Reasoning with Large Language Models on Graph Tasks: The Influence of Temperature

Wang Yiming*, Zhang Ziyang*, Chen Hanwei*, Shen Huayi†
*School of Information Science and Technology, North China University of Technology, Beijing, China
†School of Big Data and Software, Chongqing University, Beijing, China
YimingWang0619@gmail.com, zhangziyang23123@126.com, { chw123456700, theshy877}@163.com

***Abstract*—Large language models (LLMs) have garnered significant attention due to their impressive performance across various fields. Consequently, numerous researchers are exploring the potential of applying these models to graph problems. However, the effect of the temperature coefficient on graph reasoning within large models remains underexplored. To this end, we investigate the effect of temperature by using NLGraph as a benchmark. We aim to explore the effect of varying the temperature parameter in the discrete range of 0 to 1 on the models' inference performance. The experimental results show that the LLMs' sensitivity to temperature varies across tasks at different difficulty levels. In most cases, the accuracy is higher at moderate temperatures and lower at extreme temperature settings, suggesting that proper temperature tuning can improve inference performance. In addition, the effect of temperature change on accuracy is more significant in the shortest path problem. As the temperature increases, the tendency of the model to explore different solutions increases and the creativity and disorder of the response increases, leading to a decrease in accuracy and causing an increase in the rate of change.**

***Index Terms*—large language models, graph problems, graph reasoning, temperature**

I. Introduction

Recently, LLMs have gained a lot of attention due to their outstanding inference capabilities. Due to the significant expansion of the number of parameters, LLMs exhibit new abilities that did not exist in previous small-size models [1], like contextual understanding and logical reasoning ability. Many scholars have applied LLMs to the real world, such as medical area [2] and scientific research [3].

Despite the remarkable achievements of LLMs in various fields, it still faces challenges and problems in practical applications. One of them is the problem of reasoning about Graph-structure data. Recent work posted by Wang et al. (2024) [4] introduces a benchmark: NLGraph, a comprehensive suite with 29,370 graph reasoning problems across eight categories, formulated in natural language, setting the stage for assessing LLMs' graph problem-solving abilities. Fatemi, Halcrow, and Perozzi (2023) [5] delve into LLM applications in graph reasoning, offering a systematic study on encoding graph data as text and its impact on LLMs performance. However, the effect of the temperature coefficient on graph reasoning within LLMs remains underexplored, with its implications for the adaptability and efficiency of LLMs in graph-based reasoning tasks yet to be fully understood.

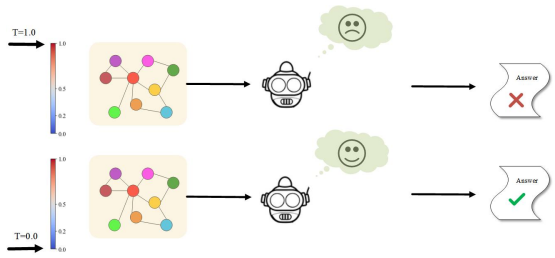

Fig. 1: Overview of our framework.

To address the aforementioned problems, we explore the impact of temperature settings on the reasoning capabilities of LLMs when tackling graph-based algorithmic challenges. Specifically, our experiments employ NLGraph as a benchmark, which serves as a comprehensive testbed for evaluating the graphical and structural reasoning capabilities of language models and their applications in natural language processing. We utilize GPT-3.5-Turbo as the experimental subject and discretize the temperature parameter in the range of 0 to 1. A series of experiments were conducted based on three fundamental graph inference tasks, including cycle detection, shortest path search, and connectivity detection. These experiments aim to elucidate the influence of the temperature parameter on the reasoning performance of LLMs and explore strategies for optimizing the model's graph reasoning capabilities by adjusting the temperature.

To summarize, the contributions of our work can be listed as follows:

- We design a series of experiments to evaluate model performance at different temperature settings. The experiments revealed that temperature sensitivity varies across different tasks and levels. Tasks like shortest path search showed significant fluctuations in accuracy at specific temperatures, while others(cycle detection and connectivity detection) were more stable.
- We observed that moderate temperatures generally yielded higher accuracy, while extreme temperatures led to lower performance. This suggests that adjusting temperature settings can enhance accuracy for some graph tasks like cycle detection and connectivity detection,

particularly for tasks where the model's complexity aligns with its capabilities.
- The impact of temperature variations is notably pronounced in the shortest path problem. At easy level, lower temperatures resulted in higher accuracy due to more certainty and consistency in the generated answer. However, as the temperature increased, creativity and disorganization lifted leading to a decrease in accuracy. Also, at hard level, the complexity of the problem often resulted in low accuracy regardless of temperature.

