Formulating and Unveiling In-Context Learning over Graphs

Anonymous ACL submission

Abstract

In-context learning (ICL) is a fascinating capability of large language models (LLMs), which can adapt to queries through demonstrations without optimizing model parameters. Although LLMs have demonstrated the ability of ICL in graph tasks, the graph in-context learning (GraphICL) mechanism is still a black box. In this paper, we introduce a novel framework for understanding and analyzing in-context learning over graphs, focusing on graph tasks, with thorough formulations, innovative mechanisms, and comprehensive benchmarks.We are the first to systematically and rigorously formalize GraphICL by explicitly defining task categories, the number of demonstrations, and graph structures in graph reasoning tasks. We reveal the mechanism of GraphICL, where the LLMs generate more accurate answers by weighting and aggregating the query representations and demonstrations representations. However, existing benchmarks lack datas with the same graph structure, which is crucial for analyzing the impact of graph structure on the GraphICL ability. We introduce two new datasets, comprising a total of 17,155 graph questions across graphs of varying sizes and multiple task categories. With these datasets, our experiments comprehensively explore for the first time how to activate GraphICL's capabilities from the perspectives of the number of demonstrations, graph structures, task categories, etc., and verify our proposed formulation and mechanism. The benchmarks and codes are available at: https://github.com/Graph-ICL/GraphICL.

1 Introduction

The rapid advancement of large language models (LLMs) such as GPT-4 (Achiam et al., 2023), Gemini (Team et al., 2024) and DeepSeek (Lu et al., 2024) have ushered in a new era of artificial intelligence, showcasing unprecedented capabilities in understanding (Nam et al., 2024), generating (Si et al., 2024), and reasoning (Hao et al., 2024) with human-like proficiency across a wide range of tasks. Graphs, with their non-Euclidean nature (Wu et al., 2021), present a particularly challenging yet promising frontier for LLMs exploration (Jin et al., 2024). GraphInstruct (Luo et al., 2024), LLM4DyG (Zhang et al., 2023) explore the graph reasoning capabilities of LLMs, finding that although LLMs have certain capabilities, they perform poorly on complex graph tasks.

Recently, one of the most exciting features emerging in LLMs is ICL (Brown et al., 2020; Nguyen et al., 2023; Wies et al., 2023), which allows LLMs to perform new tasks with several demonstrations and no additional training. This capability has sparked interest in using GraphICL to improve answer accuracies in graph tasks. NLGraph (Wang et al., 2023a) tests the GraphICL's ability of LLMs and finds that adding demonstrations does not improve the performance of LLMs in complex graph tasks (e.g., Hamiltonian paths). GPT4Graph (Guo et al., 2023) finds that in some cases, demonstrations introduce noise, bias, or incomplete information that hinders the LLMs' overall understanding. Graphwiz (Chen et al., 2024a) tests GraphICL in GPT-4 and finds that in nine types of graph tasks, the accuracy of two-demonstration is improved compared to zero-demonstration. The above studies show that the GraphICL ability of LLMs is affected by factors such as the number and quality of demonstrations and the difficulty of the problem, which also show that LLMs are very brittle with demonstrations. Although current research studies GraphICL through simple experiments, the mechanisms of GraphICL is still in a "black box" and lack theoretical explanation and sufficient verification.

In graph tasks, the complexity of node and edge information, as well as the intricacy of the graph problems themselves, significantly increases the difficulty of revealing the mechanisms of Graph-

ICL. Many studies provide explanations from different perspectives, such as Bayesian models (Panwar et al., 2024), gradient descent (Li et al., 2023a), and attention mechanisms (Dai et al., 2023), but they do not explicitly focus on graph structures. Although some works have demonstrated that LLMs can improve the accuracy of graph tasks with some demonstrations, the demonstrations in current studies are either randomly selected or fixed (Zhao et al., 2021; Lu et al., 2022). This demonstration selection method does not allow for a deeper study of the impact of the demonstrations themselves on the GraphICL ability. At the same time, current research has not systematically analyzed how graph demonstrations affect the graph reasoning ability of LLMs from a **"graph perspective"** . For example, factors such as the graph structure, task category, and graph size in the demonstrations are all necessary to study in GraphICL. However, there is currently no suitable benchmark in the graph field to fully solve the above problems, which is a major gap in current research.

In this paper, we formally define GraphICL, constructing the query and several demonstrations as a unified in-context. We explicitly represent the categories of demonstrations , number of demonstrations, the structure of graphs , etc., providing a standardized expression for GraphICL. To further reveal the intrinsic mechanism of GraphICL, we interact tthe query representations, graph question representations, and answer representations in a demonstration set, explicitly giving the intrinsic mechanism of GraphICL. In order to discuss in detail the impact of factors such as the number of demonstrations, categories, and graph structures, we construct new datasets GraphSCB and GraphTRB. GraphSCB contains three different types of graph question answering tasks, through which we can study the impact of the similarity between the graph structure in the demonstration and the graph structure in the query on the accuracy of the question. GraphTRB contains seven types of graph reasoning tasks, through which we can study the correlation between the task category in the demonstration and the category in the query, and the impact on the accuracy of the question. Our main contributions are as follows:

1. We define GraphICL for the first time, establishing a unified framework that explicitly addresses the category, number, and graph structure in demonstrations.

2. We reveal the mechanism of GraphICL, showing how the LLMs generate answers by weighting and aggregating the query representations and demonstrations representations.

3. We release two new open-source datasets, GraphSCB and GraphTRB, which provide comprehensive benchmarks for analyzing how to activate GraphICL capabilities of LLMs.

2 Related Work

2.1 Large Language Models for Graph Tasks

Recent studies explore the powerful generalization capabilities of LLMs for graph understanding. Researchers have conducted empirical performance evaluations on projects such as NLGraph (Wang et al., 2023a), GPT4Graph (Guo et al., 2023), TAPE (He et al., 2024), GraphWiz (Chen et al., 2024a), GraphTMI (Das et al., 2023) and LLM4DyG (Zhang et al., 2023), each exploring whether LLMs can understand graph-structured data. (Fatemi et al., 2023) systematically studies the impact of different graph description languages on LLMs' understanding of graph data. GraphTMI (Das et al., 2023) and (Chen et al., 2024b) explore the potential of LLMs on graph node classification tasks. The work closest to ours is done by NLgraph (Wang et al., 2023a) and GraphWiz (Chen et al., 2024a).

2.2 Demonstrations Selection for ICL

LLMs enable ICL techniques to solve different tasks with only several demonstrations. However, studies have shown that the choice of demonstrations significantly impacts performance. A promising approach to enhance ICL is demonstration selection, where the most relevant demonstrations are retrieved through a retrieval-based paradigm. (Liu et al., 2021; Li et al., 2023b) model this process using off-the-shelf retrievers that leverage sentence encoders to identify semantically similar demonstrations. The multilingual ICL can benefit from cross-lingual k-NN retrieval to improve source-target language alignment (Tanwar et al., 2023). However, the heuristic nature of these off-the-shelf retrievers and the lack of task-specific supervision make them suboptimal. To address this limitation, supervised methods have been proposed (Rubin et al., 2022; Wang et al., 2023b; Zhang et al., 2022).

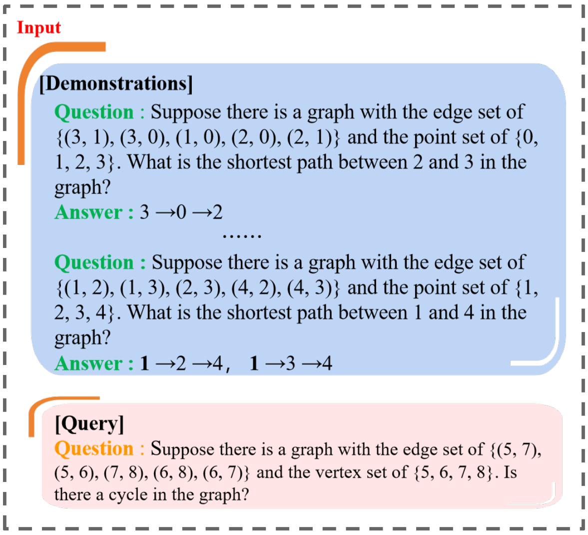

Figure 1: An example of using the ICL ability of LLMs to solve the graph reasoning problem.

