GraCoRe: Benchmarking Graph Comprehension and Complex Reasoning in Large Language Models

Zike Yuan[1,2], Ming Liu[1,2], Hui Wang[2,*], Bing Qin[1,2,*]
[1]Harbin Institute of Technology, Shenzhen, China,
[2]Peng Cheng Laboratory, Shenzhen, China
{yuanzk,wangh06}@pcl.ac.cn
{mliu,qinb}@ir.hit.edu.cn

Abstract

Evaluating the graph comprehension and reasoning abilities of Large Language Models (LLMs) is challenging and often incomplete. Existing benchmarks focus primarily on pure graph understanding, lacking a comprehensive evaluation across all graph types and detailed capability definitions. This paper presents GraCoRe, a benchmark for systematically assessing LLMs' graph comprehension and reasoning. GraCoRe uses a three-tier hierarchical taxonomy to categorize and test models on pure graph and heterogeneous graphs, subdividing capabilities into 10 distinct areas tested through 19 tasks. Our benchmark includes 11 datasets with 5,140 graphs of varying complexity. We evaluate four closed-source and eight open-source LLMs, conducting thorough analyses from both ability and task perspectives. Key findings reveal that OpenAI o1 model has amazing comprehension and reasoning capabilities, semantic enrichment enhances reasoning performance, node ordering impacts task success, and the ability to process longer texts does not necessarily improve graph comprehension or reasoning.GraCoRe is open-sourced at https://github.com/ZIKEYUAN/GraCoRe.

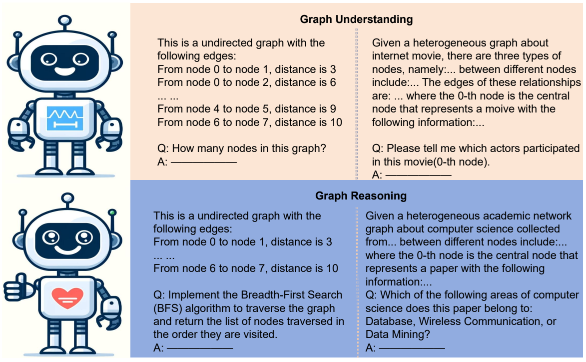

Figure 1: GraCoRe encompasses two overarching abilities and 19 distinct tasks within LLM on graph scenarios, facilitating a granular benchmarking from basic perceptivity to advanced interactivity.

1 Introduction

Graph understanding and complex reasoning are crucial capabilities of Large Language Models (LLMs), supporting applications in areas like social network analysis, drug discovery, recommendation systems, and spatiotemporal prediction (Brown et al., 2020). These abilities are particularly important for advancing Artificial General Intelligence (AGI) (Zhao et al., 2023). Graph-structured data mainly consists of homogeneous and heterogeneous graphs. Research on homogeneous graphs often addresses specific structural issues, such as protein-protein interaction prediction (Rao et al., 2014). Unlike homogeneous graphs, pure graphs are simpler, lacking node and edge attributes, as seen in graph-theoretic problems (Borgatti and Everett, 2006). For heterogeneous graphs, tasks leverage rich semantic information, like knowledge graph reasoning. However, enabling LLMs to effectively parse, understand, and reason with graph-structured data remains a significant challenge.

Firstly, the reasoning and comprehension abilities of current LLMs degrade significantly when processing complex graph-structured data with numerous nodes and edges. This decline is partly due to challenges in handling lengthy textual inputs that describe such data. Long texts increase computational burden and introduce noise, reducing the models' ability to capture essential details. Secondly, textual descriptions often involve complex entity relationships and abstract concepts, requiring models to understand explicit information and infer implicit connections. Current research primarily focuses on direct mappings from graph structure to answers (Wang et al., 2024), overlooking deeper reasoning capabilities. The absence of clear definitions and evaluation standards for graph reasoning highlights the need for a comprehensive benchmark to evaluate these abilities.

* B. Qin and H. Wang are corresponding authors.

Proceedings of the 31st International Conference on Computational Linguistics, pages 7925–7948

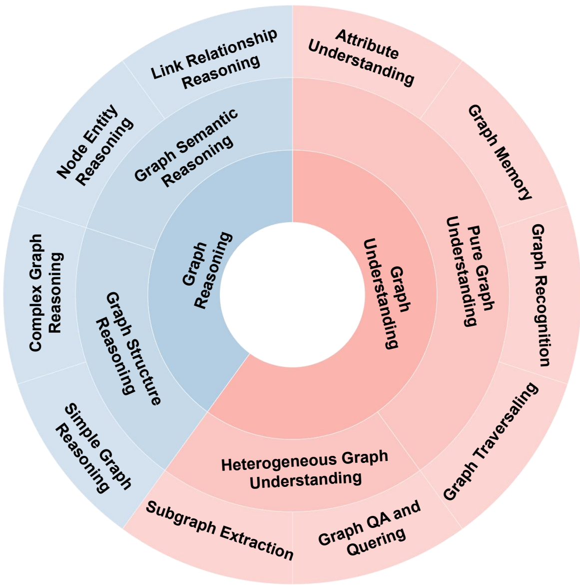

Figure 2: Our three-tier hierarchical ability taxonomy.

Previous research has introduced several benchmarks to evaluate LLMs' understanding and reasoning on graphs, but these benchmarks have notable limitations. Firstly, there is a **limited generalization** issue: most benchmarks test either pure or heterogeneous graphs separately, lacking a unified evaluation across both types. Additionally, there is a **lack of clear definition of model capabilities**: existing benchmarks are task-driven and fail to assess LLMs' specific abilities with graph data. Therefore, more comprehensive, ability-based benchmarks are needed to evaluate understanding and reasoning on graph-structured data. Lastly, there is **insufficient diversity in model types and tasks**: current benchmarks do not clearly classify tasks or test a wide range of models.

To address these challenges, we propose the GraCoRe benchmark, as shown in Figure 1, which aims to explore and evaluate the understanding and reasoning capabilities of mainstream LLMs. We have designed a **three-tier hierarchical ability taxonomy** that includes both capability-based and dataset-based categories. This taxonomy meticulously defines the model's capabilities and ensures greater generalization in the testing range. Regarding model types and task divisions, our benchmark tests multiple existing closed-source and open-source models, and divides tasks into multiple dimensions based on model capabilities. In Appendix A, we compare in detail the differences between the existing benchmark and GraCoRe, as well as our advantages.

Figure 2 presents the framework of our ability taxonomy. The first layer highlights two core capabilities: **graph understanding** and **graph reasoning**. Graph understanding reflects the model's ability to comprehend nodes, edges, and graph structure within the context, while graph reasoning builds on this, focusing on inferring implicit information from graph-structured data. The second layer categorizes LLM capabilities into four types based on data types. In the third layer, these are further divided into 10 distinct capabilities, evaluated through 19 tasks. This taxonomy provides multi-level evaluation, enabling detailed identification of model weaknesses. Each task is designed with specific prompts, structured by predefined rules to textualize graph information. GraCoRe includes 11 datasets with 5,140 graphs, with graph complexity controlled by factors such as size and network sparsity.

We conducted extensive experiments on GraCoRe to evaluate the graph understanding and reasoning abilities of LLMs, including four closed-source and eight open-source models. Key findings include:

- Graph reasoning is a major weakness in current LLMs, with most models unable to balance understanding and reasoning. OpenAI o1 model has amazing understanding and reasoning capabilities.
- LLMs perform better on graph reasoning tasks with semantic information than on purely structural tasks, showing that textual context enhances reasoning.
- Model performance is sensitive to node ordering in textual graph data, where ordered naming improves results.
- The ability to handle longer text does not impact performance, regardless of graph complexity or description length.

2 Related Work

LLMs for Graph Recent advancements in LLMs for graph tasks includes several notable contributions. (Li et al., 2023) categorizes these tasks into three types: Enhancer, Predictor, and Alignment. (Pan et al.) provides a forward-looking roadmap for the unification of LLMs and Knowledge Graphs (KGs).(Chai et al., 2023) proposes an end-to-end method for solving graph-related problems. Furthermore, (Das et al., 2023) investigates methods to improve the zero-shot reasoning ability of LLMs over structured data in a unified manner. (Yao et al., 2024) explores the graph generation capabilities of LLMs through systematic task designs and extensive experiments.

Benchmarks for LLMs on Graph Most benchmarks for evaluating LLMs on graph tasks are based on task testing. NLGraph(Wang et al., 2024) introduced a simple test dataset for eight graph tasks, while GPT4Graph(Guo et al., 2023) tested LLM capabilities on semantic tasks. (Liu and Wu, 2023) assessed the capabilities of four LLMs in graph data analysis. (Fatemi et al., 2023) proposed a method for describing graph data in text form. (Perozzi et al., 2024) designed a hint method specifically for graph tasks. GraphInstruct(Luo et al., 2024) provided diverse processes and steps for graph data generation. HiGPT(Tang et al., 2024a) proposed an evaluation method for heterogeneous graphs, and VisionGraph(Li et al., 2024) assessed the capabilities of LLMs on image graphs.

3 GraCoRe

This section begins by describing a three-tier hierarchical taxonomy for LLMs on graph data. Next, we explain the methodology used to collect the dataset. Finally, we present an analysis of the dataset's statistics.

3.1 Hierarchical Ability Taxonomy

After analyzing (Bai et al., 2024) evaluation of the LLMs in the multi-round dialogue task, we have developed a hierarchical taxonomy for classifying LLMs capabilities on graphs, essential for their evaluation. The taxonomy is structured into three levels, encompassing 19 tasks within 10 sub-capabilities. Table 1 summarizes each task with a brief description. This section elaborates on the three levels and their corresponding tasks, with detailed examples provided in the Appendix B.

