Evaluating the Structural Awareness of Large Language Models on Graphs: Can They Count Substructures?

Ly Nguyen, Yujun Yan
{ly.h.nguyen.25,yujun.yan}@dartmouth.edu
Dartmouth College
Hanover, New Hampshire, USA

ABSTRACT

Recent advancements in Large Language Models (LLMs) have showcased their ability to effectively process various tasks, including those in the domain of graph reasoning. However, most studies rely on LLMs' ability to process textual information, such as node and graph features. The exploration of LLMs' capability to incorporate and utilize structural information in graph-related tasks is limited.

One of the important tasks in structure-aware graph learning is to count substructures. Researchers use this capability to measure the expressiveness of Graph Neural Networks (GNNs). Additionally, it is essential in medical and social analysis applications, where these substructures have specific functions. Thus, in this paper, we delve into the expressive power of LLMs with a specific focus on their ability to count substructures. We extensively experiment with different prompting strategies to study the performance of LLMs in substructure counting. Additionally, we compare LLMs' performance with that of graph networks, aiming to elucidate their respective advantages and limitations in utilizing structural data. Furthermore, we investigate the out-of-distribution generalizability of LLMs in substructure counting, specifically when test graphs exceed the size of training graphs. Our research contributes to a deeper understanding of the expressiveness and generalizability of LLMs in handling diverse graph structures.

CCS CONCEPTS

• **Information systems** → *Data mining*; • **Computing methodologies** → *Natural language processing*.

KEYWORDS

graph mining, counting substructures, prompt engineering, large language models, graph neural networks

ACM Reference Format:

Ly Nguyen, Yujun Yan. 2024. Evaluating the Structural Awareness of Large Language Models on Graphs: Can They Count Substructures?. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24).* ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Traditionally, graph neural networks (GNNs) have been at the forefront of graph learning tasks. Expressiveness, a crucial property of GNNs, refers to their ability to model and differentiate between complex graph structures. This property is crucial for their effectiveness in various graph learning tasks, which can be assessed by their ability to count substructures. Despite the advancements in GNN architectures, most traditional GNNs lack sufficient expressiveness. Specifically, as shown by Chen et al. (2020) [3], certain GNN models that use message passing between nodes cannot count induced subgraphs for any connected pattern of 3 or more nodes. Although multiple Higher-Order Graph Neural Networks [15] have been proposed to improve the expressiveness regarding this problem, the impossibility remains [15].

LLMs have demonstrated promising results in tasks such as node classification and graph classification tasks [4]. However, most of these works have focused on better utilization of textual information, such as node and graph features. Their ability to utilize structural information, specifically for counting substructures, a fundamental problem in graph analysis, remains understudied. Moreover, the practical applications of this research extend significantly into fields such as computational chemistry, materials design, and pharmacy [12, 14], where the functions crucial for innovation and discovery often demand the presence or count of specific substructures, such as functional groups.

In this paper, we aim to explore the potential of LLMs to leverage graph structural information for counting substructures, intending to study how LLMs can complement or replace traditional graph network approaches. We used three GNN models [10] [17] [15] as a benchmark to evaluate the performance and capabilities of LLMs in substructure counting tasks across various graph sizes to assess generalizability and substructure complexities to assess expressiveness. Our main findings reveal that LLMs demonstrate potential in counting substructures, particularly excelling in counting 3-stars where they outperform all traditional GNNs and even higher-order GNNs mentioned above. Advanced prompting techniques, especially the combination of few-shot and least-to-most prompting with 1-hop neighbor information, significantly enhance LLMs' performance in these tasks. To elaborate, few-shot prompting provides examples to guide the model [2], least-to-most prompting helps break down complex problems into manageable steps for counting substructures associated with each node [20], and including 1-hop neighbor information enhances the model's understanding of local graph structure. This approach also achieves superior performance compared to other prompting techniques. However, LLMs show limitations in counting cycles (such as triangles and chordal 4-cycles).

They fail to generalize effectively when tasked with counting substructures in test graphs that are larger than the examples provided in the prompts.

