

# LARGE LANGUAGE MODELS AS TOPOLOGICAL THINKERS: A BENCHMARK ON GRAPH PERSISTENT HOMOLOGY

**Anonymous authors**

Paper under double-blind review

## ABSTRACT

Persistent homology offers a principled way to capture multi-scale topological structures in graphs, yet it remains unclear whether large language models (LLMs) can understand and reason about such high-order topological concepts. To address this gap, we introduce LLM4PH, the first benchmark designed to evaluate the ability of LLMs to comprehend and apply persistent homology on graphs. Our benchmark decomposes the persistent homology pipeline into four progressively challenging task levels, ranging from simplicial structure understanding to real-world graph inference. It includes 9 sub-tasks spanning 3 synthetic graph sizes and 3 real-world graph datasets, each annotated with topological features such as connected components, simplices, filtrations, and persistence diagrams. We systematically assess LLMs’ capabilities in recognizing topological features, reasoning over filtrations, designing filtration strategies, and applying persistent homology for classification. Beyond task-level evaluation, we perform cross-task ablations on prompt encoding and transfer, explore post-training effects, and construct a compositional PH pipeline to assess end-to-end performance. Our results provide the first in-depth view of how well LLMs bridge discrete graph structures with continuous topological abstraction, and offer insights into their potential for structure-aware scientific reasoning.

## 1 INTRODUCTION

Large Language Models (LLMs) have demonstrated remarkable performance across a wide range of natural language understanding and reasoning tasks (Chang et al., 2024; Hadi et al., 2023; Plaat et al., 2024). However, their capabilities on graph-structured data, common in real-world, non-Euclidean domains, remain underexplored. Recent efforts have introduced benchmarks based on classical graph-theoretic problems such as shortest paths and connectivity, achieving notable progress on both static graphs (Fatemi et al., 2024; Wang et al., 2023) and dynamic graphs (Zhang et al., 2024). These benchmarks primarily focus on local structural reasoning and rely on standard node-edge representations. However, many real-world graphs exhibit complex high-order structures that go beyond pairwise interactions and require a more global topological perspective (Bick et al., 2023; Bianconi, 2021).

High-order structures refer to non-binary relationships among multiple nodes in a graph, such as communities in social networks or clusters in protein interaction networks. Some recent studies model these structures using graph patterns or hypergraphs and explore the use of LLMs for tasks like pattern matching (Dai et al., 2025; Feng et al., 2025). However, these representations often lack a natural sense of scale or continuity, making it difficult to analyze how structures evolve. Persistent homology (PH), a method from topological data analysis, offers an alternative by capturing multi-scale topological features in graphs (Pun et al., 2022). The process begins by assigning a filtration function, such as edge weights, which determines the order in which edges or nodes are added. PH then builds a sequence of simplicial complexes from this filtration and tracks the birth and death of topological features, resulting in a persistence diagram that summarizes their lifespan. This technique has been successfully applied in domains such as social network analysis, neuroscience, molecular biology, and materials science (Chen et al., 2022; Ye et al., 2023; Li et al., 2024; Curto & Sanderson, 2025; Obayashi et al., 2022).

For example, in a social network where edge weights reflect communication frequency, a filtration can be constructed by adding edges from the weakest to the strongest ties. As shown in Figure 1, graphs  $G_a$  and  $G_b$  differ only in the weight of a single edge, yet their topological structures evolve differently. Specifically, graph  $G_a$  maintains a single connected component throughout the filtration, while graph  $G_b$  initially has two separate components that eventually merge into one. PH captures these structural differences concisely, producing barcodes that summarize when components appear, persist, and merge across different interaction thresholds. This helps distinguish stable community structures from transient interactions and noise. For more detailed explanation of PH, please refer to Appendix C.

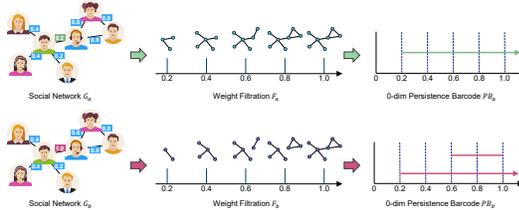


Figure 1: Social network filtration example.

Despite recent progress in benchmarking LLMs on graph-related tasks, many challenges remain in enabling language models to reason about graph data from a persistent homology (PH) perspective. From the graph side, this involves not just identifying structural patterns, but reasoning over filtration functions, edge activation thresholds, and multi-scale substructures that emerge over time. From the topological side, it demands understanding abstract mathematical notions such as simplicial complexes and homological features like connected components and cycles, and how these evolve across a sequence of filtered graphs. A model must not only understand the static shape of a graph but also track its dynamic topological evolution under varying filtrations.

To address these challenges, we introduce **LLM4PH**, a new benchmark designed to evaluate the capability of LLMs to understand, reason about, and apply PH to graph data. The benchmark decomposes the full PH workflow into interpretable sub-tasks, ranging from simple concepts like component counting and barcode reading to more complex reasoning such as filtration design, multi-step composition, and persistence-based graph comparison. We further include terminology ablation, compositional pipelines, and real-world graph settings to examine whether LLMs genuinely understand topological principles or rely on shallow pattern matching. Importantly, we also assess whether LLMs can serve as practical *assistants* in topological data analysis workflows—for example, by interpreting diagrams, selecting filtrations, or comparing barcodes in hybrid neuro-symbolic settings.

Through this structured evaluation, LLM4PH offers the first comprehensive testbed for investigating how well language models connect discrete graph structures with symbolic, algebraic, and topological abstraction.

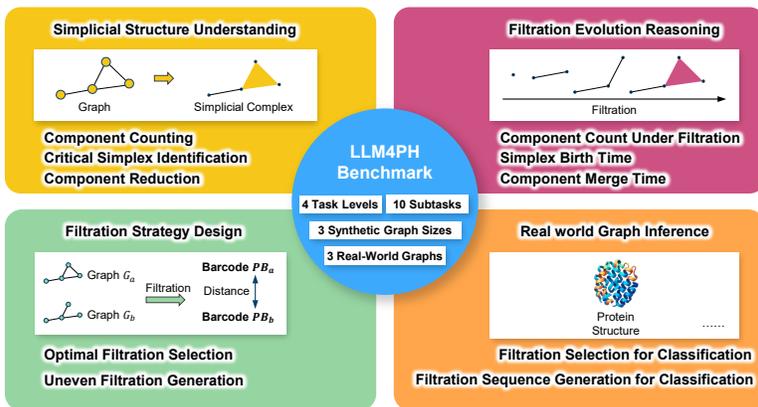


Figure 2: Benchmark Design.

## 2 BENCHMARK DESIGN

As shown in Figure 2, our benchmark is divided into four task categories, progressing from simple to complex with increasing difficulty. These tasks range from topological feature recognition to complex filtration strategy generation, and finally to practical applications, establishing a comprehensive

108 evaluation path from fundamentals to advanced concepts. The specific task designs and benchmark  
 109 statistics are as follows:

## 110 2.1 TASK DESIGN

111  
 112 To rigorously assess the capability of LLMs in understanding PH on graphs, we design a benchmark  
 113 composed of four progressively challenging task categories. In this benchmark, we employ the  
 114 Vietoris–Rips (VR) complex, widely used in related research, as the simplicial complex constructed  
 115 from graphs. For a comprehensive explanation of our task design methodology, see Appendix D.  
 116

117  
 118 **Simple Tasks (Simplicial Structure Understanding).** These tasks examine whether LLMs can  
 119 recognize topological features such as connected components and critical simplices before the  
 120 filtration. Tasks are evaluated by accuracy.

121 From the perspective of PH, these tasks correspond to the input stage before filtration begins, where  
 122 LLM must identify the fundamental simplicial elements (e.g., 0-simplices and 1-simplices) that will  
 123 later form the basis of the topological space. Accurate understanding at this level is essential for  
 124 constructing suitable simplicial complexes.

- 125 • *0D Component Counting.* Counts the number of connected components (0-dimensional homology  
 126 classes).
- 127 • *1D Simplex Counting.* Counts the number of 1-simplices (edges).
- 128 • *Component Reduction.* Modifies one edge in a given graph to reduce the number of connected  
 129 components.
- 130 • *Persistence Diagram Interpretation.* Evaluate whether LLMs can correctly parse a PD represented  
 131 in text and perform basic logical operations.
- 132 • *Topological Distance Reasoning.* Test whether LLMs can infer topological similarity between  
 133 diagrams by reasoning over Wasserstein distances.

