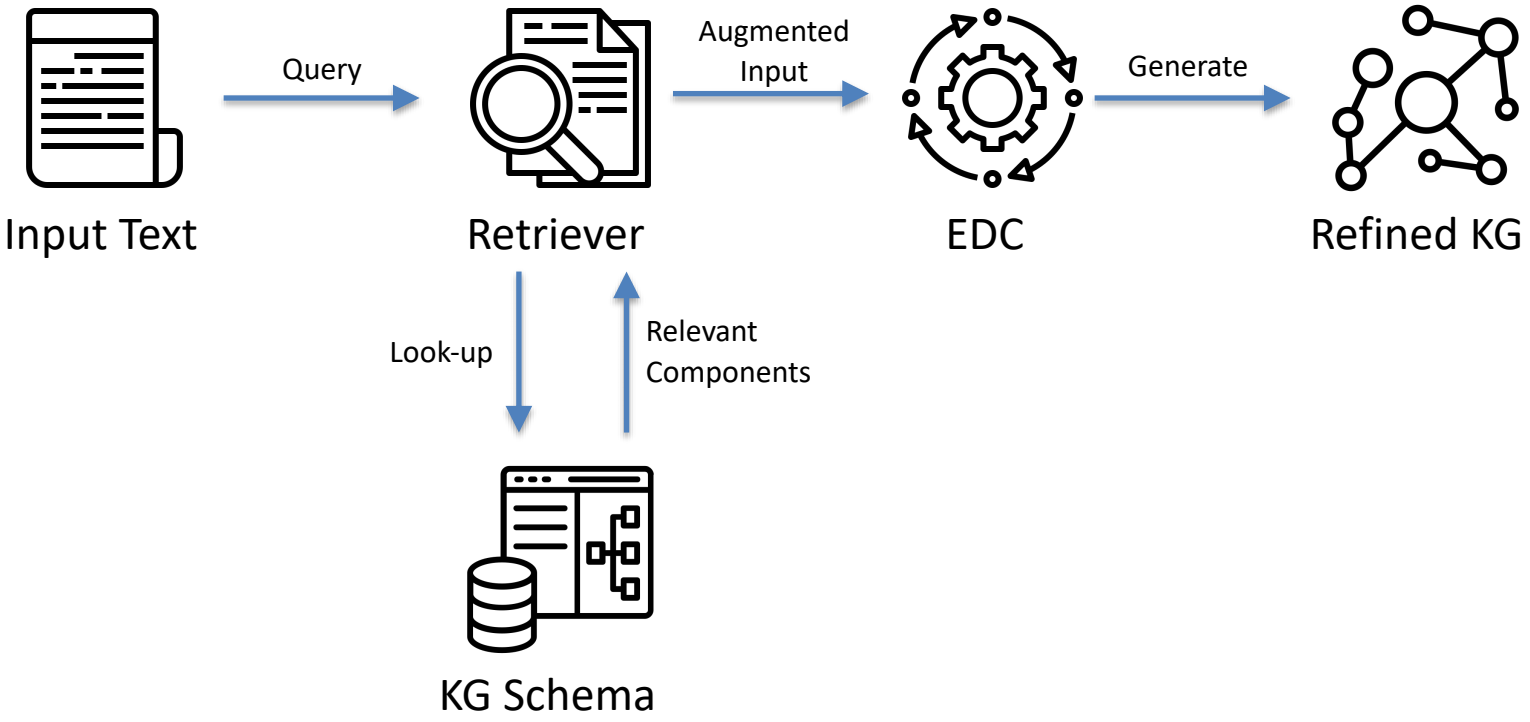

[1] https://ar5iv.labs.arxiv.org/html/2210.00305

# Extract, Define, Canonicalize: An LLM-based Framework for Knowledge Graph Construction [EMNLP'24]




[1] https://arxiv.org/pdf/2404.03868

# Extract, Define, Canonicalize: An LLM-based Framework for Knowledge Graph Construction [EMNLP'24]

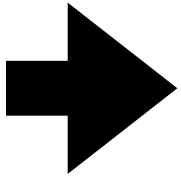



[1] https://arxiv.org/pdf/2404.03868

# Extract, Define, Canonicalize: An LLM-based Framework for Knowledge Graph Construction [EMNLP'24]

**Refined OIE Prompt**

Given a piece of text, extract relational triplets in the form of [Subject, Relation, Object] from it.
Here are some examples:
Example 1:
Text: The 17068.8 millimeter long ALCO RS-3 has a diesel-electric transmission.
Entities: ['ALCO RS-3', 'Diesel-electric transmission', '17068.8 (millimetres)']
Triplets: [['ALCO RS-3', 'powerType', 'Diesel-electric transmission'], ['ALCO RS-3', 'length', '17068.8 (millimetres)']]
...
Now please extract triplets from the following text: Alan Shepard was born on Nov 18, 1923 and selected by NASA in 1959. He was a member of the Apollo 14 crew. Entities: ['Alan Shepard', 'Nov 18, 1923', 'NASA', '1959', 'Apollo 14']
Here are some potential relations and their descriptions you may look out for during extraction:
1. birthDate: The subject entity was born on the date specified by the object entity.
2. mission: The subject entity participated in the event or operation specified by the object entity.
3. selectedByNasa: The subject entity was selected by NASA in the year specified by the object entity.
...

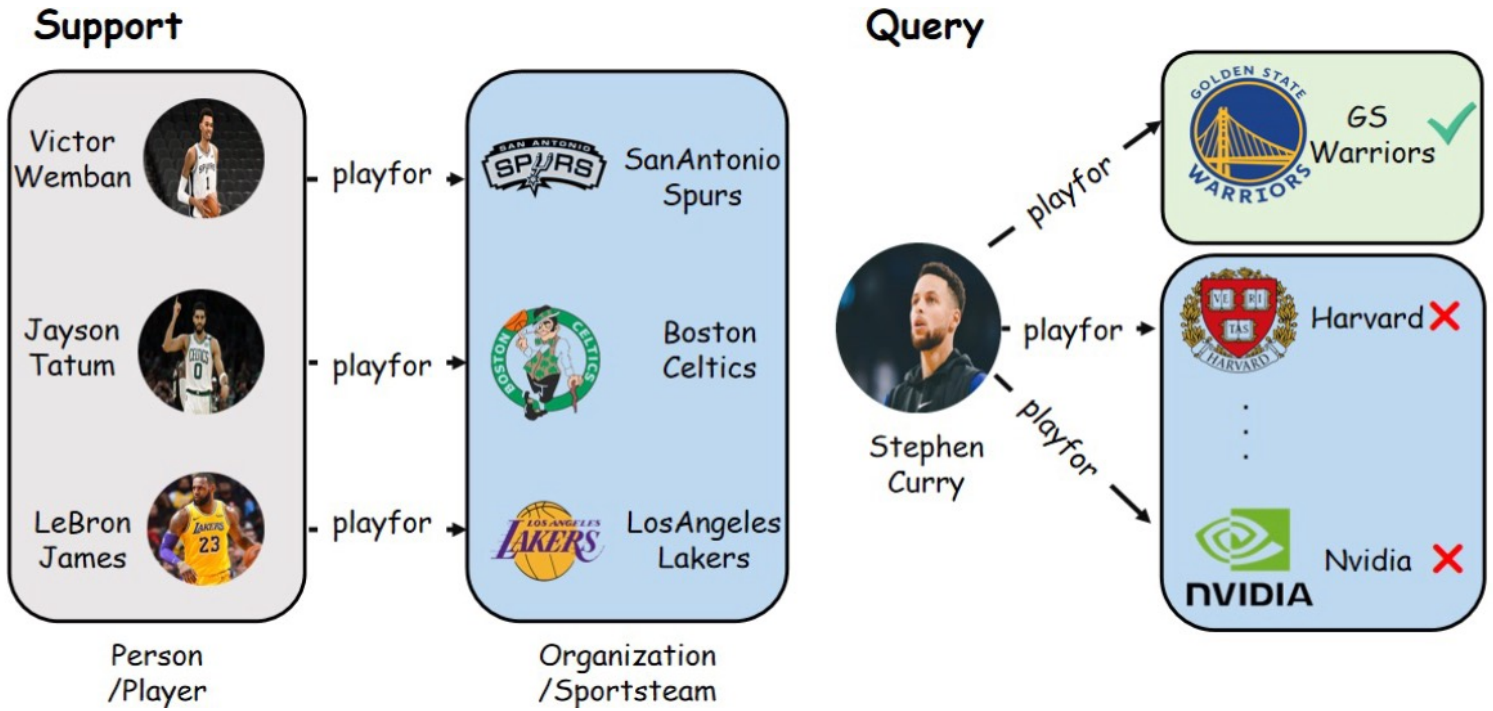


Final Knowledge Graph:

['Alan Shepard', 'birthDate', 'Nov 18, 1923']
['Alan Shepard', 'mission', 'Apollo 14']
['Alan Shepard', 'selectedByNasa', '1959']

[1] https://arxiv.org/pdf/2404.03868

# LGC-CR: Few-shot Knowledge Graph Completion via LocalGlobal Contrastive Learning and LLM-Guided Refinement [CIKM'25]

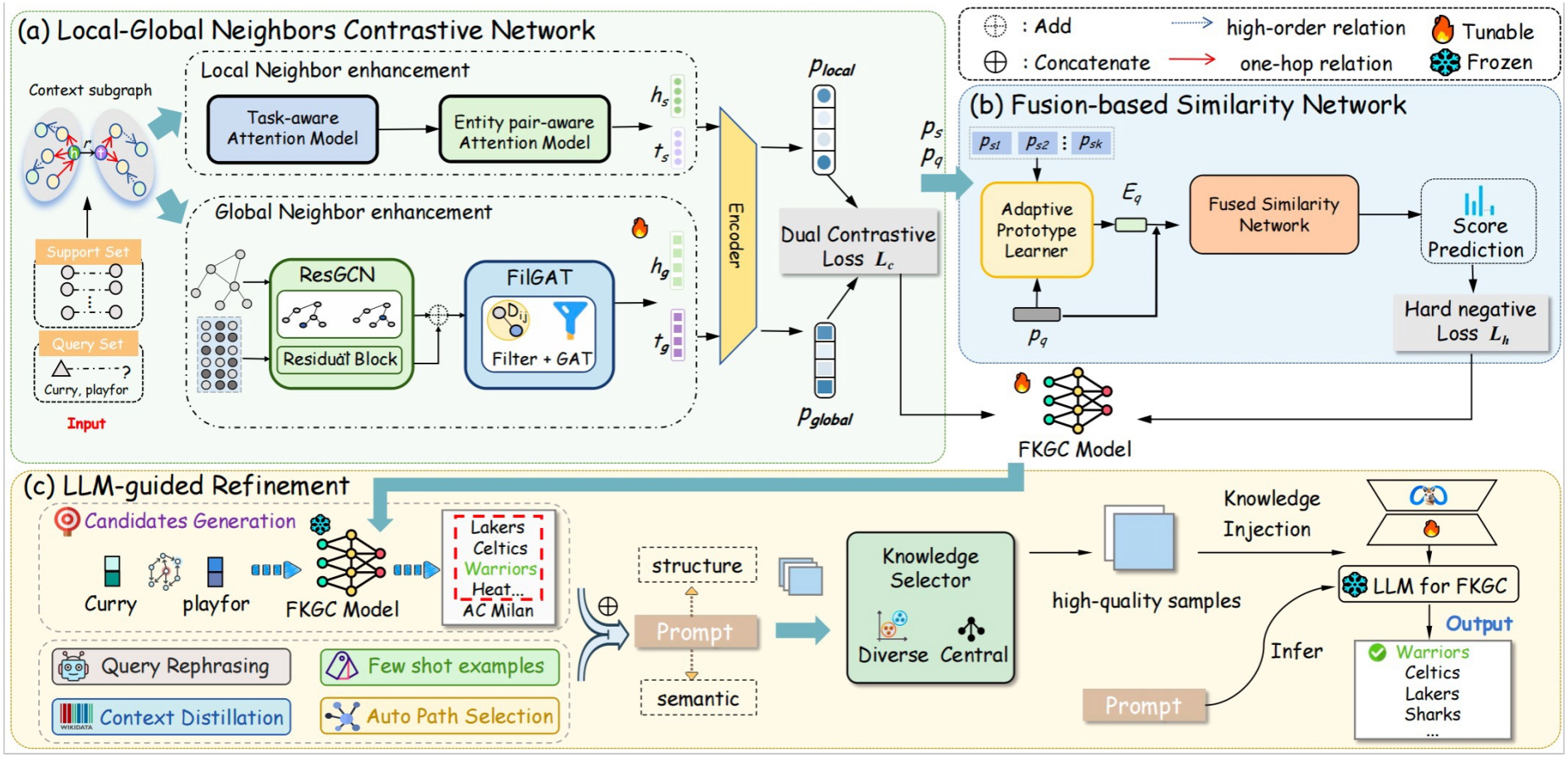



- How to effectively achieve knowledge graph completion under few-shot conditions?

few-shot relation: playfor

[1] https://dl.acm.org/doi/epdf/10.1145/3746252.3761000

# LGC-CR: Few-shot Knowledge Graph Completion via LocalGlobal Contrastive Learning and LLM-Guided Refinement [CIKM'25]

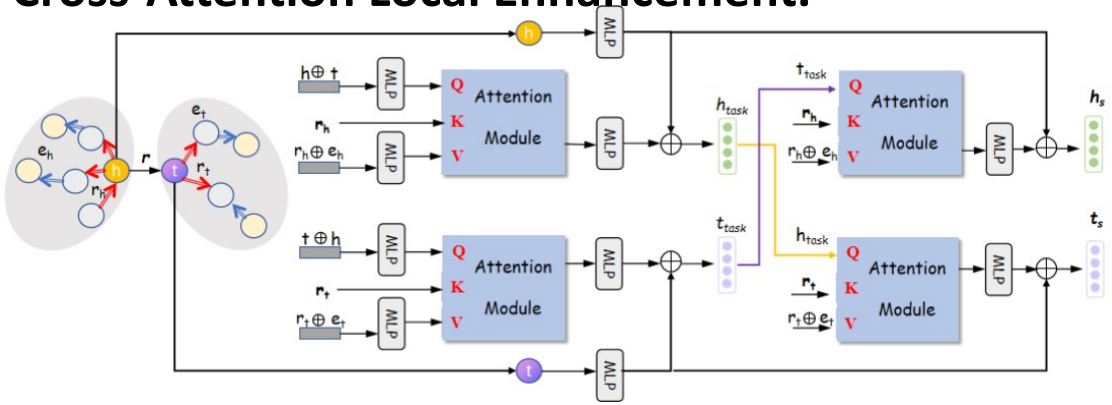



(a) Local-Global Neighbors Contrastive Network (b) Fusion-based Similarity Network (c) LLM-guided Refinement

[1] https://dl.acm.org/doi/epdf/10.1145/3746252.3761000

# LGC-CR: Few-shot Knowledge Graph Completion via LocalGlobal Contrastive Learning and LLM-Guided Refinement [CIKM'25]

**Cross-Attention Local Enhancement.**

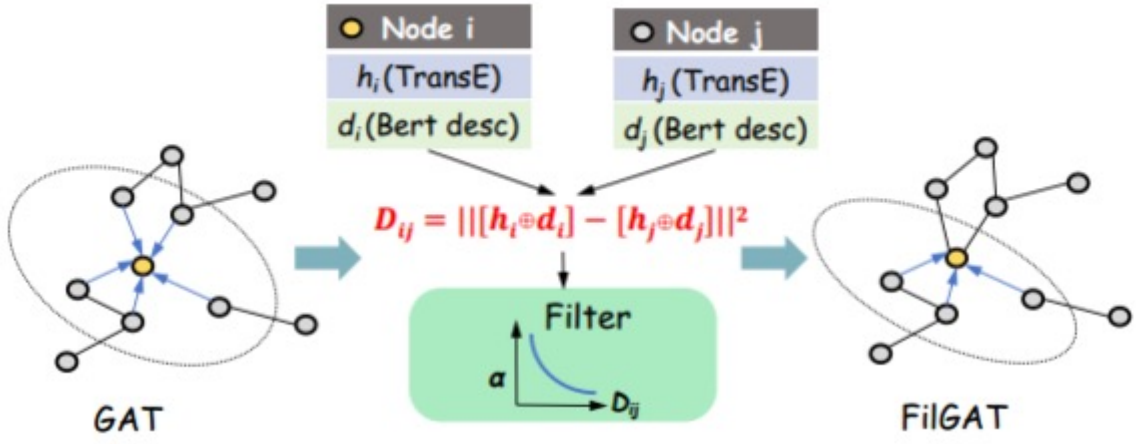



(1)Task-aware (2)Entity pair-aware

(1) Neighbors more task-relevant receive higher weights.

(2) Neighbors more similar to paired entities receive higher weights.

**ResGCN-FilGAT Global Enhancement.**

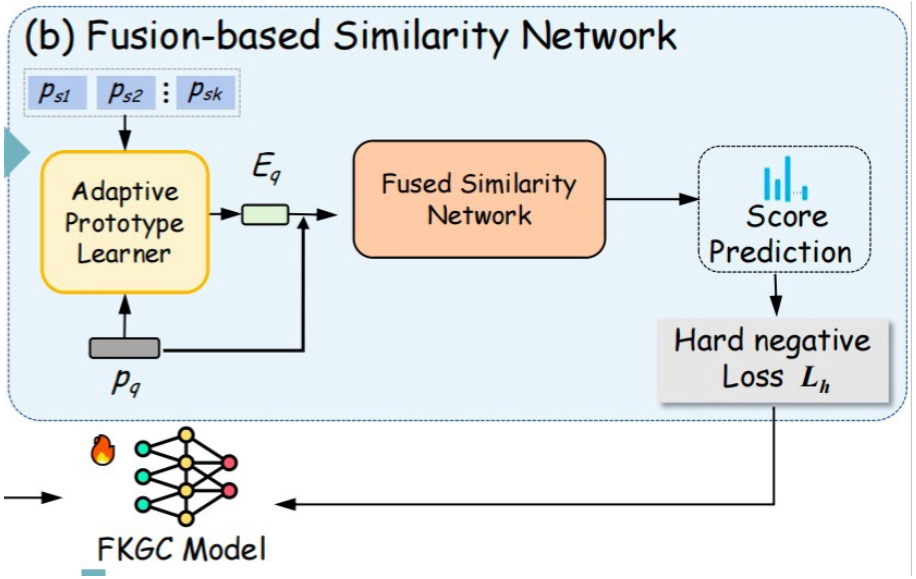



**one-hop:** stable and comprehensive

**high-order:** Not all neighbors matter: adaptive filtering based on structure-semantic dissimilarity.

[1] https://dl.acm.org/doi/epdf/10.1145/3746252.3761000

# LGC-CR: Few-shot Knowledge Graph Completion via LocalGlobal Contrastive Learning and LLM-Guided Refinement [CIKM'25]




- A fusion similarity network

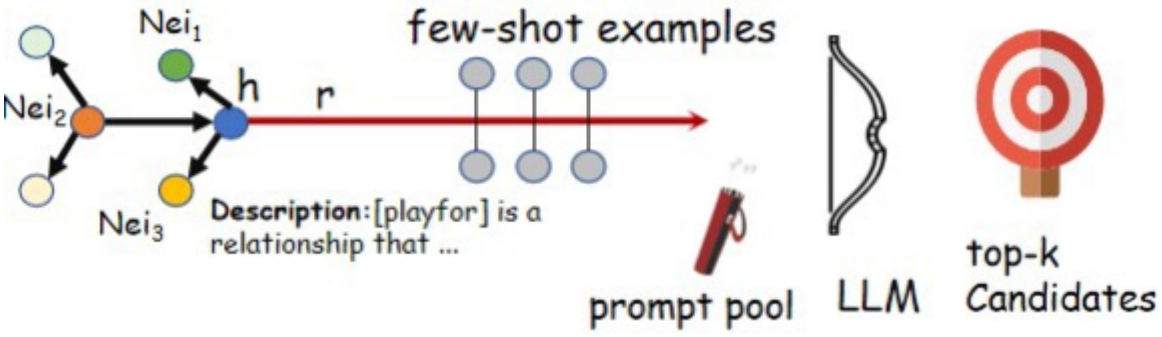
integrates three **complementary similarity metrics**, which constructs a **learnable non-linear** metric space.

- Hard Negative Loss

Instead of treat all negative samples equally, **focus more on hard negative samples** that are difficult to distinguish during training.

[1] https://dl.acm.org/doi/epdf/10.1145/3746252.3761000

# LGC-CR: Few-shot Knowledge Graph Completion via LocalGlobal Contrastive Learning and LLM-Guided Refinement [CIKM'25]




**> Prompt Template**

You are an expert in refining initial meta-learning outputs. Please refine the list of tail entities based on the given information, select 10 winnners and output as ent1 | ent2...