3 Methodology

In this section, we delve into the theoretical and practical aspects of GraphICL. It formalizes a unified framework for integrating demonstrations and query input using pre-trained language models, providing a structured foundation for graph tasks. A linear update mechanism was proposed to reveal the basic principles of GraphICL. Furthermore, assumptions are introduced to analyze how factors such as graph structure similarity, demonstration quantity, and category interaction influence the performance of LLMs in graph reasoning tasks.

3.1 Formulations of ICL over Graphs

In this paper, we focus on ICL for graph reasoning tasks using various LLMs. We assume a pre-trained language model M, which stacks L layers of the same Transformer structure, each layer consisting of an attention module and a feed-forward network. For a graph reasoning task, given a query $Q(G)$, where the graph $G = (V, E)$, including a point set V and an edge set E, we need to generate the answer A to the query based on n demonstrations $C := \left\{\left(Q_i^k(G_j), A_{ijk}\right)\right\}_{i,j,k}$. Where $Q_i^k(G_j)$ represents the questions with graph information in demonstrations, A_{ijk} refers the answers in demonstrations, i refers to the number of ICL demonstrations, j indexes the different graphs and k represents the category of graph tasks, such as Shortest path, Maximum Flow.

Formally, given a pre-trained language model M, we input demonstrations C and query $Q(G)$. The conditional probability for generating the answer A can be written as $P_M(A \mid C, Q(G))$. Specifically, the zero-demonstration prediction is denoted as $P_M(A \mid Q(G))$.

Our formulation $Q_i^k(G_j)$ introduces three dimensions i, j, and k, providing great flexibility. For example, for a query $Q(G)$, we can use multiple different graph structures and graph task categories in the demonstration set, and the number of examples i can also vary. This flexibility enables the dynamic configuration of demonstrations based on the specific requirements of graph reasoning tasks. Such an approach facilitates the adaptation of the GraphICL framework to a wide range of tasks, allowing diverse graph tasks to complement and interact effectively. The graphic representation of the GraphICL example can be seen in Figure 1. In practice, we usually format demonstrations using predefined templates and splice them into the context before the query question.

3.2 Mechanisms of ICL over Graphs

Given a query $Q(G)$, the query description text is represented by their embedding $\phi(Q(G)) \in \mathbb{R}^{d_{in}}$, where $\phi(\cdot)$ represents a text encoder. The initial prediction, denoted as $\mathcal{F}_{\text{init}}$, is obtained through a linear transformation using a pre-trained parameter matrix $\mathrm{W}_0 \in \mathbb{R}^{d_{out} \times d_{in}}$, i.e. $\mathcal{F}_{\text{init}}(G, Q) := \mathrm{W}_0\phi(Q(G))$. Here, W_0 captures the model's prior knowledge without considering any task-specific demonstrations. This transformation forms the zero-demonstration baseline for the prediction.

To adapt the predictions to task-specific requirements, demonstrations are introduced. Given the set of demonstrations $C = \{(Q_i^k(G_j), A_{ijk})\}_{i,j,k}$, where $Q_i^k(G_j)$ represents the i-th demonstration's graph problem and A_{ijk} denotes the answer to the i-th demonstration, let its representation be $\phi(Q_i^k(G_j)) \in \mathbb{R}^{d_{in}}$, with A_{ijk}'s corresponding embedding $\phi(A_{ijk}) \in \mathbb{R}^{d_{out}}$.

The model integrates the in-context information through an update matrix $\Delta\mathrm{W} \in \mathbb{R}^{d_{out} \times d_{in}}$, defined as: $\Delta\mathrm{W} := \sum_{ijk} \phi(A_{ijk}) \otimes \phi(Q_i^k(G_j))$, where $\otimes$ denotes the outer product. This update encodes the interactions between the answer representations $\phi(A_{ijk})$ and the representations of graph problems $\phi(Q_i^k(G_j))$ of the demonstrations.

The updated transformation for the query representation $\phi(Q(G))$ incorporates both the pre-trained parameter matrix W_0 and the in-context update ΔW:

$$\begin{aligned}
&\mathcal{F}_C(G,Q)\\
=&(\mathrm{W}_0+\Delta\mathrm{W})\phi\left(Q(G)\right)\\
=&\mathrm{W}_0\phi\left(Q(G)\right)+\Delta\mathrm{W}\phi(Q(G))\\
=&\mathrm{W}_0\phi\left(Q(G)\right)+\sum_{ijk}\left(\phi(A_{ijk})\otimes\phi\left(Q_i^k(G_j)\right)\right)\phi\left(Q(G)\right)\\
=&\mathrm{W}_0\phi\left(Q(G)\right)+\sum_{ijk}\phi(A_{ijk})\left(\phi(Q_i^k(G_j))^T\phi\left(Q(G)\right)\right)\\
=&\mathcal{F}_{\text{init}}(G,Q)+\sum_{ijk}\phi(A_{ijk})\left(\phi(Q_i^k(G_j))^T\phi\left(Q(G)\right)\right).
\end{aligned}$$

Here the $\mathcal{F}_{\text{init}}(G, Q)$ represents the model's prediction based solely on pre-trained knowledge, while $\Delta\mathrm{W} \cdot \phi(Q(G))$ captures the task-specific adjustments contributed by the demonstrations.

Remarkably, we explicitly demonstrate the internal mechanism of LLMs in GraphICL, where the output answer embedding $\mathcal{F}_C(G, Q)$ is determined jointly by $\phi(Q_i^k(G_j))^T\phi\left(Q(G)\right)$ and $\phi(A_{ijk})$, where $\phi(Q_i^k(G_j))^T\phi\left(Q(G)\right)$ measures the relevance between graph problem representation in query $\phi(Q(G))$ and the the graph representation in demonstration $Q_i^k(G_j)$.

Besides, we can see that the answer to the query is closely related to the graph information in the query, the graph information in the demonstration G, the task categories k, and the answer A, etc. Therefore, to better study how graph demonstrations activate the GraphICL capabilities of LLMs, we can make the following assumptions.

Assumption 1. The correctness of answers is influenced by the number of demonstrations .

For the k-th graph task category, let $C^k := \{(Q_i^k(G_j), A_{i,j,k})\}_{i,j} \subset C$ denote the subset of demonstrations C related to the k-th graph task category. Given the query $Q^m(G)$ corresponding to the m-th graph task category, we assume that the model's prediction in answering questions of the m-th task category, $P_M(A \mid C^k, Q^m(G))$, is influenced by the corresponding number of demonstrations $I^k := \max\{i \mid Q_i^k(G_j) \in C^k\}$.

Assumption 2. The correctness of answers is influenced by the structural similarity of graphs between the query and demonstrations.

When the structural similarity between the j-th graph G_j of the demonstrations and the graph G of the query varies, the accuracy of the generated answers follows a pattern defined by the relevance between the query representation $\phi(Q(G))$ and the graph representation in the demonstration $\phi(Q_i^k(G_j))$. Specifically, the prediction accuracy $P_M(A \mid C, Q(G))$ is influenced by the inner product $\phi(Q_i^k(G_j))^T\phi(Q(G))$, which reflects the similarity between graph structures in the demonstrations and graph structure in the query :

$$P_M(A \mid C, Q(G)) \propto \phi(Q_i^k(G_j))^T\phi(Q(G)).$$

We assume that the textual descriptions of the Query and Demonstrations, which include graph structure information, follow the same format, and we only focus on the variations in the graph structure information. When the graph structure of the demonstration is more similar to the graph structure of the query, the accuracy of the generated answers is higher. Particularly, a completely identical graph can significantly improve the reliability of the generated answers.