3.1.1 Graph understanding

Understanding graph structure necessitates LLMs capable of accurately answering questions about the graph's basic properties and reconstructing its structural information from extensive text descriptions. This involves two core capabilities:

Pure graph understanding: Pure graphs refer to graph data containing only structural information, representing the simplest form of graph data(Jin et al., 2023). The structural information of such graphs can be encapsulated in an adjacency matrix. Consequently, research on pure graphs often emphasizes the structural information. Assessing the ability of LLMs to understand pure graphs should thus focus on their capacity to comprehend structural information. To this end, I have identified four sub-capabilities:

- **Pure Graph Attribute Understanding:** The most intuitive measure of understanding graph structure information is the correct comprehension of its basic attributes, such as the number of nodes, average degree, and performance on several sub-tasks related to node connectivity.
- **Pure Graph Memory:** This requires the large language model to reconstruct the input graph structure data, testing its memory capacity. This is specifically evaluated by the similarity score(Lahitani et al., 2016) of the reconstructed matrix.
- **Pure Graph Recognition:** For graph data with different structures, the model must be able to recognize and distinguish them. This study uses bipartite graphs and tree graphs to evaluate this capability.
- **Graph Traversal:** Traversal is fundamental to solving many graph theory reasoning problems. The model's performance in reasoning tasks is influenced by its traversal capability. This study primarily tests whether the model can traverse a graph using Breadth-First Search (BFS)(Beamer et al., 2013).

Heterogeneous graph understanding: Unlike pure graphs, heterogeneous graphs often contain rich semantic information, with much of the data collected from real-world scenarios. Consequently, understanding heterogeneous graphs typically involves grasping their semantic information, whereas understanding their structural information is less critical. We have refined this into two sub-capabilities:

- **Graph QA and Querying:** Given the rich semantic information in heterogeneous graphs, the ability of large language models to perform question-answering and querying is crucial. This capability includes three sub-tasks: querying neighbor nodes, answering questions about node relationships, and querying the number of relationships.
- **Subgraph Extraction:** This pertains to the overall understanding of relationships and nodes within heterogeneous graphs, assessing the model's ability to extract relevant subgraphs.

3.1.2 Graph reasoning

Based on the graph understanding capabilities of LLMs, their graph reasoning abilities are also

Task	Abbr.	Description
Node Number	NN	Calculate the total number of nodes in a graph.
Average Degree	AD	Calculate the average degree of the nodes in a graph.
Connectivity Test	CT	Determine if the graph is connected, meaning there is a path between any two nodes.
Matrix Similarity	MS	Evaluate the similarity between two adjacency matrices created from LLM and target graph.
Tree Recognition	TR	Identify if a graph is binary tree.
Bipartite Recognition	BR	Identify if a graph is bipartite, meaning its nodes can be divided into two disjoint sets such that no two nodes within the same set are adjacent.
Breadth First Search	BFS	Perform a breadth-first traversal starting from a given node.
Neighborhood Query	NQ	Query all nodes that are neighbors of a specified node.
Relationship Query	RQ	Query for specific relationships between nodes in the graph.
Relation Number	RN	Count the number of relationship types in the graph.
Subgraph Extraction	SE	Extract a subgraph based on specified criteria or nodes.
Shortest Path	SP	Find the shortest path between two nodes in a graph.
Maximum Flow	MF	Calculate the maximum flow in a flow network.
Eulerian Path	EP	Determine if there is a path that visits every edge exactly once.
Hamiltonian Cycle	HC	Determine if there is a cycle that visits every node exactly once and back start node.
Traveling Salesman Problem	TSP	Find the shortest possible route that visits each node exactly once and returns to the origin node.
Graph Coloring	GC	Assign colors to the nodes of the graph so that no two adjacent nodes share the same color.
Node Classification	NC	Classify nodes into predefined categories based on their attributes or graph structure.
Link Prediction	LP	Predict whether a link (edge) exists between two nodes in the graph.

Table 1: The 19 tasks for LLMs on graph within GraCoRe.

worth exploring. This ability requires the model to infer hidden information from known graph data. It includes the following two core capabilities:

Graph structure reasoning: Graph structural information reasoning requires large language models to understand the nodes, edges, and their connections to infer the overall structural characteristics of the graph or the structural patterns of specific subgraphs. For example, the model should be able to identify cycles, paths, tree structures, and hierarchical structures within the graph, and use these structural features for further reasoning. Tasks related to structural reasoning are primarily focused on graph theory problems. We classify the complexity of these problems into two categories: **Simple Graph Theory Problems Reasoning** and **Complex Graph Theory Problems Reasoning**. The classification criterion is the time complexity of the corresponding algorithms. Simple graph theory problems are solvable in polynomial time, while complex graph theory problems are NP-complete, requiring significantly more time to solve. This classification tests the model's capability in reasoning about graph theory problems. We have selected three representative problems for testing within each of these categories.

Graph semantic reasoning: Unlike structure-based reasoning, semantic information reasoning in graphs requires large language models to deeply understand the semantic meanings of nodes and edges, and to reason based on this semantic information. This involves modeling and reasoning about entities, relationships, and their interactions within the graph. Based on these tasks, we subdivide the semantic reasoning capabilities of large language models into **Node Entity Reasoning** and **Link Relationship Reasoning**. Corresponding tasks include node classification and link prediction. Previous studies have primarily addressed these two problems using graph neural networks (GNNs) such as GCN(Kipf and Welling, 2016) and GraphSAGE(Hamilton et al., 2017). These methods typically require large structured graph data for training and cannot directly utilize graph data containing text information for inference. Consequently, investigating the use of large models for semantic reasoning is highly significant. This research will explore whether text enhancement impacts the performance of LLMs on these two tasks

Benchmark	Graph Type	#Graph	Average Node	Average Edge	#Node Type	#Edge Type	#Task
PureGra	Bipartite Graph	460	19	49	1	1	5
	Tree Structure Graph	460	19	18	1	1	5
	Graph Traversal Graph	460	19	53	1	1	5
	Shortest Path Graph	460	19	53	1	1	5
	Max Flow Graph	460	19	106	1	1	1
	Graph Coloring	460	19	53	1	1	1
	Hamiltonian Graph	460	19	72	1	1	1
	TSP Graph	460	19	193	1	1	1
	Eulerian Graph	460	19	104	1	1	1
HeterGra	IMDBText	500	27	30	3	4	4
	ACMText	500	158	171	4	8	2
GraCoRe	/	**5140**	/	/	/	/	**19**

Table 2: Benchmark Graph Statistics.

3.2 Data Collection

We first divide the dataset into pure graphs and heterogeneous graphs to test the capabilities of large language models on these two types of datasets. For pure graphs, we customize unique data generation prompts based on the specific characteristics of each task, generating corresponding graph structure data using manually set rules. The scale of the graph is defined by the number of nodes and the sparsity of the network, ensuring that the generated data meets the specific needs of each task. For heterogeneous graph data, we use the ACM(Wang et al., 2019) and IMDB(Fu et al., 2020) datasets, converting them into text-based graph data. These are constructed according to a manually specified graph structure description framework, and unique prompts are designed for each task to build the dataset.

After generating the benchmark datasets, we also designed specific few-shot prompts for each task to test the model's capabilities. These prompt datasets will be included in the benchmark data to provide additional testing options. Finally, we will provide a standard answer for each task and filter out graph data that does not meet the corresponding task requirements.

3.3 Data Statistics

Table 2 shows several key statistics of our GraCoRe benchmark. We categorized the dataset based on graph structure into two main datasets: PureGra and HeterGra. Each main dataset contains multiple sub-datasets used for corresponding task testing. In total, there are 19 tasks with up to 5,140 graphs. Detailed statistics for each task can be found in the Appendix C.

For pure graph data, the datasets include graphs with 8 to 30 nodes, with 20 test graphs per dataset. This design is intended to assess the impact of graph complexity on the performance of large language models. For heterogeneous graph data, we divided them into IMDBText and ACMText datasets. The ACMText dataset is more complex and extensive, containing more semantic information than the IMDBText dataset. Therefore, the ACMText dataset is primarily used for complex reasoning tasks, including node classification and edge prediction.

GraCoRe is the first benchmark specifically focused on the fine-grained understanding and reasoning capabilities of large language models on graph data.

3.4 Evaluation

This thesis evaluates LLM outputs using exact match accuracy across various output types, including boolean values (e.g., for graph recognition tasks), integers (e.g., path lengths), floating-point numbers (e.g., similarity and average degree), and lists of nodes (e.g., paths). For tasks with multiple valid solutions (e.g., BFS), we check whether the output is a valid solution.

Standardized global scores(Dyck et al., 2005) have become the mainstream choice in fields such as educational assessment and intelligence testing. Compared to using raw scores, employing standardized scores in the multi-task evaluation of LLMs offers the following advantages:

- Standardized scores facilitate fair comparisons across different tasks and datasets. Given the varying difficulties and sensitivities of metrics across tasks, absolute performance metrics cannot be directly compared. Standardization normalizes these scores, enabling a more meaningful assessment of model performance across diverse tasks.
- Standardized scores reduce biases from specific tasks when ranking models. Models often dis-

Model	Pure Graph							Heterogeneous Graph			
	Attribute Understanding			Graph Memory	Graph Recognition		Grap Traversaling	Graph QA and Quering			Subgraph Extraction
	NN	AD	CT	MS	TR	BR	BFS	NQ	RQ	RN	SE
OpenAI o1	57.836	**81.591**	**58.491**	**58.959**	**96.545**	**71.257**	**66.263**	**72.029**	**58.731**	62.463	**75.531**
GPT-4o	**58.978**	59.195	41.355	58.906	50.275	41.450	55.864	57.564	53.095	60.99	60.042
GPT-4	58.887	52.842	56.054	58.853	43.737	26.832	55.349	63.648	52.898	**63.128**	60.763
GPT-3.5	**58.978**	42.041	37.166	56.521	28.775	60.293	60.949	41.671	26.302	45.879	43.594
Llama3.1-ins-8b	45.869	57.845	35.643	36.175	33.679	27.517	43.522	52.908	50.327	34.094	44.614
Llama3-ins-8b	54.228	26.077	26.275	44.918	41.851	25.576	60.835	29.628	52.502	57.996	33.748
Qwen2-7b-ins	53.634	16.229	56.968	52.283	36.445	47.959	29.237	39.498	52.502	19.364	55.360
Llama2-7b-chat	17.048	35.766	53.084	38.613	33.930	25.576	38.836	29.007	0.991	35.473	28.946
Vicuna-v1.5-16k	31.755	29.175	0.000	37.818	30.410	40.993	23.808	34.718	51.118	53.720	29.486
Chatglm3-6b	23.442	33.066	55.140	37.659	33.050	25.576	28.151	13.735	21.061	15.610	17.300
Chatglm2-32k-7b	13.987	23.377	17.974	6.239	35.816	74.683	12.895	46.576	21.061	15.610	24.744
Vicuna-v1.5-7b	18.646	36.084	55.140	6.345	28.775	25.576	17.580	12.307	52.700	28.962	19.161

Table 3: Standardized performance of graph understanding.

