Large Language Models are In-Context Retrieval-Augmented Graph Learners

Anonymous ACL submission

Abstract

Large language models (LLMs) have demonstrated strong reasoning capabilities across a wide range of tasks, particularly those involving unstructured inputs such as natural language or images. However, LLMs face challenges when processing graph-structured data due to their limited understanding of non-Euclidean structures. As a result, without additional fine-tuning or adapting, their performance in graph learning tasks lags largely behind that of graph neural networks (GNNs). In this paper, we show that learning on graph data can be conceptualized as a retrieval-augmented generation (RAG) task, in which specific instances (e.g., nodes or edges) act as queries, and the graph itself functions as the retrieved context. Building on this perspective, we propose a set of in-context learning frameworks designed to improve LLM performance on graph-related tasks. Extensive evaluations demonstrate that our proposed frameworks significantly enhance the performance of LLMs in graph-based scenarios, especially when using pretrained models that cannot be fine-tuned and are accessible only via API.

1 Introduction

Large language models (LLMs) exhibit astonishing reasoning capabilities across a wide variety of real-world tasks, including text generation (Yang et al., 2025), code completion (Guo et al., 2024), and question answering (Lewis et al., 2020). The widespread success of LLMs is often attributed to their in-context learning capabilities (Luo et al., 2024). Building on top of just raw reasoning abilities, LLMs can solve a new task for which they have not been explicitly trained, by being provided with a few examples (few-shot) (Brown et al., 2020) or even with instructions describing the task (zero-shot) (Kojima et al., 2022).

Leveraging their success in handling unstructured data, LLMs have increasingly been applied to tasks involving structured data, such as graphs (Chen et al., 2023; Ye et al., 2024). Essentially, language models are trained on semantic text as next-token predictors, which are inherently capable of learning from any textual input, including graph-like structures represented as text. For example, graphs can be encoded into textual representations, such as edge lists, adjacency matrices, or even graph descriptions in natural language (Ye et al., 2024; Guo et al., 2023; Ge et al., 2025).

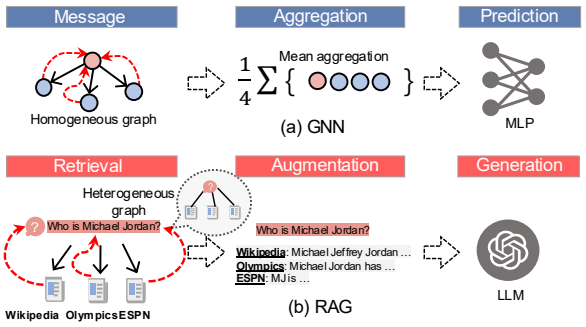


Figure 1: A technical comparison between GNN and RAG. Both leverage contextual information beyond the raw input through *homogeneous* and *heterogeneous* graph structures, respectively.

However, breaking down the graph abstraction into verbal semantic prompts (i.e., graph tokenization) leads to the loss of critical structural information (Guo et al., 2023). We argue that the intrinsic relationships and dependencies between nodes and edges require a deeper understanding of spatial and relational structures, which cannot be fully captured through textual encoding alone. As a result, despite their impressive performance on unstructured tasks, LLMs without additional fine-tuning struggle to match the effectiveness of specialized models like graph neural networks (GNNs) (Kipf and Welling, 2017; Velickovic et al., 2018).

As LLMs continue to evolve, there is growing research interest in context engineering (Mei

et al., 2025), which refers to the systematic design and management of the input context provided to LLMs in order to guide their behavior and improve task performance. Recent context engineering techniques, such as retrieval-augmented generation (RAG) (Lewis et al., 2020) and chain-of-thought (CoT) reasoning (Wei et al., 2022b), have been introduced to mitigate the limitations of LLMs in complex reasoning tasks.