134  
 135 **Medium Tasks (Filtration Evolution Reasoning).** These tasks assess whether LLMs can reason  
 136 about births and deaths of topological features. Tasks are evaluated by accuracy.

137 In the context of PH, these tasks correspond to tracking the dynamic evolution of topological features  
 138 as the filtration progresses. The goal is to test whether LLMs can associate filtration steps with  
 139 topological events such as component merging or cycle formation, which are essential for constructing  
 140 persistence diagrams.

- 141 • *Simplex Birth Time.* Given a graph and its filtration sequence, predicts the birth time of specific  
 142 simplices.
- 143 • *Component Merge Time.* Given a graph and its filtration sequence, predicts when specific connected  
 144 components merge.
- 145 • *Component Count Under Filtration.* Given a graph and its filtration sequence, determines the  
 146 number of connected components at a specified filtration value.

147  
 148 **Hard Tasks (Filtration Strategy Design).** These tasks challenge LLMs to select or generate filtration  
 149 strategies that maximize the Wasserstein distance between the persistence barcodes of two given  
 150 graphs. We consider two types of strategies: the first involves selecting a global filtration function,  
 151 while the second involves generating a non-uniform sequence of filtration thresholds under a given  
 152 filtration function. Tasks are evaluated using ranking-based metrics. From the PH perspective,  
 153 this stage focuses on the design of filtration functions, which directly determines how simplicial  
 154 complexes evolve and which topological features are revealed. LLMs are expected to reason backward  
 155 from topological goals (e.g., maximizing feature differences) to suitable filtration strategies.

- 156 • *Optimal Filtration Selection.* Given two distinct graphs, the model selects the optimal filtration  
 157 function from a set of candidates: node degree, edge weight, K-shell, closeness centrality, between-  
 158 ness centrality, or eigenvector centrality, so as to maximize the Wasserstein distance between their  
 159 resulting persistence barcodes.
- 160 • *Non-Uniform Filtration Generation.* Given two distinct graphs with edge weights ranging from 1  
 161 to 10, and a fixed filtration function (edge weight), the model generates a non-uniform filtration

sequence of length 5 (e.g., (1, 3, 4, 7, 10)) that maximizes the Wasserstein distance between their persistence barcodes.

**Real-World Tasks (Real-World Graph Inference).** These tasks evaluate the capability of LLMs to transfer and apply PH insights acquired from synthetic scenarios to real-world graph datasets. Specifically, we test whether LLMs can choose or generate suitable filtration strategies to solve real-world graph classification tasks. Tasks are evaluated by classification accuracy. The datasets are from Sutherland et al. (2003), corresponding to molecular graphs targeting benzodiazepine receptor (BZR), cyclooxygenase-2 (COX2), and dihydrofolate reductase (LDHFR), respectively.

This stage reflects the final step of the PH pipeline, where topological features are used for downstream tasks such as classification. It tests whether LLMs can operationalize abstract persistence information in practical settings.

- *Filtration Selection for Classification.* Given four graphs from two classes, the model selects the optimal filtration function from node degree, edge weight, K-shell, closeness centrality, betweenness centrality, or eigenvector centrality to group the graphs correctly by class.
- *Filtration Sequence Generation for Classification.* Given four graphs from two classes and a fixed filtration function with a set of candidate filtration values, the model generates a filtration sequence that enables correct classification of the four graphs into two groups.

Table 1: Overview of the LLM4PH Benchmark Tasks.

Task Category	Task Name	Evaluation	#Samples
Simplicial Structure Understanding	Component Counting	Accuracy	1200
	Critical Simplex Identification	Accuracy	1200
	Component Reduction	Accuracy	1200
Filtration Evolution Reasoning	Component Count Under Filtration	Accuracy	1200
	Simplex Birth Time	Accuracy	1200
	Component Merge Time	Accuracy	1200
Filtration Strategy Design	Optimal Filtration Selection	Ranking	1200
	Uneven Filtration Generation	Ranking	1200
Real-World Graph Inference	Filtration Selection for Classification	Accuracy	900
	Filtration Sequence Generation for Classification	Accuracy	900
<b>Total</b>	10		11400

### 3 EXPERIMENTS

Based on the LLM4PH benchmark, we conduct a series of experiments to evaluate the performance of various large language models across different tasks. The goal is to assess whether LLMs are capable of understanding the abstract concepts behind PH and applying this knowledge to practical graph analysis tasks.

**Experimental Setup.** We evaluate the following six LLMs: GPT-4.1 (gpt-4.1-2025-04-14), GPT-4o (gpt-4o-2024-08-06), Claude (claude-3-7-sonnet-20250219), Gemini-2.5 (gemini-2.5-flash-preview-04-17), Deepseek-R1, Deepseek-V3, Qwen3-30B, Mistral-small-3.1-24B and Llama-3-70B. The first five are proprietary models accessed via their official APIs. The last three are open-source models deployed locally on a Linux server equipped with four NVIDIA A6000 GPUs. In some experiments, Random denotes predictions sampled uniformly from the empirical label distribution.

#### 3.1 RESULTS OF SIMPLE TASKS (SIMPLICIAL STRUCTURE UNDERSTANDING)

**Performance analysis.** Simple tasks in our benchmark are conceptually related to prior LLM benchmarks on graphs that focus on counting connected components or identifying cycles Fatemi et al. (2024); Wang et al. (2023). However, instead of relying purely on graph-theoretic terminology, our benchmark frames these problems within the language of PH. For example, connected components are treated as 0-dimensional homology classes, and edges are seen as 1-simplices forming the foundation of higher-order topological features. This reframing not only allows for a more unified multi-scale perspective, but also tests whether LLMs can map familiar combinatorial structures into a topological reasoning framework.

Table 2: Results of Simple Tasks.

Size	GPT-4.1	GPT-4o	Claude	Gemini	DS-R1	DS-V3	Qwen	Mistral	Llama	Random
<b>0D Component Counting</b>										
S	0.693	0.863	0.995	<b>1</b>	0.995	0.998	0.995	0.843	0.390	0.301
M	0.518	0.733	0.915	<b>1</b>	0.915	0.995	0.978	0.680	0.335	0.303
L	0.375	0.375	0.528	0.988	0.528	<b>0.998</b>	0.535	0.455	0.335	0.353
<b>1D Simplex Counting</b>										
S	0.213	0.270	0.150	<b>0.271</b>	0.150	0.260	0.143	0.085	0.223	0.253
M	0.105	<b>0.225</b>	0.068	0.078	0.068	0.043	0.035	0.030	0.058	0.255
L	0.105	<b>0.178</b>	0.018	0.044	0.018	0.038	0.020	0.045	0	0.265
<b>Component Reduction</b>										
S	<b>1</b>	0.943	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	0.998	0.898	0.875	-
M	0.993	0.855	0.988	<b>1</b>	0.988	<b>1</b>	0.400	0.780	0.810	-
L	0.833	0.620	0.815	<b>1</b>	0.815	<b>1</b>	0.255	0.648	0.708	-

Table 3: Results of Medium Tasks.

Size	GPT-4.1	GPT-4o	Claude	Gemini	DS-R1	DS-V3	Qwen	Mistral	Llama	Random
<b>Simplex Birth Time</b>										
S	0.683	0.278	0.370	0.470	<b>0.960</b>	0.325	0.375	0.280	0.325	0.118
M	0.375	0.135	0.415	0.125	<b>0.810</b>	0.140	0.135	0.130	0.345	0.143
L	0.263	0.188	0.545	0.538	<b>0.760</b>	0.030	0	0.055	0.255	0.185
<b>Component Merge Time</b>										
S	0.333	0.425	0.270	0.180	<b>0.455</b>	0.470	0.135	0.310	0.400	0.105
M	0.218	<b>0.308</b>	0.080	0.080	0.133	0.160	0.050	0.240	0.085	0.148
L	<b>0.260</b>	0.258	0.013	0.027	0.050	0.150	0.030	0.150	0.010	0.133
<b>Component Count Under Filtration</b>										
S	0.968	0.790	<b>0.990</b>	0.885	0.988	0.838	0.535	0.515	0.448	0.173
M	0.958	0.593	0.885	0.810	<b>0.995</b>	0.660	0.448	0.422	0.240	0.120
L	0.615	0.210	0.510	0.635	<b>0.975</b>	0.335	0.285	0.130	0.113	0.252

As shown in Table 2, most LLMs perform very well on the *0D Feature Counting* task. Deepseek-V3 and Gemini reach near-perfect accuracy across all graph sizes, with open-source models such as deepseek-R1 and Qwen3-32B also achieving over 97% on small and medium graphs. This suggests that counting connected components is a well-internalized task for many models, especially when the concept is explicitly framed. Notably, the use of “0-dimensional homology” in the prompt does not significantly hinder performance, indicating that many models can align this term with the familiar notion of connectedness.