Model	Graph Structure Reasoning						Graph Semantic Reasoning		Average z	Total Score
	Simple Graph Theory Problems			Complex Graph Theory Problems			Node Entity Reasoning	Link Relationship Reasoning		
	SP	MF	EP	HC	TSP	GC	NC	LP		
OpenAI o1	**95.320**	**99.390**	**71.313**	**85.057**	**100.000**	**73.379**	66.409	53.236	**1.762**	**1403.799**
GPT-4o	52.665	35.019	61.671	41.405	37.647	37.860	**66.538**	**62.877**	0.601	993.397
GPT-4	42.456	33.045	62.255	43.303	36.700	33.954	65.435	57.641	0.528	967.778
GPT-3.5	32.852	33.834	60.867	51.788	35.872	45.581	57.908	48.830	0.251	869.702
Llama3.1-ins-8b	35.101	33.045	35.446	54.021	33.624	77.194	40.064	50.908	0.115	821.597
Llama3-ins-8b	39.081	35.809	30.698	56.477	34.334	49.306	36.041	55.563	0.028	790.943
Qwen2-7b-ins	37.005	37.389	39.537	37.498	38.238	28.957	36.625	52.903	-0.038	767.630
Llama2-7b-chat	36.486	40.153	19.594	24.882	34.570	41.039	27.994	33.786	-0.524	595.774
Vicuna-v1.5-16k	34.236	32.255	20.179	24.659	34.334	16.512	12.746	32.955	-0.595	570.875
Chatglm3-6b	23.681	40.153	42.897	24.659	36.463	40.585	32.407	21.734	-0.608	566.369
Chatglm2-32k-7b	26.103	40.153	28.141	24.659	35.872	22.871	20.013	20.072	-0.765	510.845
Vicuna-v1.5-7b	38.303	33.045	20.690	24.882	35.635	26.050	31.109	2.783	-0.756	513.774

Table 4: Standardized performance of graph reasoning.

play varying performance across tasks, making it difficult to gauge their strengths in graph understanding and reasoning. By normalizing scores, standardized metrics provide a more accurate and equitable basis for evaluating model capabilities across a broad range of tasks.

Since the metrics of different GraCoRe tasks are incomparable and differently sensitive, less experienced audiences cannot easily compare and interpret results, which is also prevalent in recent LLM benchmarks like Kola(Yu et al., 2023). Therefore, we utilized standardized global scores to evaluate the performance of LLMs on each task.

Given a task set $T = \{t_i\}_{i=1}^{|T|}$ and an evaluated model set $M = \{m_j\}_{j=1}^{|M|}$, so x_{ij} represents the performance of model m_j on task t_i. Then the standardized score z can be calculated as:

$$z_{ij} = \frac{x_{ij} - \mu\left(x_{i1}, \ldots, x_{i|M|}\right)}{\sigma\left(x_{i1}, \ldots, x_{i|M|}\right)}, \tag{1}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and standard deviation.Next, we use the Min-Max scaling(Patro and Sahu, 2015) method to adjust the scores to the range of 0-100, making it easier to observe and compare the results. The final scores are presented as:

$$s_{ij} = 100 \cdot \frac{z_{ij} - \min(z)}{\max(z) - \min(z)}, \tag{2}$$

where the functions max (z) and min (z) correspond to the maximum and minimum of all z_{ij} scores.

4 Experiments

4.1 Experimental Setup

Using the GraCoRe benchmark, we investigate whether language models can understand and reason about graph structures through textual descriptions. Furthermore, we examine whether tailored prompts can improve their performance on graph-related tasks.

Models and Settings We evaluated a total of eight popular models on the GraCoRe benchmark, including four closed-source models and eight open-source models. The closed-source models are: OpenAI o1 , GPT-4o, GPT-4, and GPT-3.5(Achiam et al., 2023). The open-source

This model is an LLMs with powerful reasoning capabilities launched by OpenAI on September 12, 2024. Due to time reasons, we only evaluated it in the main experiment.

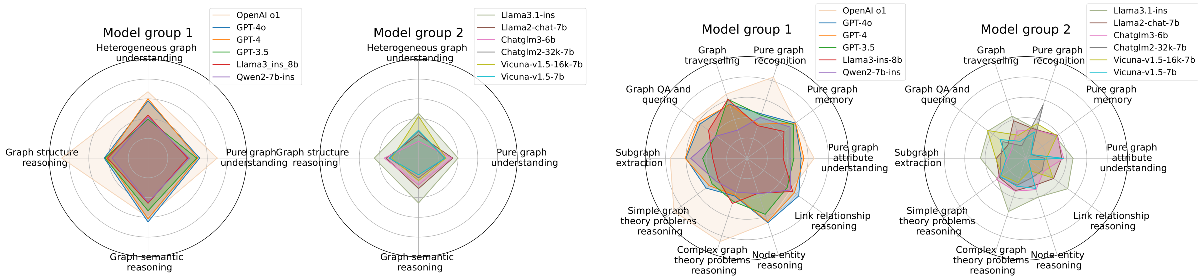

Figure 3: Performance of various LLMs for second and third layer ability dimension.

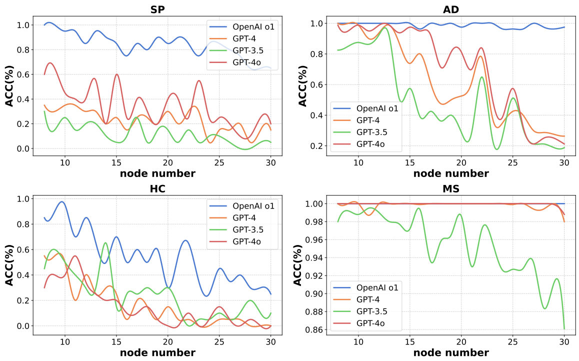

Figure 4: Effect of Graph Size.

models are: LLama3.1-ins-8b, LLama3-ins-8b, LLama2-7b-chat(Touvron et al., 2023), Chatglm3-6b, Chatglm2-32k-7b(Du et al., 2021), Vicuna-v1.5-7b, Vicuna-v1.5-16k-7b(Chiang et al., 2023) and Qwen2-7b-ins(Bai et al., 2023). More details about these models can be found in the Appendix B.

4.2 Main Results

Task Dimensional Analysis Tables 3 and 4 present the performance of various LLMs across 19 tasks in the GraCoRe benchmark. Models generally performed better on graph understanding tasks, while graph reasoning proved more challenging. Closed-source models, particularly OpenAI o1, excelled in both areas, achieving the highest score of 1403.8, significantly surpassing other closed-source models. Among open-source models, Llama3.1-8b and Qwen2-7b-ins also performed well, ranking 5th and 7th with scores of 821.6 and 767.6, respectively. In contrast, Chatglm2-7b underperformed. The overall weaker performance on reasoning tasks was anticipated.

The average z-scores reveal a significant gap between open-source and commercial models. Only the Llama3.1-ins-8b and Llama3-ins-8b models achieve z-scores above 0, indicating above-average performance. This highlights the need for increased collaboration within the open-source community to enhance large language models. Notably, these two models demonstrate graph understanding and reasoning capabilities approaching those of GPT-3.5.

Ability Dimensional Analysis We further analyze Tables 3 and 4 to assess the models' overall performance from a capability perspective, using radar charts to visually represent the second and third layers of the three-tier taxonomy. The left part of Figure 3 illustrates the performance of LLMs across four capability dimensions at the second layer, with each dimension's score averaged across relevant tasks. Most models excel in graph understanding and semantic reasoning but require improvement in structural reasoning. The OpenAI o1 model notably addresses the structural reasoning weaknesses of other OpenAI models, offering a more balanced performance. However, most open-source models underperform in heterogeneous graph understanding tasks.

The right part of Figure 3 provides a detailed analysis of LLM performance across ten capability dimensions at the third layer. It reveals that large language models generally struggle with reasoning in graph theory, especially in complex tasks. However, models like OpenAI o1, GPT-4, and GPT-4o show strong and balanced graph processing abilities, excelling in more complex tasks. Notably, the OpenAI o1 model significantly outperforms others in graph theory reasoning, with performance several times better than other OpenAI models. In contrast, other models display inconsistent performance. For instance, Vicuna-v1.5-16k-7b excels in understanding heterogeneous graph structures but falls behind in other areas, suggesting that rich semantic information may enhance graph processing capabilities in certain contexts.

Long-Text-Specific Models Since graph structure data described in text often consists of long texts, the ability of models to handle long texts is also worth noting. As shown in Table 3, models capable of processing long texts, such as Chatglm2-32k-7b and Vicuna-v1.5-16k-7b, performed poorly in graph processing tasks. Compared to other models, their performance was even lower. This suggests that despite being designed for long text input and output, these models still require further development and training to effectively enhance their graph processing capabilities.