***Query***: Which team does Curry play for?
What team is Curry currently on?

**few-shot examples:**
Lebron James, playfor, Lakers...

**Entity Des:** American basketball....
**Relation Des:** [playfor] is a relation...

***Path triples:***
Draymond Green, teammates, Curry
Curry, team_location, San Francisco, venue_of, Chase

**top-m entities:**
Lakers, Celtics, Warriors, Heat, ...

- **Instruction Retrieval**

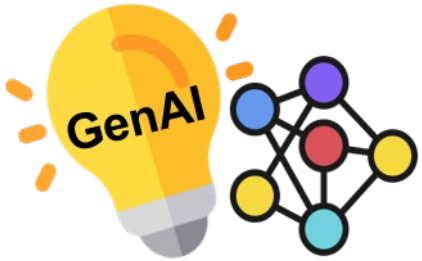
Retrieve query-relevant **structural and semantic knowledge**.

- **Knowledge Selector**

*Diversity-aware Sampling:* selects **representative relations** covering diverse KG information.

*Centrality-aware Sampling:* focusing on **high-centrality relations** while avoiding noise.

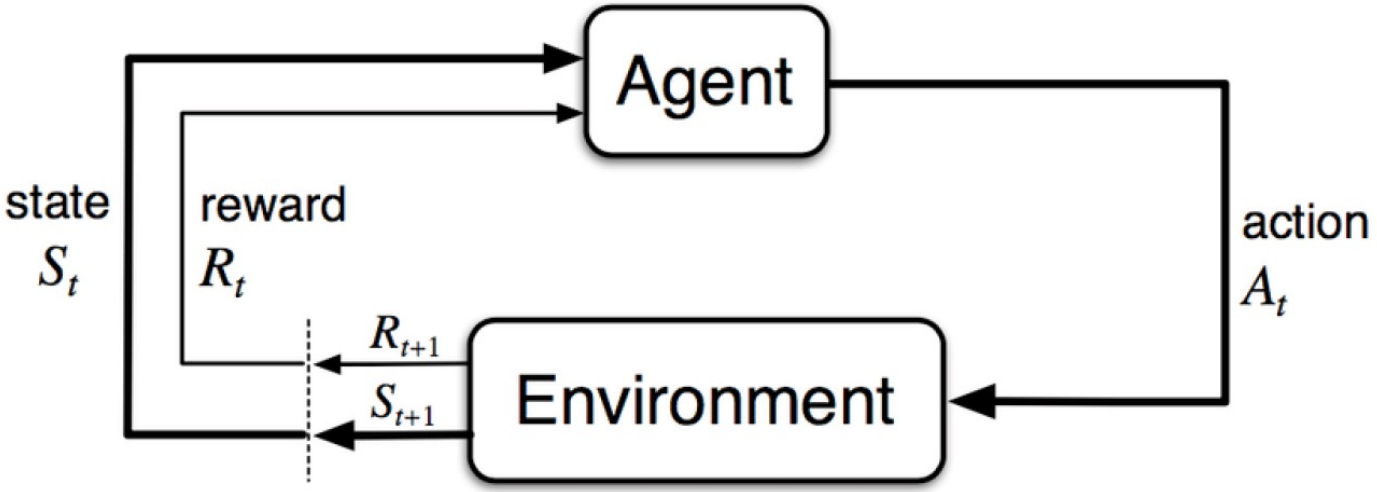
# 6. Graphs for AI Agents



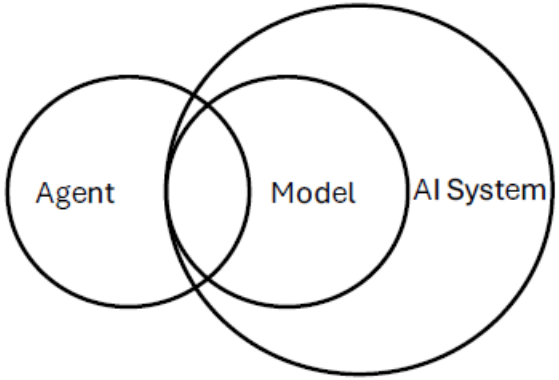

- AI Agents
- Agent Interaction With Task Planning Graphs
- Agent Interaction With Task Execution Graphs
- Agent Interaction With Memory Graphs
- Multi-Agent Coordination Graphs

Presented by Arijit Khan

# Agents - Introduction




**What would we like an agent to do? -** ACT!
Agentic systems autonomously make decisions and take actions to achieve a specified goal, operating within the constraints of their environment and design.

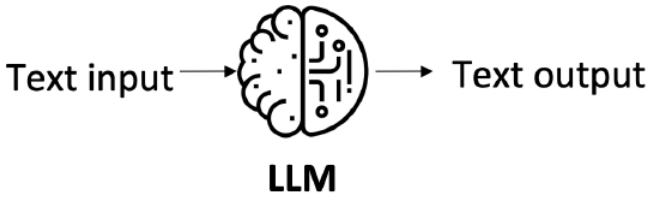


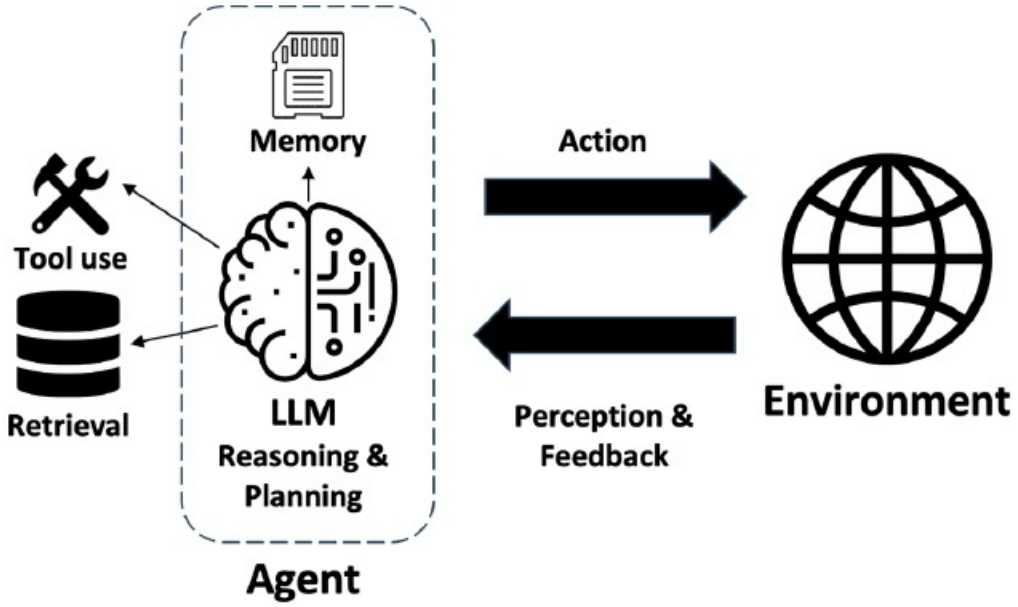

**Components of an Agent:**

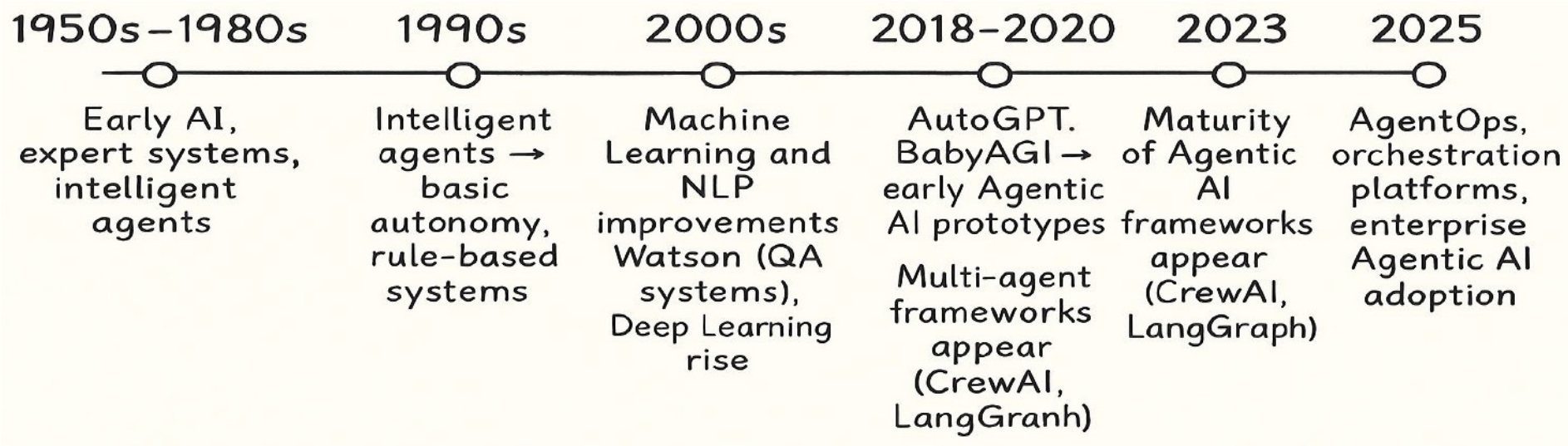
•**Required:** Grounding, Agency, Planning, Memory, Learning

•**Additional:** Embodiment, Communication, World Modeling, Multimodality

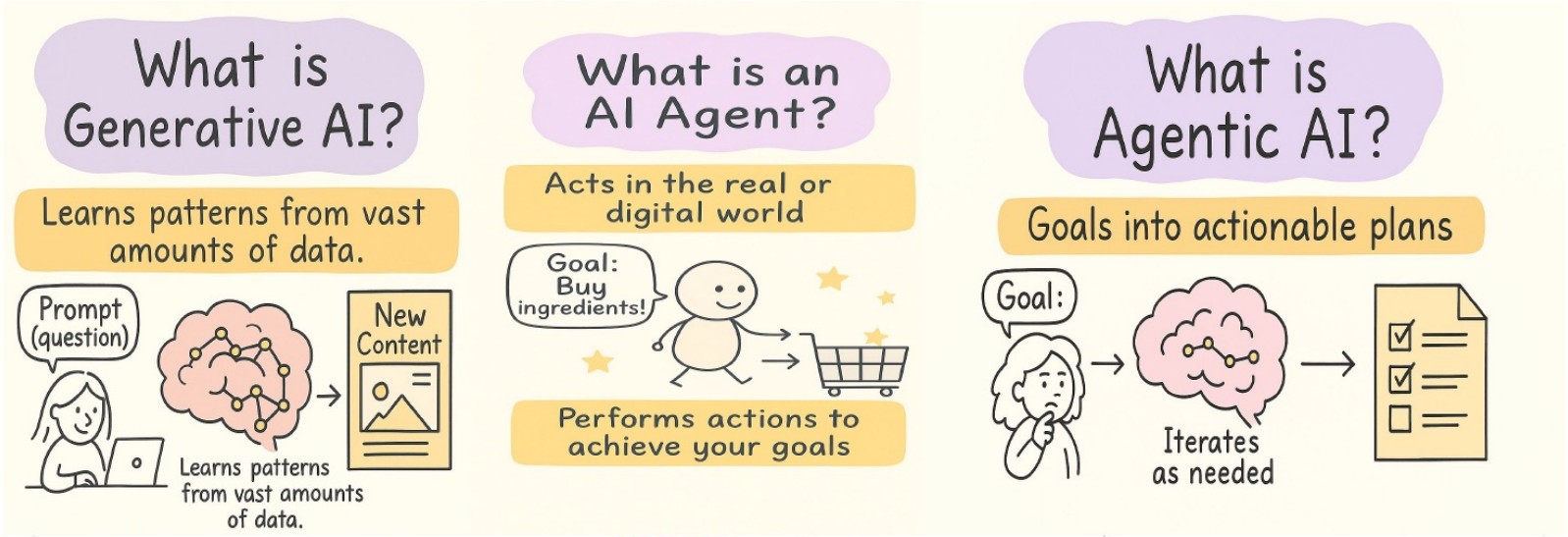
# AI Agents

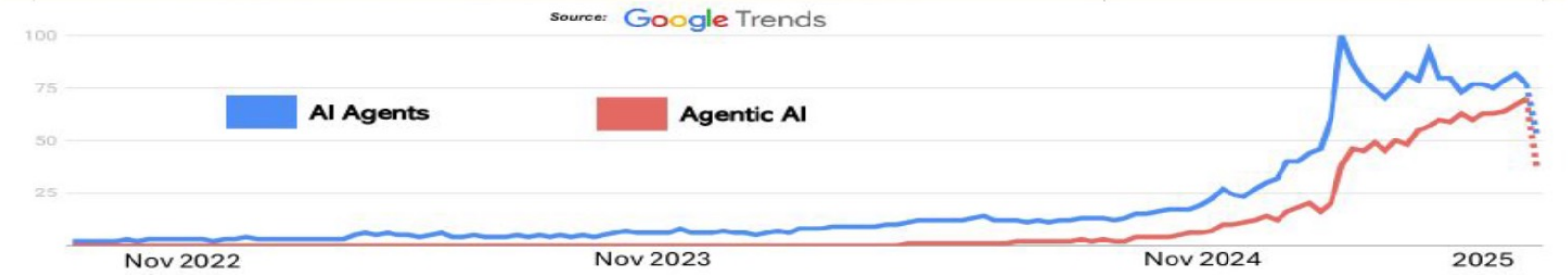

Text input
Text output
LLM


Memory
Tool use
Retrieval
LLM
Reasoning &
Planning
Agent
Action
Perception &
Feedback
Environment


1950s–1980s
Early AI, expert systems, intelligent agents
1990s
Intelligent agents → basic autonomy, rule-based systems
2000s
Machine Learning and NLP improvements Watson (QA systems), Deep Learning rise
2018–2020
AutoGPT. BabyAGI → early Agentic AI prototypes
Multi-agent frameworks appear (CrewAI, LangGranh)
2023
Maturity of Agentic AI frameworks appear (CrewAI, LangGraph)
2025
AgentOps, orchestration platforms, enterprise Agentic AI adoption

# AI Agents

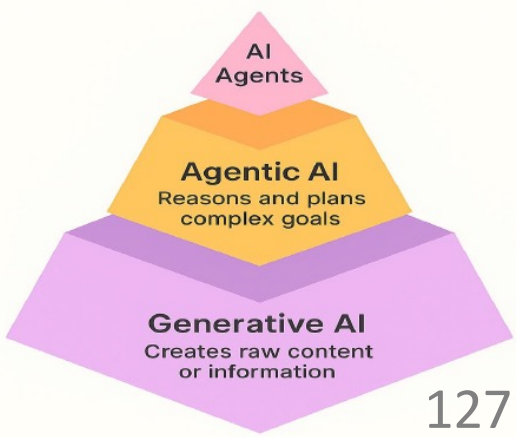


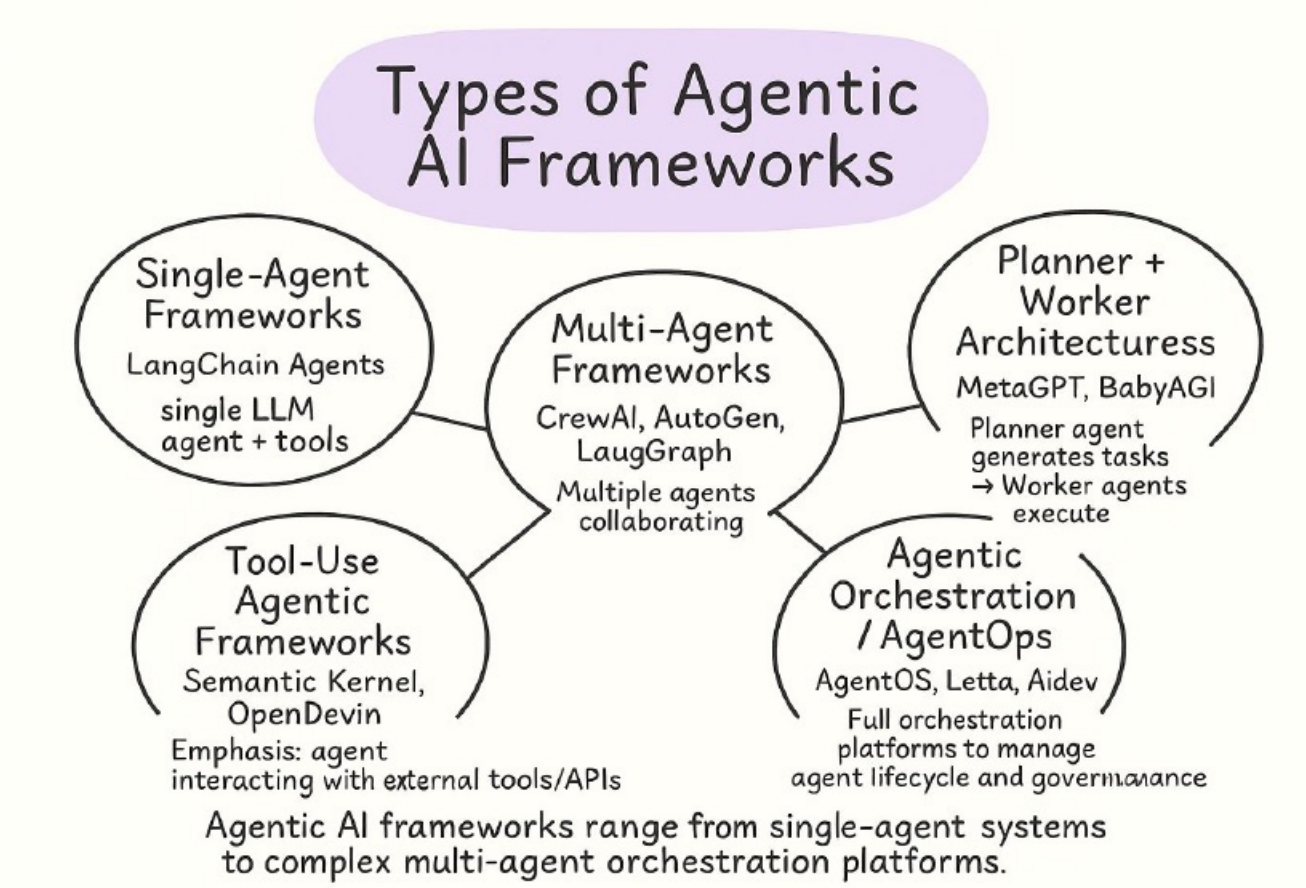

**Generative AI** produces content such as text, code, or intermediate reasoning steps.

**Agentic AI** refers to architectures that use these models to plan, reason, and orchestrate multi-step workflows.

**AI Agents** are the concrete entities that carry out those workflows by invoking tools, interacting with environments, and executing actions.



# Agentic AI Frameworks



# Graphs Meet AI Agents [Bei et al., https://arxiv.org/abs/2506.18019]

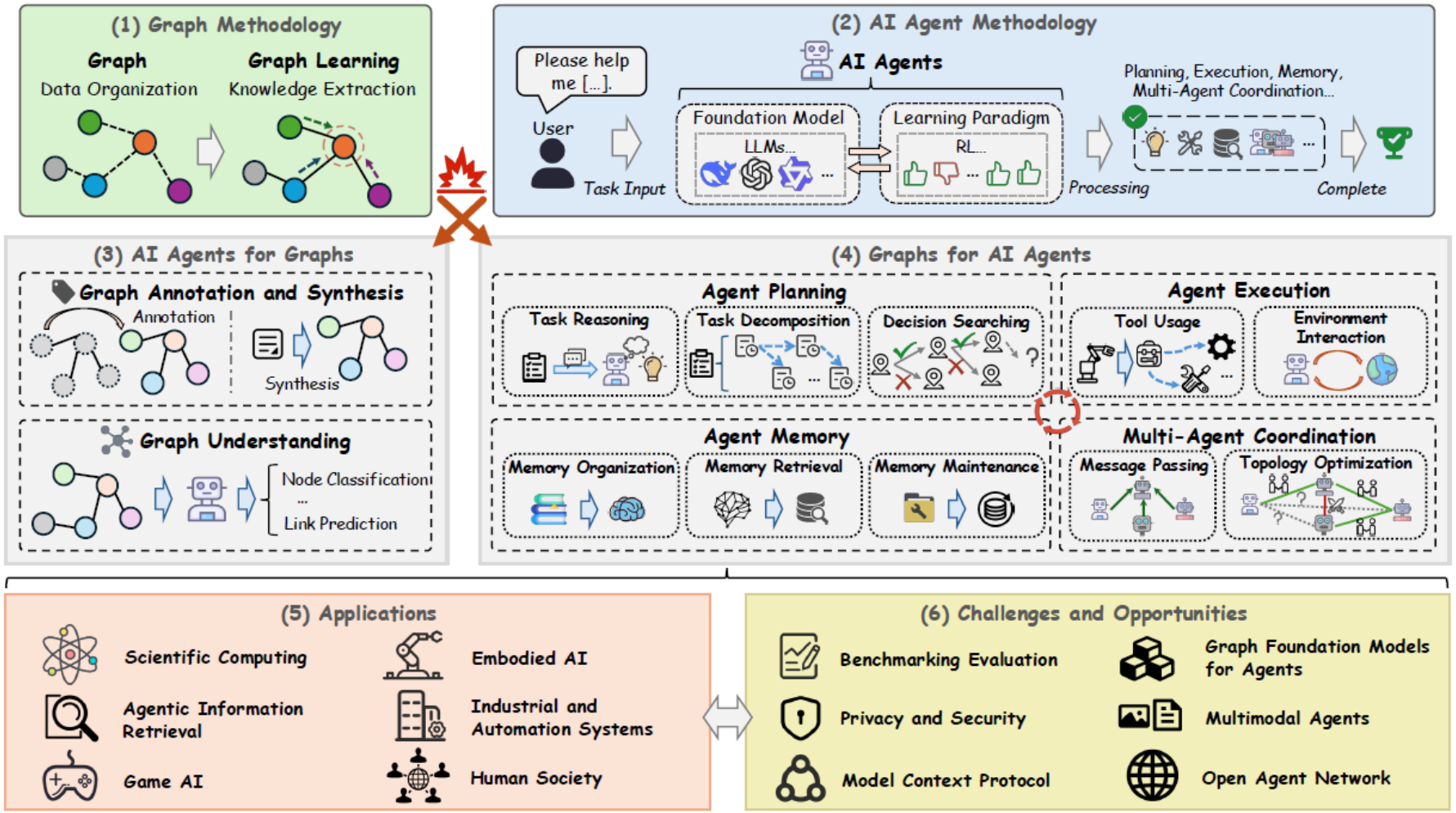



Fig. 1: An overall illustration of graphs meet AI agents. (1) Graph Methodology: graph data organization and knowledge extraction with graphs. (2) AI Agent Methodology: LLM-based foundation models and RL-based learning paradigms, forming core AI Agent pipelines. (3) AI Agents for Graphs: the powerful capabilities of agents in graph modeling and learning, such as graph annotation, synthesis, and understanding. (4) Graphs for AI Agents: the role and potential of graphs and graph learning in empowering the core functionalities of agents, including agent planning, execution, memory, and multi-agent coordination. (5) Representative Applications. (6) Challenges and Future Opportunities.