In contrast, the *1D Simplex Counting* task is considerably more challenging. Even the strongest models fail to surpass 30% accuracy on medium and large graphs. Among them, GPT-4o performs relatively better, but its results remain unsatisfactory. This difficulty likely stems from the inherently discrete nature of counting, as well as the weak association between topological terminology and natural language. Although the term “1-simplex” is mathematically precise, it is rarely used in everyday contexts, and most models may lack the prior knowledge needed to link it reliably to the concept of edges in graphs.

The *Connectivity Reduction* task sits between the two in difficulty. Most models achieve over 95% on small graphs, with Gemini maintaining strong performance even as the graph size increases. The task requires not only understanding the current number of connected components, but also modifying the graph to reduce it by one. This implies a degree of generative reasoning over topological state changes, and the performance gap between models such as Claude (81.5% on large graphs) and Qwen3 (25.5%) highlights their varying abilities to execute localized structural edits in a topological context.

**Observation.** While many LLMs perform well on classical graph tasks when expressed in plain language, only the strongest models sustain high accuracy when the same tasks are reframed using the abstract vocabulary of PH. **This reveals that true topological reasoning, understood as alignment with the PH pipeline rather than simple output matching, remains an unsolved challenge for most models.**

Table 4: Results of Hard Tasks. Subscripts indicate standard deviations.

Size	GPT-4.1	GPT-4o	Claude	Gemini	DS-R1	DS-V3	Qwen	Mistral	Llama
<b>Optimal Filtration Selection</b>									
S	4.28 <sub>1.75</sub>	4.38 <sub>1.73</sub>	4.05 <sub>1.79</sub>	4.06 <sub>1.93</sub>	4.45 <sub>1.77</sub>	4.22 <sub>1.82</sub>	4.55 <sub>1.73</sub>	3.67 <sub>1.79</sub>	<b>3.58</b> <sub>1.74</sub>
M	4.14 <sub>1.50</sub>	4.18 <sub>1.65</sub>	4.05 <sub>1.61</sub>	4.38 <sub>1.64</sub>	4.48 <sub>1.52</sub>	4.30 <sub>1.44</sub>	4.30 <sub>1.66</sub>	3.78 <sub>1.55</sub>	<b>3.74</b> <sub>1.46</sub>
L	4.23 <sub>1.40</sub>	4.24 <sub>1.43</sub>	4.21 <sub>1.43</sub>	4.40 <sub>1.50</sub>	4.46 <sub>1.48</sub>	4.20 <sub>1.46</sub>	4.27 <sub>1.60</sub>	<b>4.05</b> <sub>1.49</sub>	4.10 <sub>1.39</sub>
<b>Non-Uniform Filtration Generation</b>									
S	46.4 <sub>30.1</sub>	56.0 <sub>38.6</sub>	<b>43.4</b> <sub>29.3</sub>	58.3 <sub>34.2</sub>	60.2 <sub>35.4</sub>	54.9 <sub>34.6</sub>	53.7 <sub>34.9</sub>	46.6 <sub>31.5</sub>	50.8 <sub>31.8</sub>
M	52.6 <sub>35.1</sub>	54.2 <sub>36.6</sub>	<b>47.9</b> <sub>31.5</sub>	54.8 <sub>35.8</sub>	64.4 <sub>35.0</sub>	55.5 <sub>33.5</sub>	59.1 <sub>31.3</sub>	50.0 <sub>33.3</sub>	53.9 <sub>30.4</sub>
L	48.6 <sub>33.9</sub>	53.0 <sub>31.0</sub>	46.8 <sub>30.3</sub>	54.7 <sub>33.8</sub>	77.0 <sub>33.5</sub>	54.0 <sub>27.4</sub>	60.9 <sub>34.2</sub>	<b>46.7</b> <sub>31.1</sub>	52.6 <sub>28.6</sub>

### 3.2 RESULTS OF MEDIUM TASKS (FILTRATION EVOLUTION REASONING)

**Performance analysis.** Unlike traditional graph reasoning tasks that rely solely on fixed structures, the medium tasks in our benchmark center around filtration. This dynamic process is foundational to PH, as it defines when features appear (birth) and disappear (death). In these tasks, models must go beyond static recognition and reason about how topological features evolve across a filtration sequence. This reflects one of the most distinctive aspects of PH and poses a significantly greater challenge for LLMs, especially without explicit computation.

As shown in Table 3, model performance varies widely across tasks and sizes, highlighting both the complexity of filtration-based reasoning and the limitations of current LLMs.

In the *Simplex Birth Time* task, which requires the model to determine when specific simplices (such as edges or triangles) appear during the filtration process, the results are highly inconsistent. DeepSeek-R1 achieves strong performance on this task (up to 96%), possibly due to its robustness in handling ordering-based reasoning. However, this advantage does not consistently extend to other tasks. In contrast, other open-source models perform poorly. For instance, Qwen reaches only 10% and Mistral just 5.5% on large graphs. These results suggest that reasoning about dynamic structural evolution remains a significant challenge for large language models.

The *Component Merge Time* task, which requires models to determine when two connected components merge during a filtration, is highly challenging for all models. Even top-performing models like GPT-4.1 and Claude score below 35%, and some models, such as Claude and Gemini on large graphs, barely produce any correct predictions. Although DeepSeek-R1 achieves the best results on small graphs, its performance declines significantly as the graph size increases. This difficulty further illustrates that tracking the merging of components across filtration steps is a fundamentally hard problem for autoregressive token-based large language models.

On *Component Count Under Filtration*, DS-R1 leads across all sizes with 98.8% on small, 99.5% on medium, and 97.5% on large, showing the strongest scale robustness. Claude is marginally higher on small at 99.0%, and GPT-4.1 stays strong on small and medium at 96.8% and 95.8% but drops to 61.5% on large. GPT-4o declines with size from 79.0% to 59.3% to 21.0%. DS-V3 sits mid tier. Open-source models remain far behind, near the mid 50s on small, mid 40s on medium, and below 30% on large, with Llama near 11.0%. Overall, DS-R1 sustains near-ceiling accuracy while others degrade with scale.

**Observation.** Filtration-based reasoning adds a distinct layer of complexity that exposes the structural limitations of current LLMs. While some models manage localized success on small graphs/simple filtrations, none consistently handle the full range of tasks and sizes. **PH introduces not only unfamiliar terminology but also a conceptual framework grounded in temporal evolution and geometric abstraction which remains difficult for most models to grasp.** These results pose a critical challenge in topological reasoning where continuity and change must be understood in tandem.

### 3.3 RESULTS OF HARD TASKS (FILTRATION STRATEGY DESIGN)

**Performance analysis.** This set of tasks evaluates LLMs’ ability to reason in reverse: from desired objectives (e.g., maximizing topological difference) to designing appropriate filtration strategies. In contrast to earlier tasks that test a model’s ability to recognize topological features, these tasks assess whether a model can actively control the filtration process to induce specific topological outcomes.

We use ranking-based evaluation for both tasks. In *Optimal Filtration Selection*, each model is asked to choose a filtration function (e.g., node degree, edge weight) that maximizes the Wasserstein distance between persistence barcodes of two input graphs. Since there are six candidate functions, the model’s answer is ranked among the six: lower rank means better performance (1 is best, 6 is worst). In *Non-Uniform Filtration Generation*, the model generates a filtration sequence (a length-5 subset from values 1 to 10), and its output is ranked among 126 possible sequences based on the resulting Wasserstein distance. Again, lower rank indicates a more effective choice.

From Table 4, we observe that all models struggle with these tasks. In the *Optimal Filtration Selection* task, the small open-source models Mistral and Llama surprisingly achieve the best results, while most other models fail to produce any rankings below 3. In addition, all models exhibit large standard deviations, indicating highly unstable outputs. This suggests that their performance is only marginally better than random guessing. Overall, designing filtrations based on specific goals and graph structures remains a highly challenging task for current language models.

In the *Non-Uniform Filtration Generation* task, Claude emerges as the best-performing model across all graph sizes, achieving the lowest average rank (43.4 on small, 47.9 on medium, and 46.8 on large). This indicates a surprising strength in generating effective filtration sequences that yield topological divergence. However, the overall performance remains disappointing. All models exhibit large standard deviations that are nearly as large as their means, suggesting that the models have not truly learned any meaningful knowledge related to the task.