II. Background

A. Large Language Models

Since the advent of the Turing Test in the 1950s, efforts have been made to endow machines with the ability to grasp the intricacies of language intelligence. The models developed thus far have demonstrated remarkable performance in addressing a broad spectrum of Natural Language Processing (NLP) tasks. A key discovery in this field is the concept of model scaling: enhancing the performance of language models by increasing their size. In this way, they not only show marked performance improvements but also develop unique capabilities, which are absent in smaller-scale models. This observation led to the coining of the term "Large Language Models" to describe models with extraordinarily high numbers of parameters, ranging from tens to hundreds of billions.

Lately, the development and advancements in LLMs have become a focal point. Models such as FLAN (Wei et al., 2022) [6], OPT (Zhang et al., 2022b) [7], and PaLM (Chowdhery et al., 2022) [8], have demonstrated exceptional performance in natural language understanding(NLU) tasks. Among these models, the Generative Pre-trained Transformer (GPT) (Brown et al., 2020) [9]series, marks great progress in this field. OpenAI introduces GPT-3.5, which is based on the decoder component of the transformer architecture. GPT-3.5, while inheriting the architectural foundation of its predecessor, GPT-3, not only maintains the billion-scale parameter count that gave GPT-3 its remarkable capabilities but also introduces notable enhancements in performance and optimization. These improvements allow for more accurate language understanding and smoother text generation, pushing the boundaries of what AI algorithms can achieve in natural language processing.

B. Prompt Engineer

Although large language models, represented by OpenAI's GPT family of models, perform well on general-purpose language processing tasks, their performance may not be satisfactory in domain-specific applications, especially in tasks that require integration with domain knowledge. To overcome this challenge, researchers have begun to explore how large language models can be applied to specific domains to improve the performance of the models on specific tasks. One effective approach is the introduction of Prompt Engineering [9]–[12], a prompt is a specific input cue that guides the model to produce more accurate and targeted output by prompting it for information relevant to a specific task. In many cases, a prompt can consist of text containing a task description, background knowledge, and the desired output format, providing explicit prompts to the model enhances its comprehension of task requirements. The evolution from Discrete text prompts [13] to automatically generating prompts [12], [14], highlights the effectiveness of prompts in enhancing the model's performance on domain-specific tasks.

C. Temperature of LLMs

In neural networks, particularly in those designed for classification tasks, the softmax layer is often the final step, transforming the model's logits into a probability distribution that indicates the probability of each class being the correct one. The softmax function, given by the formula:

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{1}$$

where e is the base of the natural logarithm, ensures that the resultant probabilities add up to one. The temperature parameter T, is a key factor in adjusting this function, altering the smoothness of the output distribution. The temperature-adjusted softmax is defined as:

$$\text{softmax}_T(x_i) = \frac{e^{x_i/T}}{\sum_j e^{x_j/T}} \tag{2}$$

In this formula, T markedly influences the distribution's characteristics. A conventional softmax is realized when $T = 1$, whereas higher temperatures (i.e., $T > 1$) result in flatter, more uncertain distributions. Conversely, lower temperatures (i.e., $T < 1$)lead to sharper distributions, indicating increased model confidence. This mechanism is particularly useful in various scenarios, such as generative tasks that control output creativity, reinforcement learning to balance exploration and exploitation, and calibrating model confidence, especially in models prone to overconfidence. The temperature parameter thus plays a crucial role in fine-tuning large models, impacting their confidence levels, exploration tendencies, and the diversity of their outputs.

D. Natural Language Graph

Natural Language Graph (NLGraph) [4]is designed to assess the capabilities of LLMs for reasoning about graph structures. Utilizing a random graph generator, it constructs a base graph for each task, wherein the number of nodes and graph density serve as parameters to modulate complexity. On this basis, graph inference tasks of varying difficulty levels are designed, encompassing a broad spectrum of challenges, from connectivity and cycle detection to shortest path determination and graph neural network simulation. For each task, the base graph is adapted or modified, and corresponding queries are crafted to align with the specific requirements of the task These tasks are categorized into three levels of difficulty: easy, medium, and hard. The NLGraph benchmark consists of

a standard version (5902 problems) and an extended version (29,370 problems) that uses accuracy and partial score metrics to evaluate model performance.

III. Methodology

A. Prompt Setting

1) How to convert a graph into natural language: To convert a graph into natural language, we describe the problem setting in natural language, adjust or edit the base graph for each task, and design corresponding queries. For instance, the nodes are encoded as integers, and edges are represented by pairs of integers in parentheses. For example, if there is an edge between nodes 0 and 1, it can be represented as (0,1). So we can describe a graph like this: "In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge. The nodes are numbered from 0 to 5, and the edges are:(3,4) (3,5) (1,0) (2,5) (2,0). Question: Is there a cycle in this graph?"