# Graphs for Agentic Planning

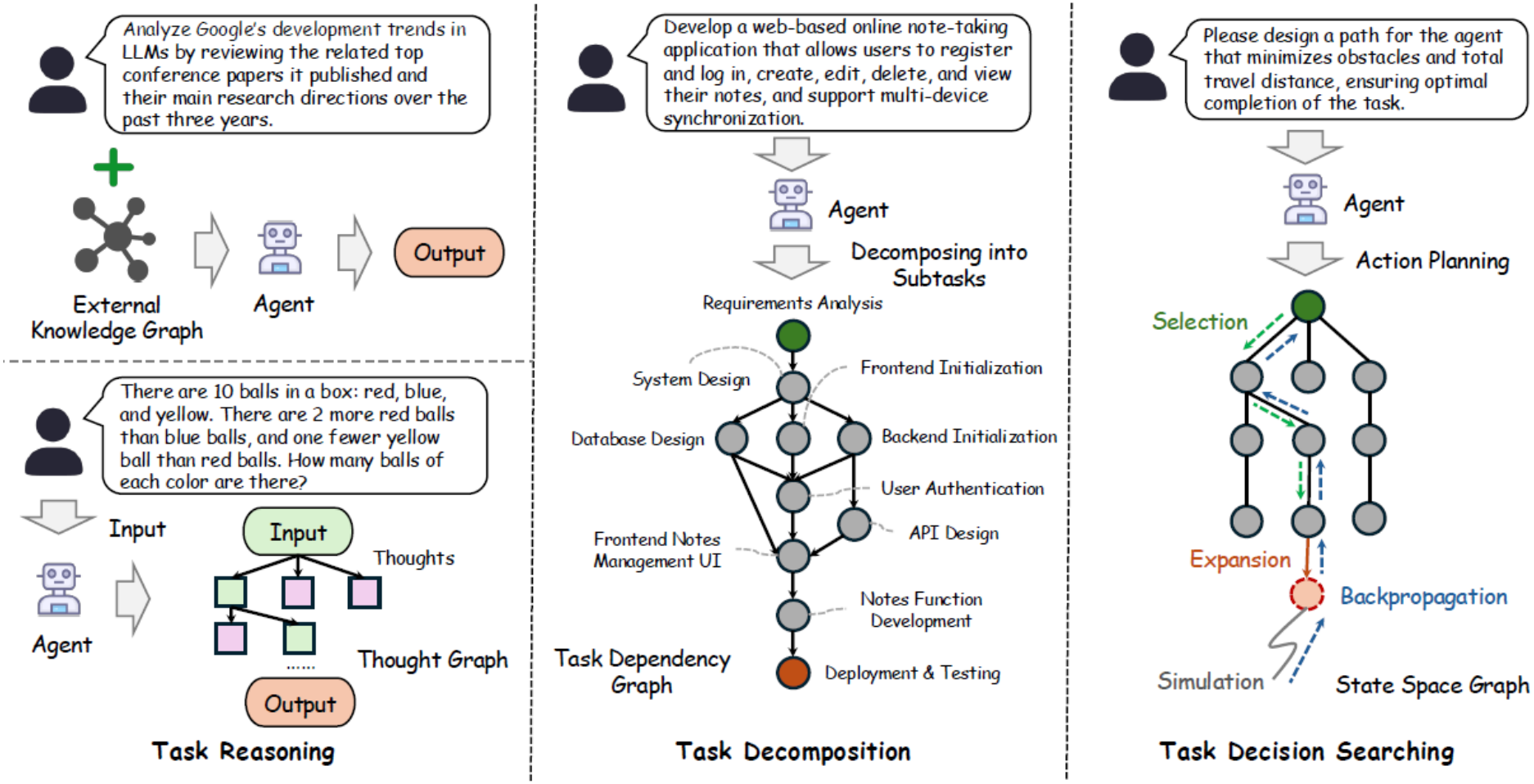



Fig. 3: An illustration of graphs for agent planning.

# Graph-of-Thought Reasoning

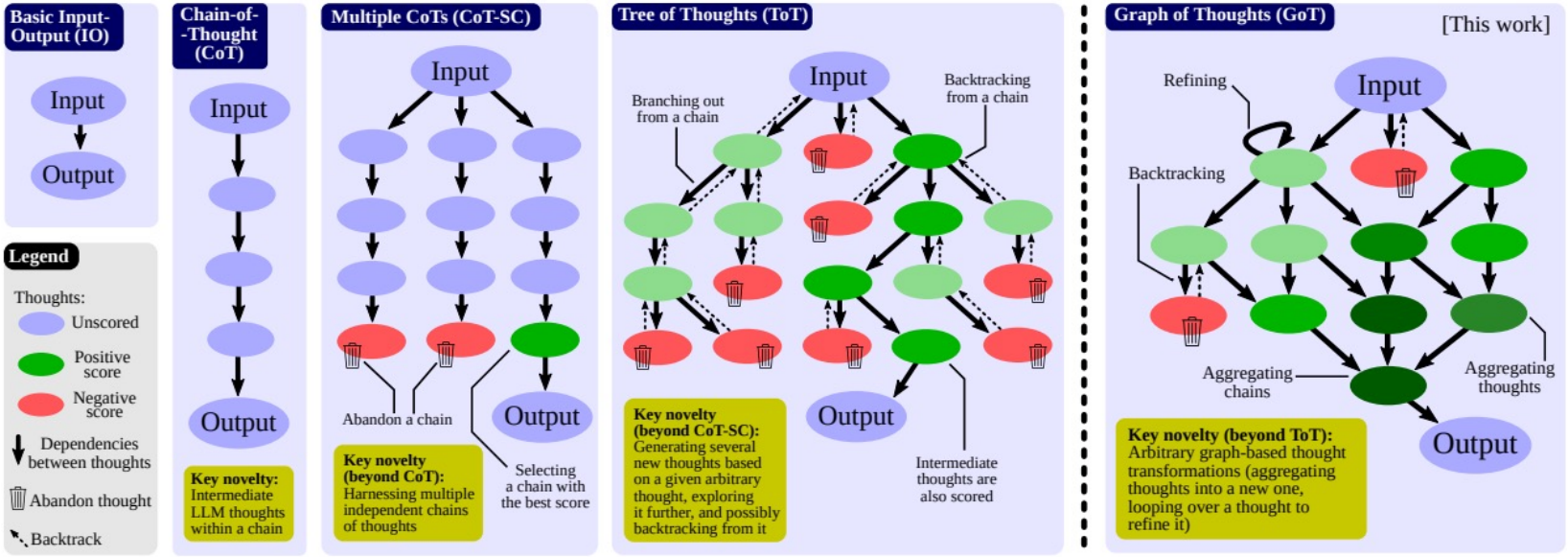



Figure 1: Comparison of Graph of Thoughts (GoT) to other prompting strategies.

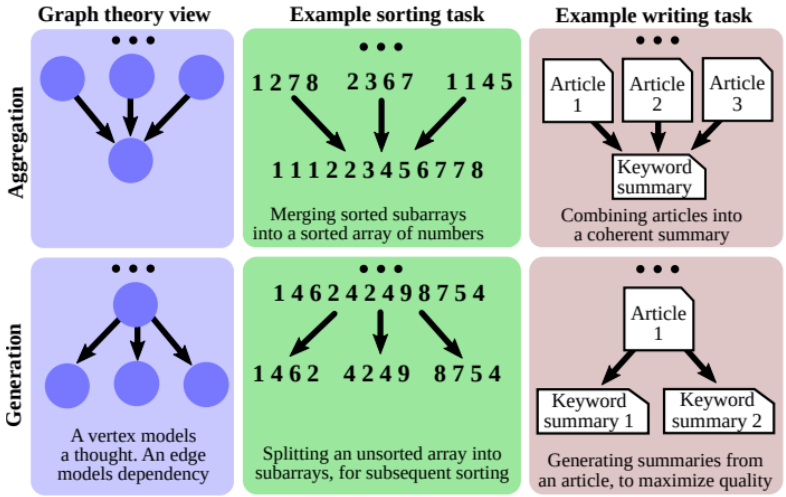



Figure 2: Examples of aggregation and generation thought transformations.

**Graph of Thoughts: Solving Elaborate Problems with Large Language Models**


**Maciej Besta[1*], Nils Blach[1*], Ales Kubicek[1], Robert Gerstenberger[1], Michał Podstawski[2], Lukas Gianinazzi[1], Joanna Gajda[3], Tomasz Lehmann[3], Hubert Niewiadomski[3], Piotr Nyczyk[3], Torsten Hoefler[1]**

[1]ETH Zurich, [2]Warsaw University of Technology, [3]Cledar

# Task Decomposition and Dependency Graph

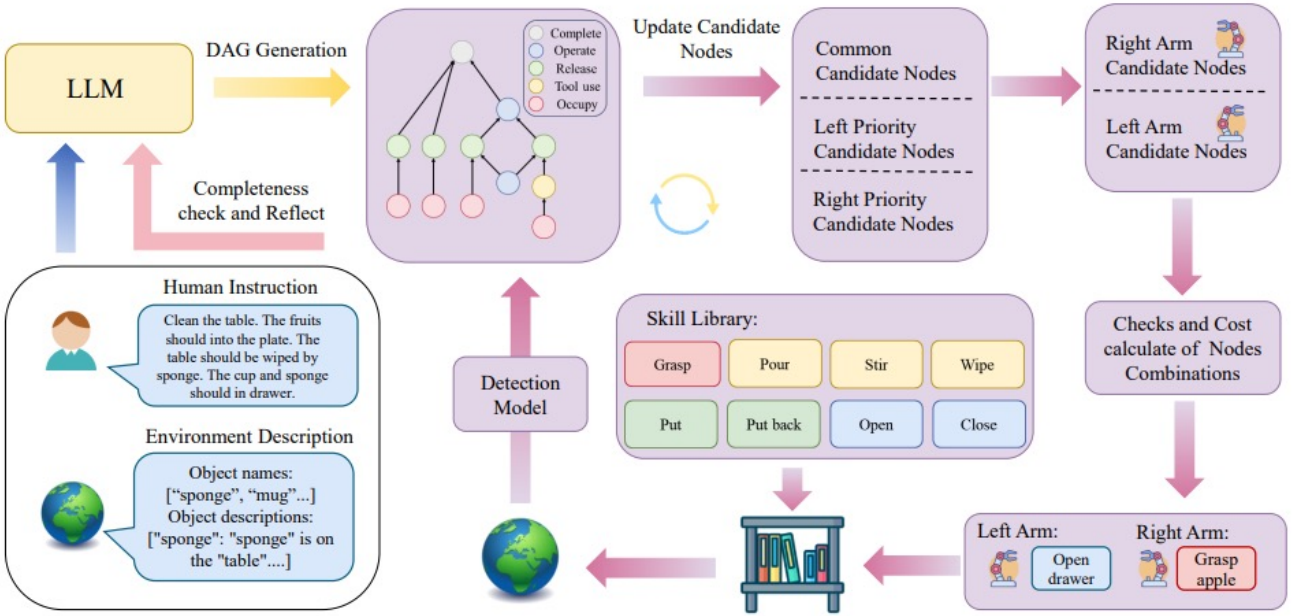
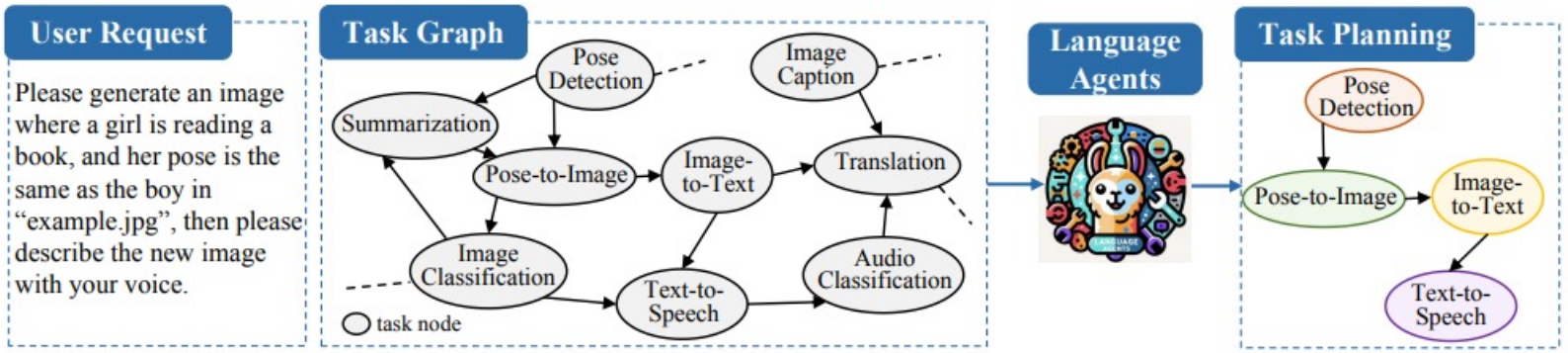



Fig. 1. An overview of DAG-Plan. The DAG-Plan generates a DAG based on human instruction and environmental description. It checks the graph's completeness and reflects the LLM to regenerate if incomplete. Once a valid DAG is obtained, DAG-Plan performs task inference to identify executable candidate nodes. The occupied arm and free arm are assigned priority candidate nodes and common candidate nodes respectively. The framework then evaluates all candidate combinations for feasibility and cost. DAG-Plan selects the nodes with the lowest cost and employs skill in library for execution. DAG-Plan updates the graph, iterating inference until the DAG is fully executed.

Z. Gao, Y. Mu, J. Qu, M. Hu, L. Guo, P. Luo, and Y. Lu, "Dag-plan: Generating directed acyclic dependency graphs for dual-arm cooperative planning," arXiv preprint arXiv:2406.09953, 2024




X. Wu, Y. Shen, C. Shan, K. Song, S. Wang, B. Zhang, J. Feng, H. Cheng, W. Chen, Y. Xiong et al., "Can graph learning improve planning in llm-based agents? NeurIPS 2024.

Figure 1: **Illustration of Task Planning in Language Agents (e.g., HuggingGPT [46])**

# Graphs for Agentic Execution

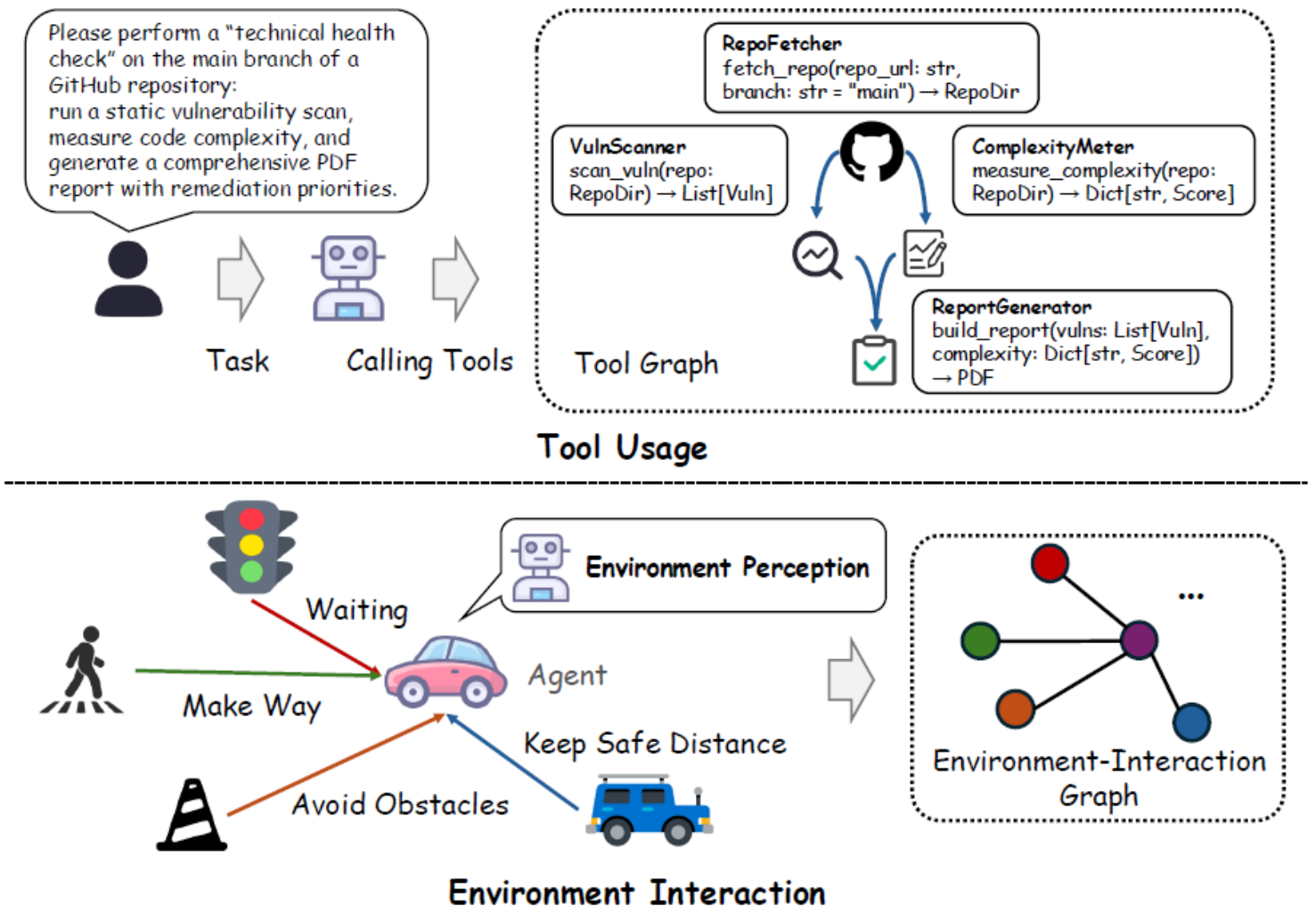



Fig. 4: An illustration of graphs for agent execution.