Assumption 3. The correctness of the answers is influenced by the interaction of task categories between demonstrations and the query.

Let $Q^k(G)$ represent questions of the k-th category, and let $\mathcal{D} := C^{k_1} \cup \cdots \cup C^{k_m}$ denote the subset of demonstrations C corresponding to m different graph task categories. We assume that the correctness of the answer A is influenced by the interaction between the problem category k and the query categories $k_1, \ldots, k_m$. The prediction $P_M(A \mid \mathcal{D}, Q^k(G))$ is governed by this interaction; when the interaction between categories $k_1, \ldots, k_m$ and k is beneficial, the accuracy of the generated answers improves. However, certain categories of demonstrations may not provide complementary information to the query, and may even introduce interference information, which can negatively impact the accuracy of the response.

4 Benchmarks

4.1 Limitations of Existing Benchmarks

In this section, we highlight the limitations of existing benchmarks in evaluating GraphICL, such as inconsistent graph structures in QA pairs and lack of detailed classification. To address these issues, we introduce two new benchmarks, Graph Structural Consistency Benchmark (GraphSCB) and Graph Task-Related Benchmark (GraphTRB), which enable more systematic analyze how graph structure and task category affect activate the ability of GraphICL in LLMs.

Inconsistent Graph Structures in QA Pairs. The graph structures in the question-answering

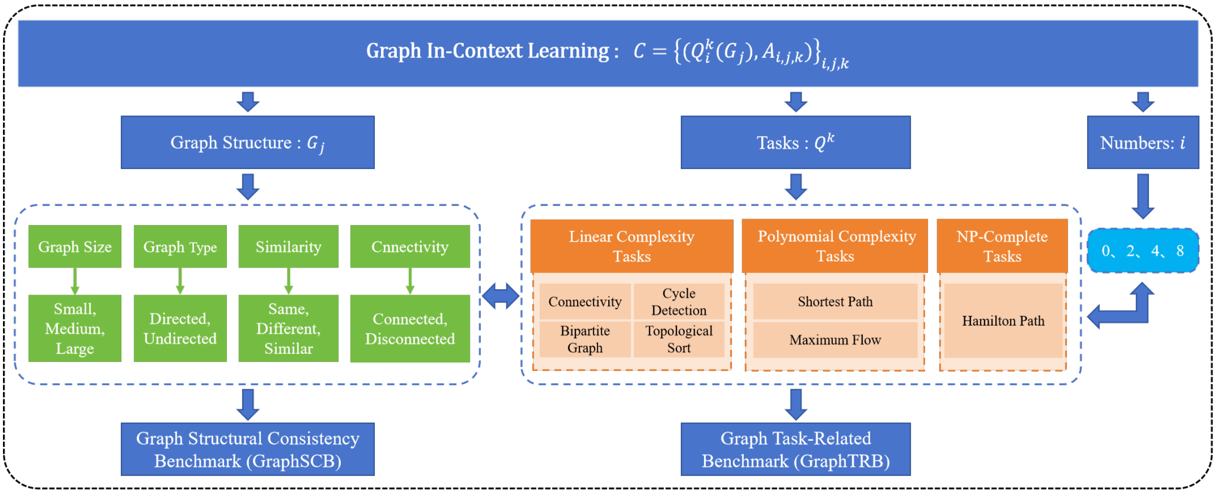

Figure 2: An overview of the GraphICL Benchmark. The two newly proposed benchmarks for evaluating GraphICL, designed to assess how graph structure and task category activate GraphICL. The GraphSCB examines how the similarity between the graph structures in demonstrations and the graph structures in query influences accuracy, while the GraphTRB explores how task categories in the demonstrations impact performance. Together, these benchmarks provide a more systematic approach to understanding the role of graph structure and task categories in activating ICL for graph-based tasks.

(QA) pairs of previous benchmarks were entirely different, preventing in-depth exploration of the impact of the graph structure in demonstrations on question accuracy. The absence of consistent graph structures hindered the study of how the similarity or difference between the graph structures in the demonstration and the query affected model performance.

Lack of Multiple Examples for the Same Graph. In previous benchmarks, each question in the QA pairs corresponded to a different graph. When exploring the interaction between different task categories, it was necessary to keep the graph the same to ensure the uniqueness of the independent variable. However, previous benchmarks lack such demonstrations, i.e., there is only one demonstration of the same category of questions, which makes it impossible to add multiple demonstrations of the same graph. This limitation prevented systematic analyze of the interaction between question categories while controlling for graph structure.

Insufficient Detailed Classification of Graph Structures and Question Categories. Previous datasets lacked detailed classification of graph structures and question categories. For example, important graph characteristics such as Connectivity and size were not considered in the classification. This makes it difficult to fully explore the impact of different graph structures and task categories on the ability to activate GraphICL.

To address these limitations and provide a more comprehensive evaluation of GraphICL, we introduce two new benchmarks: the Graph Structural Consistency Benchmark (GraphSCB) and the Graph Task-Related Benchmark (GraphTRB). These benchmarks are designed based on the definitions and concepts presented in Figure 2, enabling systematic analyze how graph structure, task category, and number of demonstrations activate GraphICL performance, thus filling the gap left by existing benchmarks.

4.2 Descriptions of the New Benchmarks

GraphSCB is a newly constructed dataset derived from GraphInstruct (Chen et al., 2024a), comprising three distinct types of graph computation tasks: Shortest Path, Maximum Flow, and Connectivity. Each task type is extracted from GraphInstruct, using data corresponding to the same task type but with varying graph sizes. The tasks is defined by categorizing graphs based on their node count: graphs with 10 to 35 nodes are considered small, those with 36 to 65 nodes are categorized as medium, and graphs with 66 to 100 nodes are classified as large. Additionally, for each data point, 16 distinct data instances are generated, each maintaining the same task type and graph structure. Consequently, the dataset contains a total of 10,891 data instances. This is the first new benchmark that can explore the impact of the graph structure in the demonstrations being the same as or different from the graph structure in the queyr on accuracy.

GraphTRB is a new dataset converted from GraphInstruct, which contains 7 types of graph computing tasks, namely Connectivity, Cycle Detection, Bipartite Graph Check, Topological Sorting, Shortest Path, Maximum Flow and Hamilton Path. First, extract data of different graph sizes from GraphInstruct, but the node range of the graph is between 10 to 30. Then generate 6 data with the same graph but different task categories for each data, and generate data with the same graph, the same problem category and different problems for each newly generated data again, for a total of 6264 data. This is the first benchmark that can explore the relationship between the categories of graph tasks in demonstrations and the categories of graph tasks in query. Our newly constructed datasets are shown in the comparison in Table 1. The detailed explanation of the tasks in the dataset is provided in Appendix A.

5 Experiments

In this section, we discuss the ICL ability of graphs by addressing the following questions. **Q1**: How do the ICL capabilities of different LLMs perform on graph reasoning tasks ? **Q2**: What is the impact of graph structure similarity between the demonstrations and the query on the accuracy of answers? **Q3**: Is there an interaction between the task categories of demonstrations and the task category of query? For example, can the Cycle task in the demonstrations promote the accuracy of the Shortest Path task in the query?

5.1 Experimental Settings

Datasets: We use GraphInstruct (Chen et al., 2024a), GraphSCB and GraphTRB as our datasets. GraphInstruct is a large-scale instruction tuning dataset that contains nine categories of graph tasks and a total of 18,125 graph questions, where each pair consists of a graph question description and a corresponding explicit reasoning path or solution.