**Observation.** Topological control tasks remain challenging for LLMs, requiring reasoning about graph-filtration interactions. Our benchmark tests active topological shaping rather than passive interpretation. Results show that even strong models lack sufficient reasoning granularity, suggesting new research directions in persistence-aware training and hybrid architectures. **Success requires aligning structural patterns with topological objectives, not just pattern recognition.**

### 3.4 RESULTS OF REAL-WORLD TASKS (OR REAL-WORLD GRAPH INFERENCE)

Table 5: Results of Real-World Tasks.

Datasets	GPT-4.1	GPT-4o	Claude	Gemini	DS-R1	DS-V3	Qwen	Mistral	Llama
<b>Filtration Selection for Classification</b>									
BZR	<b>0.350</b>	<b>0.350</b>	<b>0.350</b>	0.340	0.320	<b>0.350</b>	0.320	<b>0.350</b>	<b>0.350</b>
COX2	0.320	0.260	0.320	0.320	<b>0.360</b>	0.320	<b>0.360</b>	0.290	0.315
LDHFR	0.345	0.310	0.345	0.345	0.460	0.335	<b>0.460</b>	0.310	0.340
<b>Filtration Sequence Generation for Classification</b>									
BZR	0.595	0.650	<b>0.775</b>	0.765	0.760	0.410	0.400	0.652	0.740
COX2	0.710	0.555	0.695	<b>0.755</b>	0.180	0.160	0.480	0.655	0.655
LDHFR	0.950	0.920	<b>0.970</b>	0.958	0.760	0.880	0.740	0.860	0.945
<b>Directly Classification</b>									
BZR	<b>0.985</b>	0.940	0.990	0.969	0.920	0.980	0.820	0.525	0.995
COX2	0.985	0.920	0.995	0.990	<b>1</b>	0.995	0.740	0.460	<b>1</b>
LDHFR	<b>1</b>	0.960	0.995	<b>1</b>	<b>1</b>	0.990	0.980	0.790	<b>1</b>

This set of tasks evaluates whether LLMs can apply PH to graph classification. It includes two PH-guided tasks: *Filtration Selection for Classification* and *Filtration Sequence Generation for Classification*, as well as a direct classification baseline without topological reasoning.

In the *Filtration Selection for Classification* task, performance is notably poor and highly uniform across models and datasets, with accuracies fluctuating narrowly around 0.32 to 0.36. This indicates that models consistently choose similar filtration functions regardless of the graph structure. Further inspection reveals that most LLMs default to selecting `edge weight` as the filtration function, possibly because it appears more "interpretable" or statistically grounded. However, this uniformity fails to expose discriminative topological differences, leading to almost random classification results.

In contrast, the *Filtration Sequence Generation for Classification* task exhibits stronger performance across all models, especially on the LDHFR dataset, where top models (Claude, GPT-4.1, Gemini) achieve accuracies above 0.95. This suggests that when explicitly instructed to manipulate filtration sequences, even under fixed filtration functions, LLMs can induce more topologically meaningful

378 separations. Notably, Claude outperforms all other models on BZR and LDHFR, achieving the best  
 379 classification accuracy among all methods.  
 380

381 To evaluate the reliance on PH reasoning, we introduce a control task: *Direct Classification*, where the  
 382 model is asked to classify graphs directly without designing filtration strategies. Surprisingly, most  
 383 models achieve near-perfect accuracy (0.98–1.0), suggesting that for small 4-graph inputs, LLMs can  
 384 leverage shallow statistical patterns or implicit textual correlations to memorize or infer class labels.  
 385 However, this also reveals that the PH tasks are substantially more difficult than the direct version,  
 386 requiring abstraction, planning, and topological insight rather than pattern matching.

387 **Observation.** The weak performance on filtration function selection, contrasted with stronger  
 388 results on filtration sequence generation, suggests that LLMs are capable of expressing meaningful  
 389 topological reasoning **only when guided by structured prompts and constrained options**. In the  
 390 absence of such scaffolding, models tend to fall back on familiar heuristics, which often obscure  
 391 rather than uncover PH-based structure.  
 392

### 393 3.5 CROSS-TASK ANALYSES

394  
 395 Beyond the evaluation of individual tasks, we conduct three cross-task experiments that focus on  
 396 broader aspects of model behavior. These experiments examine the effects of input representation,  
 397 model fine-tuning, and pipeline compositionality across multiple tasks and graph sizes. Together, they  
 398 provide additional insight into the generalization and operational behavior of language models when  
 399 applied to PH reasoning. Detailed results for all cross-task experiments are provided in Appendix E.

400 **Prompt Encoding and Task Formulation Ablation** We examine how the structure of prompts  
 401 influences model behavior across tasks. This experiment includes variations in both the **graph repre-**  
 402 **sentation style** and the **task formulation style**, applied to three representative tasks: 1D Simplex  
 403 Counting, Component Merge Time, and Component Count Under Filtration on both synthetic and  
 404 real-world LDHFR data. We evaluate three styles of graph representation, i.e., text-style format, code-  
 405 like format, and matrix-based format, and three styles of task instruction formulation: topological,  
 406 graph-theoretic, and minimal. Descriptions of each style are summarized in Table 12, and examples  
 407 for each combination are provided in the appendix. From Table 8, text-style input consistently yields  
 408 the highest accuracy across models. Graph-theoretic task phrasing also outperforms both topological  
 409 and minimal formulations in most settings. In contrast, matrix-based input and minimal task prompts  
 410 tend to result in lower and less stable performance. These findings confirm that surface-level prompt  
 411 features play a substantial role in model outcomes, even when the underlying graph structure and  
 412 reasoning objectives remain unchanged. Consequently, evaluations of language models on topological  
 reasoning must account for prompt formulation as a key factor in performance variance.

413 To further investigate the effect of task formulation, we conduct a focused **terminology ablation**  
 414 **study**. For each task, we generate paired prompts with topological phrasing (e.g., H0 class) and  
 415 graph-theoretic equivalents (e.g., connected component”), keeping the underlying operations identical.  
 416 Results in Table 9 show that terminology sensitivity is task-dependent: for example, GPT-4o achieves  
 417 similar accuracy across formulations on the birth” task, but shows a significant drop in the CCUF”  
 418 task when PH-specific terminology is used (from 0.8633 to 0.45). This indicates that while domain  
 419 terminology contributes to performance variance, it does not fully explain failure modes—many errors  
 420 stem from deeper structural reasoning limitations rather than surface-level linguistic mismatches.

421 To evaluate whether prompt engineering can mitigate reasoning errors in complex settings, we  
 422 conduct an experiment on the non-uniform filtration generation task using two prompting strategies:  
 423 Few-shot Chain-of-Thought (CoT) and rule-emphasized prompting. While the former provides a  
 424 complete reasoning example to encourage mimicking, the latter enforces strict filtration and topology  
 425 rules without examples. As shown in Table 10, neither approach yields consistent improvement.  
 426 Performance remains unstable, with large variance across both models. This suggests that prompt  
 427 design alone cannot resolve the deeper challenge of maintaining accurate H0/H1 transitions in  
 428 multi-step topological reasoning.

429 **Post Training and Scale Transferability** We study whether supervised adaptation can instill stable  
 430 topological routines in an open-source model. We fine tune LLaMA-3.1-8B-Instruct on small graphs  
 431 for three representative tasks and then evaluate on medium graphs with matched distributions. The  
 protocol preserves the prompt settings used in the ablation, which allows us to measure how much

performance remains sensitive to surface representation after adaptation. As reported in Table 11, post training delivers consistent and often large gains across all tasks. On 1D Simplex Counting the best prompt improves from 0.02 to 0.20, while Code like with topological phrasing reaches 0.26. On Component Merge Time the score rises from 0.09 to 0.43 for text with graph theoretic phrasing, with similar improvements for other text prompts. On Component Count Under Filtration the text with graph theoretic phrasing increases from 0.16 to 0.66, and code like with topological phrasing reaches 0.54. Prompt sensitivity remains present after adaptation. Text prompts continue to be reliable across tasks. Code like prompts become more competitive after fine tuning and show the largest relative improvements in two tasks. Minimal phrasing improves but still trails behind explicit instructions. We observe the same trends in cross scale transfer, where a model tuned on small graphs improves from 0.005 to 0.235 on large 1D Simplex Counting, which suggests that the learned routines generalize beyond the training size. Overall, fine tuning strengthens topological reasoning and reduces variance, yet careful prompt design remains important for peak performance.

