Supplementary Materials for `GraphEval2000`

Paper ID: 1998

1 Datasheet for `GraphEval2000`

To better help users understand our dataset, we follow the questions in [Gebru et al., 2021] to create a datasheet.

1.1 Motivation

For what purpose was the dataset created? Was there a specific task in mind? Was there a specific gap that needed to be filled? Please provide a description.

This dataset is designed to evaluate and enhance the reasoning capabilities of large language models (LLMs) in addressing graph-related problems. By providing LLMs with materials such as problem statements, data examples, and code frameworks, they can generate complete code solutions. Researchers can then assess the accuracy and effectiveness of these solutions using our proposed dataset.

Who created the dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?

The dataset is created by Qiming Wu, Zichen Chen, Will Corcoran, Misha Sra and Ambuj Singh in the University of California, Santa Barbara.

Who funded the creation of the dataset? If there is an associated grant, please provide the name of the grantor and the grant name and number.

Ambuj Singh.

1.2 Composition

What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)? Are there multiple types of instances (e.g., movies, users, and ratings; people and interactions between them; nodes and edges)? Please provide a description.

The instances are graph data structure problems and graph test cases. We extract the problems from LeetCode community (`https://LeetCode.com/tag/graph/`) and generate graph test cases by python programming.

How many instances are there in total (of each type, if appropriate)?

There are 2850 graph test cases in our dataset totally. Specifically, there are 1110 directed graph samples and 1740 undirected ones.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please

Submitted to the 38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks. Do not distribute.

describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable).

GraphEval2000 contains all possible instances. However, users can scale this dataset by generating more graph test cases based on our provided code.

Is there a label or target associated with each instance? If so, please provide a description.

Yes. Each graph instance are associated with a label, which is the correct answer of that graph problem. This label is contained in the dataset.

Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? If so, please describe how these relationships are made explicit.

Yes. One instance of GraphEval2000 contains the following contents: Problem Statement, Data Examples, Code Framework, and graph test cases. The first three materials give an overview of the data structure problem to the LLMs, and graph test cases are used to evaluate the generated code.

Are there recommended data splits (e.g., training, development/validation, testing)? If so, please provide a description of these splits, explaining the rationale behind them.

Not applicable. Our dataset is mainly used in the inference phase of LLMs, it does not relate to the training phase.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? If so, please describe why

No. The problem materials are all available at the LeetCode community (https://LeetCode.com/tag/graph/) and the graph test cases are presented in the way of python list.

Does the dataset identify any subpopulations (e.g., by age, gender)? If so, please describe how these subpopulations are identified and provide a description of their respective distributions within the dataset.

No.

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals race or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)? If so, please provide a description.

No.

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor–patient confidentiality, data that includes the content of individuals' nonpublic communications)? If so, please provide a description.

No.

1.3 Collection Process

How was the data associated with each instance acquired? Was the data directly observable (e.g., raw text, movie ratings), reported by subjects (e.g., survey responses), or indirectly inferred/derived from other data (e.g., part-of-speech tags, model-based guesses for age or language)? If the data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified? If so, please describe how.

We first picked 40 graph data structure problems form the LeetCode Community (https://LeetCode.com/tag/graph/). We collect the problem statement, data examples, constraints, and code framework and store them in the jsonl files. After that, we use NetworkX (https://networkx.org/) to generate graph test cases for each of these problems.

What mechanisms or procedures were used to collect the data (e.g., hardware apparatuses or sensors, manual human curation, software programs, software APIs)? How were these mechanisms or procedures validated?

We collect the problem materials by hand, that is, directly copying and pasting the contents from the website. And for the graph test cases, we use python programming to generate them, the code is available at this webpage (`https://harrywuhust2022.github.io/GraphEval2000/`).

Over what timeframe was the data collected? Does this timeframe match the creation timeframe of the data associated with the instances (e.g., recent crawl of old news articles)? If not, please describe the timeframe in which the data associated with the instances was created.

We collect the problem materials in April 2024. At that time, all these selected problems are already available at LeetCode (`https://LeetCode.com/tag/graph/`). After we finish the data collections, we created the graph test cases.

Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)?

Qiming Wu collected the directed graph problems and Will Corcoran collected the undirected graph problems. They both funded by Ambuj Singh.

Were any ethical review processes conducted (e.g., by an institutional review board)? If so, please provide a description of these review processes, including the outcomes, as well as a link or other access point to any supporting documentation.

Unknown to the authors of the datasheet.

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)?

We collect the problem materials from the LeetCode (`https://LeetCode.com/tag/graph/`), which is a public platform that provides data structure problems.

Has an analysis of the potential impact of the dataset and its use on data subjects (e.g., a data protection impact analysis) been conducted? If so, please provide a description of this analysis, including the outcomes, as well as a link or other access point to any supporting documentation.

Not applicable.

1.4 Preprocessing/cleaning/labeling

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? If so, please provide a description. If not, you may skip the remaining questions in this section.

We mainly select the medium- and hard-level problems on LeetCode website becase the easy-level problems can be easily solved by LLMs. And then, we use the solution code of the select problem to generate labels of graph test cases. These codes are verified by the LeetCode contests.

Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)? If so, please provide a link or other access point to the "raw" data.

Yes, the dataset itself stores the raw data.

Is the software that was used to preprocess/clean/label the data available? If so, please provide a link or other access point.

Yes, it is included in our code, which is available at here (`https://harrywuhust2022.github.io/GraphEval2000/`).

1.5 Uses

Has the dataset been used for any tasks already? If so, please provide a description.

No.

Is there a repository that links to any or all papers or systems that use the dataset? If so, please provide a link or other access point.

No.

What (other) tasks could the dataset be used for?

`GraphEval2000` is designed to evaluate and improve the reasoning ability of LLMs. It can also be used to train the LLMs to better understand the graph structures.

Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses? For example, is there anything that a dataset consumer might need to know to avoid uses that could result in unfair treatment of individuals or groups (e.g., stereotyping, quality of service issues) or other risks or harms (e.g., legal risks, financial harms)? If so, please provide a description. Is there anything a dataset consumer could do to mitigate these risks or harms?

N/A.

Are there tasks for which the dataset should not be used? If so, please provide a description.

Unknown to the authors of the dataset.

1.6 Distribution

Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created? If so, please provide a description.

Yes, the dataset is available at the Internet.

How will the dataset will be distributed (e.g., tarball on website, API, GitHub)? Does the dataset have a digital object identifier (DOI)?

Users can download the dataset at the webpage here (`https://harrywuhust2022.github.io/GraphEval2000/`)

Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)? If so, please describe this license and/or ToU, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms or ToU, as well as any fees associated with these restrictions.

Yes, this dataset is licensed under a *CC BY 4.0 license*, see official instructions at `https://creativecommons.org/licenses/by/4.0/`).

Have any third parties imposed IP-based or other restrictions on the data associated with the instances? If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms, as well as any fees associated with these restrictions.

The problem materials are provided by the LeetCode, therefore, any commercial usage of this dataset is prohibited.

1.7 Maintenance

Who will be supporting/hosting/maintaining the dataset?

Qiming Wu, Will Corcoran and Zichen Chen will maintain the dataset.

How can the owner/curator/manager of the dataset be contacted (e.g., email address)?

The contact information of authors are available at the Internet.

Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)? If so, please describe how often, by whom, and how updates will be communicated to dataset consumers (e.g., mailing list, GitHub)?

Yes. The updated information will be posted at the webpage here (`https://harrywuhust2022.github.io/GraphEval2000/`).

If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were the individuals in question told that their data would be retained for a fixed period of time and then deleted)? If so, please describe these limits and explain how they will be enforced.

N/A.

Will older versions of the dataset continue to be supported/hosted/maintained? If so, please describe how. If not, please describe how its obsolescence will be communicated to dataset consumers.

Yes, all versions of the dataset is stored at the webpage here (`https://harrywuhust2022.github.io/GraphEval2000/`).

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so? If so, please provide a description.

Follow the Github regulations.