- **Exploration of Graph Structure Impact For Generalizability and Expressiveness of LLMs:** We are the first to study LLMs' ability to count substructures, which is crucial to understand their expressiveness in structural learning. Additionally, we investigate LLMs' performance across varying graph sizes and complexities to assess their generalizability to varying graph input sizes.
- **Benchmarking Against Previous Methods and Leveraging Substructure Identification:** Acknowledging the ability of graph neural networks in counting substructures [15], we serve graph neural networks as a benchmark to validate the performance of LLMs for the same tasks. We also evaluate how precise the substructures LLMs identify.
- **Experimental Findings:** Regarding expressiveness, LLMs particularly struggle with cycle-based substructures compared to GNNs. Additionally, we demonstrate LLMs' sensitivity to graph sizes, with performance declining for larger graphs.

2 RELATED WORK

Recent advancements in LLMs have sparked significant interest in their ability to reason over graph-structured data. Several studies have explored the intersection of LLMs and graph reasoning tasks, highlighting the importance of incorporating structural information into prompts to enhance performance.

LLMs Can Assist with Graph Learning Tasks. Zhang proposed Graph-Toolformer, a framework that utilizes LLMs as an interface to bridge natural language commands and GNNs [19]. Since there have been comprehensive studies on encoding graph-structured data as text for consumption by LLMs [7], researchers have followed to propose a testbed for evaluating the graph reasoning abilities of LLMs, revealing their potential capabilities in tasks such as connectivity, cycle detection, and shortest path finding [16]. Regarding node classification problems, Guo et al. explored two pipelines: LLMs-as-Enhancers, where LLMs are used to enhance text attributes before being processed by GNNs, and LLMs-as-Predictors, where LLMs directly generate predictions based on graph information provided through prompts [9]. Most of these studies focus on textual information and the integration of LLMs with GNNs for various graph reasoning tasks. However, none have utilized LLMs specifically to count substructures. Our work shifts the focus to structural information, investigating how LLMs can effectively count substructures and leverage graph structure in problem-solving.

Counting Substructures with Graph Networks. GNNs have shown limitations in counting substructures, particularly for patterns involving three or more nodes [3]. Specifically, low-order GNNs struggle to learn structural graph parameters such as clique information, diameter, conjoint cycles, or shortest cycles [8]. To address these limitations, researchers have developed more expressive GNN architectures. Tahmasebi et al. [15] introduced a recursive pooling technique called Recursive Neighborhood Pooling (RNP) can count subgraphs of size k, overcoming the limitations of low-order GNNs in Chen et al. [3]. Bouritsas et al. [1] proposed the Graph Neural Networks with Subgraph Isomorphism Counting approach, which incorporates subgraph isomorphism counting into the message passing framework, further enhancing GNNs' ability to capture structural information. You et al. [18] introduced the Identity-Aware Graph Neural Networks (ID-GNNs), which use node identifiers to break symmetries in the graph, allowing for more precise substructure counting. However, GNNs still face challenges in capturing long-range dependencies in graphs [5], which can affect their ability to accurately count substructures involving nodes that are far apart in the graph.

The novelty of our work lies in the specific focus on substructure counting using LLMs, an aspect that has not been extensively explored in previous studies. Moreover, we utilize the structural information instead of most textual information tasks performed with LLMs. We also draw comparisons between the performance of LLMs and graph network approaches in the context of substructure counting to assess expressiveness and generalizability which has been studied extensively for graph networks methods but not for LLMs. We have demonstrated that LLMs can outperform GNNs with the right tasks, prompt engineering, and graph encoding strategies, we highlight the potential strengths and weaknesses of LLMs in substructure counting and open up new avenues for future research in this area.

3 MODELS AND DESIGN

3.1 Models and Settings

We use GPT-3.5-TURBO as our primary LLM for these experiments. To achieve these goals, we designed the following experiments:

- By examining LLMs' ability to count substructures of different nature and node counts including triangles, 3-stars, and chordal 4-cycles, we can evaluate their expressiveness.
- To assess generalizability, we test LLMs on graphs of varying sizes from 10, 15, and 20, to 30 nodes, allowing us to understand how well they adapt to larger and more complex graphs.
- We employ different prompting techniques including those that incorporate local structural information (1-hop neighbors information) and step-by-step reasoning as the control factors, to understand how LLMs process and utilize graph structural data.