# Graph-based Agentic Tool Learning

01

**HuggingGPT**

Solving Diverse AI Tasks by Connecting ChatGPT with Hugging Face Models

02

**TaskBench**

A Comprehensive Benchmark for Evaluating LLM Agents on Task Automation

03

**Tool-Planner**

Enhancing Task Planning with Clusters across Multiple Specialized Tools

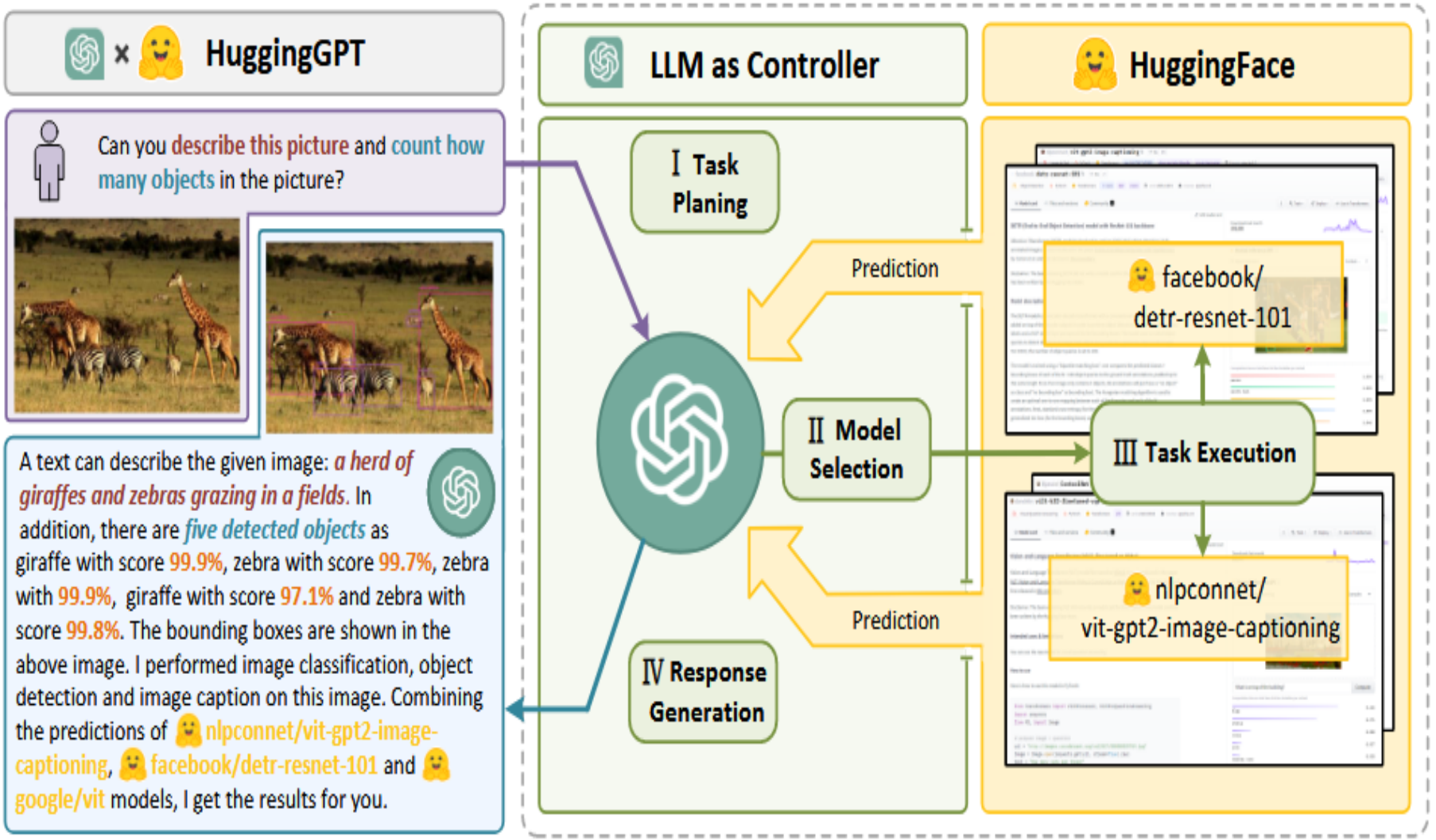
# HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face

**Yongliang Shen**[1,2,*], **Kaitao Song**[2,*,†], **Xu Tan**[2],
**Dongsheng Li**[2], **Weiming Lu**[1,†], **Yueting Zhuang**[1,†]
Zhejiang University[1], Microsoft Research Asia[2]
{syl, luwm, yzhuang}@zju.edu.cn, {kaitaosong, xuta, dongsli}@microsoft.com

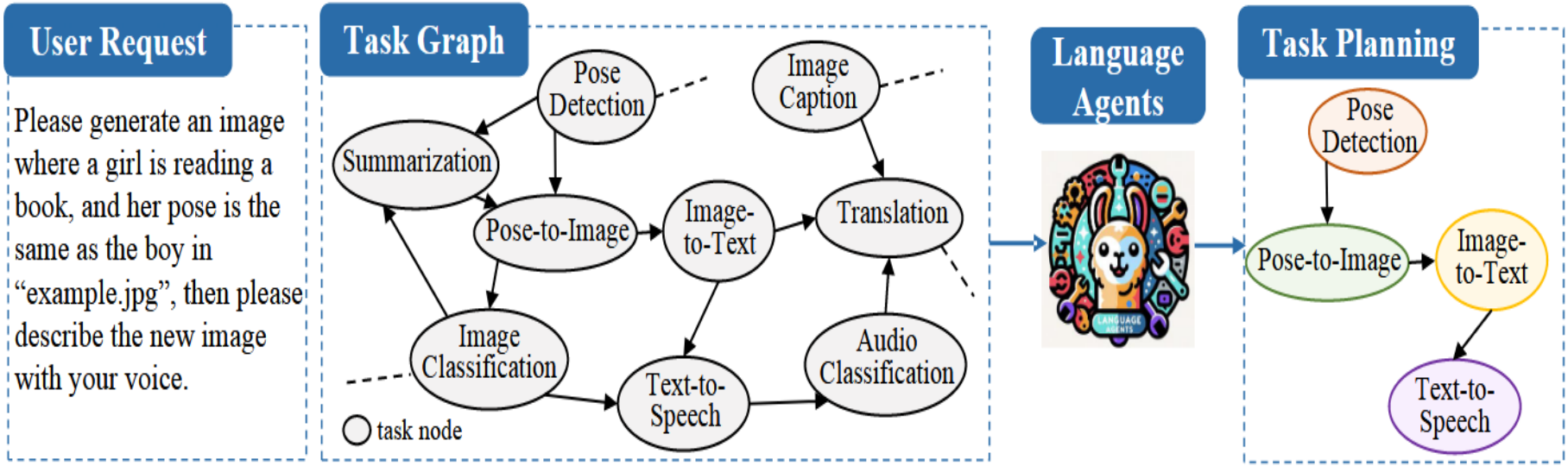
https://github.com/microsoft/JARVIS

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

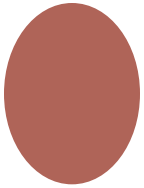
# HuggingGPT: The Four-Stage Pipeline



# Illustration of Task Planning in HuggingGPT

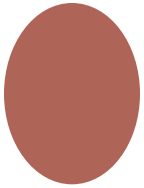


# HuggingGPT: Key Results & Findings



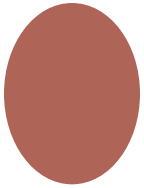
## Effectiveness on Complex Tasks

HuggingGPT successfully completed a wide range of challenging tasks that are difficult for any single model, demonstrating the power of the "LLM as a controller" paradigm.



## Performance Dependence on Controller

The choice of controller LLM (GPT-3.5 vs. GPT-4) significantly impacted results. GPT-4, with its superior reasoning ability, led to a much higher success rate in task planning and model selection.



## Key Limitations Identified

**Latency:** Significant delays from chaining multiple calls.
**Error Propagation:** Planning mistakes lead to complete failure.
**Context Limit:** Constrained by the LLM's context window size.

## Broader Impact: The Catalyst for LLM Agents

HuggingGPT demonstrated that the true power of LLMs lies not just in their internal knowledge, but in their ability to interact with and leverage external tools and models. It acted as a critical catalyst for the "LLM Agent" movement, shifting focus towards building autonomous systems that can coordinate specialized models to solve complex real-world problems.

# TaskBench: Benchmarking Large Language Models for Task Automation

**Yongliang Shen**[1], **Kaitao Song**[2†], **Xu Tan**[2], **Wenqi Zhang**[1],
**Kan Ren**[2], **Siyu Yuan**[3], **Weiming Lu**[1†], **Dongsheng Li**[2], **Yueting Zhuang**[1†]
Zhejiang University[1], Microsoft Research Asia[2], Fudan University[3]
{syl,luwm,zhangwenqi}@zju.edu.cn, syyuan21@m.fudan.edu.cn
{kaitaosong, xuta, dongsli}@microsoft.com

https://github.com/microsoft/JARVIS

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks.

# TaskBench: Motivation & Core Problem

## The Core Problem

As LLM agents become more prevalent, there is a lack of **standardized, comprehensive benchmarks** to evaluate their ability to automate complex tasks.

Existing benchmarks often focus on **isolated capabilities**(e.g., coding, reasoning) rather than the **holistic ability** to complete end-to-end tasks.

### Task Complexity

Covering a wide spectrum of difficulties to challenge agents at various skill levels.

### Tool Usage

Incorporating external tools (APIs, databases) to mimic real-world workflows.

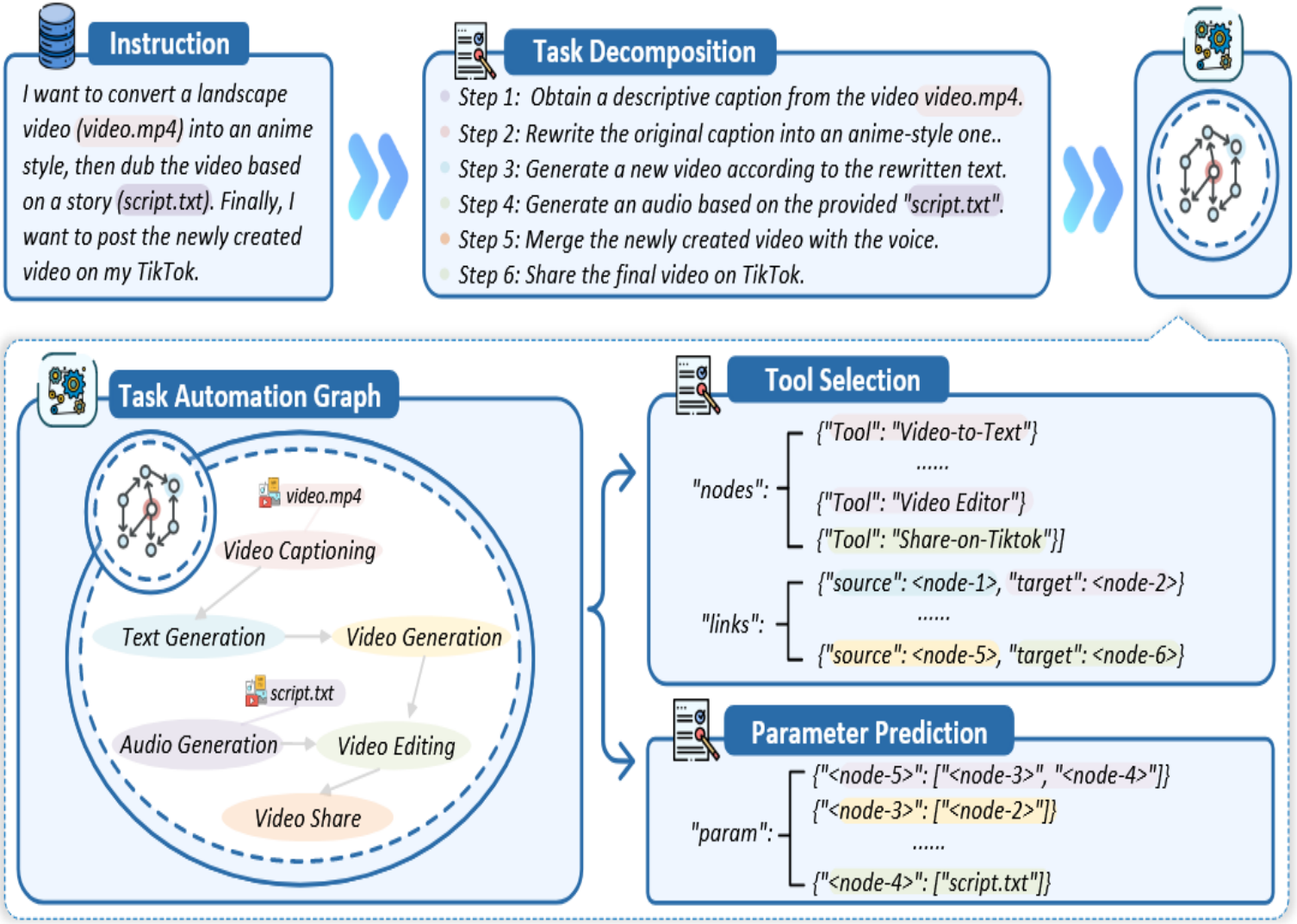
### Reasoning Steps

Requiring agents to plan, decompose, and execute long, multi-step sequences.

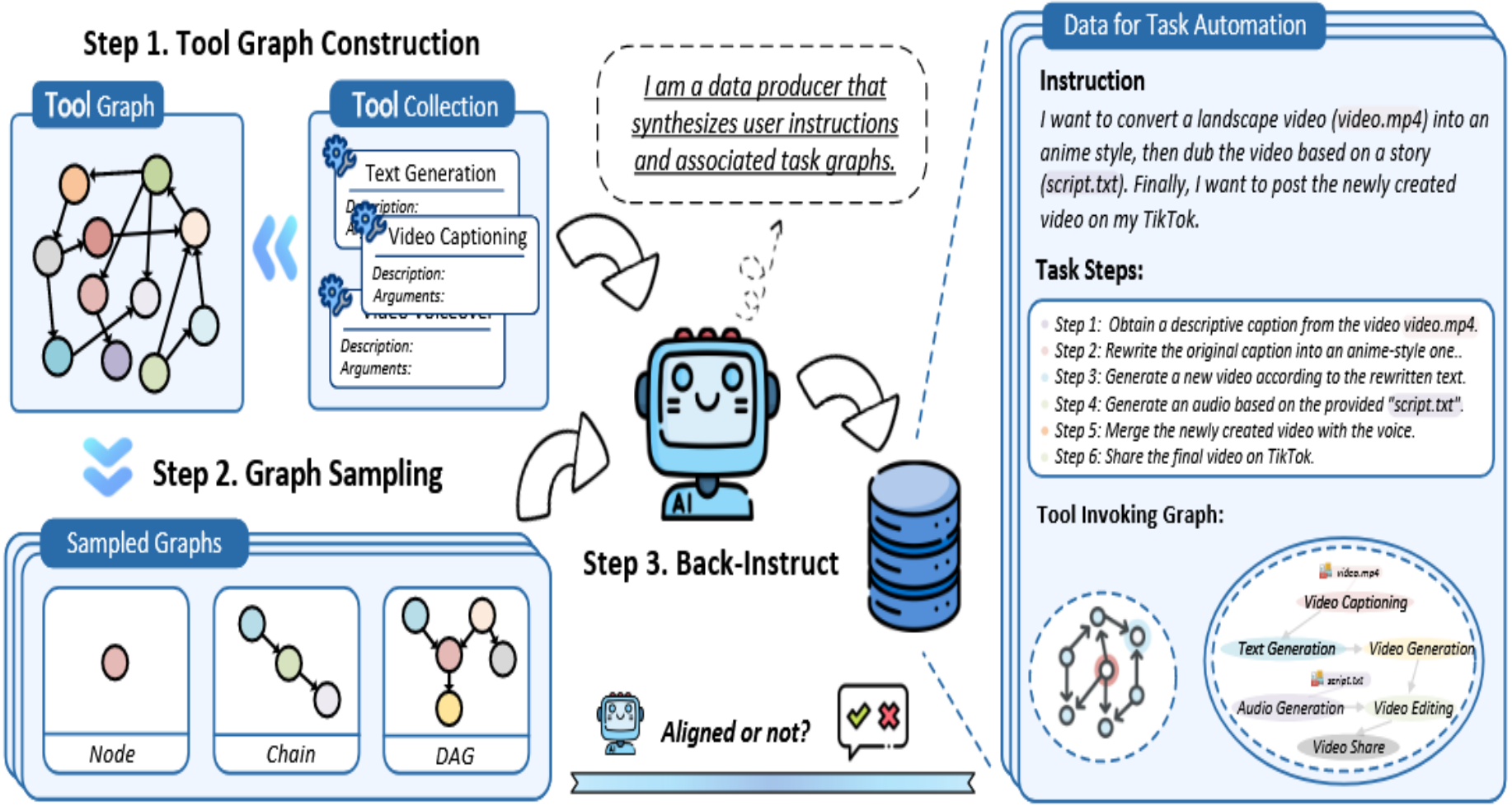
### Real-World Scenarios

Using practical, meaningful tasks that reflect actual industry and daily needs.

# LLM-Based Task Automation



# TaskBench: Benchmark Design



# TaskBench: Benchmark Design

## 01. Task Taxonomy

A comprehensive classification system defining tasks along three key dimensions:

**Domain Scope**
Software Dev, Data Analysis, Support

**Complexity Level**
Steps count, tooling needs, cognitive load

**Functional Type**
Extraction, Transformation, Decision-Making

## 02. Task Creation

Tasks are engineered for realism, mirroring real-world automation scenarios with well-defined parameters:

**Clear Definition:**Specific goal, input data & output.

**Example Scenario:**
Process CSV sales data → Calculate regional revenue → Generate chart → Write summary.

## 03. Evaluation Metrics

A multi-dimensional assessment framework to quantify agent performance:

**Success Rate**
Ability to complete the task correctly.

**Efficiency**
Minimizing steps and token consumption.

**Robustness**
Handling input variations & unexpected errors.

# TaskBench: Key Features & Capabilities Tested

## Tool Usage

**APIs:** RESTful services for real-time data retrieval.
**DBs:** SQL queries for data manipulation.
**Code:** Python execution for data processing.
**Files:** Full read/write filesystem interaction.

## Multi-step Planning

Agents must execute tasks in a **strict logical sequence**, understanding complex dependencies between steps.

Example: Calculate quarterly revenue first, then use that result to generate a trend visualization.

## Robustness & Adaptability

Designed to test resilience against **ambiguous inputs** and **simulated failures**.
Example: Clarifying vague user requests or recovering gracefully from a broken API endpoint.

# TaskBench: Example Task Walkthrough

## Task: Customer Support Ticket Resolution

### Primary Goal

Successfully resolve a customer complaint regarding a delayed order by gathering info and providing a resolution.

### Initial Input

A raw customer support ticket containing the specific Order ID and a free-text description of the delay issue.

### 01. Query Order Database

Execute SQL query using the Order ID to fetch critical details (current status, shipping address).

### 02. Check Shipping API

Call external shipping API to retrieve real-time package location and latest delivery status update.

### 03. Analyze & Decide

Evaluate status; offer discount if delay is significant.

### 04. Generate Response

Draft a personalized email with tracking info and resolution details (e.g., unique discount code).

### 05. Log Final Action

Update the support ticket database with the full resolution log for future reference.

### TASK OUTCOME

Validates agent's ability to handle DB, API, reasoning, and generation.

# TaskBench: Experimental Results

Key Findings from Initial Benchmarking

### Performance Variability

A significant performance gap exists between LLMs. Top-tier models (GPT-4, Claude 3) consistently outperformed smaller models in success rate and task efficiency.

### Tool Mastery is Key

Models demonstrating strong tool understanding performed significantly better. This includes correctly formatting API calls and accurately parsing JSON responses from tools.

### Planning vs. Execution

Capability divergence was observed: some models excelled at creating detailed plans but failed in execution (e.g., syntax errors), while others were strong executors but poor planners.

### Common Failure Modes

Key breakdowns included incorrect tool selection for subtasks, passing invalid arguments (parameter errors), and planning oversights (missing critical workflow steps).

# TaskBench: Discussion & Significance

### Standardized Evaluation

Established a common ground for comparing the task automation capabilities of diverse LLM agents objectively.

### Revealing Limitations

Systematically identified key failure modes, providing crucial insights to guide targeted R&D for future agents.

### Promoting Innovation

Encourages healthy competition and drives the evolution of LLM-based automation.

## Broader Impact & Value

### A Diagnostic Tool (Not Just a Test)

TaskBench helps developers understand**why**an agent failed a task, delivering actionable, granular feedback that accelerates the iteration and improvement cycle.

### Cornerstone of the LLM Agent Community

It has become the de facto standard for evaluating progress, enabling researchers worldwide to build on each other's work and move the field forward collectively.

Published as a conference paper at ICLR 2025

# TOOL-PLANNER: TASK PLANNING WITH CLUSTERS ACROSS MULTIPLE TOOLS

**Yanming Liu**[1], **Xinyue Peng**[2], **Jiannan Cao**[3], **Shi Bo, Yuwei Zhang, Xuhong Zhang**[1]*,
**Sheng Cheng**[1], **Xun Wang**[1], **Jianwei Yin**[1], **Tianyu Du**[1]*
[1]Zhejiang University, [2]Southeast University
[3]Massachusetts Institute of Technology
{oceann24, zhangxuhong, zjradty}@zju.edu.cn, zjuyjw@cs.zju.edu.cn,
jiannan@mit.edu, xinyuepeng@seu.edu.cn, yuweizhang@tongji.edu.cn

# Tool-Planner: Motivation & The Problem of Multi-Tool Coordination

## The Problem

Many complex tasks cannot be solved by a single tool or a simple sequence. They require the**coordinated**use of multiple tools with overlapping or complementary functionalities.

Example: Data analysis using SQL (extract) → Python (clean) → Visualization (plot).

## Key Challenges

**Identifying Relevant Tools**
Selecting the right subset from a large library for a specific task.

**Orchestrating the Flow**
Determining optimal order and interaction patterns between tools.

**Maintaining State**
Managing data and context as it flows between different tools.

## Our Goal

Develop a robust planning framework that excels at identifying and coordinating clusters of tools to efficiently solve complex, multi-faceted tasks.

**TOOL-PLANNER FRAMEWORK**

# Tool-Planner: The Core Idea - Tool Clusters

## What is a Tool Cluster?

A **tool cluster** is a cohesive group of specialized tools that are frequently used in combination to solve a specific type of problem or accomplish a well-defined goal, capitalizing on common usage patterns.