Finally, we give the measured values of each model in each task in the main experiment in the Appendix D.

4.3 Further Analysis

Effect of Graph Size

Figure 4 shows the performance of four OpenAI LLMs on two graph comprehension and two graph reasoning tasks as node count increases. Results reveal a consistent performance decline across all tasks, particularly in the average degree task for graph comprehension, indicating increased computational complexity as node count grows. Despite this, the OpenAI o1 model shows strong computational ability, with minimal impact on comprehension tasks. However, both the shortest path and Hamiltonian path tasks in graph reasoning are notably affected, even for the robust OpenAI o1 and GPT-3.5 models. These findings indicate that current LLMs require further improvement in graph reasoning, particularly with large-scale graphs, where their performance drops considerably.

Effect of Random Sort Since our data consists of randomly generated graphs with nodes named by numbers, the performance of node-related tasks may be affected by changes in node order. In this study, we examine the impact of text order on the understanding of graph structures by large language models by comparing random sorting with sequential sorting. The results in Table 5 indicate that the model's performance under sequential sorting is often superior to that under random sorting, particularly in graph path reasoning tasks, where the impact is significant. This suggests that renaming nodes and ordering them sequentially can enhance model performance in path reasoning problems. However, it also highlights the model's lack of training on graph data with random sorting. Due to the inference cost, we chose only GPT-3.5 from OpenAI's closed-source models for testing.

Effect of Text Enhancement

Heterogeneous graphs provide rich semantic information that aids in understanding and reasoning about graph-related text. To assess whether LLMs can reason purely from structural data, we excluded enhanced text features, such as titles and abstracts, from the ACMText dataset. Figure 5 illustrates the performance of various models in a node classification task. Results show that GPT-4 and GPT-4o maintain strong prediction capabilities without text, indicating effective reasoning based solely on structure. However, GPT-3.5's performance declines significantly without text, while open-source models show minimal impact from its absence.

Model		**NN**	**ST**	**BR**	**SP**	**HP**
GPT-3.5	sort	**0.993**	**0.954**	**0.504**	**0.117**	**0.243**
	random	0.968	0.865	0.346	0.070	0.048
Qwen2-7b	sort	**0.876**	0.874	**0.396**	**0.165**	**0.115**
	random	0.773	**0.885**	0.374	0.159	0.000
Llama3-ins-8b	sort	**0.889**	0.735	0.200	0.189	**0.285**
	random	0.786	**0.809**	0.200	**0.198**	0.059
Llama3.1-ins-8b	sort	**0.706**	**0.57**	**0.217**	**0.143**	**0.263**
	random sort	0.52	0.552	0.198	0.102	0.243
Chatglm3-6b	sort	**0.215**	0.598	0.200	0.011	0.000
	random sort	0.196	**0.603**	0.200	0.011	0.000
Chatglm2-32k-7b	sort	0.008	**0.005**	**0.63**	0.039	0.000
	random sort	**0.011**	0.004	0.504	**0.043**	0.000
Llama2-7b-chat-hf	sort	0.075	**0.616**	0.200	**0.159**	0.002
	random sort	**0.105**	0.413	0.200	0.133	0.002

Table 5: Performance of different models with sorted and random sorted grpah text input.

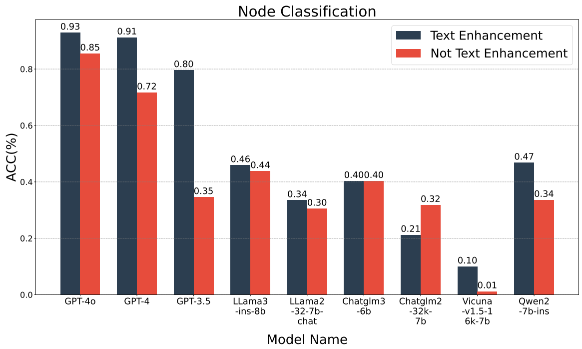

Figure 5: Effect of Text Enhancement.

5 Conclusion

This paper introduces GraCoRe, a benchmark designed to evaluate large language models' (LLMs) ability to understand and reason with graph-structured data. We present a detailed, multi-level classification system for assessing model performance on graph-based tasks. Using GraCoRe, we evaluate 12 prominent LLMs, identifying significant limitations in their graph reasoning capabilities. Our experimental results validate the benchmark's effectiveness in measuring LLM performance on graph tasks. Future research will focus on solving complex graph theory problems in large-scale pure graphs, with potential improvements in graph reasoning through methods like agents, chain-of-thought (CoT), and retrieval-augmented generation (RAG).

Limitations

As LLMs continue to develop, the volume of training data and their capacity to represent graphs are likely to increase. Our current evaluation may not encompass all their capabilities, and some models might incorporate our data for training, potentially influencing the final evaluation outcomes. In the future, we aim to continually refine and update the GraCoRe benchmark to more effectively assess the graph understanding and reasoning abilities of emerging LLMs.

Acknowledgments

The research in this article is supported by the National Key R&D Pdrogram of China (Grant No. 2021ZD0112905), the National Science Foundation of China (Grant No. U22B2059, 62276083), the Major Key Project of PCL (Grant No. PCL2023A09), the Human-Machine Integrated Consultation System for Cardiovascular Diseases (Grant No. 2023A003). We also appreciate the support from China Mobile Group Heilongjiang Co., Ltd. @ on our research,the research is jointly completed by both parties.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, et al. 2024. Mt-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. *arXiv preprint arXiv:2402.14762*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Scott Beamer, Krste Asanović, and David Patterson. 2013. Direction-optimizing breadth-first search. *Scientific Programming*, 21(3-4):137–148.

Stephen P Borgatti and Martin G Everett. 2006. A graph-theoretic perspective on centrality. *Social networks*, 28(4):466–484.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. 2023. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint arXiv:2310.05845*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, march 2023. *URL https://lmsys. org/blog/2023-03-30-vicuna*, 3(5).

Debarati Das, Ishaan Gupta, Jaideep Srivastava, and Dongyeop Kang. 2023. Which modality should i use–text, motif, or image?: Understanding graphs with large language models. *arXiv preprint arXiv:2311.09862*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. Glm: General language model pretraining with autoregressive blank infilling. *arXiv preprint arXiv:2103.10360*.

Peter J Dyck, Christopher J Boes, Donald Mulder, Clark Millikan, Anthony J Windebank, P James B Dyck, and Raul Espinosa. 2005. History of standard scoring, notation, and summation of neuromuscular signs. a current survey and recommendation. *Journal of the Peripheral Nervous System*, 10(2):158–173.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2023. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560*.

Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the web conference 2020*, pages 2331–2341.

Jiayan Guo, Lun Du, Hengyu Liu, Mengyu Zhou, Xinyi He, and Shi Han. 2023. Gpt4graph: Can large language models understand graph structured data? an empirical evaluation and benchmarking. *arXiv preprint arXiv:2305.15066*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. 2023. Large language models on graphs: A comprehensive survey. *arXiv preprint arXiv:2312.02783*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Alfirna Rizqi Lahitani, Adhistya Erna Permanasari, and Noor Akhmad Setiawan. 2016. Cosine similarity to determine similarity measure: Study case in online essay assessment. In *2016 4th International conference on cyber and IT service management*, pages 1–6. IEEE.

Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. 2023. A survey of graph meets large language model: Progress and future directions. *arXiv preprint arXiv:2311.12399*.

Yunxin Li, Baotian Hu, Haoyuan Shi, Wei Wang, Longyue Wang, and Min Zhang. 2024. Visiongraph: Leveraging large multimodal models for graph theory problems in visual context. *arXiv preprint arXiv:2405.04950*.

Chang Liu and Bo Wu. 2023. Evaluating large language models on graphs: Performance insights and comparative analysis. *arXiv preprint arXiv:2308.11224*.

Zihan Luo, Xiran Song, Hong Huang, Jianxun Lian, Chenhao Zhang, Jinqi Jiang, Xing Xie, and Hai Jin. 2024. Graphinstruct: Empowering large language models with graph understanding and reasoning capability. *arXiv preprint arXiv:2403.04483*.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap, 2023. *arXiv preprint arXiv:2306.08302*.

SGOPAL Patro and Kishore Kumar Sahu. 2015. Normalization: A preprocessing stage. *arXiv preprint arXiv:1503.06462*.

Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan Halcrow. 2024. Let your graph do the talking: Encoding structured data for llms. *arXiv preprint arXiv:2402.05862*.

V Srinivasa Rao, K Srinivas, GN Sujini, and GN Sunand Kumar. 2014. Protein-protein interaction detection: methods and analysis. *International journal of proteomics*, 2014(1):147648.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. 2024a. Higpt: Heterogeneous graph language model. *arXiv preprint arXiv:2402.16024*.

Jianheng Tang, Qifan Zhang, Yuhan Li, and Jia Li. 2024b. Grapharena: Benchmarking large language models on graph computational problems. *arXiv preprint arXiv:2407.00379*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2024. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032.

Yang Yao, Xin Wang, Zeyang Zhang, Yijian Qin, Ziwei Zhang, Xu Chu, Yuekui Yang, Wenwu Zhu, and Hong Mei. 2024. Exploring the potential of large language models in graph generation. *arXiv preprint arXiv:2403.14358*.

Jifan Yu, Xiaozhi Wang, Shangqing Tu, Shulin Cao, Daniel Zhang-Li, Xin Lv, Hao Peng, Zijun Yao, Xiaohan Zhang, Hanming Li, et al. 2023. Kola: Carefully benchmarking world knowledge of large language models. *arXiv preprint arXiv:2306.09296*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

A Comparison with existing benchmarks

In the Introduction, we outlined the advantages of the GraCoRe benchmark. First, it evaluates LLMs' understanding and reasoning abilities on rich graph-structured data while also quantifying their performance on graph-related tasks. Second, GraCoRe tests a wider range of newer models and covers more tasks compared to existing benchmarks. Table 6 highlights the key differences between the GraCoRe benchmark and other benchmarks, demonstrating its broader scope and capabilities.