3.2 Prompting techniques

To establish a comprehensive evaluation framework, we use a diverse set of prompting techniques as baselines. These are the details regarding how the prompting techniques are designed to adapt to how we assess the impact of structural information:

(1) **Zero-Shot**: Provide only the definition of the substructure.
(2) **0-COT (Zero-Shot Chain-of-Thoughts)**: Provide the definition of the substructure. Instead of giving the specific reasoning steps as in Chain-of-Thoughts (COT), we only input the prompt "Let's think step by step." in the instruction [11].

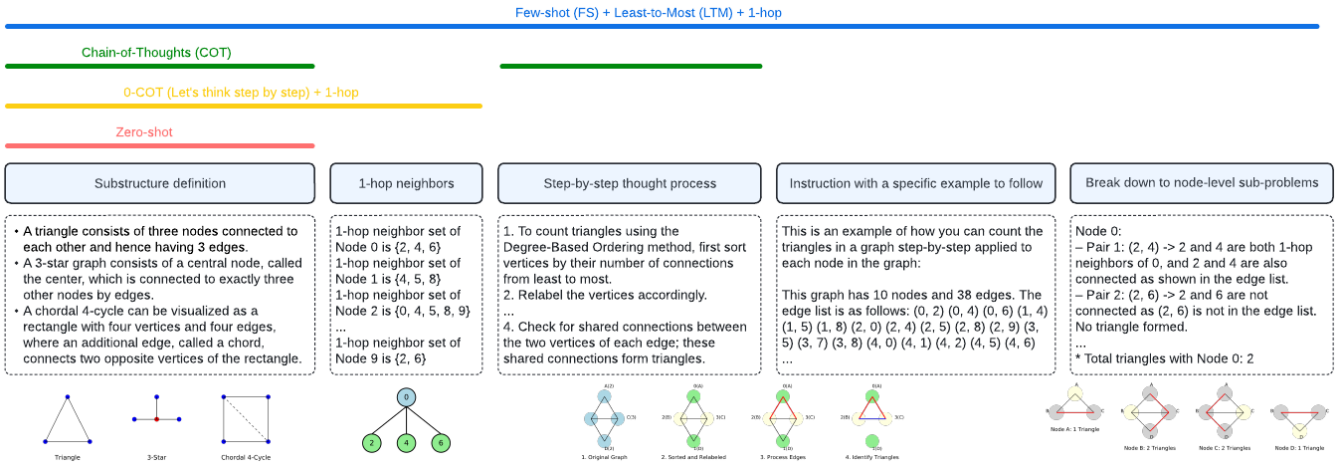


Figure 1: Prompting techniques and graph input representation.

(3) **0-COT + 1-hop**: Provide the definition of the substructure and the prompt "Let's think step by step." Note that we also list the neighbors information of each node.
(4) **FS (Few-shot) + Least-to-Most (LTM) + 1-hop**: Provide the definition of the substructures, list the neighbors of each node, give an example [2] where we break down the counting tasks into sub-problems, give a detailed example of solving these sub-problems [20]. Note that we give one example of a graph with 10 nodes.
(5) **COT (Chain-of-Thoughts)**: Provide the definition of the substructures, without listing the neighbors of each node. Then provide a series of intermediate reasoning steps.
(6) **COT + BAG (Build-a-Graph)**: Provide the definition of the substructures without listing the neighbors of each node but include the prompt "Let's build a graph." The rest is the same with COT.

We use Mean Absolute Error (MAE), Mean Squared Error (MSE) divided by the variance of the true counts, and accuracy (ACC) as the criteria for performance metrics in terms of the count number.