2 Analysis of Our Dataset `GraphEval2000`

In this section, we analyze our dataset from several perspectives: data distributions, the complexity of graph test cases, and the time and memory usage during evaluations on `GraphEval2000`.

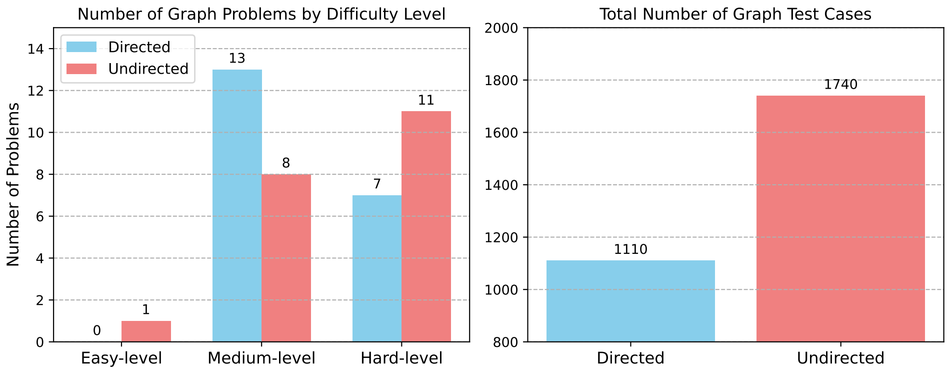


Figure 1: Distribution of graph problems on difficulty levels and the number of graph test cases in our dataset. We count the number of problems on different difficulty levels (easy-, medium- and hard-levels, which are defined by LeetCode) in the left part.We also count the number of test cases in directed and undirected graphs in the right part.

Distributions of Data in `GraphEval2000` We begin by counting the number of graph test cases, as shown on the right side of Figure 1. Additionally, we examine the distribution of problems across different difficulty levels, as illustrated on the left side of Figure 1. Figure 1 summarizes these results. Our dataset, `GraphEval2000`, includes four main graph categories (as shown in Figure 2, sparse, planar, complete, and regular) and four subcategories (connected, disconnected, cyclic, and acyclic) within each main category. For each subcategory, we generated ten graph test cases. Due to the specific characteristics of the main categories, our final dataset comprises 1,110 graph samples for directed graphs and 1,740 for undirected graphs. These samples are designed to evaluate and improve the graph reasoning abilities of large language models (LLMs).

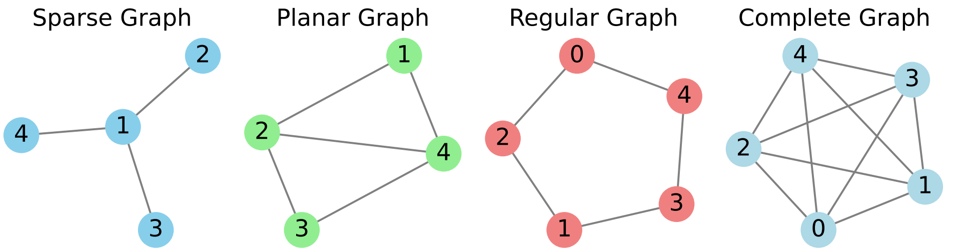


Figure 2: Graph Samples. We use NetworkX (`https://networkx.org/`) to plot the graph samples of four main categories in our dataset.

The graph problems were collected from LeetCode (`https://leetcode.com/tag/graph/`), and we calculated the distribution of problems across different difficulty levels[1]. Given the near-perfect performance of LLMs on easy-level data structure problems, we focused primarily on medium- and hard-level problems. An interesting finding is that directed graph problems tend to be more concentrated at the medium difficulty level, whereas undirected graph problems are more concentrated at the hard difficulty level.

Table 1: Complexity Analysis of Graphs. We include the average number of nodes and edges, average degree, and average degree variance to analyze the graph test cases in `GraphEval2000`.