B Details on the data generation and models

Our dataset consists primarily of two types of graph data: heterogeneous graphs and pure graphs. Based on these, we developed 19 tasks across 11 datasets to evaluate large models. This diversity ensures the comprehensiveness and robustness of our dataset.

Figures 6 to 26 provide examples of prompts and graph data descriptions for each task. To ensure that generated results meet our capability and task-specific requirements, we concatenate the prompts for each task with the initial prompts.

We evaluated 12 of the latest LLMs, including OpenAI o1 reasoning model, launched on September 12, 2024. Table 7 presents more details on the models and their versions.

C More details about each task

Due to space limitations in the main text, this section will provide a detailed explanation of the specific details for each task. First, we will discuss the classification of model diagram understanding and reasoning capabilities associated with each task, and evaluate the number of test images used for each task. The evaluation results for all tasks are measured using accuracy (ACC). Table 10 summarizes the details of each task.

D Raw measurements of each task

In order to solve the problem of being unable to quantify the capabilities of each model in graph understanding and reasoning tasks, and to compare the differences in capabilities of each model, we use the standardized global scores metric to solve the above problem. However, we will still provide the actual measurement values of the large model in each task, and table 8 and table 9 shows the actual measurement values.

Benchmark	Graph Type		Evaluation Perspective		Task #	Model #
	Pure	Heterogeneous	Task	Model		
NLGraph(Wang et al., 2024)	✔	✖	✔	✖	8	2
GPT4Graph(Guo et al., 2023)	✖	✔	✔	✖	10	4
GraphArena(Tang et al., 2024b)	✖	✔	✔	✖	10	10
GraphInstruct(Luo et al., 2024)	✔	✖	✔	✖	21	6
GraCoRe	✔	✔	✔	✔	19	12

Table 6: Differences between GraCoRe and existing benchmarks.

Model	Version	Model Link
OpenAI o1	o1-mini	`https://platform.openai.com/docs/models/o1`
GPT-4o	gpt-4o	`https://platform.openai.com/docs/models/gpt-4o`
GPT-4	gpt-4-turbo	`https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4`
GPT-3.5	gpt-3.5-turbo	`https://platform.openai.com/docs/models/gpt-3-5-turbo`
Llama3.1-ins-8b	Meta-Llama-3.1-8B-Instruct	`https://huggingface.co/meta-llama/Meta-Llama-3.1-8B-Instruct`
Llama3-ins-8b	Meta-Llama-3-8B-Instruct	`https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct`
Llama2-7b-chat	Llama-2-7b-chat-hf	`https://huggingface.co/meta-llama/Llama-2-7b-chat-hf`
Chatglm3-6b	chatglm3-6b	`https://huggingface.co/THUDM/chatglm3-6b`
Chatglm2-32k-7b	chatglm2-6b-32k	`https://huggingface.co/THUDM/chatglm2-6b-32k`
Vicuna-v1.5-16k	vicuna-7b-v1.5-16k	`https://huggingface.co/lmsys/vicuna-7b-v1.5-16k`
Qwen2-7b-ins	Qwen2-7B-Instruct	`https://huggingface.co/Qwen/Qwen2-7B-Instruct`
Vicuna-v1.5-7b	vicuna-7b-v1.5	`https://huggingface.co/lmsys/vicuna-7b-v1.5`

Table 7: More details about models.

Model	Pure Graph							Heterogeneous Graph			
	Attribute Understanding			Graph Memory	Graph Recognition		Grap Traversaling	Graph QA and Quering			Subgraph Extraction
	NN(ACC)	AD(ACC)	CT(ACC)	MS(ACC)	TR(ACC)	BR(ACC)	BFS(ACC)	NQ(ACC)	RQ(ACC)	RN(ACC)	SE(ACC)
o1-mini	0.968	**0.988**	**0.978**	**1.000**	**0.998**	**0.600**	**0.991**	**0.978**	**0.619**	0.986	**0.971**
GPT-4o	**0.993**	0.706	0.753	0.999	0.630	0.339	0.809	0.745	0.562	0.955	0.713
GPT-4	0.991	0.626	0.946	0.998	0.578	0.211	0.809	0.843	0.560	**1.000**	0.725
GPT-3.5	**0.993**	0.490	0.698	0.954	0.459	0.504	0.898	0.489	0.291	0.637	0.439
Llama3.1-ins-8b	0.706	0.689	0.678	0.570	0.498	0.217	0.593	0.670	0.534	0.389	0.456
Llama3-ins-8b	0.889	0.289	0.555	0.735	0.563	0.200	0.896	0.295	0.556	0.892	0.275
Llama2-7b-chat	0.075	0.411	0.907	0.616	0.500	0.200	0.511	0.285	0.035	0.418	0.195
Chatglm3-6b	0.215	0.377	0.934	0.598	0.493	0.200	0.324	0.039	0.238	0.000	0.001
Chatglm2-32k-7b	0.008	0.255	0.446	0.005	0.515	0.630	0.057	0.568	0.238	0.000	0.125
Vicuna-v1.5-16k	0.397	0.328	0.210	0.601	0.472	0.335	0.248	0.377	0.542	0.802	0.204
Qwen2-7b-ins	0.876	0.165	0.958	0.874	0.520	0.396	0.343	0.454	0.556	0.079	0.635
Vicuna-v1.5-7b	0.110	0.415	0.934	0.007	0.459	0.200	0.139	0.016	0.558	0.281	0.032

Table 8: Performance of graph understanding.

Model	Graph Structure Reasoning						Graph Semantic Reasoning	
	Simple Graph Theory Problems			Complex Graph Theory Problems			Node Entity Reasoning	Link Relationship Reasoning
	SP(ACC)	MF(ACC)	EP(ACC)	HC(ACC)	TSP(ACC)	GC(ACC)	NC(ACC)	LP(ACC)
o1-mini	**0.839**	**0.183**	**0.915**	**0.541**	**0.570**	**0.754**	0.927	0.621
GPT-4o	0.346	0.020	0.783	0.15	0.043	0.363	**0.929**	**0.737**
GPT-4	0.228	0.015	0.791	0.167	0.035	0.320	0.912	0.674
GPT-3.5	0.117	0.017	0.772	0.243	0.028	0.448	0.796	0.568
Llama3.1-ins-8b	0.143	0.015	0.424	0.263	0.009	0.796	0.521	0.593
Llama3-ins-8b	0.189	0.022	0.359	0.285	0.015	0.489	0.459	0.649
Llama2-7b-chat	0.159	0.033	0.207	0.002	0.017	0.398	0.335	0.387
Chatglm3-6b	0.011	0.033	0.526	0.000	0.033	0.393	0.403	0.242
Chatglm2-32k-7b	0.039	0.033	0.324	0.000	0.028	0.198	0.212	0.222
Vicuna-v1.5-16k	0.133	0.013	0.215	0.000	0.015	0.128	0.100	0.377
Qwen2-7b-ins	0.165	0.026	0.480	0.115	0.048	0.265	0.468	0.617
Vicuna-v1.5-7b	0.180	0.015	0.222	0.002	0.026	0.233	0.383	0.014

Table 9: Performance of graph reasoning.

Level 3	Task	Dataset	Graph #
Attribute Understanding	NN	BG/TSG/GTG/SPG	1840
	AD	BG/TSG/GTG/SPG	1840
	CT	BG/TSG/GTG/SPG	1840
Graph Memory	MS	BG/TSG/GTG/SPG	1840
Graph Recognition	TR	Tree Structure Graph	460
	BR	Bipartite Graph	460
Graph Traversaling	BFS	Graph Traversal Graph	460
Graph QA and Querying	NQ	IMDBText	500
	RQ	IMDBText	500
	RN	IMDBText	500
Subgraph Extraction	SE	IMDBText	500
Simple Graph Theory Problems	SP	Shortest Path Graph	460
	MF	Max Flow Graph	460
	EP	Eulerian Graph	460
Complex Graph Theory Problems	HC	Hamiltonian Graph	460
	TSP	TSP Graph	460
	GC	Graph Coloring	460
Node Entity Reasoning	NC	ACMText	500
Link Relationship Reasoning	LP	ACMText	500

Table 10: Overview of Tasks and Datasets for Graph-based Problems.

 # Initial Instructions # You are provided with a graph described by its nodes and edges. Your task is to analyze and understand the graph's properties. Identify and describe key attributes such as connectivity, degree of nodes, presence of cycles, shortest paths, and any other relevant characteristics. Provide detailed explanations and steps for your analysis.

Figure 6: The initial instructions for answer generation.

 # Initial Instructions # You are tasked with solving complex graph theory problems such as the Traveling Salesman Problem, Minimum Coloring, and Hamiltonian Path. Given a graph described by its nodes and edges, identify the problem type and provide an optimal or near-optimal solution using appropriate algorithms. Ensure your reasoning process is clear and all steps are documented.

Figure 7: The initial instructions for answer generation.

This is a undirected graph with the following edges:
From node 0 to node 6, distance is 4
From node 0 to node 1, distance is 2
From node 0 to node 3, distance is 3
From node 1 to node 2, distance is 1
From node 2 to node 7, distance is 2
From node 2 to node 4, distance is 5
From node 2 to node 5, distance is 2
From node 2 to node 3, distance is 5
From node 3 to node 6, distance is 2
From node 3 to node 4, distance is 2
From node 3 to node 5, distance is 2
From node 4 to node 6, distance is 4
From node 4 to node 5, distance is 1
From node 6 to node 7, distance is 4
Q: How many nodes in this graph? Please provide the answer directly without the reasoning process.
A: 8

Figure 8: The unique prompt for the Node Number task.