Models and Settings: We conducted experiments using open-source models such as Qwen2.5-7B, Qwen2.5-14B, and LLaMa3-8B. Our experiments were run on 8 NVIDIA A6000 GPUs with 49GB of memory. To ensure the accuracy of the experimental data, each experiment was run five times, and the average value was taken. For all tasks, we gradually adjust i from 0, 2, 4, and 8 to fully explore the impact of the number of demonstrations on the accuracies of various graph tasks. We set the temperature to 1.0 to allow for diverse and non-deterministic responses. For each query, the model generate a single response. The maximum number of tokens per response is limited to 8192 to allow sufficiently long answers without excessive output. The maximum sequence length was set to 1024 tokens, ensuring that inputs exceeding this length were truncated appropriately. Nucleus sampling with a top-p value of 0.9 is applied. Due to computational resource limitations, we use relatively small LLMs for testing on GraphInstruct as well as our extended datasets, while we encourage future research to leverage the extended version for enhanced evaluation.

Table 1: Comparison between our datasets and other state-of-the-art datasets, demonstrating its superiority and comprehensiveness over the latest 2024 datasets.

Datasets	Tasks	Node Scale	Edge Scale	Numbers	Include Same Graph?
NLGraph	8	9-35	10-30	29,370	No
GraphInstruct	9	2-100	5-500	18,125	No
GraphSCB	**3**	**10-100**	**15-500**	**10,891**	**No**
GraphTRB	**7**	**10-30**	**10-200**	**6,264**	**Yes**

5.2 Results for the Number of Demonstrations (Q1)

Increasing the number of demonstrations can better activate the ICL ability of LLMs, but more is not always better. To evaluate how the number of demonstrations can activate the Graph-ICL of LLMs and promote the development of graph reasoning tasks, we conducted extensive experiments on GraphInstruct dataset. For the query, we selected demonstrations from the dataset that are of the same task category as the query, but with different numbers of demonstrations. The reason for selecting same category graph problems is to eliminate the random errors that may be introduced by random selection (interactions between different categories of tasks or randomly selecting the same graph, etc.). The experimental results are shown in Table 2 and reveal several key findings: As the number of demonstrations increases, all models show improvement in performance, demonstrating the ability of LLMs to learn from in-context. However, for LLaMa3-8B, the accuracy reaches its peak with four demonstrations, and adding more demonstrations causes a decline in performance. This could

Table 2: Test accuracies (%) of different LLMs with different numbers of demonstrations in nine types of graph question answering tasks. We highlight the best results in each column in bold.

Models	ICL-Examples	Tasks									Average
		Cycle	Connectivity	Bipartite	Topology	Shortest	Triangle	Flow	Hamilton	Subgraph	
Qwen2.5-7B	0	98.45	77.11	73.27	60.21	28.30	85.38	37.04	**65.57**	56.31	56.83
	2	99.26	88.28	75.01	81.54	30.75	91.58	39.01	58.08	**59.92**	61.79
	4	99.26	88.50	75.11	83.26	31.54	**93.11**	41.23	59.13	59.17	71.42
	8	**99.30**	**88.80**	**73.57**	**87.96**	**33.48**	92.20	**41.73**	60.04	59.48	**77.69**
Qwen2.5-14B	0	97.72	78.45	67.46	65.83	35.34	80.7	44.44	**54.51**	65.38	65.53
	2	98.05	78.56	**77.00**	77.18	47.99	91.87	64.20	52.03	68.96	72.87
	4	98.12	79.23	75.16	92.09	49.57	94.70	62.96	51.88	69.77	74.83
	8	**98.45**	**80.05**	74.91	**92.43**	**50.50**	**95.83**	**64.69**	52.07	**70.68**	**75.51**
LLaMa3-8B	0	81.19	83.10	54.94	51.15	27.73	26.60	34.07	**69.19**	42.31	52.25
	2	91.65	84.26	53.35	82.11	28.23	67.27	38.52	57.89	53.74	61.89
	4	**93.26**	**85.97**	60.71	**91.28**	**30.03**	67.49	**39.26**	55.08	**56.72**	**64.42**
	8	82.11	85.30	**62.30**	90.48	28.66	**71.52**	37.28	33.33	28.47	57.71

Table 3: Test accuracies (%) on the three-class graph question answering tasks. The best result in each row, corresponding to the same number of demonstrations, is highlighted in bold.

ICL-Examples	Connectivity							
	SameGraph				DifferentGraph			
	Small	Medium	Large	Average	Small	Medium	Large	Average
0	79.24	75.12	72.58	75.65	79.24	75.12	72.58	75.65
2	89.24	75.25	71.43	**78.64**	86.71	75.61	66.96	76.43
4	90.00	76.35	70.52	78.96	89.24	79.17	69.52	**79.31**
ICL-Examples	Maximum Flow							
	SameGraph				DifferentGraph			
	Small	Medium	Large	Average	Small	Medium	Large	Average
0	31.01	21.32	19.92	24.08	31.01	21.32	19.92	24.08
2	38.75	23.90	21.29	**27.98**	37.12	19.49	17.97	24.86
4	46.25	24.26	24.41	**31.64**	39.38	20.22	19.73	26.44
ICL-Examples	Shortest Path							
	SameGraph				DifferentGraph			
	Small	Medium	Large	Average	Small	Medium	Large	Average
0	30.03	7.00	5.65	14.23	30.03	7.00	5.65	14.23
2	46.15	13.88	11.61	**23.88**	42.56	11.75	8.33	20.88
4	47.05	16.00	11.31	**24.79**	42.82	10.88	6.70	20.13

Notice: In the case of zero-demonstration, we used the same data in SameGraph and DifferentGraph. Since zero-demonstration does not provide any in-context information, our experiments directly count the accuracy of small, medium, and large graph questions in the GraphSCB dataset.

be due to the LLMs being limited by the maximum token length, which negatively impacts its ability to answer effectively with longer contexts.

There are also significant differences between LLMs in terms of tasks accuracy. For instance, in the Subgraph task, Qwen2.5-7B achieves an accuracy of 56.31% with zero examples, Qwen2.5-14B reaches 65.38%, while LLaMa3-8B only scores 42.31%. Some tasks experience a notable performance boost after adding two demonstrations, such as the Topology and Triangle tasks. However, for the Hamilton task, the accuracy of all three LLMs decreases as more demonstrations are added.

For the same task, different LLMs have varying requirements for the number of demonstrations. In the Bipartite task, Qwen2.5-7B shows a consistent increase in accuracy as more demonstrations are provided, reaching the highest accuracy with eight demonstrations. In contrast, Qwen2.5-14B achieves optimal performance with just two demonstrations.

5.3 Results for the Graph Structures (Q2)

The Graph Structure in Demonstrations Has a Significant Impact on Activating the GraphICL Ability of LLMs. In order to further study how to activate the GraphICL ability of the LLMs, we conducted experiments on the newly constructed **GraphSCB**. Through comparative experiments, we selected two types of demonstrations: one with the same graph structure as the query and one with a different graph structure. Here, different demonstrations refer to demonstrations extracted from different graphs in the same question category. The experimental results are shown in Table 3. We found that when the graph structure in the demonstrations is the same as the graph structure in the query, it significantly activates the GraphICL.

When the graph structure in demonstration is the same as the graph structure in the query, the accuracy of answers in 2 demonstrations and 4 demonstrations is the highest, which is verified in the Maximum Flow and Shortest Path tasks. For Connectivity, when the number of demonstrations is 2, the accuracy of the same graph structure is 2.21% higher than that of different graph structures. When the number of demonstrations is 4, the average accuracy is 79.31%, which is slightly higher than the same graph structure, indicating that for computationally simple tasks such as Connectivity, demonstrations with different graph structures may provide a wider range of graph structure features, which can better predict.