**Compositional PH Pipeline** Beyond evaluating LLM performance on individual subtasks, we construct a full pipeline to assess how well LLMs perform when guiding an integrated PH workflow. We run a *Filtration Selection for Classification* experiment on the real-world BZR dataset. The pipeline follows the standard five-step process used by PH practitioners: (1) select a filtration function, (2) compute filtration values, (3) generate persistence diagrams, (4) compare diagram distances, and (5) determine the final class label. We also include results obtained directly from persistent homology tools, using four filtration functions: Weight, Degree, Betweenness, and Closeness.

In Table 6, LLM indicates that a step is completed entirely by the language model through prompt-based reasoning, while `code` denotes that the step is handled by standard PH libraries such as `Gudhi`. We test four settings that vary the boundary between model reasoning and code execution. In Setting 1 the model selects the filtration and also executes diagram generation and distance comparison, while code handles filtration values and the final classifier. In Setting 2 the model selects the filtration and all downstream steps use code. In Setting 3 the model selects the filtration and performs distance comparison, while code generates the diagrams. In Setting 4 the model selects the filtration and generates diagrams, while code performs distance comparison and the final classifier.

Table 6 shows that the pure code-based configuration (filtration by the model, all downstream steps by code) produces stable mid-level accuracy across models, providing a consistent baseline. In contrast, hybrid designs where the model participates in distance comparison achieve the highest scores for Claude and DS-V3. Settings that require the model to generate diagrams are less reliable, often leading to performance drops. These results suggest that the language model is most effective in filtration selection and, for some models, in distance reasoning, while code remains essential for diagram generation and overall stability. The results obtained directly from persistent homology tools show that although PH tool based pipelines consistently achieve higher end to end accuracy than LLM driven configurations, hybrid pipelines can occasionally match or even surpass tool only performance in specific settings. For instance, Claude reaches an accuracy of 0.840 when combined with symbolic components. These findings illustrate the potential of hybrid reasoning loops and highlight particular interfaces, such as filtration selection, where LLMs can provide meaningful contributions.

**Large-Graph Scalability via Subgraph Sampling** Since current context windows cannot contain full large graphs, we evaluate scalability through controlled subgraph sampling on the ego-Facebook dataset Leskovec & Mcauley (2012). We use random walk sampling to build six collections with 10, 20, 50, 80, 100, and 150 subgraphs, each subgraph containing 50 nodes. We run the 1D Simplex Counting task five times for GPT-4o and GPT-4.1-mini and report mean accuracy (standard deviation) in Table 7. Both models improve as the number of sampled subgraphs grows. GPT-4.1-mini increases monotonically from 14% to 24%, and GPT-4o shows an overall upward trend.

**Error analysis: Non-Uniform Filtration Generation** To better understand model failure on this task, we conduct a systematic error analysis and identify six recurring patterns for the Non-Uniform Filtration Generation task. These fall into three broad categories: loss of prior context, incomplete procedural reasoning, and chain-of-thought instability.

1. *Fact forgetting*. The model fails to retain previously mentioned edge information. For instance, it may first note that node 1 is connected to node 7, but later treat them as belonging to separate components. This occurs because earlier facts drift out of the model’s attention window and are no longer integrated into downstream predictions.

2. *Rule forgetting.* After explicitly acknowledging rules, such as “isolated nodes are not counted as components” or “a merge occurs when two components join”. This reflects the absence of persistent logical constraints; once the model’s focus shifts, prior rules are no longer enforced.
3. *Incomplete algorithm emulation.* The model often attempts to reproduce the output of traversal-based algorithms (such as depth-first search) but does so heuristically, without internal state tracking. This results in invalid outputs, such as calling a walk with repeated nodes a cycle or skipping key edges during merge detection.
4. *Blurry conceptual boundaries.* Topological concepts are applied too loosely. For example, any path resembling a loop may be labeled as a “hole” or “cycle,” even if it includes repeated vertices or fails basic structural criteria. This suggests the model relies on surface-level statistical cues rather than precise graph-theoretic definitions.
5. *Interrupted reasoning.* In longer examples requiring multiple stages of analysis, the model’s output may terminate mid-thought (e.g., “Now we check for cycles. . .” with no continuation). This typically occurs when the chain of thought becomes too long to maintain internal coherence, especially in graphs with many nodes or edge weights.
6. *Analysis paralysis.* When faced with multiple plausible merge paths or interpretations, the model may enter loops of self-revision. For instance, it might state “the correct answer is 4.00,” then immediately consider “perhaps 10.00,” and continue revising. This failure to converge indicates difficulty in managing uncertainty and resolving conflicting hypotheses.

These failure modes reveal structural limitations in current LMs. Robust reasoning under non-uniform filtrations needs persistent memory, explicit constraint handling, and the ability to simulate structured procedures.

## 4 CONCLUSION

This work introduces LLM4PH, the first benchmark designed to evaluate large language models’ ability to understand, reason about, and apply PH on graphs. By decomposing the PH pipeline into interpretable sub-tasks and organizing them into a progression of difficulty, we assess both the conceptual alignment and reasoning depth of LLMs in a topological setting. Our results reveal that while some models can handle low-dimensional features or structured prompts, most struggle with dynamic evolution and filtration design, indicating a fundamental gap between current LLM capabilities and the demands of topological abstraction. We hope this benchmark provides a foundation for future advances in topology-aware reasoning, hybrid neuro-symbolic architectures, and geometric representation learning with language models. Improving LLMs’ ability to engage with topological structures could unlock new directions in scientific discovery, graph-based inference, and interpretable AI systems grounded in geometric reasoning.

540 **Ethics Statement** This study does not involve human subjects, sensitive personal data, or any  
541 proprietary or confidential information. All datasets used in this work are synthetic or publicly  
542 available and are pre-processed to remove any personally identifiable information. For real-world  
543 graph datasets, we follow standard usage and citation practices and do not modify the underlying  
544 node or edge semantics.

545  
546 **Reproducibility Statement** To ensure full reproducibility, we provide all datasets, prompts, task  
547 definitions, and evaluation scripts in an anonymous GitHub repository: <https://anonymous.4open.science/r/LLM4PH-7015/>. Detailed descriptions of each benchmark task, including  
548 data generation methods, prompt templates, and scoring metrics, are included in the main paper and  
549 the appendix.  
550

551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593

## REFERENCES

- 594  
595  
596 Mehmet E Aktas, Esra Akbas, and Ahmed El Fatmaoui. Persistence homology of networks: methods  
597 and applications. *Applied Network Science*, 4(1):1–28, 2019.
- 598  
599 László Babai and Paolo Codenotti. Isomorphism of hypergraphs of low rank in moderately exponential  
600 time. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pp. 667–676.  
IEEE, 2008.
- 601  
602 Ginestra Bianconi. *Higher-order networks*. Cambridge University Press, 2021.
- 603  
604 Christian Bick, Elizabeth Gross, Heather A Harrington, and Michael T Schaub. What are higher-order  
605 networks? *SIAM review*, 65(3):686–731, 2023.
- 606  
607 Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang  
608 Yang. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint  
arXiv:2310.05845*, 2023.
- 609  
610 Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan  
611 Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM  
transactions on intelligent systems and technology*, 15(3):1–45, 2024.
- 612  
613 Yuzhou Chen, Yulia Gel, and H Vincent Poor. Time-conditioned dances with simplicial complexes:  
614 Zigzag filtration curve based supra-hodge convolution networks for time-series forecasting. *Ad-  
vances in Neural Information Processing Systems*, 35:8940–8953, 2022.
- 615  
616 Yuzhou Chen, Jose Frias, and Yulia R Gel. Topogcl: Topological graph contrastive learning. In  
617 *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 11453–11461, 2024a.
- 618  
619 Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei  
620 Yin, Wenqi Fan, Hui Liu, et al. Exploring the potential of large language models (llms) in learning  
621 on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61, 2024b.
- 622  
623 Carina Curto and Nicole Sanderson. Topological neuroscience: Linking circuits to function. *Annual  
Review of Neuroscience*, 48, 2025.
- 624  
625 Yuri Dabaghian, Facundo Mémoli, Loren Frank, and Gunnar Carlsson. A topological paradigm for  
626 hippocampal spatial map formation using persistent homology. 2012.
- 627  
628 Xinnan Dai, Haohao Qu, Yifei Shen, Bohang Zhang, Qihao Wen, Wenqi Fan, Dongsheng Li, Jiliang  
629 Tang, and Caihua Shan. How do large language models understand graph patterns? a bench-  
630 mark for graph pattern comprehension. In *The Thirteenth International Conference on Learning  
Representations*, 2025. URL <https://openreview.net/forum?id=CkKEuLmRnr>.
- 631  
632 Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large  
633 language models. In *The Twelfth International Conference on Learning Representations*, 2024.  
URL <https://openreview.net/forum?id=IuXR1CCrSi>.
- 634  
635 Yifan Feng, Jiashu Han, Shihui Ying, and Yue Gao. Hypergraph isomorphism computation. *IEEE  
Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3880–3896, 2024.
- 636  
637 Yifan Feng, Chengwu Yang, Xingliang Hou, Shaoyi Du, Shihui Ying, Zongze Wu, and Yue Gao.  
638 Beyond graphs: Can large language models comprehend hypergraphs? In *The Thirteenth Inter-  
national Conference on Learning Representations*, 2025. URL [https://openreview.net/  
forum?id=28qOQwjuma](https://openreview.net/forum?id=28qOQwjuma).
- 639  
640
- 641  
642 Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muham-  
643 mad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. A survey on large language  
models: Applications, challenges, limitations, and practical usage. *Authorea Preprints*, 2023.
- 644  
645 Christoph D Hofer, Roland Kwitt, and Marc Niethammer. Learning representations of persistence  
646 barcodes. *Journal of Machine Learning Research*, 20(126):1–45, 2019.
- 647  
Danijela Horak, Slobodan Maletić, and Milan Rajković. Persistent homology of complex networks.  
*Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03034, 2009.