3.3 Baselines

To compare the performance of LLMs with traditional graph neural network architectures, we include Graph Isomorphism Network (GIN) [17] and Graph Convolutional Network (GCN) [10] as representative models. Furthermore, we also include a higher-oder GNN (RNP-GNN) [15] as a representative base model, which has been an advancement compared to 2 above mentioned models.

4 EXPERIMENTAL DETAILS

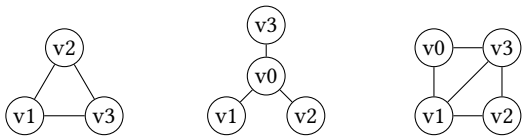
4.1 Datasets

In our study, we employ two datasets as set up in Chen et al. [3] to evaluate the performance of LLMs on substructure counting tasks: the Erdős-Renyi dataset and the Noisy Random Regular dataset. Both datasets also include labels of ground-truth counts for substructures considered in the experiments: (a) 3-star (b) triangle (c) chordal cycle.

4.1.1 Erdős-Renyi Dataset. The Erdős-Renyi dataset consists of 5,000 randomly generated graphs following the Erdős-Renyi model [6]. In this model, a graph is constructed by connecting nodes randomly with a fixed probability. Each graph in the dataset contains 10 nodes, and the edges are randomly distributed among these nodes. The results obtained from the Erdős-Renyi dataset serve as a benchmark for LLMs' ability to count substructures in graphs with varying connectivity.

4.1.2 Noisy Random Regular Dataset. The Noisy Random Regular dataset consists of 5,000 graphs generated using the random regular graph model [13] with added noise. In a random regular graph, each node has the same degree, meaning that all nodes have the same number of connections. The graphs in this dataset have varying sizes, with either 10, 15, 20, or 30 nodes. The maximum number of edges is 90. To introduce noise and make the graphs more challenging, we randomly delete n edges from each graph, where n is equal to the number of nodes in the graph.

4.2 Expressiveness Assessment



(a) Triangle **(b) 3-Star** **(c) Chordal 4-Cycle**

Figure 2: Substructures considered in the experiments: (a) triangle (b) 3-star (c) chordal 4-cycle.

These tasks are formulated as graph regression problems, where the input is a graph representation (including node and edge information), and the output is the count of specific substructures and the list of specific substructures.

- **Task 1: Counting Triangles** Given an undirected graph $G = (V, E)$, count the number of triangles in G. A triangle is defined as a set of three nodes $(v_1, v_2, v_3) \in V$ such that

(v_1, v_2), (v_2, v_3), and $(v_3, v_1) \in E$. The task is to compute $CI(G; T)$, the induced-subgraph-count of a triangle pattern T in G, where each triangle contributes to the count only if all three edges that form the triangle are present in E.

- **Task 2: Counting 3-Stars** Count the number of 3-star substructures in an undirected graph $G = (V, E)$. A 3-star is a central node v_0 connected to three distinct nodes v_1, v_2, and v_3 (i.e., v_0 is the central hub). This task aims to calculate $CS(G; S_3)$, the subgraph-count of the 3-star pattern S_3 in G, which counts each occurrence of S_3 regardless of additional edges among the leaves v_1, v_2, and v_3.
- **Task 3: Counting Chordal 4-Cycles** Given an undirected graph $G = (V, E)$, count the number of chordal 4-cycles present in G. A chordal 4-cycle is defined as a set of four nodes $(v1, v2, v3, v4) \in V$ that form a cycle, such that $(v1, v2)$, $(v2, v3)$, $(v3, v4)$ and $(v4, v1) \in E$, with at least one additional internal edge (chord) that connects a pair of non-consecutive vertices within the cycle, making the cycle chordal. This task requires the count of $CI(G; C)$, the induced-subgraph-count of a chordal 4-cycle pattern C in G, where each chordal 4-cycle contributes to the count only if all four edges that form the cycle and at least one chordal edge are present in E. Including chordal 4-cycles further enhances our ability to assess the expressive power of the LLM approach in handling complex graph structures, involving both cyclic connectivity and internal chords together with increasing number of nodes and edge features, as they occur as subgraphs or induced subgraphs within the graph.
- **Task 4: Validate node sets of triangles, 3-stars, and chordal 4-cycles output**. For each graph in the datasets, we task the LLM model with listing out the unique sets of nodes that form triangular, 3-star, and chordal 4-cycle substructures. This step allows us to evaluate the LLM's ability to correctly identify and enumerate the specific instances of these substructures within the graph. Due to the lack of ground truth exact nodes associated with the substructures present in the graph, we calculate the precision by checking all the substructures listed by the LLMs and come up with the proportion of correctly identified substructures. By doing this, we identify areas where LLMs excel, either in correctly identifying substructures or accurately counting their occurrences.