Table 4: Test accuracies (%) on the seven-class graph tasks, demonstrating the impact of demonstration categories. We bold the largest number in each row and underline the second largest number. We use zero-demonstration as the benchmark, light red indicates values greater than zero-demonstration, and light blue indicate values less than zero-demonstration.

Query	Demonstrations							
	Cycle	Connectivity	Bipartite	Topology	Shortest	Flow	Hamilton	Zero-Demonstration
Cycle	**96.52**	81.59	83.95	83.05	93.27	93.15	81.93	<u>94.84</u>
Connectivity	<u>92.33</u>	90.67	86.56	90.44	88.00	89.67	**94.11**	90.44
Bipartite	90.57	91.25	**94.39**	<u>92.37</u>	81.26	31.65	73.51	90.46
Topology	5.95	6.06	7.74	**30.19**	6.73	7.63	8.53	<u>17.40</u>
Shortest	**42.33**	38.89	36.67	39.67	<u>39.89</u>	36.89	39.33	38.11
Flow	17.11	32.00	20.89	17.56	28.89	**39.22**	20.22	<u>34.56</u>
Hamilton	68.80	<u>73.29</u>	60.04	69.36	70.93	69.70	**93.49**	66.33

When the graph structures are different, adding demonstrations can also improve the generalization ability of the model, but the effect is slightly inferior to the same graph structure. This shows that the model can still learn useful information from different graph structures, but the learning effect is limited. At the same time, for all three tasks, we experimented with the size of the graph. For small graph tasks, whether providing demonstrations of the same graphs or different graphs, the accuracy of the question was greatly improved. When the scale of the graph increases from small to large, the accuracy gradually decreases, but even on the largest graph tasks, adding demonstrations can significantly improve accuracy.

5.4 Results for Task Categories (Q3)

The Category of the Demonstrations Interacts with the Category of the Query. In order to better evaluate the ability of GraphICL, we conduct experiments on the **GraphTRB** dataset from the perspective of task categories. We fix the number of demonstrations to 4, and the experimental results are shown in the Table 4. For all seven categories of tasks, selecting demonstrations of the same category as the query being asked can significantly promote the question accuracy compared with Zero-Demonstration. For Cycle, Bipartite, Topology, Flow, and Hamilton questions, using demonstrations of the self category (i.e., the query and the demonstration category are the same) can achieve the highest accuracy. It is worth noting that for Connectivity and Shortest questions, cross-category demonstrations selection can also have a positive impact on the accuracy of the query, which may be due to the certain complementarity between different tasks, so that different category demonstrations can also provide valuable information. For example, the performance of the Connectivity task under the Cycle and Hamilton demonstration categories is higher than its performance under the self-category demonstration (90.67%).

However, for Cycle, Flow, and Hamilton questions, when selecting demonstrations that are different from the category of the query, the accuracy drops significantly compared to Zero-Demonstration. This shows that for these three categories of tasks, other categories cannot provide effective complementary information and may even introduce noise, thus affecting the performance of the model. Through this experiment, we show the mutual influence between task categories and point out the importance of demonstration selection strategies in GraphICL.

6 Conclusion

In this paper, we formalize GraphICL and propose a unified framework that integrates the query representations along with the graph question representations and answer representations derived from the demonstration set through a linear mechanism. We introduce two new datasets, GraphSCB and GraphTRB, to evaluate GraphICL. Our experiments show that LLMs exhibit capabilities of GraphICL, and its performance is greatly affected by graph structure similarity and task complementarity. Same or similar graph structures improve accuracy, while cross-task demonstrations may introduce noise unless the task information is complementary. Our work establishes foundational insights and benchmarks for advancing the state-of-the-art in GraphICL, highlights the need for improving demonstrations selection strategies, and lays the foundation for future explorations of GraphICL on more complex and diverse graph tasks.

7 Limitations

Despite systematically exploring GraphICL and providing key insights, our study still has some limitations. Our experiments are conducted entirely on open source models, the performance of proprietary, larger-scale LLMs (e.g., GPT-4o) remains unexplored, and large-scale batch testing is limited by the instability of API-based reasoning, which highlights the need for local deployment to ensure reproducibility and efficiency. In addition, graph question descriptions are usually long, and the maximum token length limit hinders the performance of LLMs when incorporating multiple demonstrations. Our experiments are limited by computational resources, which restricts the exploration of larger graphs and longer contexts.

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*.

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. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nuo Chen, Yuhan Li, Jianheng Tang, and Jia Li. 2024a. Graphwiz: An instruction-following language model for graph problems. *arXiv preprint arXiv:2402.16029*. Version 5.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and Jiliang Tang. 2024b. Exploring the potential of large language models (llms)in learning on graphs. *SIGKDD Explor. Newsl.*, 25(2):42–61.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada. Association for Computational Linguistics.

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*.

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

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*. Version 2.

Shibo Hao, Yi Gu, Haotian Luo, Tianyang Liu, Xiyan Shao, Xinyuan Wang, Shuhua Xie, Haodi Ma, Adithya Samavedhi, Qiyue Gao, et al. 2024. Llm reasoners: New evaluation, library, and analysis of step-by-step reasoning with large language models. *arXiv preprint arXiv:2404.05221*. Version 2.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Bryan Hooi, et al. 2024. Explanations as features: Llm-based features for text-attributed graphs. *arXiv preprint arXiv:2305.19523*. Version 5.

Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. 2024. Large language models on graphs: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 36(12):8622–8642.

Shuai Li, Zhao Song, Yu Xia, Tong Yu, and Tianyi Zhou. 2023a. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023b. Unified demonstration retriever for in-context learning. *arXiv preprint arXiv:2305.04320*. Version 2.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. 2024. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*. Version 2.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*. Version 2.

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

Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. 2024. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on*

Software Engineering, ICSE '24, pages 1–13, New York, NY, USA. Association for Computing Machinery.

Thao Nguyen, Yuheng Li, Utkarsh Ojha, and Yong Jae Lee. 2023. Visual instruction inversion: Image editing via image prompting. In *Advances in Neural Information Processing Systems*, volume 36, pages 9598–9613. Curran Associates, Inc.

Madhur Panwar, Kabir Ahuja, and Navin Goyal. 2024. In-context learning through the bayesian prism. In *The Twelfth International Conference on Learning Representations*. OpenReview.net.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*. Version 2.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. *arXiv preprint arXiv:2409.04109*.

Eshaan Tanwar, Subhabrata Dutta, Manish Borthakur, and Tanmoy Chakraborty. 2023. Multilingual llms are better cross-lingual in-context learners with alignment. *arXiv preprint arXiv:2305.05940*. Version 3.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2024. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*. Version 4.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023a. Can language models solve graph problems in natural language? In *Advances in Neural Information Processing Systems*, volume 36, pages 30840–30861. Curran Associates, Inc.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2023b. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 15614–15638. Curran Associates, Inc.

Noam Wies, Yoav Levine, and Amnon Shashua. 2023. The learnability of in-context learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 36637–36651. Curran Associates, Inc.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022. Active example selection for in-context learning. *arXiv preprint arXiv:2211.04486*.

Zeyang Zhang, Xin Wang, Ziwei Zhang, Haoyang Li, Yijian Qin, Simin Wu, and Wenwu Zhu. 2023. Llm4dyg: Can large language models solve problems on dynamic graphs? *arXiv preprint arXiv:2310.17110*. Version 3.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *In Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

A TASK DEFINITION

1. **Cycle.** In an undirected graph $\mathcal{G} = (V, E)$, the task is to detect whether a cycle exists. A cycle is a sequence of vertices $v_1, v_2, \ldots, v_k$ with $k \geq 3$, forming a closed loop where $v_1 = v_k$. Each vertex v_i must be distinct for all $1 \leq i < k$, and there must be an edge connecting v_i to v_{i+1}.

2. **Connectivity.** Given an undirected graph $\mathcal{G} = (V, E)$, the task is to determine if two randomly selected nodes $u, v \in V$ are connected through a sequence of edges in E.