- 648 Johanna Immonen, Amauri Souza, and Vikas Garg. Going beyond persistent homology using  
649 persistent homology. *Advances in neural information processing systems*, 36:63150–63173, 2023.
- 650  
651 Jure Leskovec and Julian McAuley. Learning to discover social circles in ego networks. *Advances in*  
652 *neural information processing systems*, 25, 2012.
- 653 Hao Li, Hao Jiang, Fan Jiajun, Dongsheng Ye, and Liang Du. Dynamic neural dower network:  
654 Approximating persistent homology in dynamic directed graphs. In *Proceedings of the 30th ACM*  
655 *SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1554–1564, 2024.
- 656 Zhenyu Meng, D Vijay Anand, Yunpeng Lu, Jie Wu, and Kelin Xia. Weighted persistent homology  
657 for biomolecular data analysis. *Scientific reports*, 10(1):2079, 2020.
- 658  
659 Ippei Obayashi, Takenobu Nakamura, and Yasuaki Hiraoka. Persistent homology analysis for  
660 materials research and persistent homology software: Homcloud. *journal of the physical society of*  
661 *japan*, 91(9):091013, 2022.
- 662 Aske Plaat, Annie Wong, Suzan Verberne, Joost Broekens, Niki van Stein, and Thomas Back.  
663 Reasoning with large language models, a survey. *arXiv preprint arXiv:2407.11511*, 2024.
- 664  
665 Chi Seng Pun, Si Xian Lee, and Kelin Xia. Persistent-homology-based machine learning: a survey  
666 and a comparative study. *Artificial Intelligence Review*, 55(7):5169–5213, 2022.
- 667 Jeffrey J Sutherland, Lee A O’Brien, and Donald F Weaver. Spline-fitting with a genetic algorithm:  
668 A method for developing classification structure- activity relationships. *Journal of chemical*  
669 *information and computer sciences*, 43(6):1906–1915, 2003.
- 670 Yanchao Tan, Hang Lv, Xinyi Huang, Jiawei Zhang, Shiping Wang, and Carl Yang. Musegraph:  
671 Graph-oriented instruction tuning of large language models for generic graph mining. *arXiv*  
672 *preprint arXiv:2403.04780*, 2024.
- 673  
674 Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang.  
675 Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th*  
676 *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp.  
677 491–500, 2024.
- 678 Jacob Townsend, Cassie Putman Micucci, John H Hymel, Vasileios Maroulas, and Konstantinos D  
679 Vogiatzis. Representation of molecular structures with persistent homology for machine learning  
680 applications in chemistry. *Nature communications*, 11(1):3230, 2020.
- 681  
682 Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov.  
683 Can language models solve graph problems in natural language? *Advances in Neural Information*  
*Processing Systems*, 36:30840–30861, 2023.
- 684  
685 Kelin Xia and Guo-Wei Wei. Persistent homology analysis of protein structure, flexibility, and folding.  
686 *International journal for numerical methods in biomedical engineering*, 30(8):814–844, 2014.
- 687  
688 Zuoyu Yan, Tengfei Ma, Liangcai Gao, Zhi Tang, and Chao Chen. Link prediction with persistent  
689 homology: An interactive view. In *International conference on machine learning*, pp. 11659–11669.  
PMLR, 2021.
- 690  
691 Dongsheng Ye, Hao Jiang, Ying Jiang, and Hao Li. Stable distance of persistent homology for  
dynamic graph comparison. *Knowledge-Based Systems*, 278:110855, 2023.
- 692  
693 Yifan Zhang, Shaoyi Du, Yifan Feng, Shihui Ying, and Yue Gao. Reinterpreting hypergraph kernels:  
694 Insights through homomorphism analysis. *IEEE Transactions on Pattern Analysis and Machine*  
695 *Intelligence*, 2025.
- 696  
697 Zeyang Zhang, Xin Wang, Ziwei Zhang, Haoyang Li, Yijian Qin, and Wenwu Zhu. Llm4dyg: can  
698 large language models solve spatial-temporal problems on dynamic graphs? In *Proceedings of the*  
*30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4350–4361, 2024.
- 699  
700 Jianan Zhao, Le Zhuo, Yikang Shen, Meng Qu, Kai Liu, Michael M. Bronstein, Zhaocheng Zhu, and  
701 Jian Tang. Graphtext: Graph reasoning in text space. In *Adaptive Foundation Models: Evolving AI*  
*for Personalized and Efficient Learning*, 2024. URL <https://openreview.net/forum?id=6cRM00T03F>.

## THE USE OF LARGE LANGUAGE MODELS (LLMs)

We used large language models (LLMs), such as ChatGPT, to assist with writing refinement, code debugging, and literature exploration. LLMs helped improve the clarity and academic tone of the manuscript, assisted in refactoring evaluation scripts, and supported the search for relevant background references. All core research ideas, experimental design, implementation, and analysis were conducted independently by the authors. No text or results were generated solely by LLMs.

## A RELATED WORK

**LLM for Graphs.** Recent efforts have sought to evaluate the reasoning capabilities of large language models (LLMs) on graph-structured data through benchmark datasets. Early benchmarks focus on static graph tasks such as shortest path computation, cycle detection, node classification, and graph traversal Fatemi et al. (2024); Wang et al. (2023). More recent work expands this scope to dynamic graphs, introducing temporal reasoning tasks such as dynamic link prediction and event ordering Zhang et al. (2024). Other studies have explored graph pattern recognition Dai et al. (2025) and subgraph isomorphism through natural language prompts. These benchmarks typically emphasize local structure, edge-level logic, or symbolic reasoning over discrete structures. However, they often overlook the global, multi-scale, and high-dimensional characteristics inherent in many real-world graphs. Our work complements and extends this line of research by introducing PH as a testbed for evaluating topological reasoning beyond pairwise relationships.

Recent work has explored using LLMs for traditional graph learning tasks Chen et al. (2024b). Methods like Graph-LLM Chai et al. (2023) and GraphText Zhao et al. (2024) convert graphs into natural language for few-shot inference, while others combine LLMs with GNNs Tang et al. (2024); Tan et al. (2024). However, these approaches focus on low-order features rather than high-dimensional topological structures. Our work complements this by targeting PH features and assessing LLMs’ ability to reason about them without neural encoders.

**Persistent Homology on Graphs.** Persistent homology, a core tool in topological data analysis (TDA), provides a framework to extract multi-scale topological features from data (Pun et al., 2022). When applied to graphs, it captures global structures such as connected components, cycles, and voids across a filtration of the graph Aktas et al. (2019); Immonen et al. (2023); Horak et al. (2009). Graph-based persistent homology has been widely used in fields like neuroscience Dabaghian et al. (2012); Curto & Sanderson (2025), biology Xia & Wei (2014); Meng et al. (2020); Townsend et al. (2020), and materials science Obayashi et al. (2022), often serving as a shape descriptor or structural feature for downstream tasks. In machine learning, persistent diagrams have been integrated into graph neural networks Hofer et al. (2019); Yan et al. (2021); Chen et al. (2024a) as interpretable features. Despite its growing adoption, prior work assumes full access to the topological pipeline and does not explore whether LLMs can internalize or reason over such structures directly. Our work bridges this gap by proposing a benchmark that tests whether LLMs can understand, predict, and generate persistent homology concepts from graph inputs without explicit computation.