4.3 Generalizability Assessment

To evaluate the generalizability of LLMs in counting substructures, we instruct LLMs with a graph example of 10 nodes and then test the models on graphs of increasing sizes categorized as easy, medium, and hard. The graphs in these categories contain 10, 15-20, and 30 nodes, respectively.

Table 1: Task Specifics for Counting Triangles, 3-Star Structures, and Chordal 4-Cycle

Task	Few-shot Example	Easy	Medium	Hard
# Nodes	10	10	15-20	30
Total	/	1,228	2,507	1,265

5 RESULTS

In assessing LLM's ability to count substructure, We discuss and establish three key observations focusing on its expressiveness and generalizability:

(1) Of all prompting techniques tailored to count substructures, FS+LTM+1-hop has the best results across 3-stars, triangles, and chordal 4-cycles.
(2) LLMs show signs of sensitivity to the size of the input graphs.
(3) LLMs show signs of sensitivity to the substructure.

5.1 Step-by-step reasoning instruction is critical for optimizing LLMs.

Observation 1: FS+LTM+1-hop is the most effective prompting technique for enhancing LLMs' ability to count substructures.

In counting 3-stars across different levels (Table 4), FS+LTM+1-hop achieves precision levels of 93.19%, 96.78%, and 98.16% for easy, medium, and hard graphs, respectively. Similarly, for counting triangles, FS+LTM+1-hop improves precision to 52.02%, 29.01%, and 11.34% for easy, medium, and hard difficulty levels, respectively. Other prompting methods stay behind. Tables 3 and 4 show that 0-COT+1-hop prompting barely increases counting accuracy and precision of substructure identification compared to zero-shot and 0-COT. Table 5 and Table 6 present performance metrics for counting 3-stars and counting triangles across different graph sizes. Table 5 and Table 6 show that zero-shot or 0-COT yield surprisingly low accuracy for counting 3-stars (0.00% for both in this experiment and the lowest in most other cases).

Table 2: Precisions Across Optimal Methods for Counting Substructures Tasks.

Methods	Easy	Medium	Hard	Avg.
Counting 3-Stars				
FS+LTM+1-hop	93.19	96.78	98.16	96.25
COT+BAG	63.15	59.63	56.22	59.63
COT	69.65	66.82	64.90	67.03
Counting Triangles				
FS+LTM+1-hop	52.02	29.01	11.34	30.19
COT+BAG	39.97	17.78	6.70	20.43
COT	41.77	18.48	6.97	22.41

5.2 LLMs struggle with counting cycles while outperforming GNNs in counting 3-stars.

In general, LLMs have shown potential to outperform GNNs in certain substructure counting tasks, particularly for 3-stars. They demonstrate a unique ability to identify substructures with high precision. However, LLMs face challenges with cyclic structures such as triangles and chordal 4-cycles, sharing similar difficulties with traditional GNNs in these areas.

Observation 2: LLMs can outperform all graph networks models, including even Higher-Order GNNs (RNPGNN) for counting 3-stars.

Table 3: Performance metrics for the task for counting triangles. ACC is calculated for the count, while Precision is calculated for the validated substructures. Gray denotes the best result. Slash (/) denotes no available results.