3. **Bipartite Graph Check.** For a directed graph $\mathcal{G} = (V, E)$, the task is to check if the graph is bipartite. A graph is bipartite if its nodes can be divided into two disjoint sets $U, V \subseteq V$ such that no two nodes within the same set are adjacent.

4. **Topology Sort.** The task is to find a valid topological ordering of a directed graph $\mathcal{G} = (V, E)$. In a topological sort, the nodes are linearly ordered such that for every directed edge $(u, v) \in E$, node u comes before v in the ordering.

5. **Shortest Path.** In an undirected, weighted graph $\mathcal{G} = (V, E, w)$ where $w : E \rightarrow \mathbf{R}^+$ assigns a positive weight to each edge, the task is to find the path of minimal total weight between any two nodes $s, t \in V$.

6. **Maximum Weight Triangle.** Given an undirected, weighted graph $\mathcal{G} = (V, E, l)$ where $l : V \rightarrow \mathbf{R}^+$ assigns a positive weight to each vertex, the task is to identify a triangle (a cycle

of three connected vertices (v_1, v_2, v_3)) that maximizes the sum $l(v_1) + l(v_2) + l(v_3)$.

7. **Maximum Flow.** For a directed, weighted graph $\mathcal{G} = (V, E, c)$ where $c : E \to \mathbf{R}^+$ assigns a positive capacity to each edge, the task is to maximize the flow from a source node $s \in V$ to a sink node $t \in V$.

8. **Hamiltonian Path Detection.** The task is to ascertain the presence of a Hamiltonian path in an undirected graph $\mathcal{G} = (V, E)$. A Hamiltonian path is a path that visits each vertex in V exactly once.

9. **Substructure Matching.** Given two graphs $\mathcal{G} = (V, E)$ and $\mathcal{G}' = (V', E')$, the task is to determine if there is a subgraph of $\mathcal{G}$ that is isomorphic to $\mathcal{G}'$.

B ADDITIONAL EXPERIMENTS

B.1 Impact of Graph Size in Demonstrations on Problem Accuracy

The experimental results presented in Table 5 reveal several key insights into the impact of graph size in demonstrations on the accuracy of Connectivity task answers. Firstly, for questions involving small, medium, and large graphs, the optimal performance is consistently achieved when the graphs in the demonstrations are relatively small. This suggests that smaller graph demonstrations provide a more effective learning signal for the model, enabling it to better grasp the underlying connectivity concepts. Secondly, regardless of the size of the graphs in the questions, the accuracy of the answers decreases as the size of the graphs in the demonstrations increases. This indicates that larger graph demonstrations may introduce unnecessary complexity, hindering the model's ability to generalize effectively. Thirdly, as the size of the graphs in the qurey increases, the overall accuracy of the answers decreases, which is consistent with the findings in Table 2. This trend highlights the challenge of maintaining high accuracy when dealing with larger and more complex graph structures. Lastly, the results show that the accuracy of the answers improves with an increasing number of demonstrations, underscoring the importance of providing sufficient demonstrations to enhance the model's performance. These findings emphasize the need to carefully select demonstrations with smaller graph sizes to maximize the model's accuracy in answering connectivity questions, while also considering the number of demonstrations provided. In this experiment, we explored the effect of graph size in demonstrations on performance on connectivity tasks and showed that smaller graph examples generally improve accuracy on tasks of varying graph sizes. However, this trend may not be universally applicable to all graph tasks. Future work can further investigate how graph size and other factors influence the ICL ability for various graph tasks to develop more effective demonstration selection strategies.

B.2 Discussion of Tasks Group Performance

To further explore the interactions between different categories of graph tasks in GraphICL, we conducted task grouping experiments, with the results presented in Table 6. We fixed the number of demonstrations at 4 and randomly selected demonstrations to ensure the law of large numbers. Based on the experimental results in Table 4, we selected six categories of tasks and grouped them according to their graph task types and the nature of the graph algorithms involved. Group 1 includes tasks such as Cycle, Bipartite, and Hamilton, which focus more on graph traversal and structural properties. Group 2 includes tasks such as shortest path, maximum flow, and connectivity, which focus more on graph connectivity and optimization. This grouping enables us to systematically analyze how tasks within the same group or between different groups interact and affect each other's performance. The results show that the relevance of demonstrations to tasks significantly impacts performance. Tasks within the same group are more likely to benefit from each other's demonstrations, while using demonstrations from different groups may introduce noise and reduce the model's ability to generalize effectively. The grouped experiments emphasize the importance of task relevance in the selection of graph demonstrations for ICL. Future research should focus on developing more sophisticated methods for selecting and formatting demonstrations to maximize their utility and improve the model's performance on graph reasoning tasks.

B.3 Performance Gain of ICL over Zero-Demonstration for Graph Tasks

We processed the data in Table 4, and Figure 3 presents a heatmap that illustrates the gain in accuracy (percentage) when using several demonstrations compared to zero demonstration learning in various graph tasks. Heatmaps provide a vi-

Table 5: Test accuracies (%) on the connectivity graph question answering tasks. We highlight the best result in bold.

ICL-Examples	Connectivity								
	Small			Medium			Large		
	Small	Medium	Large	Small	Medium	Large	Small	Medium	Large
2	**87.99**	87.48	87.67	**79.33**	77.23	70.98	**73.55**	67.55	65.59
4	**89.14**	88.05	88.18	**80.57**	74.69	74.13	**75.32**	68.88	72.03
8	**89.67**	89.07	87.22	**81.54**	77.54	73.27	**76.14**	71.59	69.32

Table 6: Test accuracies (%) Comparison of Different Task Groups. We highlight the best result in bold.

Group		Group1			Group2		
		Cycle	Bipartite	Hamilton	Shortest	Flow	Connectivity
Group1	Cycle Bipartite Hamilton	**86.57**			63.49		
Group2	Shortest Flow Connectivity	41.00			**48.15**		

sual representation of performance improvement or degradation, offering valuable informations into the effectiveness of different example categories for specific tasks.

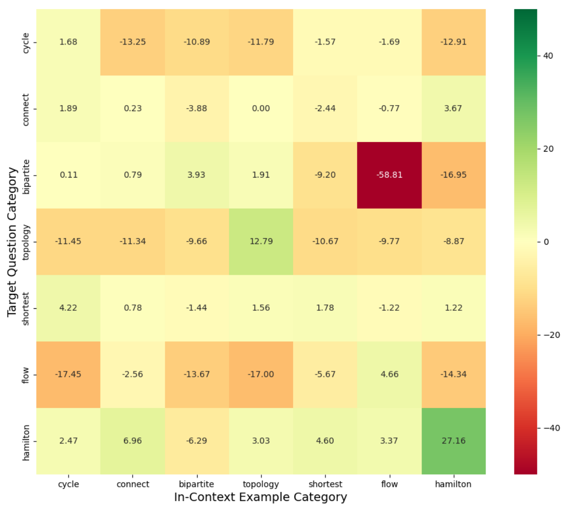

Figure 3: Performance Gain of In-Context Learning over Zero-Demonstration for Graph Tasks. Green shades indicate positive accuracy gains, with darker green showing higher improvements, while red shades represent negative gains, with darker red highlighting significant degradation.

C PROMPT CASES

In this section, we provide all the prompts for testing the performance of LLMs in different tasks. Table 7, 8, 9, 10, 11, 12, 13, 14, 15 provide the prompts for tasks in Table 2, Table 16, 17 provide the prompts for tasks in Table 3, and Table 18, 19 provide the prompts for tasks in Table 4.

Table 7: Prompts of Cycle Task.

Prompt of Cycle Tasks (2-Demonstration)

The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints.