Indeed, GNNs propagate and aggregate information from neighboring nodes (Gilmer et al., 2017) in an inductive manner, a process that is conceptually similar to in-context learning (Hamilton et al., 2017). While LLMs are proficient at interpreting and reasoning with unstructured data, their ability to perform in-context learning on graph-structured data remains largely unexplored. This naturally raises the question: *Are LLMs effective in-context graph learners?*

In this work, we investigate the in-context learning capabilities of LLMs on graph-structured data. Our findings reveal that message passing—the core mechanism of GNNs—can be interpreted as a recursive, node-level instantiation of RAG, where each node queries its local neighborhood as contextual input (see Figure 1 for an illustration). Framed in this way, message passing aligns naturally with principles of context engineering. We further show that this RAG-like mechanism enables even simple models, such as multilayer perceptrons (MLPs), to achieve competitive performance on graph-based tasks when supplied with well-engineered graph context during inference.

Building on this insight, we propose a series of RAG-based in-context frameworks, i.e., QUERYRAG, LABELRAG, and FEWSHOTRAG, which leverage graph structure as inherent context to enhance the in-context learning capabilities of LLMs. Unlike standard RAG approaches that require external retrieval mechanisms, our methods inherently retrieve relevant context by utilizing the local graph neighborhood of each node. Experimental results show that LLMs augmented with our in-context RAG frameworks significantly outperform their zero-shot and standard few-shot counterparts.

We outline our main contributions below.

- **GNNs can be conceptualized as RAG networks.** We are the first to relate message-passing GNNs to RAG, both of which are powerful tools for facilitating context-aware tasks. By drawing this connection, we provide new insights into leveraging contextual information for LLMs from proximal queries and corresponding labels.
- **In-context RAG frameworks.** Motivated by the connection between RAG and message-passing GNNs, we introduce three specialized in-context RAG frameworks: QUERYRAG, LABELRAG, and FEWSHOTRAG. These frameworks leverage the query, label, or both from retrieved nodes/edges to enhance the understanding of LLMs on graph data.
- **Experimental results.** We conduct extensive experiments with LLMs on node/edge classification tasks to evaluate the effectiveness of our proposed frameworks. Experimental results show that, with our frameworks, an off-the-shelf LLM without additional fine-tuning can match or even surpass the performance of supervised GNNs and fine-tuned graph LLMs.

2 Related Work

In-context learning. In-context learning (ICL) is a task adaptation strategy where an LLM is provided with examples or demonstrations within the input prompt to guide its responses for various tasks (Brown et al., 2020). ICL enables LLMs to adapt to unseen tasks by observing a few demonstrations in their input, without requiring additional training data or fine-tuning (Luo et al., 2024). Since in-context demonstrations are crucial for ICL performance, various strategies have been proposed to improve demonstration selection, including retrieving semantically similar examples (Liu et al., 2022), employing chain-of-thought reasoning (Kojima et al., 2022), and decomposing tasks into subproblems using least-to-most prompting (Zhou et al., 2023). Additionally, research has shown that the effectiveness of ICL can be significantly improved through fine-tuning (Wei et al., 2022a) or adaptation during pretraining (Min et al., 2022), even for smaller language models (Schick and Schütze, 2021).

Retrieval-augmented generation. Retrieval-augmented generation (RAG) is a context-based strategy that enhances LLMs by dynamically retrieving relevant information from a knowledge base or document corpus (Lewis et al., 2020). Unlike ICL, which relies solely on in-prompt demonstrations, RAG provides external context

to guide LLM responses, making it particularly effective for tasks that require up-to-date or domain-specific information where static prompts may be insufficient (Fan et al., 2024). In addition, RAG can be viewed as a variant of ICL, where the input context is enriched by retrieving relevant external knowledge to support downstream reasoning or generation. More recently, several studies have proposed graph-aware retrieval mechanisms to enhance RAG by providing structured, graph-based contextual information (Edge et al., 2024; He et al., 2024b; Wang et al., 2025).

3 Preliminary

Notations. We define a text-attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ with N nodes, where $\mathcal{V}$ is the set of nodes and $\mathcal{E} \subseteq (\mathcal{V} \times \mathcal{V})$ is set of edges connecting them. For each node $v_i \in \mathcal{V}$ is associated with a textual attribute x_i, and $\mathcal{X} = \{x_i | v_i \in \mathcal{V}\}$ is the attribute set. We consider a fundamental task, namely node classification, which predicts the class label of a node v_i. This task can be conceptualized and approached as a text classification problem, but with additional graph context. For simplicity, let $\mathcal{Y} = \{y_i\}_{i=1}^{N}$ denote the set of ground-truth labels of nodes or edges.