Method	MAE (↓)	MSE/σ^2 (↓)	ACC (↑)	PRE (↑)
Zero-shot	5.58	23.59	6.68	36.36
0-COT	4.40	6.66	8.35	36.61
0-COT+1-hop	3.17	2.62	10.65	39.76
FS+LTM+1-hop	2.28	1.29	15.05	44.97
COT	4.03	4.59	9.74	35.48
COT+BAG	4.26	5.17	9.09	34.48
GCN	2.07	0.98	15.48	/
GIN	0.63	0.10	50.04	/
RNPGNN	0.26	0.02	85.56	/

Table 4: Performance metrics for the task for counting 3-stars. ACC is calculated for the count, while Precision is calculated for the validated substructures. Gray denotes the best result. Slash (/) denotes no available results.

Method	MAE (↓)	MSE/σ^2 (↓)	ACC (↑)	PRE (%) (↑)
Zero-shot	18.08	1.97	3.07	72.91
0-COT	16.53	1.73	3.51	71.08
0-COT+1-hop	16.99	1.79	3.33	74.64
FS+LTM+1-hop	0.35	0.02	94.44	93.24
COT	7.82	0.97	27.78	80.72
COT+BAG	6.82	0.67	25.79	71.44
GCN	0.98	13.28	19.20	/
GIN	0.36	7.1E-4	79.76	/
RNPGNN	1.10	0.01	35.96	/

As shown in Table 3, our most effective LLM approach (FS+LTM+1-hop) surpasses the performance of all GNN models including GCN, GIN, and RNPGNN in counting 3-stars. Notably, LLMs even outperform the Recursive Neighborhood Pooling GNN (RNPGNN), a higher-order GNN designed to enhance expressiveness in substructure counting tasks [15]. With the FS+LTM+1-hop approach, LLMs achieve 94.44% accuracy in counting 3-stars, compared to 79.76% for GIN and 35.96% for RNPGNN.

Observation 3: LLMs exhibit limited expressive power compared to GNNs when it comes to cycles. Chen et al. [3] demonstrated that Message Passing Neural Networks (MPNNs) and second-order Invariant Graph Networks (2-IGNs) cannot perform induced-subgraph-count for any connected substructure consisting of 3 or more nodes, except for star-shaped patterns. Our results show that LLMs face similar challenges with cyclic structures. In counting triangle tasks, we observe that LLMs' accuracy is around 15%, while the higher-order GNN (RNPGNN) achieves 85.56%. The results in Table 7 further demonstrate that LLMs face significant challenges in accurately counting chordal 4-cycles, a more complex cyclic substructure. Even with explicit instructions to first count the induced sub-graph of a 4-cycle and then check for a chord, the best-performing LLM approach (FS+LTM+1-hop) achieves only 19.79% accuracy and 16.61% precision. This limit can be partially explained by Observation 4.

Observation 4: LLMs' brittleness leads to incorrect counting and identification. LLMs are indeed surprisingly vulnerable to spurious correlations in structured reasoning, leading to incorrect counts, particularly for cycles, and identification of substructures.

Consider the following response for a graph where Node 0 is connected to nodes 2, 4, 5, 7; Node 2 is connected to nodes 0, 4, 7; Node 4 is connected to nodes 0, 1, 2, 6, 9.

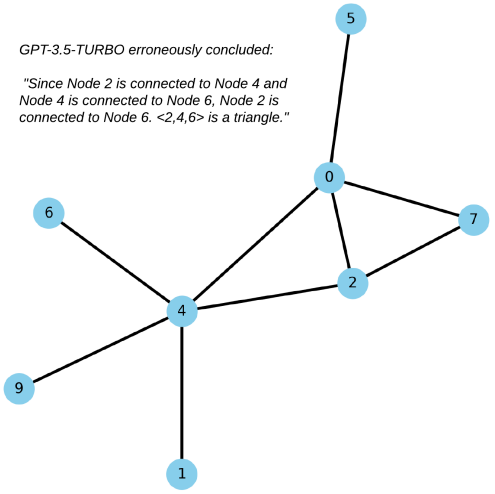


Figure 3: Graph visualization of the case where LLMs make a fundamental mistake in detecting edges in cycles.

The ground truth shows that there is no direct connection between Node 2 and Node 6.