### Example Clusters

**Data Analysis Cluster**
`SQLQuery · PandasTransform · MatplotlibPlot`

**Web Scraping Cluster**
`HTTPRequest · BeautifulSoupParse · WriteFile`

**Code Dev:** `CodeGenerator · TestRunner · Linter`

## The Tool-Planner Approach

**01**

### Cluster Identification

LLM first maps the user's task description to the most relevant pre-defined tool cluster(s).

**02**

### Intra-Cluster Planning

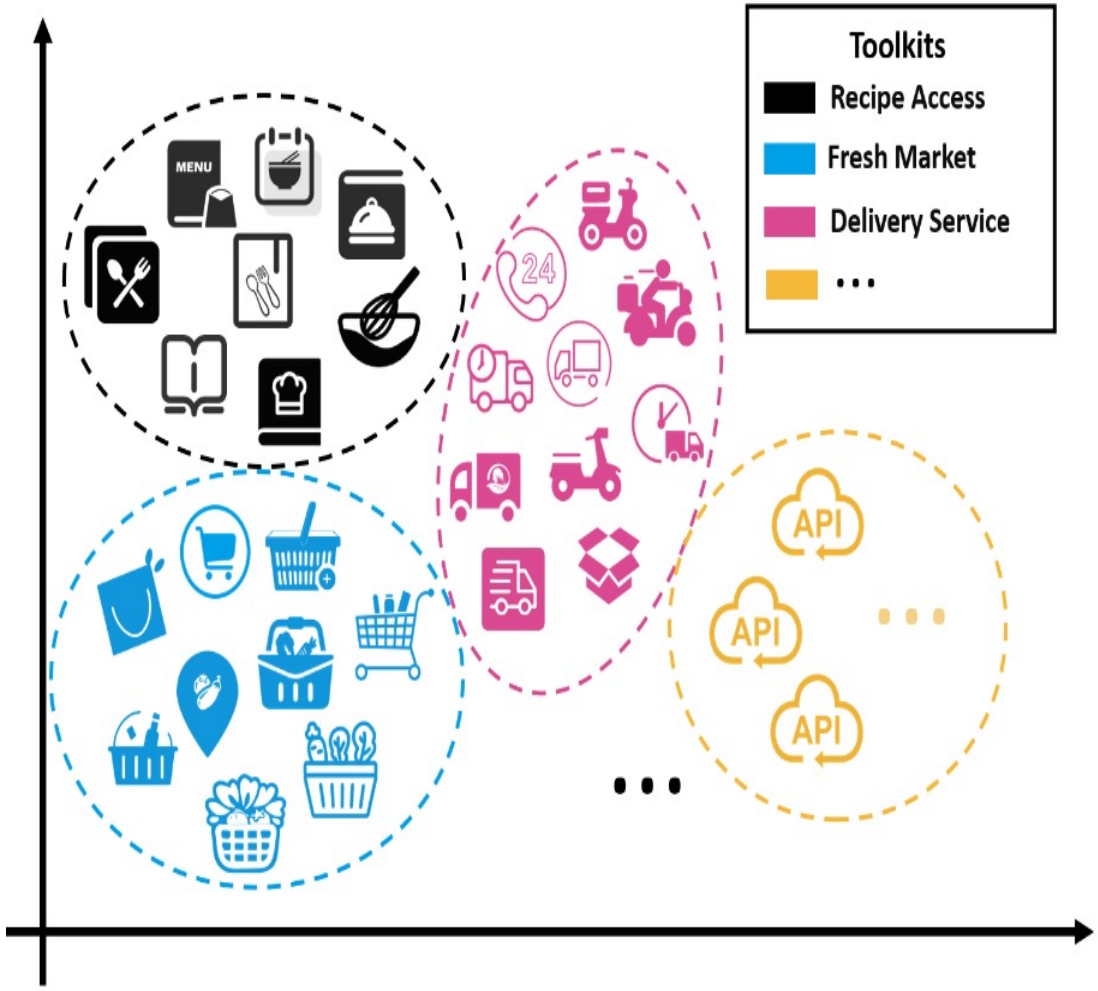
Strategizes the specific, optimal sequence of tool invocations required *within* the identified cluster.

**03**

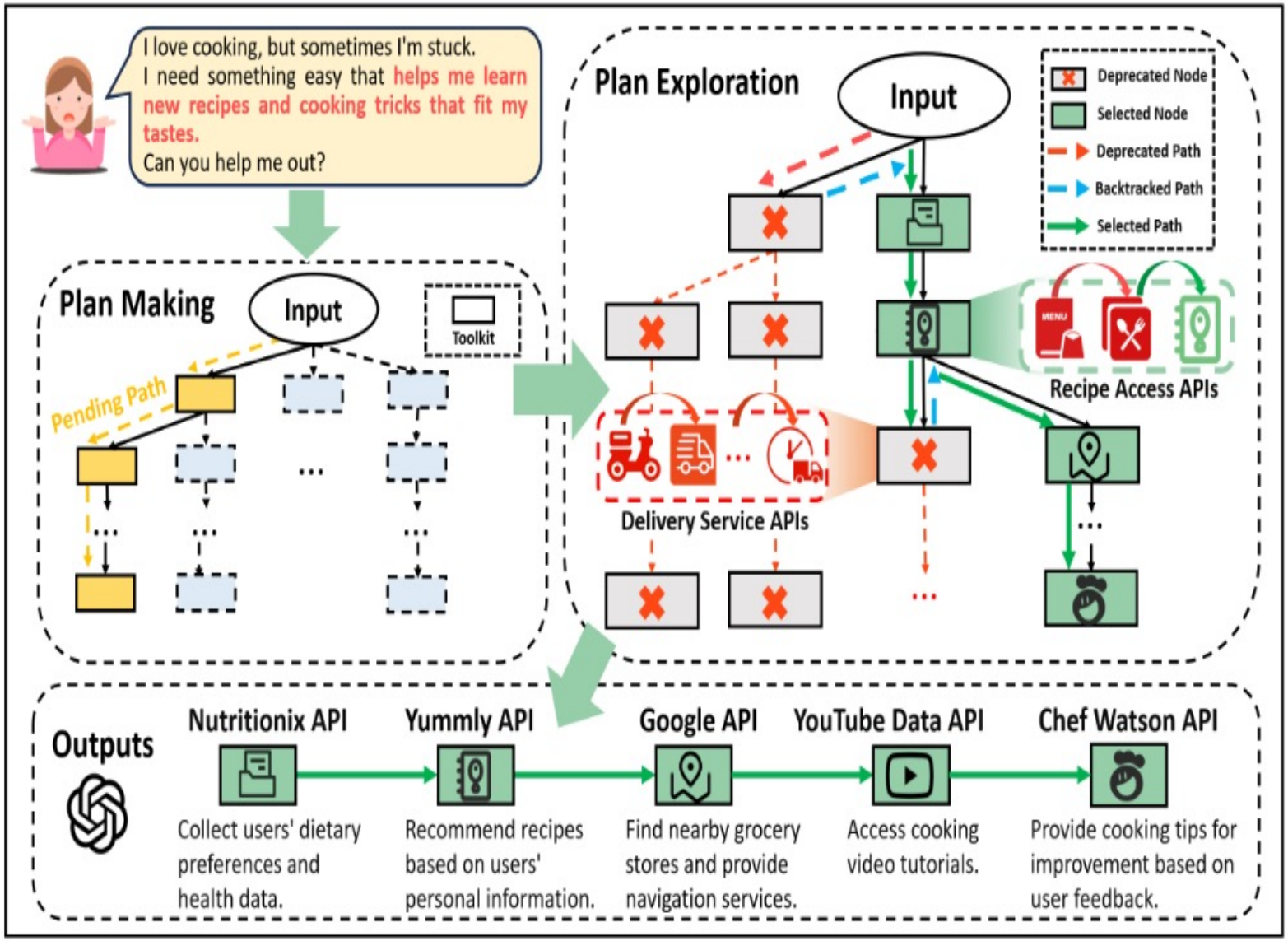
### Inter-Cluster Coordination

Orchestrates data flow and context sharing between multiple clusters for complex, multi-stage tasks.

# Tool Clustering



# Overview of Tool-Planner



# Tool-Planner: Mechanism - The Planning Process

## 01 Cluster Selection

The LLM parses the user's task and leverages its understanding of pre-defined clusters to select the most semantically relevant cluster(s) for execution.

### PRACTICAL EXAMPLE

For "Analyze sales data & generate a report", it selects "Data Analysis" and "Report Generation" clusters.

## 02 Workflow Instantiation

For each selected cluster, the LLM retrieves its manifest and instantiates a concrete workflow template, dynamically filling in task-specific parameters.

### PRACTICAL EXAMPLE

Takes the "Data Analysis" template and specifies the exact input file path, analysis algorithm type, and output format.

## 03 Inter-Cluster Chaining

When multiple clusters are involved, the LLM plans the logical flow, defining how the output artifact of one cluster is passed as input to the next.

### PRACTICAL EXAMPLE

Ensures the CSV output from "Data Analysis" is automatically converted/formatted to match the required input schema for "Report Generation".

# Tool-Planner: Example Workflow

## TASK OBJECTIVE

"Scrape the latest headlines from a news website, summarize the top 3 stories, and post a summary to a social media platform."

### 01. Cluster Selection

Identified core functional groups:
**[Web Scraping] · [Text Summarization] · [Social Media Posting]**

### 02. Instantiation

- Scrape: HTTPReq → SoupParse → ExtractHeadlines
- Summarize: LLMSummarizer(count=3)
- Post: FormatPost → SocialAPIPost

### 03. Inter-Cluster Flow

Data is pipelined sequentially: Extract → Summarize → Format & Post

## OUTCOME: AGENT CAPABILITY

The Tool-Planner agent efficiently combines three distinct clusters of tools to decompose and execute a complex, multi-step task with clear data flow.

# Tool-Planner: Experimental Results

## Improved Performance on Complex Tasks

Outperformed baseline planners on multi-tool coordination tasks, achieving significantly higher success rates and task execution efficiency compared to flat list approaches.

## Accelerated Planning Efficiency

Leveraging pre-defined clusters drastically reduced the search space. The LLM considered far fewer tool combinations, leading to a significantly faster decision-making process.

## Enhanced Generalization Capability

The cluster-based framework enabled the agent to reuse known workflows. This resulted in strong generalization to novel tasks that aligned with the existing cluster patterns.

## Robustness to Tool Variations

The system demonstrated resilience to tool updates or failures. When a specific tool was unavailable, the agent effectively adapted by substituting another compatible tool within the same cluster.

# Tool-Planner: Discussion & Significance

### Coordinated Multi-Tool Use

A structured framework for tackling tasks requiring the coordinated orchestration of multiple diverse tools.

### Leveraging Usage Patterns

Learning and exploiting tool usage clusters significantly simplifies the planning process for complex goals.

### Improved Robustness

Cluster-based approaches yield agent behaviors that are far more robust and generalizable across tasks.

## Broader Impact & Future Vision

### Beyond Sequential Execution

Tool-Planner advances AI capabilities from simple linear task chains to sophisticated tool orchestration. This represents a critical evolutionary step in agent design.

### Real-World Applicability

This work paves the way for building agents that can perform complex, multi-faceted real-world jobs. It demonstrates how AI can integrate diverse software tools seamlessly, mirroring the way human experts combine different applications to achieve a goal.

# Tool-Planner: Critical Analysis & Future Directions

## CRITICAL ANALYSIS

### Dependency on Cluster Quality

Planning effectiveness is directly tied to the definition of learned clusters. Poorly structured clusters lead to flawed planning outcomes.

### Limitation in Novel Tasks

Performs optimally for tasks fitting existing clusters, but struggles with highly novel or unstructured scenarios requiring ad-hoc tool combinations.

### Dynamic & Adaptive Clusters

Empower the agent to create or modify clusters on-the-fly, adapting to new task structures or evolving tool usage patterns in real-time.

### Hierarchical Meta-Planning

Develop a high-level planner that decides whether to leverage pre-defined clusters or to generate entirely new, ad-hoc tool combinations from scratch.

### Continuous Cluster Evolution

Enable the system to continuously update and refine its cluster knowledge base as it accumulates more experience and data from tool interactions.

# Graph-based Agentic Memory

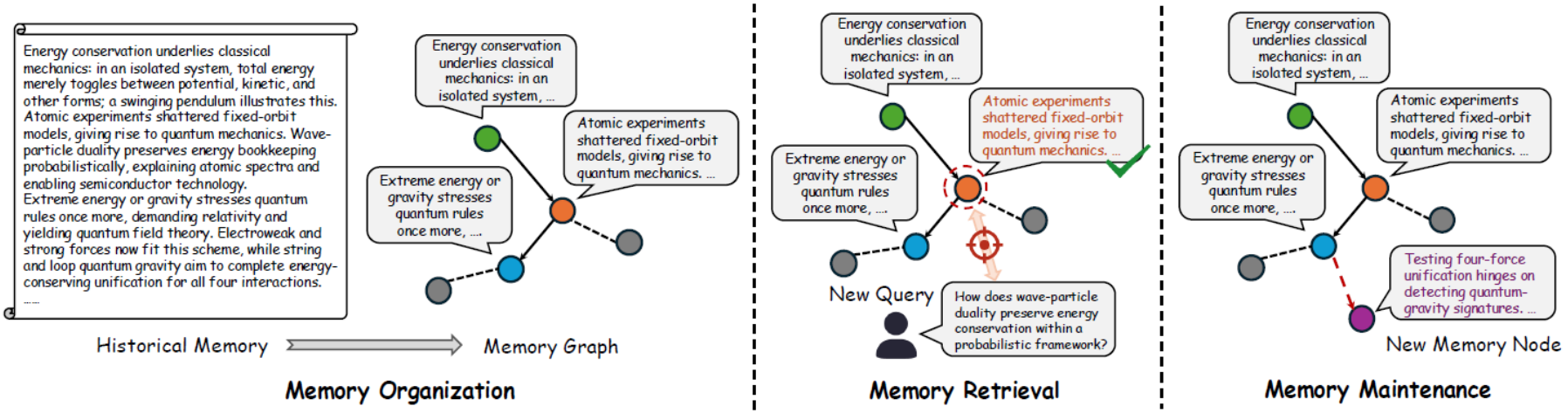
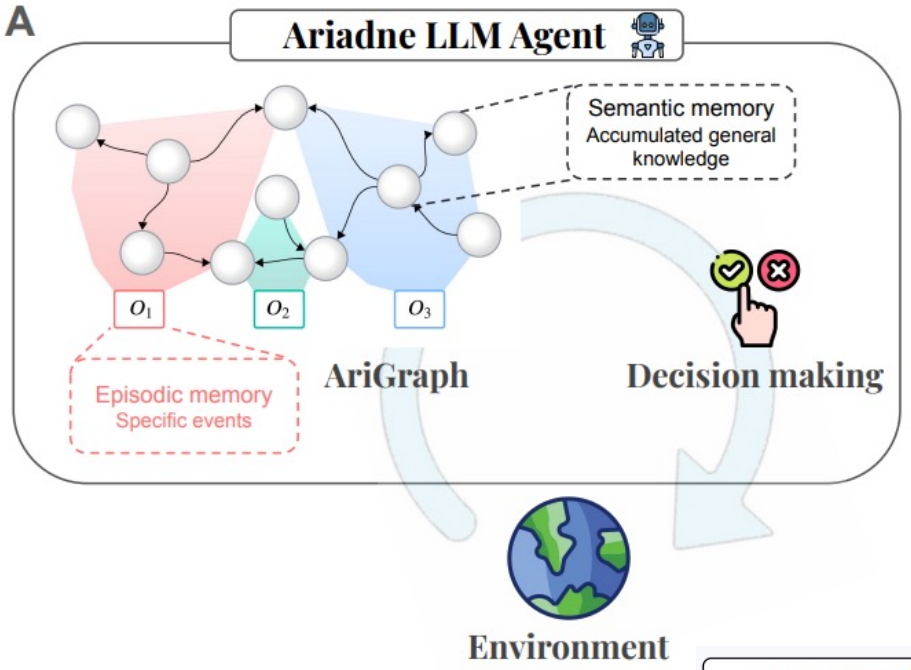



Fig. 5: An illustration of graphs for agent memory.

# Memory Types in Agentic Systems

| Memory Type | What It Represents | Primary Substrate | Why This Mapping Makes Sense |
|---|---|---|---|
| **Semantic Memory** | Facts, concepts, stable knowledge | **Knowledge Graphs** | Semantic memory is structured, relational, and context-free — exactly what knowledge graphs encode: entities, attributes, and relations. |
| **Episodic Memory** | Time-stamped experiences, interactions, tool calls | **Vector Stores** | Episodic traces are unstructured, high-dimensional, and sequential. Vector stores excel at storing and retrieving such embeddings efficiently. |
| **Reflective Memory** | Lessons learned, distilled insights, strategies | **Agentic Self-Reflection Loops** | Reflection is produced by analyzing episodes and extracting reusable rules. This is implemented through self-critique loops that generate new meta-knowledge. |

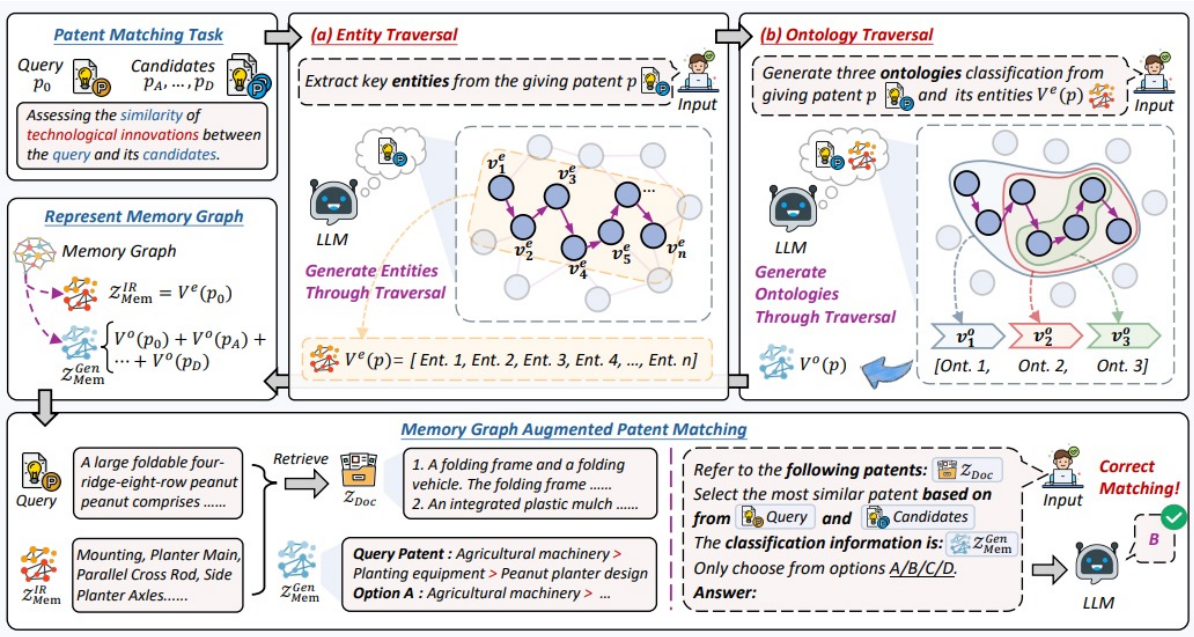
# Graphs for Agentic Memory




The architecture of our Ariadne agent, equipped with AriGraph memory. AriGraph integrates both semantic knowledge graph and past experiences. Memory in the form of a semantic knowledge graph extended with episodic vertices and edges significantly enhances the performance of LLM-agent in text-based games.

P. Anokhin, N. Semenov, A. Sorokin, D. Evseev, M. Burtsev, and E. Burnaev, "Arigraph: Learning knowledge graph world models with episodic memory for llm agents, https://arxiv.org/pdf/2407.04363

Q. Xiong, Z. Xu, Z. Liu, M. Wang, Z. Chen, Y. Sun, Y. Gu, X. Li, and G. Yu, "Enhancing the patent matching capability of large language models via memory graph," in ACM SIGIR 2025.