This is a undirected graph with the following edges:
From node 0 to node 6, distance is 4
From node 0 to node 1, distance is 2
From node 0 to node 3, distance is 3
From node 1 to node 2, distance is 1
From node 2 to node 7, distance is 2
From node 2 to node 4, distance is 5
From node 2 to node 5, distance is 2
From node 2 to node 3, distance is 5
From node 3 to node 6, distance is 2
From node 3 to node 4, distance is 2
From node 3 to node 5, distance is 2
From node 4 to node 6, distance is 4
From node 4 to node 5, distance is 1
From node 6 to node 7, distance is 4
Q: What's the average degree of this graph? Round the result to one decimal place. Please provide the answer directly without the reasoning process.
A: The average degree of the graph is 3.5.

Figure 9: The unique prompt for the Average Degree task.

This is a undirected graph with the following edges:
From node 0 to node 6, distance is 4
From node 0 to node 1, distance is 2
From node 0 to node 3, distance is 3
From node 1 to node 2, distance is 1
From node 2 to node 7, distance is 2
From node 2 to node 4, distance is 5
From node 2 to node 5, distance is 2
From node 2 to node 3, distance is 5
From node 3 to node 6, distance is 2
From node 3 to node 4, distance is 2
From node 3 to node 5, distance is 2
From node 4 to node 6, distance is 4
From node 4 to node 5, distance is 1
From node 6 to node 7, distance is 4
Q: Is this graph a connected graph? Please provide the answer directly without the reasoning process.
A: Yes, this graph is a connected graph.

Figure 10: The unique prompt for the Connectivity Test task.

This is a undirected graph with the following edges:
From node 0 to node 6, distance is 4
From node 0 to node 1, distance is 2
From node 0 to node 3, distance is 3
From node 1 to node 2, distance is 1
From node 2 to node 7, distance is 2
From node 2 to node 4, distance is 5
From node 2 to node 5, distance is 2
From node 2 to node 3, distance is 5
From node 3 to node 6, distance is 2
From node 3 to node 4, distance is 2
From node 3 to node 5, distance is 2
From node 4 to node 6, distance is 4
From node 4 to node 5, distance is 1
From node 6 to node 7, distance is 4
Q: What are the triplets in this graph? Please output the triplet (start_node,end_node,weight) of this graph in LIST format.For example:[(1,2,3),(4,5,6)]
A: The triplets (start_node, end_node, weight) represent the edges in the undirected graph with their respective weights. Based on the given edges, the list format of triplets is as follows:[(0, 6, 4), (0, 1, 2), (0, 3, 3), (1, 2, 1), (2, 7, 2), (2, 4, 5), (2, 5, 2), (2, 3, 5), (3, 6, 2), (3, 4, 2), (3, 5, 2), (4, 6, 4), (4, 5, 1), (6, 7, 4)]

Figure 11: The unique prompt for the Matrix Similarity task.

This is a undirected graph with the following edges:
From node 0 to node 6, distance is 4
From node 0 to node 1, distance is 2
From node 0 to node 3, distance is 3
From node 1 to node 2, distance is 1
From node 2 to node 7, distance is 2
From node 2 to node 4, distance is 5
From node 2 to node 5, distance is 2
From node 2 to node 3, distance is 5
From node 3 to node 6, distance is 2
From node 3 to node 4, distance is 2
From node 3 to node 5, distance is 2
From node 4 to node 6, distance is 4
From node 4 to node 5, distance is 1
From node 6 to node 7, distance is 4
Q: What is the shortest path from node 4 to node 1? How long is it? Please provide the answer directly without the reasoning process and output it in JSON format. For example: {"path":PATH,"distance":DISTANCE}
A: {"path":[1,2,4],"distance":6}

Figure 12: The unique prompt for the Shortest Path task.

This is a undirected graph with the following edges:
From node 0 to node 4, distance is 3
From node 0 to node 5, distance is 5
From node 0 to node 6, distance is 2
From node 1 to node 4, distance is 5
From node 2 to node 4, distance is 3
From node 2 to node 5, distance is 2
From node 2 to node 6, distance is 4
From node 2 to node 7, distance is 4
From node 3 to node 5, distance is 2
From node 3 to node 7, distance is 3

Q: A bipartite graph is a special type of graph where the vertex set can be divided into two disjoint subsets such that every edge connects a vertex in one subset to a vertex in the other subset. Please determine if the given graph is bipartite.Please provide the answer directly without the reasoning process .

A: Yes, the given graph is bipartite.

Figure 13: The unique prompt for the Bipartite Recognition task.

This is a undirected graph with the following edges:
From node 0 to node 3, distance is 4
From node 0 to node 5, distance is 3
From node 0 to node 6, distance is 3
From node 0 to node 7, distance is 1
From node 1 to node 3, distance is 4
From node 1 to node 6, distance is 5
From node 2 to node 6, distance is 2
From node 2 to node 7, distance is 4
From node 5 to node 7, distance is 4
From node 6 to node 7, distance is 3

Q: An Eulerian circuit in this graph is a cycle that visits every edge exactly once and returns to the starting node.Please determine if an Eulerian circuit exists in this graph. Please provide the answer directly without the reasoning process . Yes or No.
A: No, it does not have an Eulerian circuit.

Figure 14: The unique prompt for the Eulerian Path task.

This is a undirected graph with the following edges:
From node 0 to node 6, distance is 5
From node 0 to node 3, distance is 5
From node 1 to node 7, distance is 3
From node 1 to node 6, distance is 5
From node 1 to node 2, distance is 1
From node 1 to node 3, distance is 5
From node 2 to node 6, distance is 4
From node 2 to node 5, distance is 1
From node 2 to node 3, distance is 2
From node 3 to node 5, distance is 3
From node 3 to node 4, distance is 5
From node 4 to node 7, distance is 5
From node 4 to node 5, distance is 5
From node 6 to node 7, distance is 3
Q: Use a greedy algorithm with the "largest_first" strategy to determine the minimum number of colors needed to color the graph, ensuring that no two adjacent nodes share the same color.Please provide the answer directly without the reasoning process .
A: Minimum number of colors required: 4

Figure 15: The unique prompt for the Graph Coloring task.

This is a undirected graph with the following edges:
From node 0 to node 7, distance is 5
From node 0 to node 3, distance is 2
From node 0 to node 2, distance is 2
From node 1 to node 5, distance is 5
From node 1 to node 2, distance is 1
From node 1 to node 7, distance is 1
From node 1 to node 3, distance is 4
From node 2 to node 3, distance is 2
From node 3 to node 6, distance is 4
From node 4 to node 5, distance is 2
From node 4 to node 7, distance is 2
From node 6 to node 7, distance is 4

Q: Implement the Breadth-First Search (BFS) algorithm to traverse the graph and return the list of nodes traversed in the order they are visited.Visited the graph start node 0.You can choise A,B,C or D.
A. [0, 2, 5, 3, 1, 6, 4, 7] B. [0, 7, 3, 2, 6, 1, 4, 5] C. [0, 4, 2, 7, 6, 3, 5, 1] D. [0, 1, 2, 3, 5, 4, 6, 7] Please provide the answer directly.
A: B.

Figure 16: The unique prompt for the Breadth First Search task.

This is a directed graph with the following edges:
From node 3 to node 0, distance is 2
From node 3 to node 6, distance is 2
From node 0 to node 5, distance is 3
From node 0 to node 2, distance is 4
From node 0 to node 7, distance is 2
From node 5 to node 7, distance is 3
From node 5 to node 4, distance is 5
From node 7 to node 1, distance is 5
From node 7 to node 6, distance is 2
From node 7 to node 4, distance is 2
From node 1 to node 2, distance is 5
From node 1 to node 7, distance is 4
From node 1 to node 3, distance is 4
From node 1 to node 4, distance is 5
From node 1 to node 6, distance is 4
From node 2 to node 4, distance is 4
From node 2 to node 5, distance is 1
From node 2 to node 6, distance is 4
From node 4 to node 6, distance is 3
From node 4 to node 5, distance is 3
From node 6 to node 3, distance is 3
From node 6 to node 4, distance is 1
Q: A Hamiltonian cycle in this graph is a cycle that visits each node exactly once and returns to the starting node. A Hamiltonian cycle exists in this graph, provide the sequence of nodes that form this cycle. Please provide the answer directly without the reasoning process and output it in JSON format. For example: {"Hamiltonian_cycle": CYCLE}
A: {"Hamiltonian_cycle": [3, 0, 5, 7, 4, 6, 1, 2, 3]}

Figure 17: The unique prompt for the Hamiltonian Cycle task.

This is a directed graph with the following edges:
From node 0 to node 2, distance is 2
From node 0 to node 5, distance is 5
From node 0 to node 6, distance is 3
From node 0 to node 7, distance is 3
From node 0 to node 1, distance is 3
From node 0 to node 3, distance is 5
From node 1 to node 2, distance is 4
From node 1 to node 4, distance is 1
From node 1 to node 7, distance is 4
From node 2 to node 3, distance is 2
From node 2 to node 4, distance is 2
From node 2 to node 5, distance is 5
From node 2 to node 6, distance is 1
From node 3 to node 4, distance is 1
From node 3 to node 5, distance is 5
From node 3 to node 6, distance is 1
From node 4 to node 5, distance is 3
From node 4 to node 6, distance is 3
From node 4 to node 7, distance is 3
From node 5 to node 6, distance is 5
From node 5 to node 7, distance is 2
From node 6 to node 7, distance is 3
Q: What is the maximum flow from node 0 to node 7?Please provide the answer directly without the reasoning process.
A: The maximum flow is 14. The flow paths are [(0, 2, 2), (0, 6, 2), (0, 7, 3), (0, 1, 3), (0, 3, 4), (1, 7, 3), (2, 4, 2), (3, 4, 1), (3, 5, 2), (3, 6, 1), (4, 7, 3), (5, 7, 2), (6, 7, 3)].

Figure 18: The unique prompt for the Maximum Flow task.