5.3 LLMs are sensitive to the size of the input graphs.

We track LLM's accuracy in the count number and its precision in validating substructures across different graph sizes.

Observation 5: There are generalization limits for LLMs. As depicted in Table 3, there is a noticeable downtrend in both accuracy and precision metrics as the graph sizes expand from 10, 15, and 20, to 30 nodes. The LLMs' performance clearly declines with increasing graph sizes for counting triangles while in the previous task of counting 3-stars, the performance remains fairly consistent across all graph sizes. This decline suggests LLMs' performance diminishes with larger or more complex graph structures. To get a sense of how sensitive it is to graph sizes, the Hard collection consists graphs with 30 nodes while the Easy consists graph with 10 nodes. Still, the LLMs performance remain consistent across all difficulties in counting 3-stars.

6 CONCLUSION

Our experiments show that LLMs can count substructures and even outperform Higher-Order GNNs in counting 3-stars. Advanced prompting techniques, such as least-to-most, significantly improve

Table 5: Performance Metrics Across Different Methods for Counting 3-Stars in 3 Levels of Difficulty. Gray denotes the best result.

Difficulty	Metric	Zero-shot	0-COT	0-COT+1-hop	FS+LTM+1-hop	COT	COT+BAG
Easy	MAE (↓)	49.76	48.44	47.96	1.81	18.54	15.83
	MSE/σ^2 (↓)	60.23	57.35	56.23	0.80	17.79	12.13
	Accuracy (↑) (%)	0.00	0.00	0.00	76.65	4.18	4.80
	Precision (%)	89.27	85.64	88.14	93.19	69.65	63.15
Medium	MAE (↓)	56.05	48.48	53.50	4.69	34.16	26.27
	MSE/σ^2 (↓)	6.69	42.33	6.23	0.12	34.38	21.75
	Accuracy (↑) (%)	0.00	0.00	0.00	24.17	1.02	1.53
	Precision (%)	81.44	81.24	80.81	96.78	66.82	59.63
Hard	MAE (↓)	140.71	48.48	49.28	8.19	97.26	173.83
	MSE/σ^2 (↓)	165046.18	42.33	44.07	11.94	7251.30	95789.06
	Accuracy (↑) (%)	0.08	0.00	0.00	2.63	0.56	1.29
	Precision (%)	69.50	74.18	75.10	98.16	64.90	56.22

Table 6: Performance Metrics Across Different Methods for Counting Triangles in 3 Levels of Difficulty. Gray denotes the best result.

Difficulty	Metric	Zero-shot	0-COT	0-COT+1-hop	FS+LTM+1-hop	COT	COT+BAG
Easy	MAE (↓)	6.05	5.94	5.26	4.07	5.36	6.07
	MSE/σ^2 (↓)	32.43	36.58	28.62	10.15	24.82	31.70
	Accuracy (↑) (%)	8.07	7.84	7.38	7.96	5.97	7.53
	Precision (%)	47.99	46.88	49.03	52.02	41.77	39.97
Medium	MAE (↓)	12.65	12.32	12.21	5.06	11.73	15.91
	MSE/σ^2 (↓)	94.24	64.58	44.73	7.45	42.09	118.00
	Accuracy (↑) (%)	2.84	3.45	3.71	7.51	3.50	3.29
	Precision (%)	21.95	21.67	22.13	29.01	18.48	17.78
Hard	MAE (↓)	27.15	53.68	112.97	16.30	32.22	85.66
	MSE/σ^2 (↓)	778.79	72149.39	272265.99	627.03	3021.48	140615.24
	Accuracy (↑) (%)	0.58	0.38	0.96	7.21	2.40	1.33
	Precision (%)	8.20	8.03	7.84	11.34	6.97	6.70

Table 7: Performance metrics for the task for counting Chordal 4-Cycles. ACC is calculated for the count, while Precision is calculated for the validated substructures. Gray denotes the best result. Slash (/) denotes no available results.