In-context learning. ICL is a paradigm that enables language models to learn tasks using only a few examples as demonstrations. Formally, given a pretrained LLM, the model is expected to generate an output r for a new query input q by conditioning on a prompt that includes in-context examples along with the query input, i.e.,

$$r = \text{LLM}(q, \mathcal{C}_K), \tag{1}$$

where $\mathcal{C}_K = (q_1, r_1, ..., q_K, r_K)$ is the demonstration set and K is the number of demonstrations. Each pair of input-output examples $(q_i, r_i) \in \mathcal{C}_K$, also called *shots*, is sampled from a known distribution or a training dataset. Without loss of generality, we omit the task-specific instructions for the generation task hereafter for simplicity.

Retrieval-augmented generation. RAG combines a retrieval mechanism with a generative model to generate responses by leveraging external knowledge. Formally, given a query q and a corpus of documents $\mathcal{D}$, the RAG process can be formulated as:

$$r = \text{LLM}(q, \{c_i\}_{i=1}^{K}), \text{ s.t. } D(q, c_i) < \epsilon, \tag{2}$$

where $\{c_i\}_{i=1}^{K}$ represents the set of top-K documents or chunks retrieved from the corpus $\mathcal{D}$. We use the distance metric $D(\cdot)$ and a sufficiently small constant ϵ to represent the top-K retrieval step, which computes the relevance of the textual corpus in $\mathcal{D}$ to the query q and selects the top-K documents or chunks to serve as additional context for LLMs. The distance function $D(\cdot)$ can be implemented as cosine similarity $D(x, y) = \frac{x \cdot y}{\|x\|\|y\|}$.

Graph neural networks. Most GNNs rely on iteratively propagating information between neighboring nodes in the graph, i.e., message passing (Gilmer et al., 2017). In general, the message-passing step in GNNs is formalized as:

$$h_i = \text{UPD}\left(\{x_i, \text{AGG}\left(\{x_j : v_j \in \mathcal{N}(v_i)\}\right)\}\right), \tag{3}$$

where $\mathcal{N}(v_i)$ denotes the neighborhood set of node v_i. AGG and UPD are the aggregation and update function, respectively. Basically, both AGG and UPD can be learnable functions or neural networks, which leads to different variants of GNNs (Velickovic et al., 2018; Wu et al., 2019; Hamilton et al., 2017). Since an MLP network can be considered a special case of a GNN, we can formulate the MLP below based on Eq. (3) as $h_i = \text{UPD}(x_i)$.

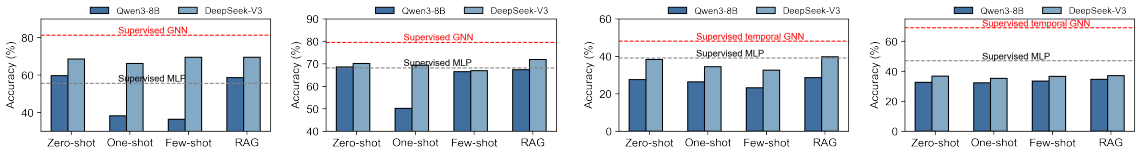
4 Present work

In this section, we begin with an empirical study to investigate the in-context learning capabilities of LLMs in graph learning tasks. Next, we establish the connection between RAG and GNNs. Finally, we introduce three novel in-context RAG frameworks designed to improve LLMs' understanding of graph data.

4.1 Empirical study

We introduce concrete node classification and edge classification tasks on graphs and investigate prompting techniques in in-context learning settings. Our goal is to evaluate the in-context learning capabilities of LLMs on graph data.