**Simplicial Complex Identification** Recent efforts in understanding and identifying higher-order topological structures have extended classical graph methods to the realm of hypergraphs and simplicial complexes Babai & Codenotti (2008) analyzed the complexity of hypergraph isomorphism, showing that while the general problem is NP-complete, isomorphism between bounded-rank hypergraphs, such as low-dimensional simplicial complexes, admits moderately exponential-time algorithms via group-theoretic methods. To capture structural similarity beyond exact isomorphism, recent work has introduced polynomial-time heuristics based on combinatorial refinement. Feng et al. (2024) proposed a generalization of the Weisfeiler–Lehman (WL) test to hypergraphs and constructed WL-based hypergraph kernels, enabling efficient comparisons between high-order structures through subtree and hyperedge patterns. Building on this, Zhang et al. (2025) reinterpreted such kernels in terms of homomorphism counts, revealing that many existing methods implicitly focus on acyclic patterns and introducing the Subtree-Cycle Kernel to enhance expressiveness by capturing cyclic features. These approaches provide both theoretical insights and practical tools for

756 identifying and comparing simplicial complexes through a combination of isomorphism heuristics  
 757 and homomorphism-informed embeddings.  
 758

## 759 B DATASET CONSTRUCTION

760 To support a controlled and interpretable evaluation of persistent homology understanding, we  
 761 construct both synthetic and real-world datasets tailored to each task category.  
 762

### 763 B.1 BENCHMARK STATISTICS

764 The statistical overview of our benchmark is illustrated in Table 1. Each synthetic dataset task  
 765 category’s samples are composed of an equal number of large, medium, and small graphs (10 nodes,  
 766 15 nodes, 30 nodes). For example, out of 1200 samples, 400 are large graphs, 400 are medium graphs,  
 767 and 400 are small graphs. The real-world datasets BZR, COX2, and LDHFR have an equal number  
 768 of samples. For more detailed information about dataset construction, please refer to Appendix B.  
 769

### 770 B.2 SYNTHETIC GRAPH GENERATION

771 All synthetic graphs are generated using randomized network models with fixed size constraints.  
 772 Specifically, small, medium, and large graphs are set to contain 10, 15, and 30 nodes respectively. For  
 773 each graph, we randomly sample edges to ensure structural variability while maintaining connectivity  
 774 and topological relevance.  
 775

### 776 B.3 TASK-SPECIFIC FILTERING AND SELECTION

777 For **simple** and **medium** tasks, we compute the Vietoris–Rips complex from each candidate graph and  
 778 evaluate its corresponding Betti numbers and persistence diagrams across standard filtration functions.  
 779 Graphs are selected based on whether they yield diverse and interpretable topological outcomes, such  
 780 as distinct connected components, nontrivial 1-dimensional features, and clear merging events.  
 781

782 For **hard tasks**, we compute pairwise Wasserstein distances between persistence diagrams of graph  
 783 pairs under all candidate filtration strategies. Each strategy (e.g., node degree, edge weight, closeness  
 784 centrality) is evaluated, and we retain the ranking of each strategy based on the resulting distance.  
 785 This ranking provides the ground truth supervision for tasks involving filtration design.  
 786

### 787 B.4 REAL-WORLD GRAPH SELECTION

788 For **real-world tasks**, we curate graphs from public datasets (e.g., BZR, COX2, LDHFR) and sample  
 789 small subsets of graphs from two distinct classes. For each problem instance, we evaluate the  
 790 Wasserstein distance between diagrams produced under each candidate filtration strategy, and label  
 791 tasks according to whether the correct class separation can be achieved. Graphs are allowed to appear  
 792 in at most two tasks to avoid overfitting and ensure evaluation diversity.  
 793

794 This construction pipeline ensures that each benchmark task is grounded in verifiable topological  
 795 variation and aligned with persistent homology computation standards.  
 796

## 797 C ADDITIONAL EXPLANATION OF PERSISTENT HOMOLOGY

800 Persistent homology is a core method in topological data analysis (TDA) that enables the extraction  
 801 of multi-scale topological features from data. It generalizes classical homology theory by introducing  
 802 the concept of a *filtration*, which allows one to track how topological features evolve across scales.  
 803

804 A filtration is a nested sequence of simplicial complexes:

$$805 K_0 \subseteq K_1 \subseteq \dots \subseteq K_T,$$

806 typically constructed by gradually adding simplices based on a filtration function (e.g., edge weights in  
 807 a graph). As the filtration evolves, features such as connected components (0-dimensional homology),  
 808 cycles (1-dimensional), and voids (2-dimensional) appear and eventually disappear.  
 809

810 Key terms used throughout this work include:

- 811
- 812 • **Simplex:** A generalization of a vertex (0-simplex), edge (1-simplex), triangle (2-simplex),
- 813 or higher-dimensional face. Simplices are the building blocks of a simplicial complex.
- 814
- 815 • **Simplicial Complex:** A finite set of simplices closed under the subset operation. It defines
- 816 a discrete topological space amenable to homology computation.
- 817
- 818 • **Filtration Function:** A function  $f$  that assigns a real-valued threshold to each simplex (often
- 819 indirectly through edges or vertices). The function governs the order in which simplices are
- 820 added during the filtration.
- 821
- 822 • **Birth and Death:** A topological feature (e.g., a connected component or a cycle) *births* at
- 823 the filtration index where it first appears and *dies* when it is merged into a larger feature or
- 824 filled in.
- 825
- 826 • **Persistence Diagram / Barcode:** A multiset of intervals  $\{[b_i, d_i]\}$  representing the lifespan
- 827 of topological features. The length  $d_i - b_i$  reflects the *persistence* or significance of each
- 828 feature.
- 829
- 830 • **Betti Number  $\beta_k$ :** The rank of the  $k$ -th homology group  $H_k$ , counting  $k$ -dimensional holes:
- 831  $\beta_0$  for components,  $\beta_1$  for loops, etc.
- 832
- 833 • **Wasserstein Distance:** A distance metric between persistence diagrams that quantifies
- 834 how topological structures differ across filtrations. It is often used to evaluate the output of
- 835 persistence-aware models.

832 By computing and analyzing these topological signatures, persistent homology provides a robust,

833 noise-tolerant summary of data structure that is invariant to continuous transformations and particu-

834 larly suited to graph-structured domains.

## 837 D DETAILS OF TASK DESIGN

838

839 This appendix provides detailed explanations for each task in the LLM4PH benchmark, framed

840 through the mathematical language of persistent homology and simplicial complex theory.

### 842 D.1 SIMPLE TASKS (SIMPLICIAL STRUCTURE UNDERSTANDING)

843

844 These tasks correspond to the initial stage of persistent homology, prior to any filtration, where the

845 simplicial structure of the graph must be identified. Each task tests whether an LLM can understand

846 basic homological objects from a combinatorial graph input.

847

848 **0D Component Counting.** Given an undirected graph  $G = (V, E)$ , the model is asked to compute

849  $\beta_0(G)$ , the 0-th Betti number, which equals the number of connected components in the graph. This

850 is equivalent to computing the rank of the 0-dimensional homology group  $H_0(G)$ .

851

852 **1D Simplex Counting.** Given a graph  $G$ , the task requires counting the number of 1-simplices,

853 which correspond to edges in the 1-skeleton of the simplicial complex induced by  $G$ . This measures

854 the cardinality of the 1-simplex set  $\Sigma_1$ .

855

856 **Component Reduction.** Given  $G$ , the model is asked to identify a single edge  $e \notin E$  such that

857  $E' = E \cup \{e\}$  decreases  $\beta_0(G)$  by one. This requires reasoning about connected component merges

858 via edge addition.

### 860 D.2 MEDIUM TASKS (FILTRATION EVOLUTION REASONING)

861

862 These tasks test an LLM’s ability to reason over a filtration  $\{K_t\}_{t \in T}$  of simplicial complexes built

863 from a weighted graph  $G$ , where  $K_t$  denotes the simplicial complex at filtration threshold  $t$ . The aim

is to understand topological evolution: the birth and death of homology classes across  $t$ .