# Graphs for Multi-Agent Coordination

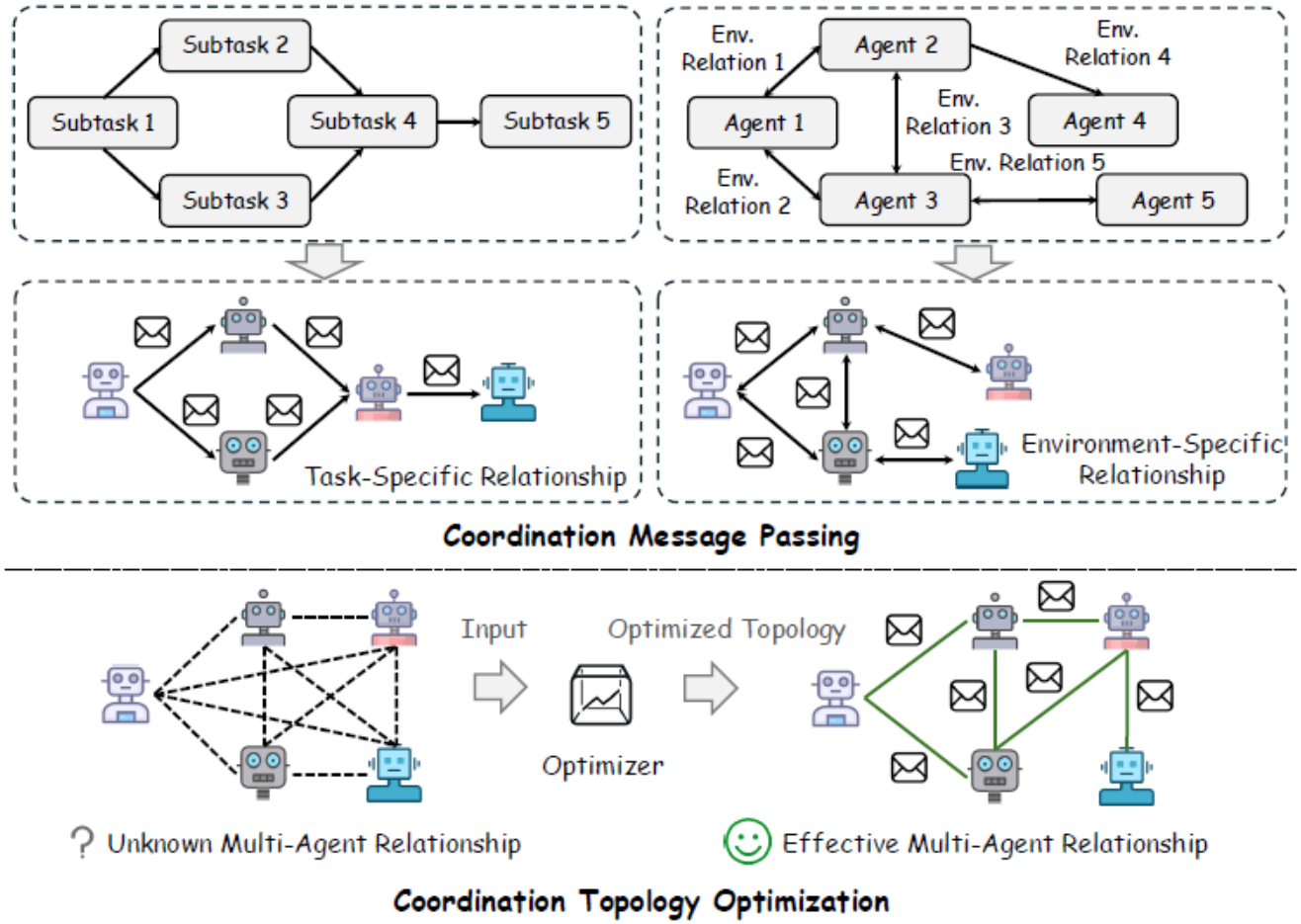
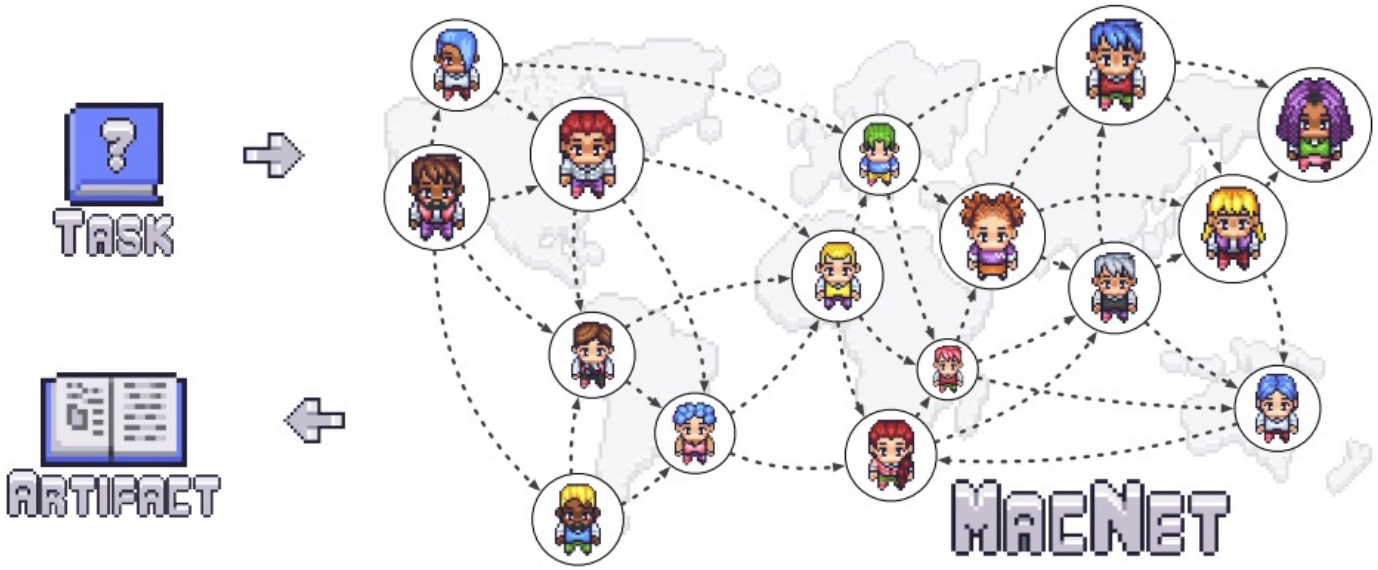



Fig. 6: An illustration of graphs for multi-agent coordination.

# Graphs for Multi-Agent Coordination




C. Qian, Z. Xie, Y. Wang, W. Liu, K. Zhu, H. Xia, Y. Dang, Z. Du, W. Chen, C. Yang, Z. Liu, and M. Sun, "Scaling large language model-based multi-agent collaboration," in ICLR 2025.

Figure 1: Multi-agent collaboration network (MACNET) uses directed acyclic graphs to arrange agents for collaborative interactions, facilitating autonomous task-solving through collective reasoning.

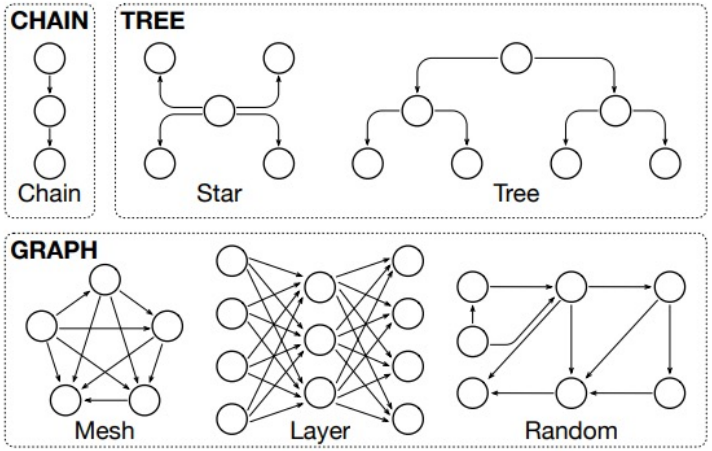



Figure 2: Representative topologies.

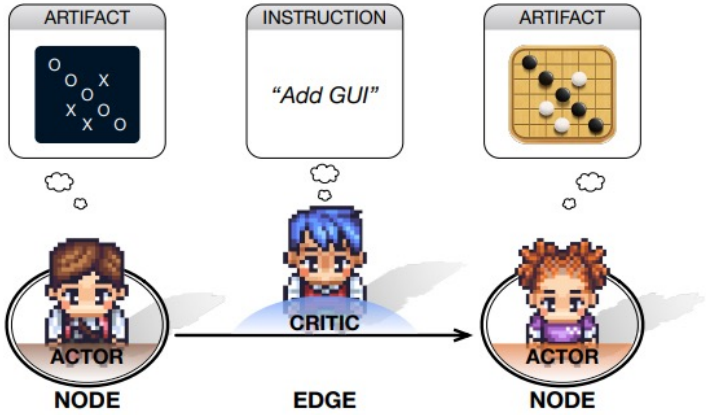
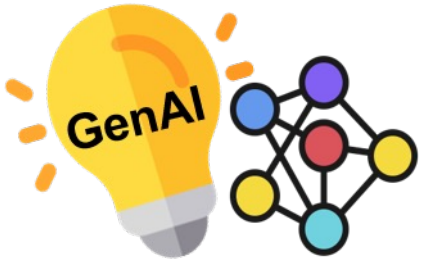



Figure 3: Assign functionally bipartite agents on nodes and edges, respectively.

# 6. Graphs for AI Agents

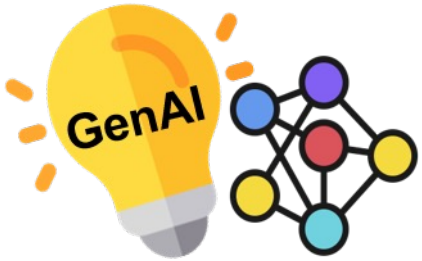



- AI Agents
- Agent Interaction With Task Planning Graphs
- Agent Interaction With Task Execution Graphs
- Agent Interaction With Memory Graphs
- Multi-Agent Coordination Graphs

Questions?

Presented by Arijit Khan

# 7. AI Agents for Graphs




A. General Graph Problem Solving

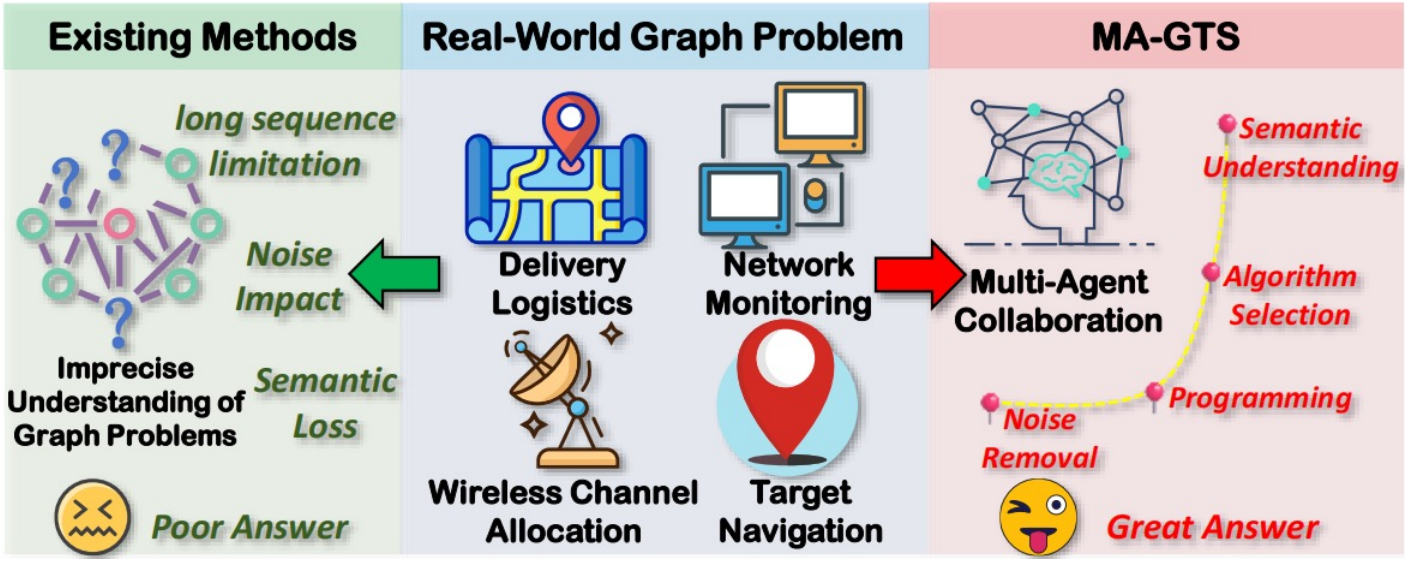
B. Tool-Augmented Graph Algorithmic Reasoning

C. Domain-Specific KG Construction

D. Community Search

Presented by Longxu Sun

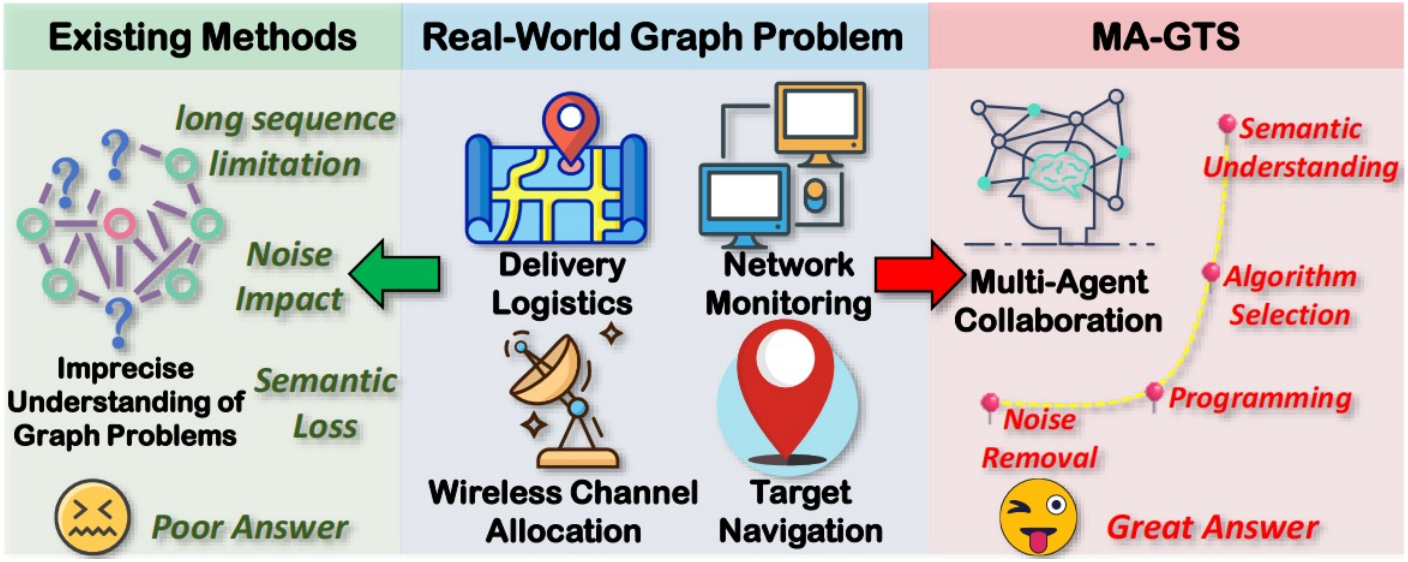
# MA-GTS: A Multi-Agent Framework for Graph Problems (EMNLP 2025)




- **Problem:** Real-world graph problems (logistics, networks, traffic) are complex, noisy, and irregular
- **Limitations of LLMs:** limited accuracy, input length constraints, suboptimal algorithm selection

[1] https://aclanthology.org/2025.emnlp-main.973.pdf

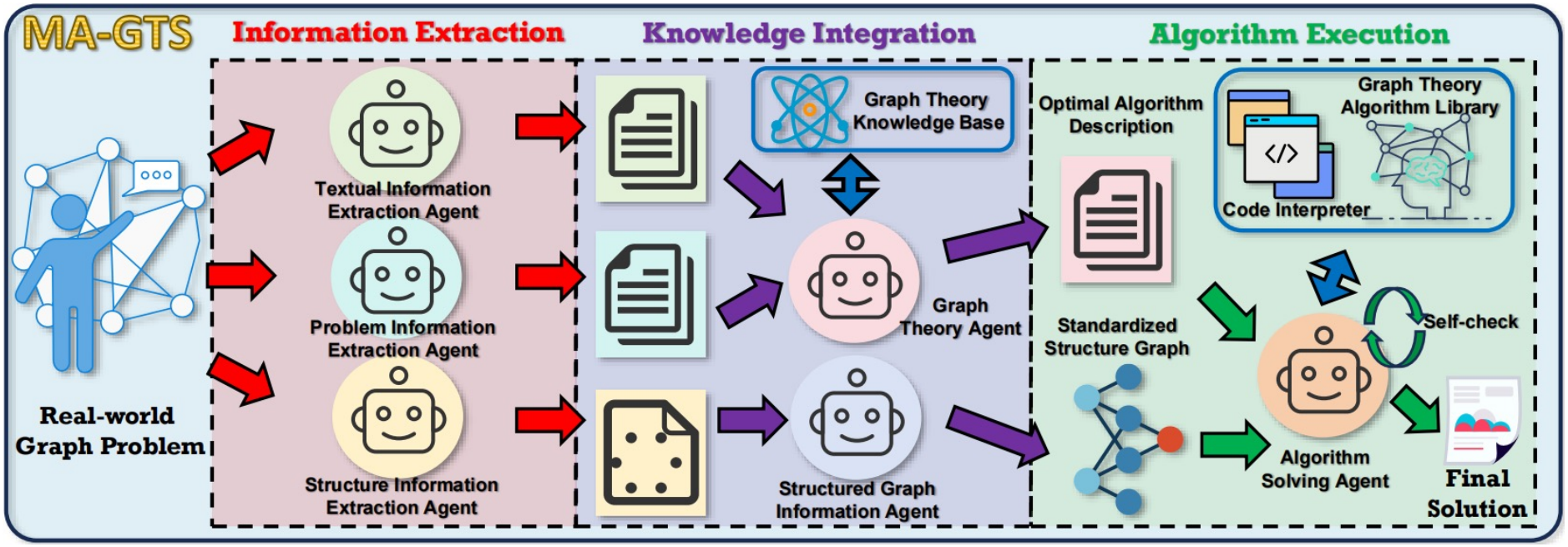
# MA-GTS: A Multi-Agent Framework for Graph Problems (EMNLP 2025)




- **Solution:** MA-GTS decomposes complex problems via agent collaboration

[1] https://aclanthology.org/2025.emnlp-main.973.pdf

# MA-GTS: A Multi-Agent Framework for Graph Problems (EMNLP 2025)




- **Pipeline:** Text-based graph → structured representation → dynamic algorithm selection

[1] https://aclanthology.org/2025.emnlp-main.973.pdf

# MA-GTS: A Multi-Agent Framework for Graph Problems (EMNLP 2025)

| Model | Method | G-REAL | | | | GraCoRe | NLGraph | |
|---|---|---|---|---|---|---|---|---|
| | | Delivery Logistics Problem (TSP) | Wireless Channel Allocation Problem (Coloring) | Network Monitoring Problem (Vetex Cover) | Target Navigation Problem (Shortest Path) | TSP | Shortest Path | Cycle |
| *o3-mini* | **Direct** | 11.8% | 80.1% | 68.7% | 47.1% | 79.7% | **100.0%** | 97.3% |
| | **CoT** | 12.9% | 83.1% | 72.8% | 41.8% | 80.0% | 98.4% | 97.8% |
| *GPT-4o-mini* | **Direct** | 2.5% | 23.4% | 0.3% | 7.1% | 1.1% | 27.3% | 50.9% |
| | **CoT** | 3.1% | 25.1% | 0.0% | 6.4% | 1.1% | 27.6% | 51.1% |
| *GPT-3.5* | **Direct** | 0.1% | 0.7% | 2.5% | 4.0% | 1.9% | 30.5% | 50.0% |
| | **CoT** | 2.1% | 7.6% | 4.8% | 3.6% | 1.6% | 34.7% | 49.9% |
| *Qwen2.5-7B* | **Direct** | 0.6% | 16.2% | 17.4% | 4.6% | 3.8% | 22.1% | 49.6% |
| | **CoT** | 0.6% | 8.8% | 8.5% | 5.8% | 3.0% | 27.3% | 52.7% |
| *Llama3-7B* | **Direct** | 3.6% | 10.1% | 7.2% | 4.3% | 0.3% | 12.6% | 53.7% |
| | **CoT** | 4.1% | 14.3% | 6.7% | 4.2% | 0.3% | 19.4% | 50.9% |
| *Deepseek-V3-660B* | **Direct** | 4.9% | 27.2% | 21.1% | 11.4% | 10.5% | 50.8% | 78.1% |
| | **CoT** | 5.5% | 28.3% | 22.2% | 32.2% | 18.8% | 92.9% | 77.8% |
| *OWL (GPT-4o-mini)* | **Multi-Agent** | 10.2% | 47.4% | 7.8% | 19.1% | 4.4% | 36.3% | 49.7% |
| *GraphTeam (GPT-4o-mini)* | **Multi-Agent** | 8.8% | 90.0% | 12.0% | 87.7% | 84.4% | 98.4% | **100.0%** |
| ***MA-GTS (Deepseek-V3-660B)*** | **Multi-Agent** | 76.2% | 88.2% | **99.1%(↑26.3%)** | 88.2% | 93.1% | 93.8% | **100.0%** |
| ***MA-GTS (GPT-4o-mini)*** | **Multi-Agent** | **94.9%(↑82%)** | **94.5%(↑4.5%)** | 93.2% | **91.7%(↑4%)** | **96.9%(↑12.5%)** | 97.8%(↓2.2%) | 98.9% |