Method	MAE (↓)	MSE/σ^2 (↓)	ACC (↑)	PRE (%) (↑)
Zero-shot	8.31	13.48	2.53	/
0-COT	10.54	197.50	2.70	15.79
0-COT+1-hop	15.19	19264.46	2.97	6.78
FS+LTM+1-hop	2.32	1.53	19.79	11.11
COT	4.98	7.95	17.59	16.61
COT+BAG	5.15	8.24	15.13	16.53
GCN	0.65	0.11	57.76	/
GIN	2.16	0.98	12.16	/
RNPGNN	0.71	0.15	59.80	/

their performance. LLMs also show preliminary ability to identify substructures with precision.

However, LLMs struggle with larger and more complex graphs and particularly counting cycles. They are prone to errors due to spurious correlations. These findings highlight the need for further research to improve LLMs' expressiveness and generalizability in adapting LLMs to counting substructure applications.

REFERENCES

[1] Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M. Bronstein. 2023. Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 1 (2023), 657–668. https://doi.org/10.1109/TPAMI.2022.3154319

[2] 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 (2020), 1877–1901.

[3] Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. 2020. Can Graph Neural Networks Count Substructures? *arXiv* (2020). arXiv:2002.04025 [cs.LG]

[4] Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and Jiliang Tang. 2024. Exploring the Potential of Large Language Models (LLMs) in Learning on Graphs. *arXiv* (2024). arXiv:2307.03393 [cs.LG]

[5] Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. 2023. Long Range Graph Benchmark. arXiv:2206.08164 [cs.LG]

[6] Paul Erdos and Alfred Renyi. 1960. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci., Ser. A* 5 (1960), 17–61. [Reviewer: K. Culik].

[7] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2023. Talk like a Graph: Encoding Graphs for Large Language Models. *arXiv* (2023). arXiv:2310.04560 [cs.LG]

[8] Vikas K. Garg, Stefanie Jegelka, and Tommi Jaakkola. 2020. Generalization and Representational Limits of Graph Neural Networks. arXiv:2002.06157 [cs.LG]

[9] 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* (2023). arXiv:2305.15066 [cs.AI]

[10] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907 [cs.LG]

[11] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=e2TBb5y0yFf

[12] Tiago Sousa, João Correia, Vítor Pereira, and Miguel Rocha. 2021. Generative Deep Learning for Targeted Compound Design. *Journal of Chemical Information and Modeling* 61, 11 (Oct. 2021), 5343–5361. https://doi.org/10.1021/acs.jcim.0c01496 Copyright © 2021 The Authors. Published by American Chemical Society. This publication is licensed under.

[13] A. Steger and N.C. Wormald. 1999. Generating Random Regular Graphs Quickly. *Combinatorics, Probability and Computing* 8, 4 (1999), 377–396. https://doi.org/10.1017/S0963548399003867

[14] Mengying Sun, Sendong Zhao, Coryandar Gilvary, Olivier Elemento, Jiayu Zhou, and Fei Wang. 2020. Graph convolutional networks for computational drug development and discovery. *Briefings in Bioinformatics* 21, 3 (May 2020), 919–935. https://doi.org/10.1093/bib/bbz042

[15] Behrooz Tahmasebi, Derek Lim, and Stefanie Jegelka. 2021. Counting Substructures with Higher-Order Graph Neural Networks: Possibility and Impossibility Results. *arXiv* (2021). arXiv:2012.03174 [cs.LG]

[16] Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2024. Can Language Models Solve Graph Problems in Natural Language? *arXiv* (2024). arXiv:2305.10037 [cs.CL]

[17] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks? arXiv:1810.00826 [cs.LG]

[18] Jiaxuan You, Jonathan Gomes-Selman, Rex Ying, and Jure Leskovec. 2021. Identity-aware Graph Neural Networks. *CoRR* abs/2101.10320 (2021). arXiv:2101.10320 https://arxiv.org/abs/2101.10320

[19] Jiawei Zhang. 2023. Graph-ToolFormer: To Empower LLMs with Graph Reasoning Ability via Prompt Augmented by ChatGPT. arXiv:2304.11116 [cs.AI]

[20] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. arXiv:2205.10625 [cs.AI]