Graph Type	**Average Number of Nodes**	**Average Number of Edges**	**Average Degree**	**Average Degree Variance**
Directed Sparse	98.63	247.83	4.69	3.65
Directed Planar	41.67	44.68	2.31	1.25
Undirected Sparse	88.55	89.47	1.96	0.72
Undirected Planar	68.67	69.75	1.97	0.72
Undirected Regular	91.91	258.57	5.53	0.55
Undirected Complete	92.13	6845.71	91.13	0.0

Complexity Analysis of Graphs in `GraphEval2000` To analyze the complexity of the graph test cases in `GraphEval2000`, we include several key metrics: the average number of nodes and edges, the average degree, and the average degree variance. These metrics provide a comprehensive understanding of the structural characteristics of the graphs in our dataset, allowing for a more detailed assessment of the complexity and diversity of the test cases. The results are summarized in Table 1.

To begin with, we first introduce the evaluation metrics. Consider N_i to be number of nodes in the i-th graph, E_i represents the number of edges in the i-th graph. Then, average number of nodes $\overline{N}$ across m graph test cases is

$$\overline{N} = \frac{1}{m}\sum_{i=1}^{m} N_i, \tag{1}$$

and average number of edges $\overline{E}$ across m graph test cases is

$$\overline{E} = \frac{1}{m}\sum_{i=1}^{m} E_i. \tag{2}$$

[1] The difficulty level is defined by LeetCode.

The degree of a node in a graph is the number of edges connected to it. The average degree of a graph is the mean degree of all its nodes. For an undirected graph with N nodes and E edges, the average degree $\overline{d}$ is

$$\overline{d} = \frac{2E}{N}. \tag{3}$$

For a directed graph, the average degree is

$$\overline{d} = \frac{E}{N}. \tag{4}$$

Across multiple graphs, the average degree is the mean of the average degrees of each graph. The degree variance measures the variability of the degrees of the nodes in a graph. For a graph with N nodes and degrees $d_1, d_2, \ldots, d_N$, the variance of the degrees is calculated as:

$$\text{Var}(d) = \frac{1}{N}\sum_{i=1}^{N}(d_i - \overline{d})^2 \tag{5}$$

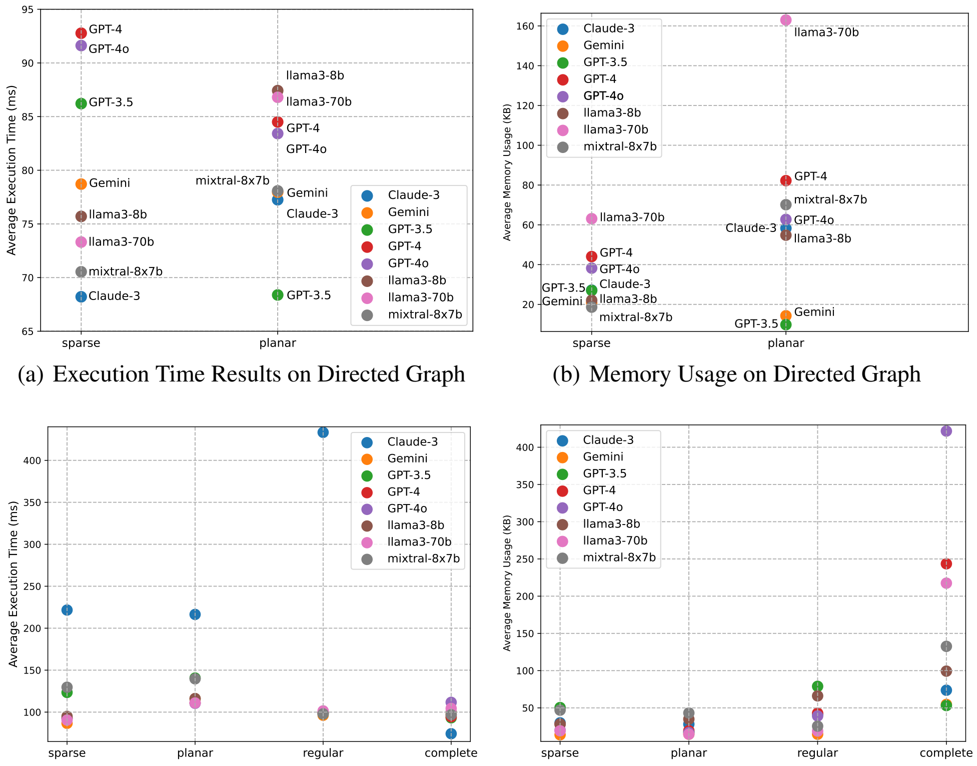
Here, $\overline{d}$ is the average degree of the nodes in that graph. Across multiple graphs, the average degree variance is the mean of the degree variances of each graph.