## Results

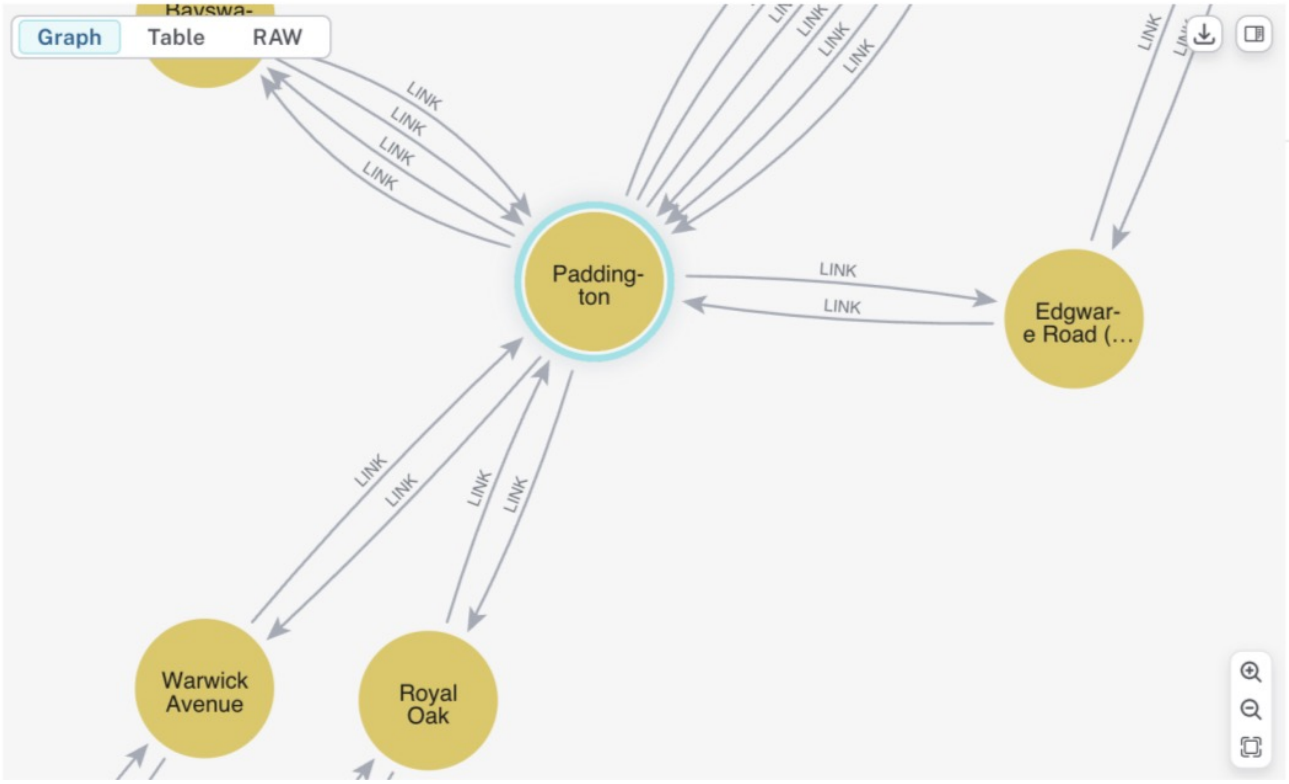
**93.6%**
G-REAL

**96.9%**
GraCoRe

**98.4%**
NLGraph

[1] https://aclanthology.org/2025.emnlp-main.973.pdf

# GDS Agent for Graph Algorithmic Reasoning




**Node details**

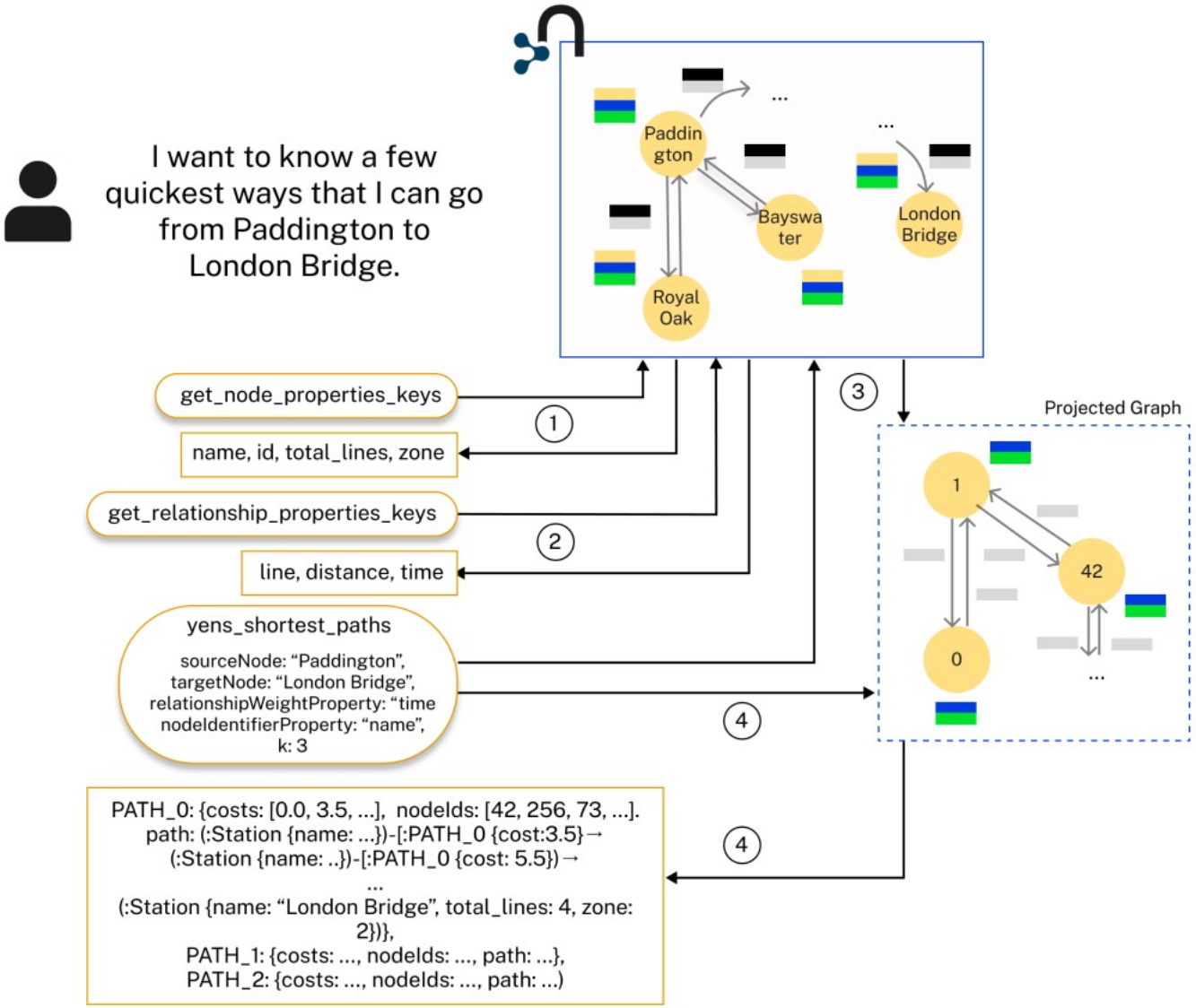
UndergroundStation

| Key | Value |
|---|---|
| <id> | 4:9fc80800-f0b8-4863-bdc7-294698216e7d:167 |
| display_name | "Paddington" |
| id | "193" |
| latitude | 51.5154 |
| longitude | -0.1755 |
| name | "Paddington" |
| rail | 1 |
| total_lines | 4 |
| zone | 1 |

- GDS Agent is a graph algorithm reasoning agent designed to address the challenges LLMs face when processing and reasoning over large-scale graph-structured data.

[1] https://arxiv.org/pdf/2508.20637

# GDS Agent for Graph Algorithmic Reasoning

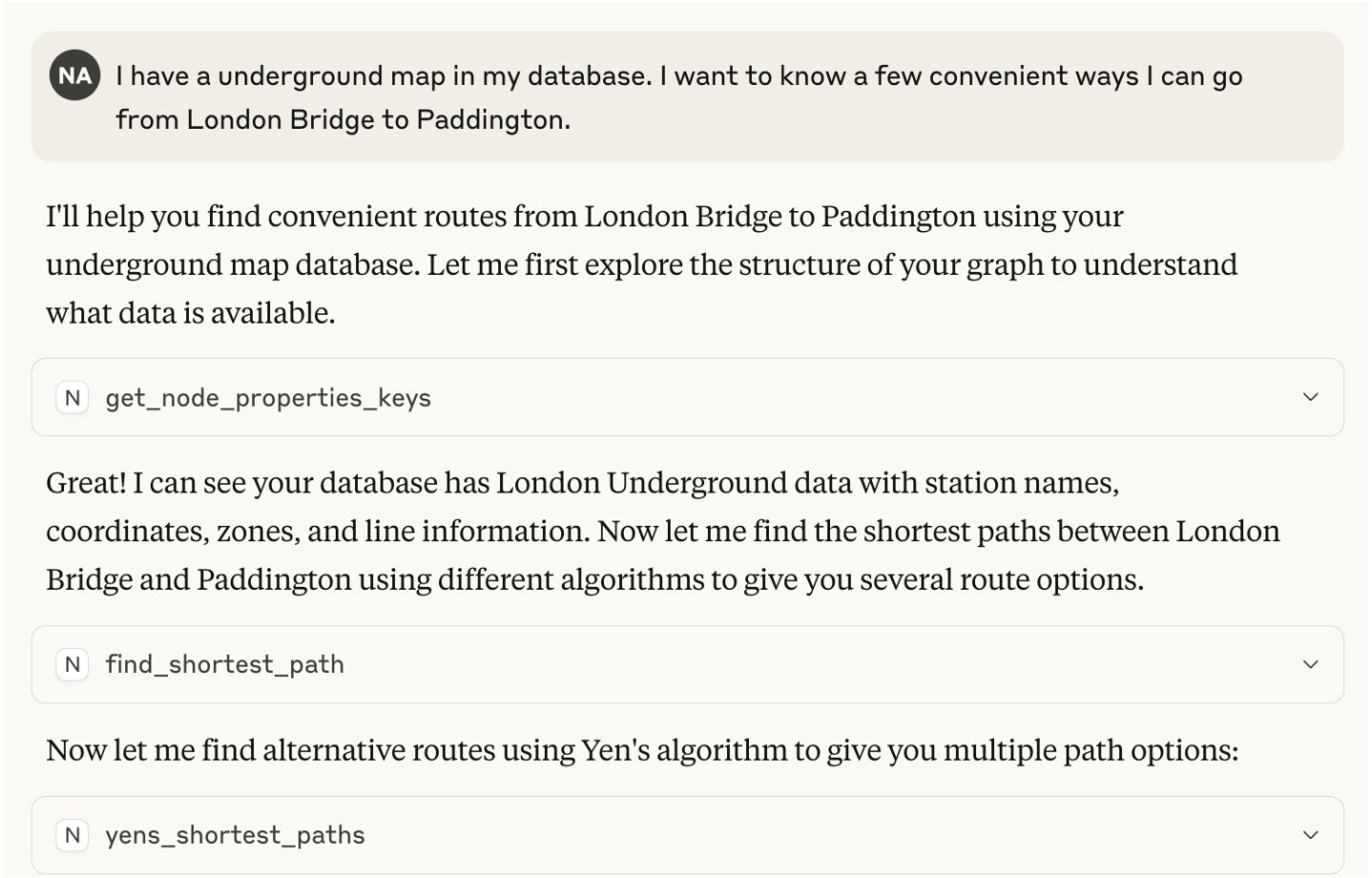



**Procedure：**

1.User Query
2.Tool Invocation
3.Algorithm Selection
4.Parameter
5.Cypher Projection & GDS Execution
6.Result & Final

[1] https://arxiv.org/pdf/2508.20637

# GDS Agent for Graph Algorithmic Reasoning




- An example where an LLM with GDS Agent is able to pick shortest path and Yen's algorithm to answer my question about travel plan:

[1] https://arxiv.org/pdf/2508.20637

# GDS Agent for Graph Algorithmic Reasoning

Based on your underground map database, here are the convenient ways to travel from London Bridge to Paddington:

**Route 1 (Most Common Path - 8 stops):**

London Bridge → **Bank** → Waterloo → Westminster → Green Park → Bond Street → Baker Street → Edgware Road → **Paddington**

**Route 2 (Alternative via Southwark - 8 stops):**

London Bridge → **Southwark** → Waterloo → Westminster → Green Park → Bond Street → Baker Street → Edgware Road → **Paddington**

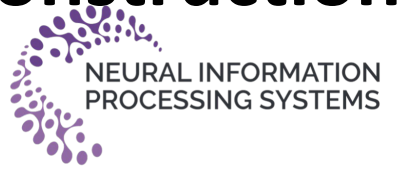
Both routes are equally efficient with 8 stops total. The key differences:

**Route 1** goes via **Bank** - this connects you to the Northern/Central lines and gives you more

- LLMs equipped with GDS agent can decide and accurately execute the appropriate parameterised graph algorithms over the graph you have in your Neo4j database.

[1] https://arxiv.org/pdf/2508.20637

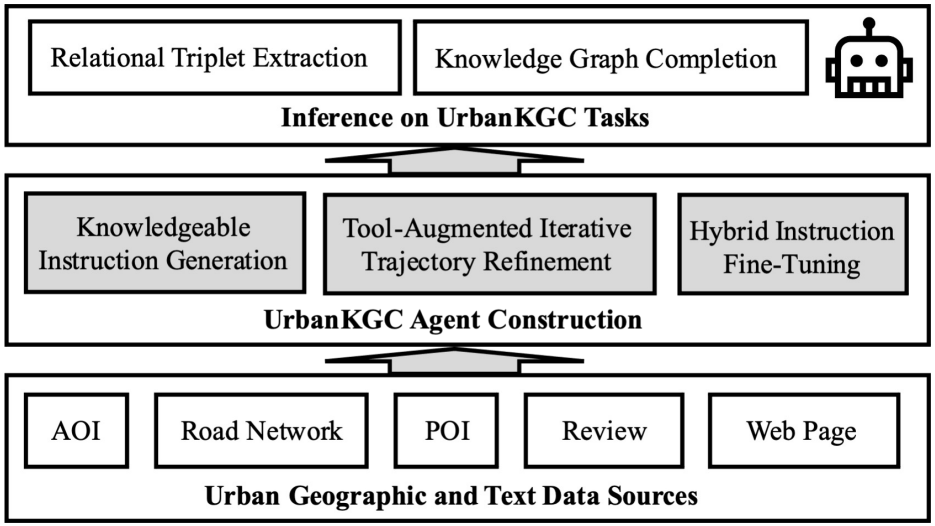
# UrbanKGent: A Unified Large Language Model Agent Framework for Urban Knowledge Graph Construction




- **Previous UrbanKG Construction (UrbanKGC) Studies**
  - Manually designed methods: 1) rely on deep understanding of the application domain; 2)labor-intensive.
  - Language model based methods: 1) rely on annotated corpus; 2) need model retraining.
- **Motivation**
  - Leverage the remarkable zero-shot capability of LLM in autonomous domainspecific task completion.
  - Construct tailored LLM agent compatible with various UrbanKGC tasks to address the aforementioned limitations in UrbanKGC.

[1] https://arxiv.org/pdf/2402.06861

# UrbanKGent: A Unified Large Language Model Agent Framework for Urban Knowledge Graph Construction

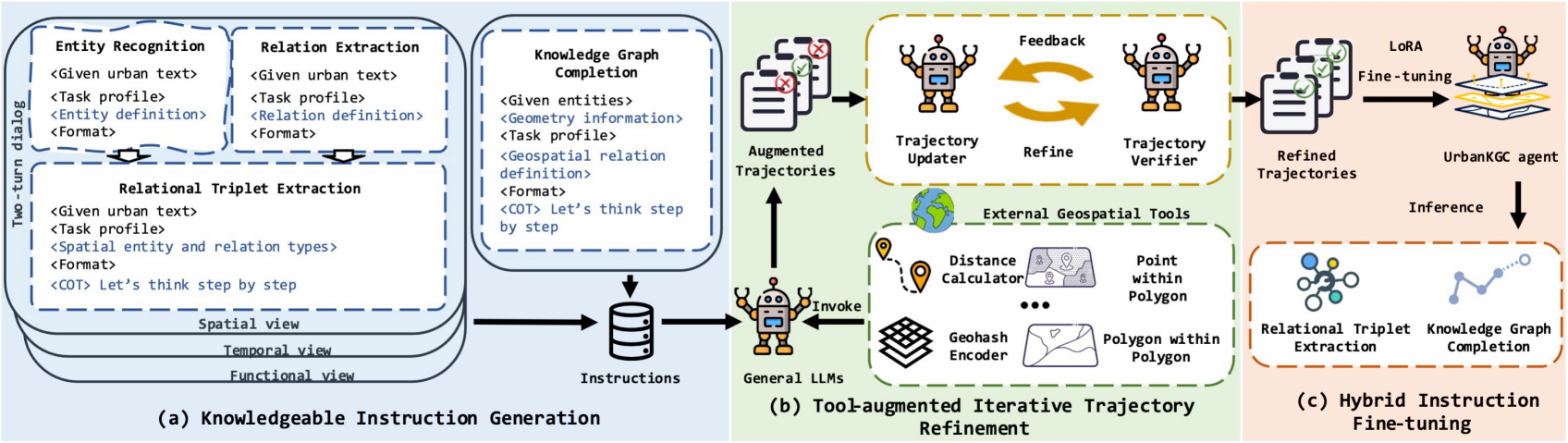



## Overview

- A unified LLM agent framework for automatic UrbanKG construction.
- Three steps: 1) urban data collection; 2) UrbanKGC agent construction; 3)Inference on UrbanKGC task.

[1] https://arxiv.org/pdf/2402.06861

# UrbanKGent: A Unified Large Language Model Agent Framework for Urban Knowledge Graph Construction

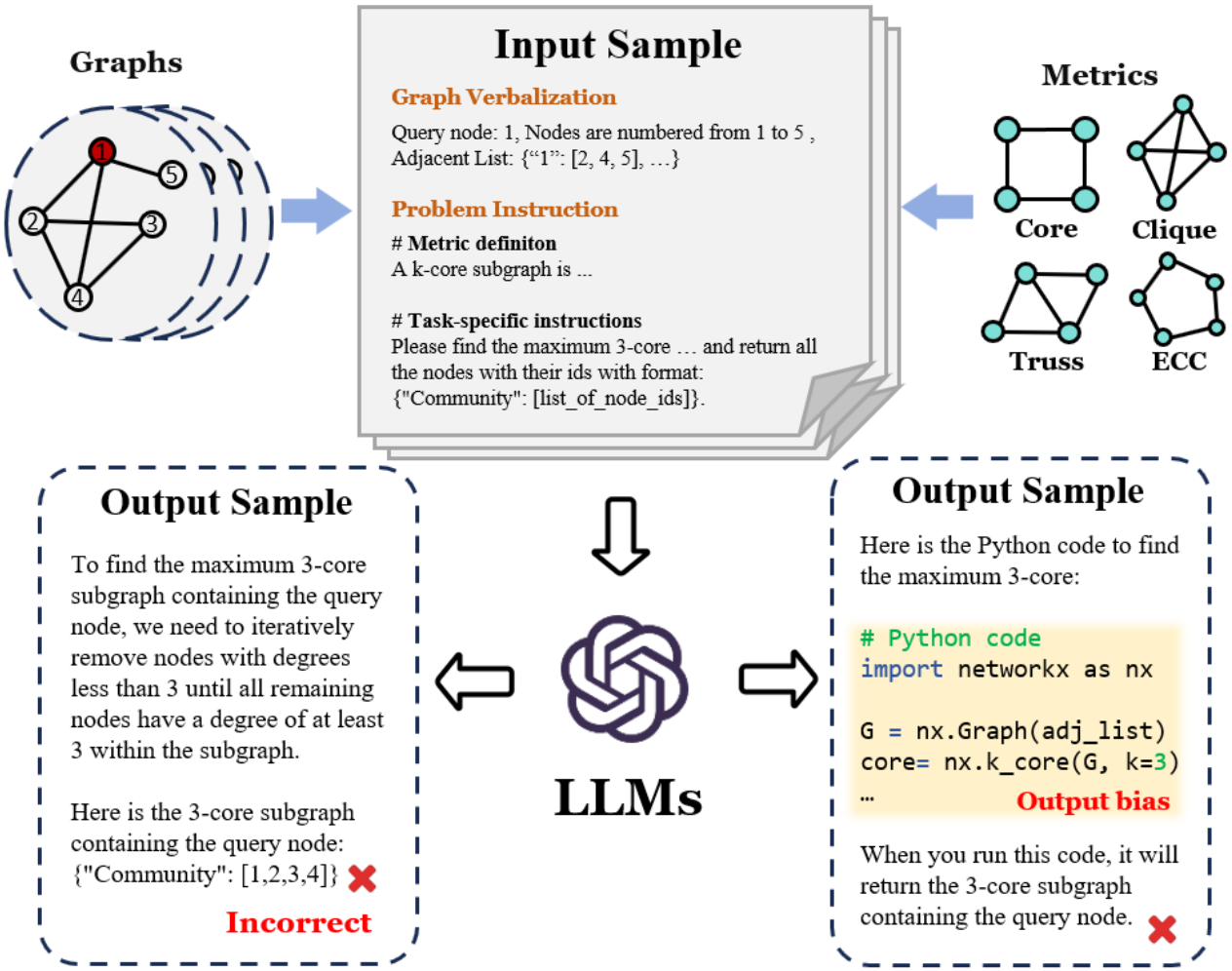



## UrbanKGC Agent Construction

- Knowledgeable instruction generation for aligning LLM to UrbanKGC tasks;
- Tool-augmented iterative trajectory refinement to enhance and refine generated trajectory;
- Hybrid Instruction Fine-tuning for cost-effectively completing UrbanKGC tasks.