2) Chain of Thought: Chain of Thought (CoT) [8]is added for guiding LLMs to generate coherent chains of reasoning, designed to enhance model performance in solving complex tasks. CoT prompts encourage models to think step-by-step and explain their reasoning processes. [16]In simple graph reasoning tasks, the application of CoT can significantly improve model performance. Specifically, in the task of determining whether there is a cycle in a graph, the model needs to check whether there exists a path in a given graph such that the start and end points of the path are the same. By employing CoT prompts, the model is directed to generate an inference chain describing how it progressively checks the edges and nodes in the graph to determine whether such a path exists. This reasoning chain can include steps such as checking whether each node has been visited multiple times, or whether an edge connects a node back to a node that has been visited. Through this step-by-step reasoning process, the model is able to determine more accurately whether a cycle exists in the graph. In this way, the model not only needs to consider the relationship between nodes and edges in the graph but also needs to perform step-by-step reasoning based on the CoT hints to determine whether a ring exists or not. The CoT hints not only improve the model's accuracy on the connectivity task but also enhance the model's interpretativeness so that the researchers and the users can better understand the model's reasoning process. This approach shows potential in some simple graph inference tasks, but its effectiveness may be limited when dealing with more complex graph inference problems.

B. Tasks

Leveraging a random graph generator, we craft graphs for three distinct graph reasoning tasks, each varying in complexity. The process initiates with the creation of base graphs, categorized as easy, medium, or hard, by adjusting the number of nodes and graph sparsity. Subsequently, these graphs are refined and adapted for individual tasks, serving as the groundwork for the formulation of queries.

· **Task 1: Connectivity** In an undirected graph $G = \{V, E\}$, nodes u and $v$are considered connected if there exists a sequence of edges set E that links u to v. To assess the connectivity properties of graph G, we randomly select pairs of nodes u, v from the node-set V and investigate whether these two nodes are connected. This inquiry is formalized as a boolean query, a true/false question, to determine if a path exists from u to v. Through this approach, we systematically evaluate and understand the connectivity structure of undirected graphs. We ensure that the set of questions is balanced: half of the questions are about connected node pairs, and the other half are about disconnected node pairs. To achieve this balance, we discard some extra questions.

· **Task 2: Cycle** In an undirected graph $G = \{V, E\}$, a cycle is defined as a non-empty path consisting of a sequence of edges $(e_1, e_2, \ldots, e_n)$ and the corresponding sequence of nodes$(v_1, v_2, \ldots, v_n, v_1)$, where the starting and ending node is the same. To investigate the presence of cycles in graph G,, we analyze the given undirected graph and pose true/false questions to determine whether at least one such cycle exists in the graph. This approach aims to deeply understand the cyclic structural features of the graph, evaluating how nodes and edges form closed paths. We use base graphs without cycles as the False subset and randomly add edges to these base graphs to generate graphs with cycles as the True subset. We ensure that the dataset is balanced with an equal number of cyclic and non-cyclic graphs.

· **Task 3: Shortest Path** The shortest path problem involves finding a path between two nodes in a graph where the sum of the edge weights is minimized. In the given undirected graph $G = \{V, E\}$ with a positive weight w assigned to each edge, the objective is to determine the shortest path between two specific nodes u and v and its path length. To increase the difficulty level, we filter the generated base graphs to ensure that the number of nodes on the correct shortest path reaches a predefined threshold, varying within the range from 'min' to 'max' for each question as an additional measure of difficulty control.

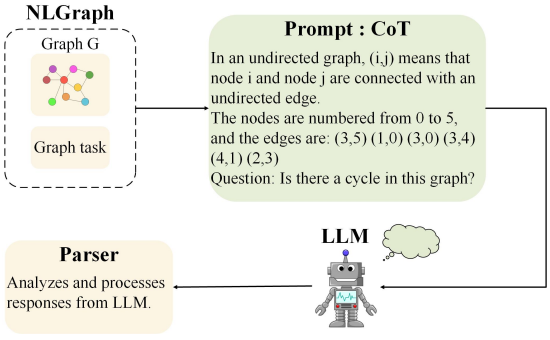

Fig. 3: The pipeline of our experiment

C. Pipeline

With the input of the NLGraph, we commence by converting the graph into natural language, including the description

TABLE I: This table presents the experimental results of the graph reasoning tasks at different temperature settings (0, 0.2, 0.5, and 1). The tasks are categorized into Cycle detection, Shortest Path, and Connectivity detection, with each having different levels (Easy, Medium, and Hard). The table shows the accuracy for each task and level combination.