This is a undirected graph with the following edges:
From node 0 to node 1, distance is 4
From node 0 to node 2, distance is 5
From node 1 to node 3, distance is 1
From node 1 to node 4, distance is 2
From node 2 to node 5, distance is 4
From node 2 to node 6, distance is 5
From node 2 to node 3, distance is 4
From node 3 to node 7, distance is 1

Q: A binary tree is a tree data structure in which each node has at most two children, referred to as the left child and the right child. Based on the following description, determine if the given graph is a binary tree.
Answer:(Yes or No) No, it is not a binary tree.

Figure 19: The unique prompt for the Tree Recognition task.

This is a undirected graph with the following edges:
From node 0 to node 1, distance is 2
From node 0 to node 2, distance is 1
From node 0 to node 3, distance is 1
From node 0 to node 4, distance is 4
From node 0 to node 5, distance is 2
From node 0 to node 6, distance is 3
From node 0 to node 7, distance is 3
From node 1 to node 2, distance is 1
From node 1 to node 3, distance is 2
From node 1 to node 4, distance is 3
From node 1 to node 5, distance is 5
From node 1 to node 6, distance is 5
From node 1 to node 7, distance is 2
From node 2 to node 3, distance is 1
From node 2 to node 4, distance is 5
From node 2 to node 5, distance is 1
From node 2 to node 6, distance is 3
From node 2 to node 7, distance is 3
From node 3 to node 4, distance is 4
From node 3 to node 5, distance is 1
From node 3 to node 6, distance is 4
From node 3 to node 7, distance is 5
From node 4 to node 5, distance is 1
From node 4 to node 6, distance is 4
From node 4 to node 7, distance is 3
From node 5 to node 6, distance is 2
From node 5 to node 7, distance is 3
From node 6 to node 7, distance is 5
Q: The goal is to find the shortest possible route that visits each node exactly once and returns to the starting node.Please determine the optimal solution for this Traveling Salesman Problem (TSP).You can use Nearest Neighbor Algorithm solve this problem. Provide the sequence of nodes that form this shortest route and the total distance of this route.Start from node 0.Please provide the answer directly without the reasoning process and output it in JSON format. For example: {"length":LENGTH, "path": PATH}
A:The TSP path is 17 and path is [0, 2, 1, 3, 5, 4, 7, 6, 0].

Figure 20: The unique prompt for the TSP task.

Given a heterogeneous graph about internet movie, there are three types of nodes, namely: ['movie', 'director', 'actor']. The relationships (meta paths) between different nodes include: [('movie', 'to', 'director'), ('movie', 'to', 'actor'), ('director', 'to', 'movie'), ('actor', 'to', 'movie')]. The edges of these relationships are: {\"('movie', 'to', 'director')\": [(1, 18), (2, 18), (3, 18), (4, 18), (0, 18), (5, 18), (6, 18), (7, 18), (8, 18), (9, 18)], \"('movie', 'to', 'actor')\": [(10, 19), (0, 19), (11, 19), (12, 20), (0, 20), (0, 21), (13, 21), (14, 21), (15, 21), (16, 21), (17, 21)], \"('director', 'to', 'movie')\": [(18, 0)], \"('actor', 'to', 'movie')\": [(19, 0), (20, 0), (21, 0)]}.By performing random sampling of 2-hop 10 neighbors centered on the target movie node, a heterogeneous subgraph is obtained. In the subgraph, \"movie\" nodes: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17],and the type of these movies are: {'action': [2, 13], 'comedy': [1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17], 'drama': []},
where the 0-th node is the central node that represents a moive with the following information:
Name: The Three Stooges
Director's name: Bobby Farrelly \nActors' names: Kate Upton|Larry David|Sean Hayes
Plot keywords: mule|nun|orphanage|tennis court|the three stooges
"actor" nodes:[19, 20, 21];"director\" nodes:[18]
Question: How many relationships classes in this graph?
A:4

Figure 21: The unique prompt for the Relation Number task.

Given a heterogeneous graph about internet movie, there are three types of nodes, namely: ['movie', 'director', 'actor']. The relationships (meta paths) between different nodes include: [('movie', 'to', 'director'), ('movie', 'to', 'actor'), ('director', 'to', 'movie'), ('actor', 'to', 'movie')]. The edges of these relationships are: {\"('movie', 'to', 'director')\": [(1, 15), (2, 15), (0, 15), (3, 15), (4, 15), (5, 15), (6, 15)], \"('movie', 'to', 'actor')\": [(0, 16), (0, 17), (7, 18), (8, 18), (0, 18), (9, 18), (10, 18), (11, 18), (1, 18), (12, 18), (13, 18), (14, 18)], \"('director', 'to', 'movie')\": [(15, 0)], \"('actor', 'to', 'movie')\": [(16, 0), (17, 0), (18, 0)]}.By performing random sampling of 2-hop 10 neighbors centered on the target movie node, a heterogeneous subgraph is obtained. In the subgraph, \"movie\" nodes: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],and the type of these movies are: {'action': [5], 'comedy': [1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14], 'drama': []},
where the 0-th node is the central node that represents a moive with the following information:Name: The Longest Yard
Director's name: Peter Segal
Actors' names: Adam Sandler|Steve Reevis|Dalip Singh \nPlot keywords: coach|convict|football|prison|warden
"actor\" nodes:[16, 17, 18];\"director\" nodes:[15]
Question: Did actor 17 act in movie 5? Please provide the answer directly without the reasoning process.
A: No

Figure 22: The unique prompt for the Neighborhood Query task.

Given a heterogeneous graph about internet movie, there are three types of nodes, namely: ['movie', 'director', 'actor']. The relationships (meta paths) between different nodes include: [('movie', 'to', 'director'), ('movie', 'to', 'actor'), ('director', 'to', 'movie'), ('actor', 'to', 'movie')]. The edges of these relationships are: {\"('movie', 'to', 'director')\": [(1, 15), (2, 15), (0, 15), (3, 15), (4, 15), (5, 15), (6, 15)], \"('movie', 'to', 'actor')\": [(0, 16), (0, 17), (7, 18), (8, 18), (0, 18), (9, 18), (10, 18), (11, 18), (1, 18), (12, 18), (13, 18), (14, 18)], \"('director', 'to', 'movie')\": [(15, 0)], \"('actor', 'to', 'movie')\": [(16, 0), (17, 0), (18, 0)]}.By performing random sampling of 2-hop 10 neighbors centered on the target movie node, a heterogeneous subgraph is obtained. In the subgraph, \"movie\" nodes: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],and the type of these movies are: {'action': [5], 'comedy': [1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14], 'drama': []},
where the 0-th node is the central node that represents a moive with the following information:Name: The Longest Yard
Director's name: Peter Segal
Actors' names: Adam Sandler|Steve Reevis|Dalip Singh \nPlot keywords: coach|convict|football|prison|warden
"actor\" nodes:[16, 17, 18];\"director\" nodes:[15]
Question: Please tell me which actors participated in this movie(0-th node).Please provide the answer directly without the reasoning process and return a node list.
A:[16, 17, 18]

Figure 23: The unique prompt for the Relationship Query task.

Given a heterogeneous graph about internet movie, there are three types of nodes, namely: ['movie', 'director', 'actor']. The relationships (meta paths) between different nodes include: [('movie', 'to', 'director'), ('movie', 'to', 'actor'), ('director', 'to', 'movie'), ('actor', 'to', 'movie')]. The edges of these relationships are: {\"('movie', 'to', 'director')\": [(1, 15), (2, 15), (0, 15), (3, 15), (4, 15), (5, 15), (6, 15)], \"('movie', 'to', 'actor')\": [(0, 16), (0, 17), (7, 18), (8, 18), (0, 18), (9, 18), (10, 18), (11, 18), (1, 18), (12, 18), (13, 18), (14, 18)], \"('director', 'to', 'movie')\": [(15, 0)], \"('actor', 'to', 'movie')\": [(16, 0), (17, 0), (18, 0)]}.By performing random sampling of 2-hop 10 neighbors centered on the target movie node, a heterogeneous subgraph is obtained. In the subgraph, \"movie\" nodes: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],and the type of these movies are: {'action': [5], 'comedy': [1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14], 'drama': []},
where the 0-th node is the central node that represents a moive with the following information:Name: The Longest Yard
Director's name: Peter Segal
Actors' names: Adam Sandler|Steve Reevis|Dalip Singh \nPlot keywords: coach|convict|football|prison|warden
"actor\" nodes:[16, 17, 18];\"director\" nodes:[15]
Question: Please extract the edges connected to this movie(0-th node).Please provide the answer directly without the reasoning process and return a edge list. For example: \\[(1,2),(4,5)]
A: [(0, 15), (0, 16), (0, 17), (0, 18),(15,0),(16,0),(17,0),(18,0)]

Figure 24: The unique prompt for the Subgraph Extraction task.