Determine whether or not there is a cycle in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge.
Q: The nodes are numbered from 0 to 3, and the edges are: (0, 1) (1, 3) (1, 2). Is there a cycle in this graph?
A: Starting from node 0, we can only reach node 1. From node 1, we can reach nodes 0, 3, and 2. From node 2, we can only reach node 1. From node 3, we can only reach node 1. There is no way to form a cycle with these connections without revisiting an edge. Thus, with no cycle formation possible without reusing edges, there is no cycle in this graph.
Q: The nodes are numbered from 0 to 3, and the edges are: (0, 2) (1, 3). Is there a cycle in this graph?
A: There are two disjoint paths in this graph: [0→2] and [1→3]. There is no way to form a cycle with only two edges, because a cycle requires at least three edges and a closed loop. Therefore, there is no cycle in this graph.

Input:
Q: he nodes are numbered from 0 to 3, and the edges are: (0, 3) (2, 3). Is there a cycle in this graph?
A: <Answer >

Table 8: Prompts of Connectivity Task.

Prompt of Connectivity Tasks (2-Demonstration)

The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints.

Determine whether two nodes are connected in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge.
Q: The nodes are numbered from 0 to 4, and the edges are: (1, 4) (1, 2) (2, 3) (3, 4). Is there a path between node 3 and node 4?
A: Node 3 is connected to node 4 directly by an edge, so the answer is yes.
Q: The nodes are numbered from 0 to 6, and the edges are: (0, 3) (0, 2) (1, 6) (2, 6). Is there a path between node 1 and node 6?
A: Node 1 is connected to node 6 directly by an edge. We can follow the path: [1→6], so the answer is yes.

Input:
Q: The nodes are numbered from 0 to 4, and the edges are: (2, 4) (2, 3). Is there a path between node 0 and node 4?
A: <Answer >

Table 9: Prompts of Bipartite Task.

Prompt of Bipartite Tasks (2-Demonstration)

The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints.

Determine whether or not a graph is bipartite. In a directed graph, (i→j) means that node i and node j are connected with an directed edge from node i to node j.
Q: The nodes are numbered from 0 to 3, and the edges are: (0→2) (1→3) (1→2). Is this graph bipartite?
A: With this coloring, we can see that no two nodes with the same color have a direct edge, and the graph can be divided into two groups: (0,1) and (2.3). Therefore, the graph is bipartite.
Q: The nodes are numbered from 0 to 5, and the edges are: (0→4) (0→3) (0→1) (1→2) (4→5). Is this graph bipartite?
A: We can see that with this assignment, no two nodes in the same group are directly connected by an edge, and every edge connects a node from group (0,2,5) to a node from group (1,3,4). Therefore, the graph is bipartite.

Input:
Q: The nodes are numbered from 0 to 3, and the edges are: (0→2) (0→1) (1→3). Is this graph bipartite?
A: <Answer >

Table 10: Prompts of Topology Task.

Prompt of Topology Tasks (2-Demonstration)

The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints.

Find one of the topology sorting paths of the given graph. In a directed graph, (i→j) means that node i and node j are connected with a directed edge from node i to node j.
Q: The nodes are numbered from 0 to 3, and the edges are: (0→2) (0→1) (0→3) (1→3). Give one topology sorting path of this graph.
A: We can place node 0 first, as it has no incoming edges. Then, we can place node 1 second, as its only incoming edge is from node 0. Next, we can place node 2 third, as its only incoming edge is from node 0. Finally, we can place node 3 last, as it has incoming edges from nodes 0 and 1. Therefore, a valid topological sorting path for this graph is [0, 1, 2, 3].
Q: The nodes are numbered from 0 to 3, and the edges are: (0→2) (0→1) (0→3) (2→3). Give one topology sorting path of this graph.
A: Initially, node 0 is the only node that has no incoming edges, so it is the first node in the path. Next, node 1 becomes a candidate, as its only incoming edge is from node 0, which has already been removed from the graph. Then, node 2 is also a valid choice, as its only incoming edge is from node 0 as well. Finally, node 3 is the last node to be added to the path, as it has incoming edges from nodes 0 and 2, which have already been placed in the path. Therefore, a valid topology sorting path for this graph is [0, 1, 2, 3].

Input:
Q: The nodes are numbered from 0 to 3, and the edges are: (0→1) (0→2) (1→3) (1→2). Give one topology sorting path of this graph.
A: <Answer >

Table 11: Prompts of Shortest Task.

Prompt of Shortest Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Find the shortest path between two nodes in an undirected graph. In an undirected graph, (i,j,k) means that node i and node j are connected with an undirected edge with weight k. Q: The nodes are numbered from 0 to 5, and the edges are: (0,4,10) (1,4,4) (1,3,3) (4,5,2). Give the weight of the shortest path from node 3 to node 0. A: The weight of path 3,1,4,0 is the smallest, so the shortest path from node 3 to node 0 is [3,1,4,0] with a total weight of 17. Q: The nodes are numbered from 0 to 4, and the edges are: (0,4,1) (0,3,1) (1,3,10) (2,3,5). Give the weight of the shortest path from node 0 to node 2. A: The weight of path 0,3,2 is the smallest, so the shortest path from node 0 to node 2 is [0,3,2] with a total weight of 6. Input: Q: The nodes are numbered from 0 to 4, and the edges are: (0,1,3) (0,3,1) (1,2,4). Give the weight of the shortest path from node 1 to node 3. A: <Answer >

Table 12: Prompts of Triangle Task.

Prompt of Triangle Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Find the maximum sum of the weights of three interconnected nodes. In an undirected graph, [i, k] means that node i has the weight k. (i,j) means that node i and node j are connected with an undirected edge. Q: The nodes are numbered from 0 to 3, weights of nodes are: [0, 5] [1, 2] [2, 8] [3, 10], and the edges are: (0, 2) (0, 3) (2, 3). What is the maximum sum of the weights of three nodes? A: Nodes 0, 2, and 3 are the only interconnected nodes in this graph, each directly connected to the other two. The sum of their weights is [5 (Node 0) + 8 (Node 2) + 10 (Node 3) = 23]. There are no other groups of three interconnected nodes in this graph. Therefore, the maximum sum of the weights of three interconnected nodes in this graph is 23. Q: The nodes are numbered from 0 to 3, weights of nodes are: [0, 9] [1, 10] [2, 1] [3, 1], and the edges are: (0, 1) (0, 3) (1, 3). What is the maximum sum of the weights of three nodes? A: There is only one group of three interconnected nodes in this graph, which is Nodes 0, 1, and 3. The sum of their weights is [9 (Node 0) + 10 (Node 1) + 1 (Node 3) = 20]. There are no other groups of three interconnected nodes in this graph. Therefore, the maximum sum of the weights of three interconnected nodes in this graph is 20. Input: Q: The nodes are numbered from 0 to 3, weights of nodes are: [0, 4] [1, 10] [2, 4] [3, 6], and the edges are: (1, 2) (1, 3) (2, 3). What is the maximum sum of the weights of three nodes? A: <Answer >

Table 13: Prompts of Flow Task.

Prompt of Flow Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Find the maximum flow between two nodes in a directed graph. In a directed graph, (i→j,k) means that node i and node j are connected with an directed edge from node i to node j with weight k. Q: The nodes are numbered from 0 to 5, and the edges are: (0→4,2) (1→2,1) (2→3,1) (3→4,7) (3→5,3). What is the maximum flow from node 1 to node 3? A: The only direct path from node 1 to node 3 is through node 2, which has a capacity of 1 unit. There are no other paths that can increase the flow from node 1 to node 3, since node 0 and node 4 are not reachable from node 1. Therefore, the maximum flow from node 1 to node 3 is 1 unit. Q: The nodes are numbered from 0 to 9, and the edges are: (0→3,7) (2→3,9) (3→6,6) (4→9,10). What is the maximum flow from node 0 to node 6? A: The only path from node 0 to node 6 is through node 3, so the maximum flow is limited by the capacity of the edge (0→3) and the edge (3→6). Since the edge (0→3) has a capacity of 7 units and the edge (3→6) has a capacity of 6 units, the bottleneck is the edge (3→6). Therefore, the maximum flow from node 0 to node 6 is 6 units. Input: Q: The nodes are numbered from 0 to 7, and the edges are: (0→1,7) (2→4,2) (2→7,3) (3→4,2) (4→5,7). What is the maximum flow from node 3 to node 5? A: <Answer >

Table 14: Prompts of Hamilton Task.