Experimental settings. We experiment with four representative text-attributed graph datasets: Cora and Pubmed (Sen et al., 2008) for node classification, and Stack elec and Stack ubuntu for edge classification. We use QWEN3-8B(Yang et al., 2025) and DEEPSEEK-V3(Liu et al., 2024a) as base models, representing state-of-the-art LLMs with small



(a) Node Classification (Cora and Pubmed)

(b) Edge Classification (Stack elec and Stack ubuntu)

Figure 2: Classification results on four datasets. GNN and MLP are supervisedly trained on partial nodes/edges.

and large scales, respectively. Our approach to in-context learning follows previous works (Chen et al., 2023; Brown et al., 2020). Specifically, we include zero-shot, one-shot, few-shot (Brown et al., 2020), and RAG (Lewis et al., 2020) for standard settings of LLMs. For comparison, we include supervised GNNs (i.e., GCN (Kipf and Welling, 2017) for node classification and DyGFormer (Xu et al., 2020) for edge classification) and MLP as baselines. Both models are trained in a supervised manner on the training set. Due to space limitations, we defer the detailed experimental settings to the Appendix C, along with results for other graph datasets and additional LLMs.

Observation I: LLMs are not inherently effective in-context graph learners. The in-context learning results of QWEN3-8B and DEEPSEEK-V3 on text-attributed graphs are presented in Figure 2. Both LLMs perform significantly worse than supervised baselines such as GNNs in the zero-shot setting, highlighting the challenge of graph-based reasoning without explicit in-context examples. More surprisingly, in the one-shot and few-shot settings, both LLMs exhibit a further performance drop compared to zero-shot, particularly on the Cora and Stack elec datasets. This suggests that LLMs may struggle to effectively utilize in-context examples, and in some cases, these examples can even negatively impact their performance.

Observation II: RAG enhances LLM performance on graph learning. Compared to the zero-shot and few-shot settings, RAG achieves higher accuracy for both LLMs, indicating that incorporating graph node-specific textual content as context benefits LLM-based reasoning. LLMs augmented with RAG show improved performance and even surpass the supervised MLP on both datasets. This indicates the importance of context-aware retrieval for improving the downstream performance of LLMs. However, RAG still falls significantly behind supervised GNNs, which highlights the limitations of in-context learning for graph tasks compared to models explicitly trained on node features. This underscores the gap between in-context learning and fully supervised methods in graph learning, suggesting that while retrieval-based augmentation helps, it does not yet fully bridge the performance disparity.

Overall, the findings suggest that LLMs have limited but promising in-context learning capabilities for the node and edge classification tasks. This highlights the need for further research to improve their reasoning abilities on graph-structured data without requiring additional fine-tuning.

4.2 Connecting GNNs with RAG

In our empirical experiments, RAG generally achieves the best performance among in-context baselines, although it still significantly underperforms supervised GNNs. This motivates us to explore potential improvements in RAG for graph learning tasks.

In the literature, it has been shown that improving the quality of retrieved context significantly enhances the performance of LLMs (Ram et al., 2023). However, in RAG, the textual attributes of nodes (i.e., the *queries*) are treated as an external *corpus* to retrieve relevant textual context for each node, which differs from traditional RAG methods (Lewis et al., 2020). In this regard, the objective of retrieving 'neighborhood' information from the graph has led us to explore the connections between RAG and GNNs.

Both RAG and GNNs are context-based methods that leverage contextual information in a similar way. Specifically, RAG retrieves relevant information from an external corpus based on the query to serve as context, while GNNs aggregate information from the local neighborhood of the query node within a graph. Given these similarities, we are motivated to establish a formal connection between RAG and GNNs for further exploration. To this end, we first rewrite the formulations of RAG and

GNNs below.

$$r_i = \text{GEN}\left(\left\{q_i, \text{AUG}\left(\left\{c_j : c_j \in \mathcal{D}(q_i)\right\}\right)\right\}\right), \tag{4}$$

$$h_i = \text{UPD}\left(\left\{x_i, \text{AGG}\left(\left\{x_j : v_j \in \mathcal{N}(v_i)\right\}\right)\right\}\right), \tag{5}$$

where GEN and AUG are the generation and augmentation step in RAG. Typically, GEN is implemented with an off-the-shelf LLM while AUG involves prompt engineering to condition retrieved contexts. $\mathcal{D}(q_i) = \{q_j | D(q_i, q_j) < \epsilon\}$ is the set of retrieved documents that are close to the query q_i. From Eqs. (4) and (5), we observe that RAG and GNNs share a similar formulation across the *retrieval*, *augmentation*, and *generation* steps.