Summary of Analysis In our dataset, the directed graph has two categories: sparse and planar. The undirected graph has four categories: sparse, planar, regular and complete. We summarized the experimental results based on these categories:

- Directed sparse graphs have a moderate number of nodes and edges. The average degree is relatively low, indicating sparsity. The degree variance suggests variability in node connectivity.
- Directed planar graphs have fewer nodes and edges compared to sparse graphs. The average degree and variance are low, indicating even sparser connectivity with less variation in node degree.
- Undirected sparse graphs have a similar number of nodes to directed sparse graphs but significantly fewer edges. The average degree is just below 2, with low degree variance, indicating very sparse and consistent connectivity.
- Undirected planar graphs are similar to undirected sparse graphs in terms of average degree and degree variance but have fewer nodes and edges. They maintain sparse and consistent connectivity.
- Undirected regular graphs have a higher average number of edges and degree compared to sparse and planar graphs, indicating denser connectivity. The low degree variance suggests uniformity in node connectivity.
- Undirected complete graphs have a very high number of edges, as expected. Each node is connected to every other node, resulting in the maximum possible average degree for the given number of nodes. The degree variance is zero, indicating perfect uniformity in connectivity.

Time and Memory Usage Analysis In addition to evaluating the passing rates of LLMs on our dataset, we also present the time and memory usage data, summarized in Figure 3. These metrics are crucial for assessing the efficiency of the code generated by LLMs.

Results on Directed Graphs For directed graphs, GPT-3.5 stands out as the most efficient model, demonstrating the lowest average execution times and memory usage across planar and sparse graphs. Claude-3 also performs well, particularly for sparse graphs, achieving the lowest execution time. Gemini and mixtral-8x7b show good memory efficiency, with mixtral-8x7b being particularly efficient for sparse graphs. Conversely, llama3-70b consistently exhibits the highest memory usage, making it the least efficient in terms of memory management. Although GPT-4 is powerful, it has higher execution times and memory usage compared to GPT-3.5. Overall, GPT-3.5 proves to be the most balanced and efficient model across the tested scenarios, while llama3-70b demonstrates significant inefficiencies, especially in memory usage.



(a) Execution Time Results on Directed Graph

(b) Memory Usage on Directed Graph

(c) Execution Time Results on Undirected Graph

(d) Memory Usage on Undirected Graph

Figure 3: Time and Memory usage results of LLMs on `GraphEval2000`.

Results on Undirected Graphs The performance of models on undirected graphs varies significantly across different graph types. Claude-3 displays notably high execution times for planar, sparse, and regular graphs but performs exceptionally well with complete graphs, showing the lowest execution time albeit with relatively higher memory usage. Gemini demonstrates consistently low memory usage and good execution times, particularly excelling with sparse graphs. GPT-3.5 and GPT-4 show balanced performance, with GPT-3.5 achieving better execution times for planar graphs and GPT-4 managing memory more efficiently except for complete graphs, where it uses significantly more memory. GPT-4o is similar to GPT-4 but with slightly improved times and memory efficiency. Llama3-8b and llama3-70b exhibit higher memory usage, especially for complete graphs, with llama3-70b showing the highest memory usage for complete graphs. Mixtral-8x7b presents high execution times and memory usage across all graph types but performs moderately well with regular graphs. Overall, Gemini and GPT-3.5 emerge as the most efficient models in terms of execution time and memory usage across various graph types, while Claude-3 and GPT-4 show significant variations based on the graph type.

It is important to note that high efficiency in code execution does not always guarantee a high passing rate. Occasionally, code may produce incorrect answers, which can result in accelerated program execution.

3 Ref

References

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12): 86–92, 2021.