Task	Level	Temperature			
		0	0.2	0.5	1
Cycle	Easy	0.513	0.520 [+0.07]	0.513 [+0.00]	0.473 [-0.04]
	Medium	0.610	0.573 [-0.037]	0.567 [-0.043]	0.558 [-0.052]
	Hard	0.538	0.523 [-0.015]	0.495 [-0.043]	0.518 [-0.02]
Shortest Path	Easy	0.417	0.606 [+0.189]	0.633 [+0.216]	0.600 [+0.183]
	Hard	0.200	0.220 [+0.02]	0.180 [-0.02]	0.165 [-0.035]
Connectivity	Easy	0.760	0.778 [+0.018]	0.761 [+0.001]	0.781 [+0.021]
	Medium	0.660	0.673 [+0.013]	0.686 [+0.026]	0.701 [+0.041]
	Hard	0.608	0.597 [-0.011]	0.612 [+0.004]	0.641 [+0.033]

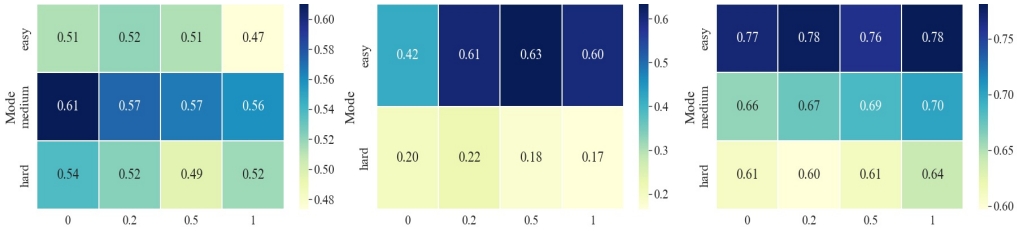

Fig. 2: The heatmap visualizes the accuracy metrics across three distinct tasks under varying difficulty levels.

of the graph and the formulation of the task. And we use the CoT prompts to guide the LLMs through the reasoning. Temperature parameters are set discretely in the range of 0 to 1 to appraise their consequential effects on the LLMs' performance. For each graph inference task, the LLMs is asked to respond to reflect its understanding and reasoning about the problem posed. The LLMs' tasks vary in complexity from simple cycle detection to complex shortest path determination, and the LLMs' outputs are then parsed and analyzed to assess the accuracy and consistency of the solutions provided. This process allows us to accurately assess how temperature settings affect the inference capabilities of the LLMs in the area of graph understanding and reasoning.

D. Experimental settings

We use GPT-turbo-3.5 as the default large language model due to its notable performance on various tasks. For all graph tasks, The temperature parameter is discretely adjusted within the range of 0 to 1, specifically at values 0, 0.2, 0.5, and 1, to explore its impact on performance. For the CoT, the input prompts consist of k examples selected from previous extended versions of the problem in question. For the connectivity task and cycle task, we set k to 4, for the shortest task, we set k to 5 due to the context size limit. The token length for all tasks is uniformly set at 1200.

IV. Result

The experimental results show that there is a differential sensitivity to temperature across various tasks, contingent upon their assigned difficulty levels. In cycling detection, the medium difficulty level was very sensitive to temperature changes, with accuracy decreasing with increasing temperature. Contrastingly, the easy and hard levels appeared to be less affected by temperature changes. For the shortest path task, optimal performance in the easy level was predominantly observed at moderate thermal settings (0.2 and 0.5), with a decrement at the extremes of the temperature spectrum (0 and 1). The accuracy for the hard level peaked at a temperature of 0.2, indicating some sensitivity to temperature changes, but overall there was less variation across temperatures. For the connectivity task, as the temperature increases, there is an overall trend of improved accuracy, most notably at the easy level. The model's peak performance suggests that moderate to higher temperatures facilitate better performance for less complex tasks. At the hard level, the model's performance is relatively stable across different temperatures, indicating that the task's complexity might overshadow the influence of temperature settings. These emergent trends suggest that while model performance on easy tasks can be optimized over a range of temperatures, temperature changes have little effect on more challenging scenarios. This may be due to the inherent difficulty of the task exceeding the model's capabilities. Furthermore, this reinforces the notion that the accuracy of the model and its propensity to explore different solutions can be affected by temperature, especially when the

complexity of the task is within the model's capabilities.