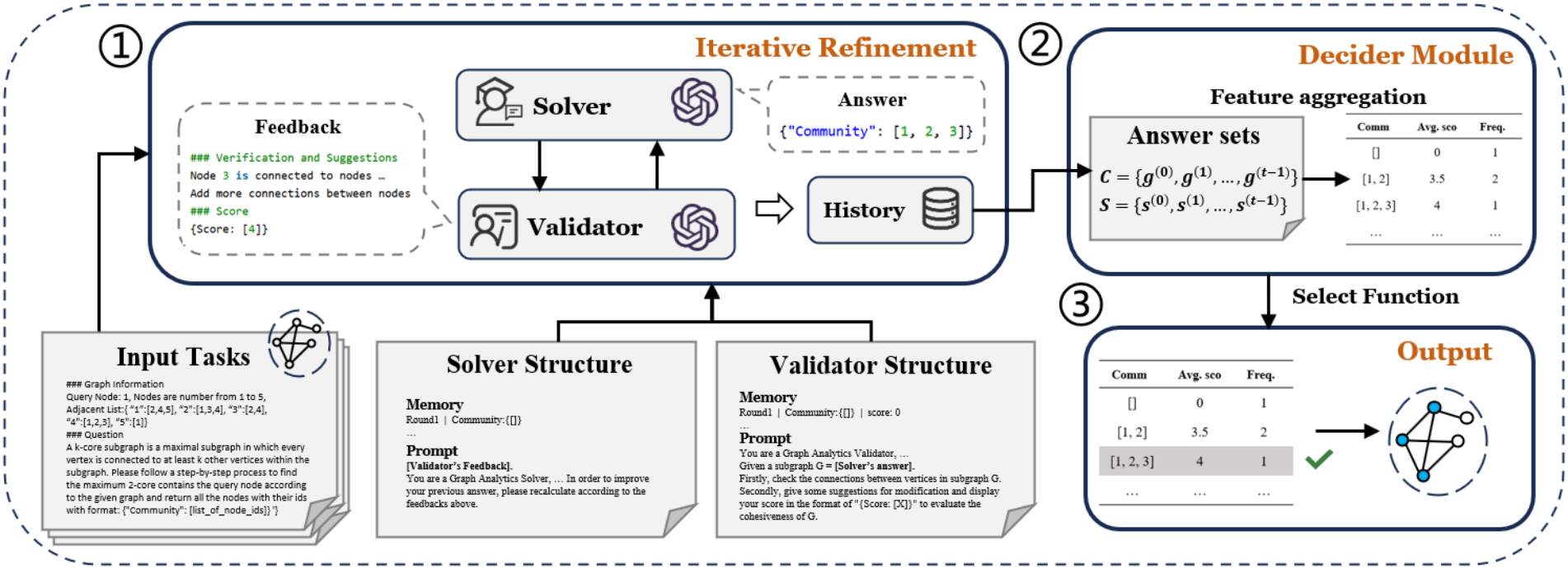
[1] https://arxiv.org/pdf/2402.06861

# CS-Agent: LLM-based Community Search via Dual-agent Collaboration



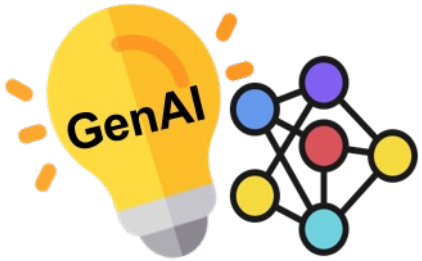

[1] https://arxiv.org/html/2508.09549v2

# CS-Agent: LLM-based Community Search via Dual-agent Collaboration




[1] https://arxiv.org/html/2508.09549v2

# 8. Future Directions



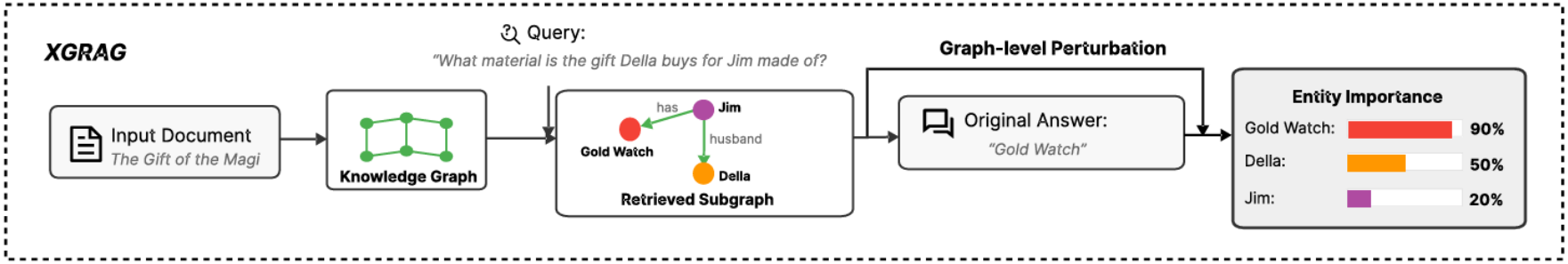

- Explainability, Responsibility, and Reliability
- Security and Privacy
- Knowledge Graph-based Agentic Memory
- AI Agents with Graph DB & Graph Analytics
- Domain-specific Applications

Presented by Arijit Khan

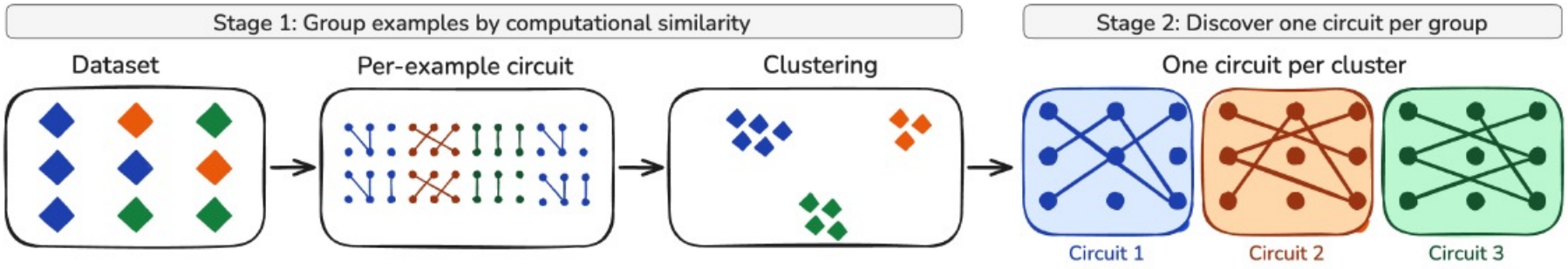
# Explainability, Responsibility, and Reliability

- Explainability over GraphRAG

XGRAG: A Graph-Native Framework for Explaining KG-based Retrieval-Augmented Generation. Zhuoling Li, Ha Linh Hong Tran Nguyen, Valeria Bladinieres, Maxim Romanovsky




- Circuits for Mechanistic Interpretability: computational subgraphs responsible for the model's behavior.

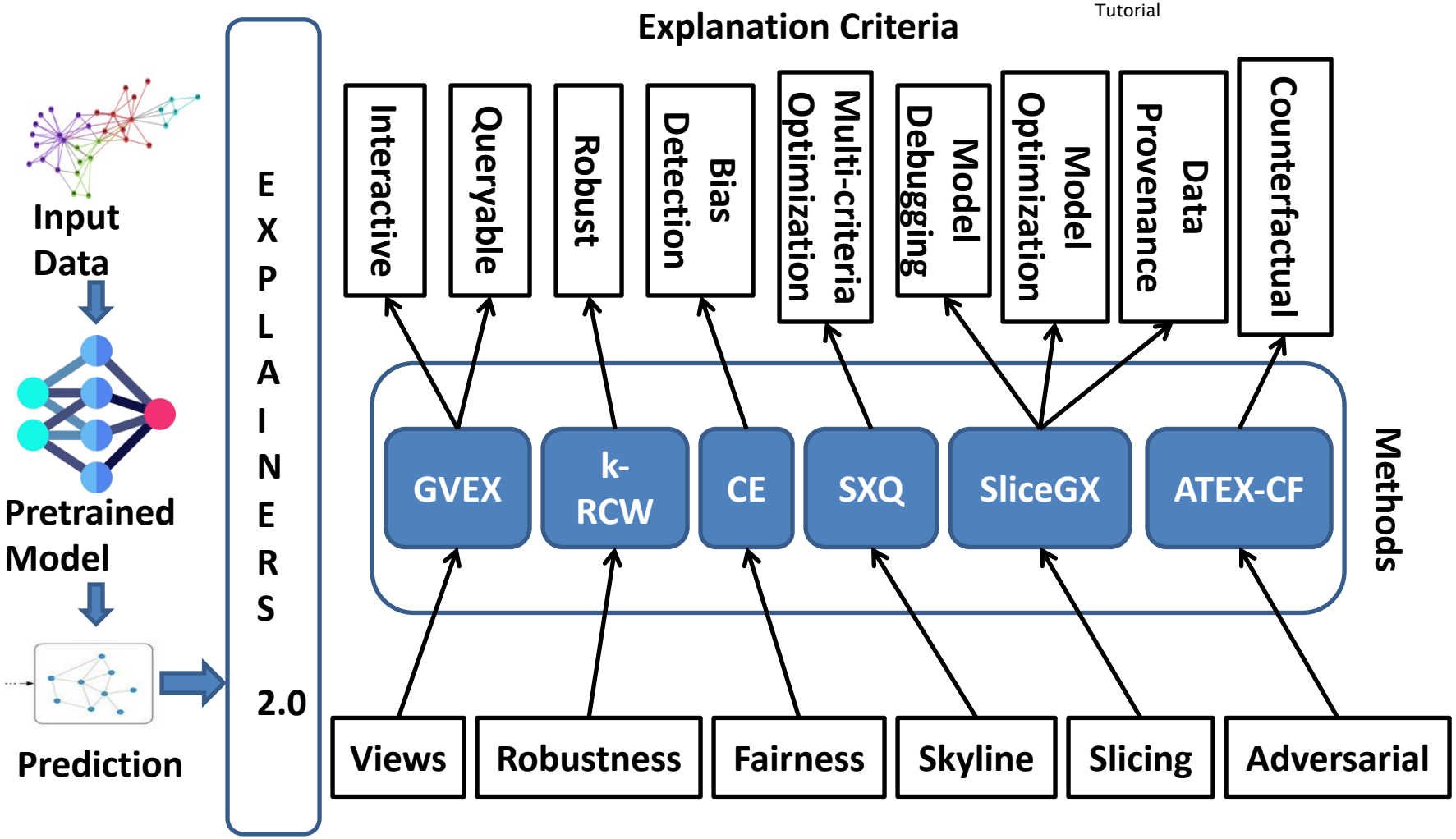



Data-driven Circuit Discovery for Interpretability of Language Models, Daking Rai, Mor Geva, Ziyu Yao

# Explainability, Responsibility, and Reliability

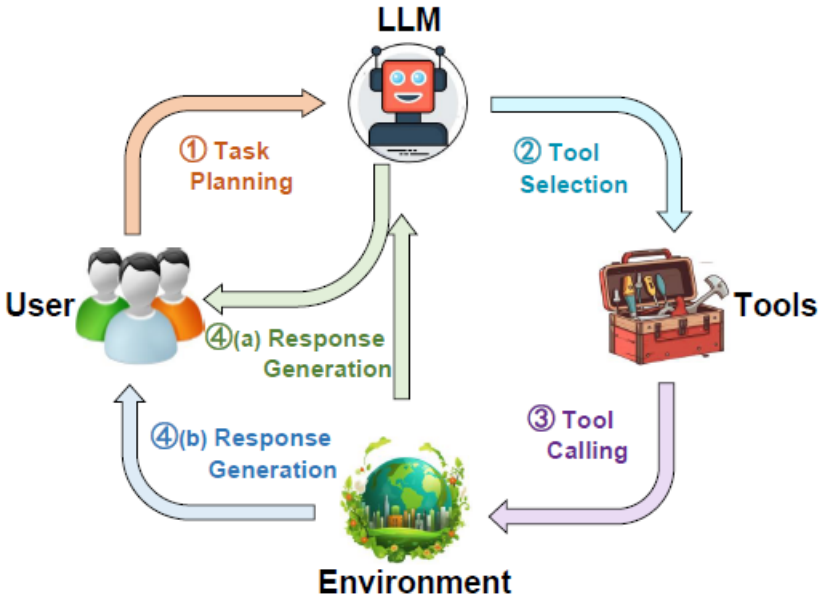
- Data-driven and User-focused Explanations

Arijit Khan, Xiangyu Ke, Yinghui Wu, and Francesco Bonchi, "GNN Explainers 2.0: User-centric and Data Driven Insights", WSDM 2026 Tutorial



# Explainability, Responsibility, and Reliability

- Evaluation science that measures and explains failures at the process level and identifies where misalignment emerges inside graph-structured reasoning.



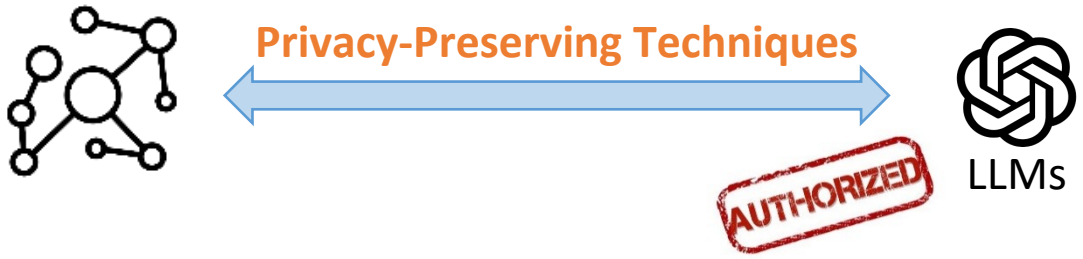

**Workflow of tool learning**

- Current benchmarks mainly focus on final answer accuracy and do not assess intermediate capabilities such as sequential graph comprehension or quantitative conflict detection between tool outputs and model knowledge.

| Aspect | GraphToolBench (Ours) | ShortcutsBench [1] | ToolEyes [2] | SoAyBench [3] | ToolHop [4] | StableToolBench [5] | TaskBench [6] | ToolBench [7] | ToolSword [8] | APIBench [9] |
|---|---|---|---|---|---|---|---|---|---|---|
| Sequential Graph Comprehension ? | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Quantitative Conflict Identification ? | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| One-time Multi-tool Invocation ? | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Offline Evaluation ? | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |

# Security and Privacy

- Security & Privacy:
  - Unifying domain-specific KGs raises privacy risks.




- Differential Privacy
- Federated Learning
- Anonymization
- Access Control

- **Compositional guarantees** — enforce policies across agents, models, and workflows.
- **Policy-aware retrieval** — ensure queries respect access controls and governance rules.
- **Leakage-aware design** — prevent sensitive data exposure in embeddings, memory, and intermediate states.
- **Cross-user privacy** — manage shared context without violating individual confidentiality.
- **Sustained interactions** — long-running, context-rich exchanges demand new abstractions for secure state management.
- **Auditable compliance** — log agent intent, provenance, and data use for accountability.

# Knowledge Graph-based Agentic Memory

**Unified graph memory** — Knowledge graphs unify text, logs, sensor streams, documents, and multimodal data, aligning heterogeneous signals into a coherent temporal-causal context.

**Long-horizon, multi-session workloads** — KG memory maintains evolving state, schema, and causal dependencies across sessions, making it uniquely suited for sustained reasoning beyond single queries.

**Transparency & auditability** — Provenance-rich KG traces provide explainable reasoning, causal attribution, and compliance, ensuring trustworthy agentic planning.

**Corrective knowledge** — Counterfactual reasoning captures why past actions failed, encoding reusable graph-linked constraints that prevent agents from repeating errors.

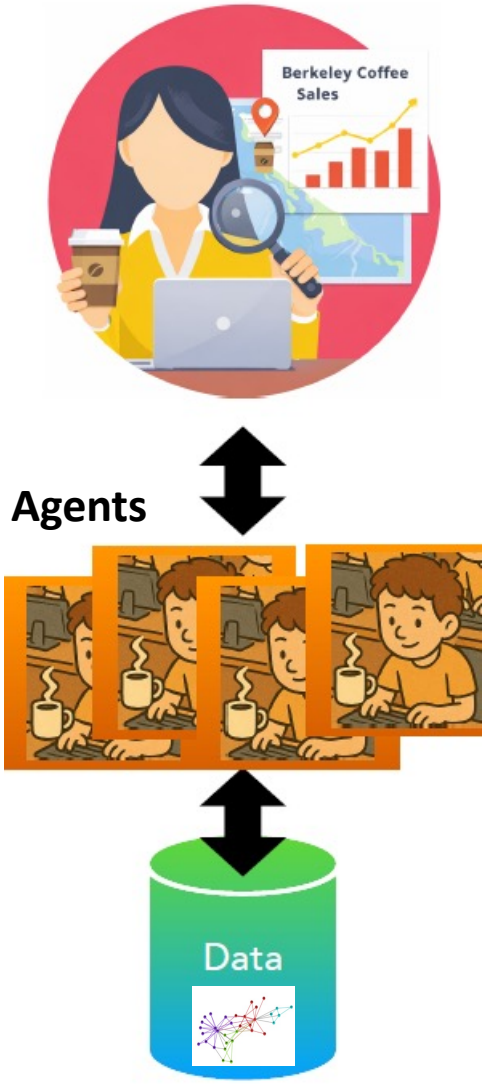
Shubham Agarwal, Asim Biswal, Sepanta Zeighami, Alvin Cheung, Joseph Gonzalez, and Aditya G. Parameswaran, "Arming Data Agents with Tribal Knowledge"

# AI Agents with Graph DB & Analytics

- **Agent-driven hypothesis exploration** — data analysts will orchestrate swarms of agents to probe diverse hypotheses across complex datasets.

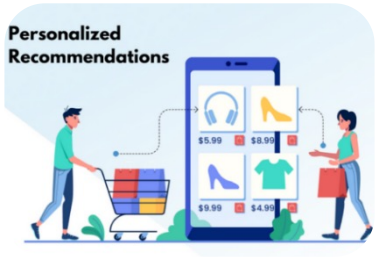
- **Combinatorial query space** — agents must navigate an exponential landscape of possible queries, including NP-hard graph problems.

- **Graph DBs for agentic workloads** — future systems must evolve to support adaptive, multi-agent query planning, optimization, and analytics at scale.



# Domain-specific Applications

- The combination of LLMs and KGs leverages **LLMs' natural language understanding** and **KGs' structured knowledge** to enhance applications like:

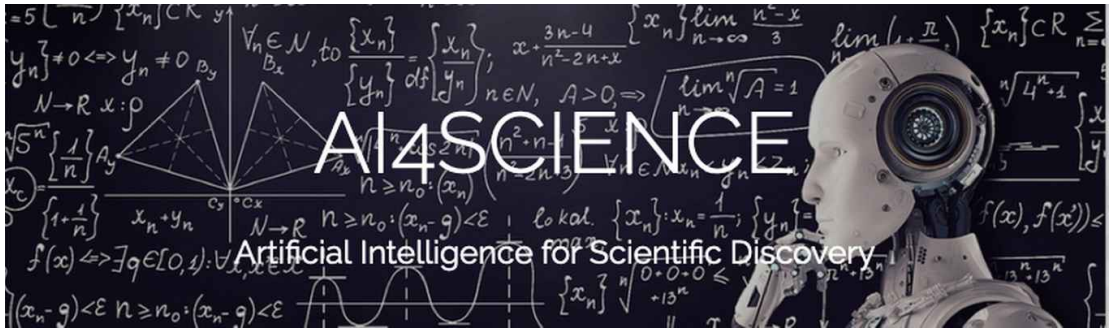



Personalized Recommendations


Customer Service

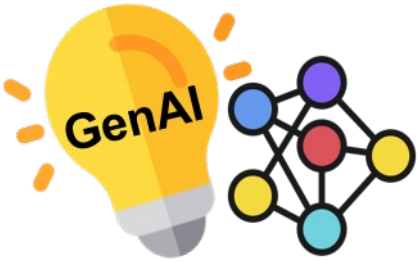
Medical Diagnostics

Financial Decision-Making




- **Future**: Smarter, knowledge-rich solutions across domains.

# Thank You!





https://seucoin.github.io/workshop/ag2026/