Given a heterogeneous academic network graph about computer science collected from Association for Computing Machinery (ACM), there are four types of nodes, namely: ['paper', 'subject', 'term', 'author']. The relationships (meta paths) between different nodes include: [('paper', 'cite', 'paper'), ('paper', 'ref', 'paper'), ('paper', 'to', 'author'), ('author', 'to', 'paper'), ('paper', 'to', 'subject'), ('subject', 'to', 'paper'), ('paper', 'to', 'term'), ('term', 'to', 'paper')]. The edges of these relationships are: {\"('paper', 'cite', 'paper')\": [], \"('paper', 'ref', 'paper')\": [], \"('paper', 'to', 'author')\": [(0, 106)], \"('author', 'to', 'paper')\": [(106, 0)], \"('paper', 'to', 'subject')\": [(0, 95)], \"('subject', 'to', 'paper')\": [(95, 0)], \"('paper', 'to', 'term')\": [(1, 96), (2, 96), (3, 96), (4, 96), (5, 96), (6, 96), (7, 96), (8, 96), (9, 96), (0, 96), (10, 97), (11, 97), (12, 97), (13, 97), (14, 97), (15, 97), (16, 97), (17, 97), (18, 97), (0, 97), (19, 98), (20, 98), (21, 98), (22, 98), (23, 98), (24, 98), (25, 98), (26, 98), (27, 98), (28, 98), (29, 99), (30, 99), (31, 99), (32, 99), (33, 99), (34, 99), (35, 99), (36, 99), (37, 99), (38, 99), (39, 100), (40, 100), (41, 100), (42, 100), (43, 100), (0, 100), (44, 100), (45, 100), (46, 100), (47, 100), (48, 101), (49, 101), (50, 101), (51, 101), (52, 101), (53, 101), (54, 101), (55, 101), (56, 101), (57, 101), (58, 102), (59, 102), (60, 102), (61, 102), (62, 102), (63, 102), (64, 102), (65, 102), (66, 102), (67, 102), (68, 103), (69, 103), (70, 103), (71, 103), (72, 103), (73, 103), (74, 103), (75, 103), (41, 103), (76, 103), (77, 104), (78, 104), (79, 104), (80, 104), (81, 104), (82, 104), (83, 104), (84, 104), (85, 104), (55, 104), (86, 105), (87, 105), (88, 105), (89, 105), (90, 105), (91, 105), (2, 105), (92, 105), (93, 105), (94, 105)], \"('term', 'to', 'paper')\": [(96, 0), (97, 0), (98, 0), (99, 0), (100, 0), (101, 0), (102, 0), (103, 0), (104, 0), (105, 0)]}.By performing random sampling of 2-hop 10 neighbors centered on the target paper node, a heterogeneous subgraph is obtained. In the subgraph, \"paper\" nodes: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94],and the type of these papers are: {'Database': [1, 2, 5, 11, 12, 13, 16, 19, 28, 29, 30, 31, 32, 34, 35, 37, 38, 39, 41, 42, 46, 49, 51, 53, 59, 61, 62, 68, 71, 75, 78, 80, 83, 89], 'Wireless Communication': [3, 4, 6, 15, 27, 36, 55, 57, 60, 63, 64, 67, 70, 72, 77, 79, 81, 82, 84, 86, 91, 92, 94], 'Data Mining': [7, 8, 9, 10, 14, 17, 18, 20, 21, 22, 23, 24, 25, 26, 33, 40, 43, 44, 45, 47, 48, 50, 52, 54, 56, 58, 65, 66, 69, 73, 74, 76, 85, 87, 88, 90, 93]}, where the 0-th node is the central node that represents a paper with the following information:\nTitle and Abstract: Foundation matters This talk is meant as a wake-up call ... The foundation of the database field is, of course, the relational model. Sad to say, however, there are some in the database community--certainly in industry, and to some extent in academia also--who do not seem to be as familiar with that model as they ought to be; there are others who seem to think it is not very interesting or relevant to the day-today business of earning a living; and there are still others who seem to think all of the foundation-level problems have been solved. Indeed, there seems to be a widespread feeling that \"the world has moved on,\" so to speak, and the relational model as such is somehow passé. In my opinion, nothing could be further from the truth! In this talk, I want to sketch the results of some of my own investigations into database foundations over the past twenty years or so; my aim is to convey some of the excitement and abiding interest that is still to be found in those investigations, with a view--I hope--to inspiring others in the field to become involved in such activities.   First of all, almost all of the ideas I will be covering either are part of, or else build on top of, The Third Manifesto [1]. The Third Manifesto is a detailed proposal for the future direction of data and DBMSs. Like Codds original papers on the relational model, it can be seen as an abstract blueprint for the design of a DBMS and the language interface to such a DBMS. Among many other things: • It shows that the relational model--and I do mean the relational model, not SQL--is a necessary and sufficient foundation on which to build \"object/relational\" DBMSs (sometimes called universal servers). • It also points out certain blunders that can unfortunately be observed in some of todays products (not to mention the SQL:1999 standard). • And it explores in depth the idea that a relational database, along with the relational operators, is really a logical system and shows how that idea leads to a solution to the view updating problem, among other things.\n\"subject\" nodes:[95];\"term\" nodes:[96, 97, 98, 99, 100, 101, 102, 103, 104, 105];\"author\" nodes:[106]\

Question: Which of the following areas of computer science does this paper belong to: Database, Wireless Communication, or Data Mining?\n Please provide the answer directly without the reasoning process.

A: Database

Figure 25: The unique prompt for the Node Classification task.

Given a heterogeneous academic network graph about computer science collected from Association for Computing Machinery (ACM), there are four types of nodes, namely: ['paper', 'subject', 'term', 'author']. The relationships (meta paths) between different nodes include: [('paper', 'cite', 'paper'), ('paper', 'ref', 'paper'), ('paper', 'to', 'author'), ('author', 'to', 'paper'), ('paper', 'to', 'subject'), ('subject', 'to', 'paper'), ('paper', 'to', 'term'), ('term', 'to', 'paper')]. The edges of these relationships are: {\"('paper', 'cite', 'paper')\": [], \"('paper', 'ref', 'paper')\": [], \"('paper', 'to', 'author')\": [(0, 106)], \"('author', 'to', 'paper')\": [(106, 0)], \"('paper', 'to', 'subject')\": [(0, 95)], \"('subject', 'to', 'paper')\": [(95, 0)], \"('paper', 'to', 'term')\": [(1, 96), (2, 96), (3, 96), (4, 96), (5, 96), (6, 96), (7, 96), (8, 96), (9, 96), (0, 96), (10, 97), (11, 97), (12, 97), (13, 97), (14, 97), (15, 97), (16, 97), (17, 97), (18, 97), (0, 97), (19, 98), (20, 98), (21, 98), (22, 98), (23, 98), (24, 98), (25, 98), (26, 98), (27, 98), (28, 98), (29, 99), (30, 99), (31, 99), (32, 99), (33, 99), (34, 99), (35, 99), (36, 99), (37, 99), (38, 99), (39, 100), (40, 100), (41, 100), (42, 100), (43, 100), (0, 100), (44, 100), (45, 100), (46, 100), (47, 100), (48, 101), (49, 101), (50, 101), (51, 101), (52, 101), (53, 101), (54, 101), (55, 101), (56, 101), (57, 101), (58, 102), (59, 102), (60, 102), (61, 102), (62, 102), (63, 102), (64, 102), (65, 102), (66, 102), (67, 102), (68, 103), (69, 103), (70, 103), (71, 103), (72, 103), (73, 103), (74, 103), (75, 103), (41, 103), (76, 103), (77, 104), (78, 104), (79, 104), (80, 104), (81, 104), (82, 104), (83, 104), (84, 104), (85, 104), (55, 104), (86, 105), (87, 105), (88, 105), (89, 105), (90, 105), (91, 105), (2, 105), (92, 105), (93, 105), (94, 105)], \"('term', 'to', 'paper')\": [(96, 0), (97, 0), (98, 0), (99, 0), (100, 0), (101, 0), (102, 0), (103, 0), (104, 0), (105, 0)]}.By performing random sampling of 2-hop 10 neighbors centered on the target paper node, a heterogeneous subgraph is obtained. In the subgraph, \"paper\" nodes: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94],and the type of these papers are: {'Database': [1, 2, 5, 11, 12, 13, 16, 19, 28, 29, 30, 31, 32, 34, 35, 37, 38, 39, 41, 42, 46, 49, 51, 53, 59, 61, 62, 68, 71, 75, 78, 80, 83, 89], 'Wireless Communication': [3, 4, 6, 15, 27, 36, 55, 57, 60, 63, 64, 67, 70, 72, 77, 79, 81, 82, 84, 86, 91, 92, 94], 'Data Mining': [7, 8, 9, 10, 14, 17, 18, 20, 21, 22, 23, 24, 25, 26, 33, 40, 43, 44, 45, 47, 48, 50, 52, 54, 56, 58, 65, 66, 69, 73, 74, 76, 85, 87, 88, 90, 93]}, where the 0-th node is the central node that represents a paper with the following information:\nTitle and Abstract: Foundation matters This talk is meant as a wake-up call ... The foundation of the database field is, of course, the relational model. Sad to say, however, there are some in the database community--certainly in industry, and to some extent in academia also--who do not seem to be as familiar with that model as they ought to be; there are others who seem to think it is not very interesting or relevant to the day-today business of earning a living; and there are still others who seem to think all of the foundation-level problems have been solved. Indeed, there seems to be a widespread feeling that \"the world has moved on,\" so to speak, and the relational model as such is somehow passé. In my opinion, nothing could be further from the truth! In this talk, I want to sketch the results of some of my own investigations into database foundations over the past twenty years or so; my aim is to convey some of the excitement and abiding interest that is still to be found in those investigations, with a view--I hope--to inspiring others in the field to become involved in such activities.   First of all, almost all of the ideas I will be covering either are part of, or else build on top of, The Third Manifesto [1]. The Third Manifesto is a detailed proposal for the future direction of data and DBMSs. Like Codds original papers on the relational model, it can be seen as an abstract blueprint for the design of a DBMS and the language interface to such a DBMS. Among many other things: • It shows that the relational model--and I do mean the relational model, not SQL--is a necessary and sufficient foundation on which to build \"object/relational\" DBMSs (sometimes called universal servers). • It also points out certain blunders that can unfortunately be observed in some of todays products (not to mention the SQL:1999 standard). • And it explores in depth the idea that a relational database, along with the relational operators, is really a logical system and shows how that idea leads to a solution to the view updating problem, among other things.\n\n\"subject\" nodes:[95];\"term\" nodes:[96, 97, 98, 99, 100, 101, 102, 103, 104, 105];\"author\" nodes:[106]\
Question: Please predict if Paper 72 cited Paper 25.Answer Yes or No. Please provide the answer directly without the reasoning process.
A: No

Figure 26: The unique prompt for the Link Prediction task.