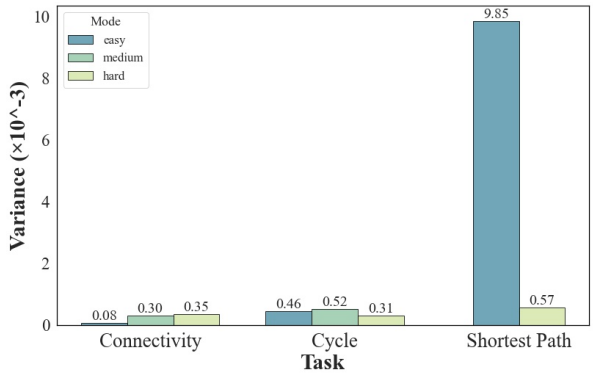

Fig. 4: The bar graph illustrates the variance in performance across three different tasks: Connectivity, Cycle, and Shortest Path, stratified by difficulty level.

In the Shortest Path problem, we observed a wide variation in accuracy in the easy level, while the accuracy in the difficult level is more consistent. This phenomenon can be attributed to the inherent simplicity of the task at the easy level, where the model can quickly identify the correct solution at lower temperature settings, resulting in a higher accuracy. However, as the temperature increases, the model's tendency to explore different solutions increases, which may increase the creativity and disorganization of the responses, thereby decreasing accuracy and introducing greater variability. Conversely, at the difficult level, the complexity of the task may exceed the processing capacity of the model, leading to generally lower accuracy, which is less affected by changes in temperature parameters. In essence, unlike binary classification tasks such as loop detection or connectivity, the shortest path task requires more than a simple "yes" or "no" answer; it requires the precise determination of the sequence of nodes that constitute the optimal path. Any deviation from the optimal path, no matter how small, may result in an incorrect answer, even if the model is close to the correct solution.

V. Conclusion

In this work, we investigate the effect of temperature on the reasoning ability of LLMs in addressing graph problems. Experimental results show differences in sensitivity to temperature across tasks of differing complexity levels. Notably, in most scenarios, the accuracy tends to be higher at moderate temperatures and diminishes at extreme temperatures, adjusting the temperature appropriately can improve inference performance. Additionally, the influence of temperature variations on the accuracy of the shortest path problem is more pronounced across different difficulty levels. As the temperature increased, the model increased its tendency to explore different solutions, and the creativity and disorder of the responses increased. Concurrently, the complexity of the problem intensifies, resulting in a decrease in accuracy and a subsequent rise in variance. Future research could further explore the effects of temperature parameters and other factors on the reasoning ability of LLMs on graph problems, with a focus on optimizing model settings to enhance their performance in such tasks.

References

[1] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.

[2] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, D. S. W. Ting, "Large language models in medicine," *Nature medicine*, vol. 29, no. 8, pp. 1930–1940, 2023, Nature Publishing Group US New York.

[3] H. Wang, T. Fu, Y. Du, W. Gao, K. Huang, Z. Liu, P. Chandak, S. Liu, P. Van Katwyk, A. Deac, *et al.*, "Scientific discovery in the age of artificial intelligence," *Nature*, vol. 620, no. 7972, pp. 47–60, 2023, Nature Publishing Group UK London.

[4] H. Wang, S. Feng, T. He, Z. Tan, X. Han, Y. Tsvetkov, "Can language models solve graph problems in natural language?," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[5] B. Fatemi, J. Halcrow, B. Perozzi, "Talk like a graph: Encoding graphs for large language models," *arXiv preprint arXiv:2310.04560*, 2023.

[6] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, Q. V. Le, "Finetuned language models are zero-shot learners," *arXiv preprint arXiv:2109.01652*, 2021.

[7] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, *et al.*, "Opt: Open pre-trained transformer language models," *arXiv preprint arXiv:2205.01068*, 2022.

[8] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[10] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen, *et al.*, "Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models," *arXiv preprint arXiv:2203.06904*, 2022.

[11] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023.

[12] T. Schick, H. Schmid, and H. Schütze, "Automatically identifying words that can serve as labels for few-shot text classification," *arXiv preprint arXiv:2010.13641*, 2020.

[13] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing, "Toward controlled generation of text," in *International Conference on Machine Learning*, pp. 1587–1596, 2017.

[14] T. Schick and H. Schütze, "Exploiting cloze questions for few-shot text classification and natural language inference," *arXiv preprint arXiv:2001.07676*, 2020.

[15] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al., "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824–24837, 2022.

[16] S. Narang, C. Raffel, K. Lee, A. Roberts, N. Fiedel, and K. Malkan, "Wt5?! Training text-to-text models to explain their predictions," *arXiv preprint arXiv:2004.14546*, 2020.