**Simplex Birth Time.** Given a weighted graph  $G$  and a filtration order (e.g., by edge weight), the model predicts the value  $t$  at which a given simplex  $\sigma \in K_t$  appears. This corresponds to the birth time  $b(\sigma)$  in the persistence diagram.

**Component Merge Time.** For a pair of initially disconnected vertices  $(u, v)$ , the model predicts the threshold  $t$  at which  $u$  and  $v$  are first included in the same connected component, i.e., when their representatives in  $H_0$  are merged.

**Component Count Under Filtration.** At a given threshold  $t$ , the model is asked to compute  $\beta_0(K_t)$ , the number of connected components in the complex  $K_t$ .

### D.3 HARD TASKS (FILTRATION STRATEGY DESIGN)

These tasks involve reasoning from desired topological divergence toward filtration design. Given two graphs  $G_1$  and  $G_2$ , the objective is to select or generate a filtration function  $f : E \rightarrow \mathbb{R}$  (or a sequence of filtration thresholds) that induces persistence diagrams  $D_1, D_2$  with maximal Wasserstein distance  $W(D_1, D_2)$ .

**Optimal Filtration Selection.** The model selects a global filtration function  $f \in \mathcal{F}$  from a predefined set (e.g., node degree, edge weight, K-shell, etc.) such that the resulting persistence diagrams maximize  $W_p(D_1, D_2)$  under a  $p$ -Wasserstein metric.

**Non-Uniform Filtration Generation.** Under a fixed filtration function  $f$ , the model selects a non-uniform filtration sequence  $\{t_1, t_2, \dots, t_5\}$  from a discrete set (e.g.,  $\{1, 2, \dots, 10\}$ ) to maximize  $W(D_1, D_2)$ , effectively constructing a coarse but topologically expressive filtration.

### D.4 REAL-WORLD TASKS (REAL-WORLD GRAPH INFERENCE)

These tasks examine whether an LLM can apply persistent homology insights to downstream tasks such as graph classification. Let  $\mathcal{G} = \{G_1, G_2, G_3, G_4\}$  be a set of graphs belonging to two classes. The goal is to design a filtration such that the resulting persistence-based representations  $\{D_i\}$  allow for correct class separation.

**Filtration Selection for Classification.** The model selects a filtration function  $f \in \mathcal{F}$  that yields persistence diagrams best aligned with the class partition of the graphs, typically optimizing for inter-class distance or clustering.

**Filtration Sequence Generation for Classification.** Under a given filtration function  $f$ , the model generates a filtration sequence  $\{t_1, \dots, t_5\}$  that maximizes topological separability between graphs from different classes, typically measured by diagram-level distance or homology-aware clustering accuracy.

## E SUPPLEMENTARY TABLES FOR CROSS-TASK ANALYSES

This appendix provides supplemental materials to support the results presented in the Sec. 3.5. We include:

- **Prompt Style Summary (Table 12)**, outlining the different ways graph structure and task instructions are encoded.
- **Prompt Ablation Results (Table 8)**, showing performance across multiple representation and instruction combinations.
- **Post-Training Evaluation (Table 11)**, reporting transfer results after supervised tuning on small graphs.
- **Compositional Pipeline Evaluation (Table 6)**, examining model performance in hybrid workflows across persistent homology steps.

- **Subgraph Scaling (Table 7)**, assessing accuracy trends as the number of sampled subgraphs increases.

Table 6: Compositional PH pipeline on BZR. The model chooses the filtration in every row.

Settings					Accuracy				
Filtration selection	Filtration values	Diagram generation	Distance comparison	Classifier	GPT-4o	GPT-4.1	Claude	DS-V3	Mistral
LLM	CODE	LLM	LLM	CODE	0.150	0.180	0.780	0.290	0.210
LLM	CODE	CODE	CODE	CODE	0.350	0.350	0.350	0.350	0.350
LLM	CODE	CODE	LLM	CODE	0.250	0.280	0.840	0.780	0.520
LLM	CODE	LLM	CODE	CODE	0.300	0.320	0.190	0.025	0.150
<b>WEIGHT</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>			<b>0.790</b>		
<b>DEGREE</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>			<b>0.610</b>		
<b>BETWEENNESS</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>			<b>0.890</b>		
<b>CLOSENESS</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>	<b>CODE</b>			<b>0.800</b>		

Table 7: Subgraph sampling on ego-Facebook for 1D Simplex Counting. Values are mean accuracy  $\pm$  standard deviation over five runs.

Subsamples	10	20	50	80	100	150
GPT-4o	0.0600 $\pm$ 0.08	0.1400 $\pm$ 0.08	0.1280 $\pm$ 0.04	0.1325 $\pm$ 0.03	0.1733 $\pm$ 0.02	0.1960 $\pm$ 0.02
GPT-4.1-mini	0.1400 $\pm$ 0.10	0.1400 $\pm$ 0.08	0.1520 $\pm$ 0.03	0.1850 $\pm$ 0.03	0.2260 $\pm$ 0.02	0.2413 $\pm$ 0.04

Table 8: Prompt encoding and task formulation ablation shown in a single horizontally merged table across three tasks.

Prompt setting	1D Simplex Counting				Component Merge Time				Component Count Under Filtration			
	Synthetic		LDHFR		Synthetic		LDHFR		Synthetic		LDHFR	
	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude
Text style + Graph theoretic	<b>0.270</b>	0.150	0.170	0.260	0.425	0.270	<b>0.250</b>	0.090	0.785	<b>0.983</b>	0.173	<b>0.780</b>
Text style + Minimal	0.176	0.078	0.273	<b>0.670</b>	0.443	0.225	0.160	0.023	0.093	0.025	0.233	0.085
Text style + Topological	0.160	0.268	0.250	0.250	<b>0.510</b>	0.090	0.125	0.020	0.633	0.790	0.148	0.655
Matrix style + Topological	0.125	0.224	0.118	0.230	0.245	0.090	0.018	0.000	0.643	0.870	0.180	0.610
Code like + Topological	0.128	0.210	0.118	0.160	0.445	0.115	0.220	0.110	0.833	0.980	0.120	0.570

Table 9: Terminology ablation results comparing topological vs. graph theoretic phrasing across three tasks.

Condition	GPT 4.1 mini	GPT 4o
Graph terminology birth	0.67	0.33
Topology terminology birth	0.6533	0.3233
Graph terminology merge	0.02	0.5533
Topology terminology merge	0.0066	0.57
Graph terminology CCUF	0.99	0.8633
Topology terminology CCUF	1.0	0.45

Table 10: Performance on non uniform filtration generation with different prompting strategies (Mean Rank  $\pm$  std)

Prompt Type	GPT 4o	Claude 3 7
Few shot CoT	52.09 $\pm$ 32.84	48.76 $\pm$ 29.99
Rule emphasized	53.11 $\pm$ 34.86	40.17 $\pm$ 31.31

Table 11: Post training on LLaMA-3.1-8B-Instruct with supervision on small graphs and evaluation on medium graphs.

Prompt setting	1D Simplex Counting		Component Merge Time		Component Count Under Filtration	
	Zero-shot	Post-train	Zero-shot	Post-train	Zero-shot	Post-train
Text style + Graph theoretic	0.020	0.200	0.090	0.430	0.160	0.660
Text style + Minimal	0.000	0.160	0.040	0.290	0.090	0.400
Text style + Topological	0.000	0.040	0.080	0.430	0.330	0.480
Matrix style + Topological	0.000	0.060	0.140	0.250	0.210	0.480
Code like + Topological	0.000	0.260	0.010	0.370	0.220	0.540

Table 12: Summary of graph representation and task instruction styles. Full prompt examples are provided in the appendix.

Prompt Style	Description
<i>Graph Representation Styles</i>	
Text-style format	Nodes and edges are described using plain natural language sentences, following human-like narration.
Code-like format	Graph elements are structured using syntax that resembles programming or configuration files, such as lists or dictionaries.
Matrix-based format	The graph is represented by a numerical adjacency matrix, requiring structural inference from tabular data.
<i>Task Instruction Styles</i>	
Topological formulation	Instructions are written using formal terminology from algebraic topology, such as simplex, homology, and filtration.
Graph-theoretic formulation	Tasks are described using standard graph terminology, avoiding domain-specific topological language.
Minimal formulation	Prompts are short and generic, with little or no explicit context, requiring the model to infer the task from minimal cues.