Prompt of Hamilton Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Determine whether or not there is a Hamiltonian path in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge. Q: The nodes are numbered from 0 to 2, and the edges are: (0, 2) (1, 2). Is there a Hamiltonian path in this graph? A: To determine if a Hamiltonian path exists in an undirected graph, we need to check if there's a path that visits each node exactly once. In this simple graph with three nodes and two edges, the only possible Hamiltonian path is: [0, 2, 1] or [1, 2, 0]. Therefore, there is a Hamiltonian path in this graph. Q: The nodes are numbered from 0 to 3, and the edges are: (0, 1) (0, 2) (1, 2) (2, 3). Is there a Hamiltonian path in this graph? A: To determine if a Hamiltonian path exists in an undirected graph, we need to check if there's a path that visits each node exactly once. We can start at node 0. As node 0 is connected with node 1 and node 2, we can choose either of them as the next node. Suppose we choose node 1.As node 1 is connected with node 0 and node 2, we can choose node 2 as the next node.As node 2 is connected with node 0, node 1, and node 3, we can choose node 3 as the next node.As node 3 is connected with node 2, we have no more choices and we have visited all the nodes. So, one possible Hamiltonian path is: [0, 1, 2, 3]. Therefore, there is a Hamiltonian path in this graph. Input: Q: The nodes are numbered from 0 to 3, and the edges are: (0, 2) (0, 1) (1, 3). Is there a Hamiltonian path in this graph? A: <Answer >

Table 15: Prompts of Subgraph Task.

Prompt of Subgraph Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Determine if a smaller graph is present as an exact match within a larger graph. In a directed graph, (i→j) means that node i and node j are connected with a directed edge from node i to node j. Q: The nodes of graph G are numbered from 0 to 3, and the edges are: (0→1) (1→2) (2→3) (0→3). The nodes of subgraph G' are numbered from a to e, and the edges are: (a→e) (a→c) (a→d) (b→d). Is subgraph G' present within graph G as a direct substructure? A: Subgraph G' has more nodes and edges than graph G, so it cannot be present as a direct substructure within G. Q: The nodes of graph G are numbered from 0 to 6, and the edges are: (0→4) (0→1) (1→3) (5→6). The nodes of subgraph G' are numbered from a to d, and the edges are: (a→c) (a→b) (b→d) (c→d). Is subgraph G' present within graph G as a direct substructure? A: No node in G satisfies the conditions of subgraph G', so subgraph G' is not present within graph G as a direct substructure. Input: Q: The nodes of graph G are numbered from 0 to 4, and the edges are: (0→4) (1→2) (1→3) (2→3). The nodes of subgraph G' are numbered from a to d, and the edges are: (a→b) (a→d) (b→c) (b→d). Is subgraph G' present within graph G as a direct substructure? A: <Answer >

Table 16: Prompts of Connectivity Task using the same graphs as examples.

Prompt of Connectivity Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Determine whether two nodes are connected in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge. Q: The nodes are numbered from 0 to 4, and the edges are: (0, 4) (2, 3) (3, 4). Is there a path between node 3 and node 0? A: To see if there's a path between node 3 and node 0, we can trace the connections: From node 3, you can go to node 4 (via the edge (3, 4)). From node 4, you can go to node 0 (via the edge (0, 4)). So, yes, there is a path between node 3 and node 0, which goes through node 4. Q: The nodes are numbered from 0 to 4, and the edges are: (0, 4) (2, 3) (3, 4). Is there a path between node 3 and node 4? A: Yes, there is a direct path between node 3 and node 4. This is given by the edge (3, 4), meaning node 3 is directly connected to node 4. Input: Q: The nodes are numbered from 0 to 4, and the edges are: (0, 4) (2, 3) (3, 4). Is there a path between node 0 and node 4? A: <Answer >

Table 17: Prompts of Connectivity Task using the different graphs as examples.

Prompt of Connectivity Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Determine whether two nodes are connected in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge. Q: The nodes are numbered from 0 to 4, and the edges are: (1, 4) (1, 2). Is there a path between node 4 and node 2? A: We can trace the possible connections: Node 4 is connected to node 1 (via the edge (1, 4)). Node 1 is connected to node 2 (via the edge (1, 2)). So, there is a path from node 4 to node 2, and it goes through node 1. Q: The nodes are numbered from 0 to 4, and the edges are: (0, 3) (0, 2) (1, 3). Is there a path between node 3 and node 1? A: Let's examine possible paths from node 3 to node 1: Start at node 3. From the edge (1, 3), we see that node 3 is connected to node 1. So yes, there is a direct path between node 3 and node 1 via the edge (1, 3). Input: Q: The nodes are numbered from 0 to 4, and the edges are: (0, 4) (2, 3) (3, 4). Is there a path between node 0 and node 4? A: <Answer >

Table 18: Prompts of Connectivity Task using the same tasks as examples.

Prompt of Connectivity Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Determine whether two nodes are connected in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge. Q: The nodes are numbered from 0 to 9, and the edges are: (0, 3) (2, 3) (3, 6) (4, 9). Is there a path between node 2 and node 5? A: Node 2 is connected to node 3 (via the edge (2, 3)). From node 3, there is a connection to node 6 (via the edge (3, 6)). There is no direct or indirect connection from node 6 or any other node to node 5, based on the given edges. So, there is no path between node 2 and node 5. Q: The nodes are numbered from 0 to 9, and the edges are: (0, 3) (2, 3) (3, 6) (4, 9). Is there a path between node 4 and node 5? A: Node 4 is connected to node 9 (via the edge (4, 9)). There is no direct or indirect connection between node 9 and node 5 based on the edges provided. So, there is no path between node 4 and node 5. Input: Q: The nodes are numbered from 0 to 9, and the edges are: (0, 3) (2, 3) (3, 6) (4, 9). Is there a path between node 1 and node 5? A: <Answer >

Table 19: Prompts of Connectivity Task using the different tasks as examples.

Prompt of Connectivity Tasks (2-Demonstration)
The following is a Graph problem. Your task is to think about this problem and provide a solution based on the [ICL Examples] I provide as hints. Find the maximum flow between two nodes in a directed graph. In a directed graph, (i→j,k) means that node i and node j are connected with a directed edge from node i to node j with weight k. Q: The nodes are numbered from 0 to 9, and the edges are: (0→3,7) (2→3,9) (3→6,6) (4→9,10). What is the maximum flow from node 3 to node 6? A: Let's focus on the path from node 3 to node 6, which is the edge (3→6) with a capacity of 6. Since node 3 to node 6 is the only direct path and the edge has a capacity of 6, the maximum flow from node 3 to node 6 is 6. Q: The nodes are numbered from 0 to 9, and the edges are: (0→3,7) (2→3,9) (3→6,6) (4→9,10). What is the maximum flow from node 0 to node 3? A: For the flow from node 0 to node 3, we focus on the edge (0→3) with a capacity of 7. There is no other edge directly connecting node 0 to node 3, so the maximum flow is limited by this edge's capacity. Thus, the maximum flow from node 0 to node 3 is 7. Input: Determine whether two nodes are connected in an undirected graph. In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge. Q: The nodes are numbered from 0 to 9, and the edges are: (0, 3) (2, 3) (3, 6) (4, 9). Is there a path between node 0 and node 6? A: <Answer >