4.3 Graph-guided in-context RAG frameworks

Building on the inherent connections between GNNs and RAG, we propose three graph-guided in-context RAG frameworks: QueryRAG, LabelRAG, and FewshotRAG, as illustrated in Figure 3. Specifically, QueryRAG utilizes the *query* context of neighboring nodes from the graph, while LabelRAG incorporates the *label* information. Combining these two approaches, FewshotRAG leverages both *query* and *label* information from graph-guided contexts.

From GNN to QueryRAG. Although we have demonstrated that GNNs can be conceptualized as RAG networks, there are key differences between them. Unlike RAG, where context is dynamically retrieved from an external corpus, the context in GNNs is inherently relational, relying on the graph's topology and interactions between nodes. This makes GNNs particularly well-suited for graph learning tasks, where relationships and dependencies play a critical role. Furthermore, the retrieved context in GNNs is more akin to retrieving a *similar query* based on node features or representations, rather than relying on an external corpus, as is typical in vanilla RAG. In this regard, we propose the concept of QueryRAG, which utilizes the queries from other queries as the external corpus for retrieval:

$$r_i = \text{GEN}\left(\left\{q_i, \text{AUG}\left(\left\{q_j : v_j \in \mathcal{N}(v_i)\right\}\right)\right\}\right). \tag{6}$$

In QueryRAG, we take inspiration from GNNs and utilize the text attributes of neighboring nodes $\mathcal{N}$ as inherently retrieved context. Each neighboring node contributes to the contextual understanding of the query node. By treating the local graph neighborhood as a source of context, QueryRAG combines the strengths of GNN-style aggregation with the flexibility of generative models. This design enables QueryRAG to effectively handle tasks that require relational reasoning. Additionally, this inherently relational retrieval mechanism simplifies the workflow, as it does not depend on an external corpus or complex similarity-based retrieval pipelines. In QueryRAG, q_i can represent a specific query in natural language understanding tasks or the textual attributes of a node in node classification tasks.

From label propagation to LabelRAG. Since we have established connections between RAG and GNNs, particularly message-passing GNNs, we can conceptualize graph learning as an RAG-specific task and leverage different message-passing architectures to develop advanced RAG variants. For instance, QueryRAG utilizes only the 'query' information from neighboring or retrieved nodes, leaving label information unexplored. In fact, propagating label information among adjacent nodes is an effective message-passing strategy. This process, commonly known as label propagation, iteratively updates node labels by aggregating information from their neighbors (Zhou et al., 2003). Formally, label propagation at each iteration can be expressed in a message-passing framework:

$$\begin{aligned}\hat{y}_i &= (1-\alpha)y_i + \frac{\alpha}{|\mathcal{N}(v_i)|}\sum_{v_j \in \mathcal{N}(v_i)} y_j, \\ &= \text{UPD}\left(\{y_i, \text{AGG}\left(\{y_j : v_j \in \mathcal{N}(v_i)\}\right)\}\right),\end{aligned} \tag{7}$$

where α is a hyperparameter balancing the influence of neighboring labels and the initial label. Recent studies have also shown that propagating node labels during message passing in GNNs can be an effective strategy (Shi et al., 2021; Huang et al., 2021; Li et al., 2023). Building on this insight, we propose LabelRAG, which extends the concept of label propagation to RAG learning:

$$r_i = \text{GEN}\left(\left\{q_i, \text{AUG}\left(\left\{r_j : v_j \in \mathcal{N}(v_i)\right\}\right)\right\}\right), \tag{8}$$

where r_j is the ground-truth response of query q_i. In LabelRAG, we incorporate label information as part of the retrieved context to enhance

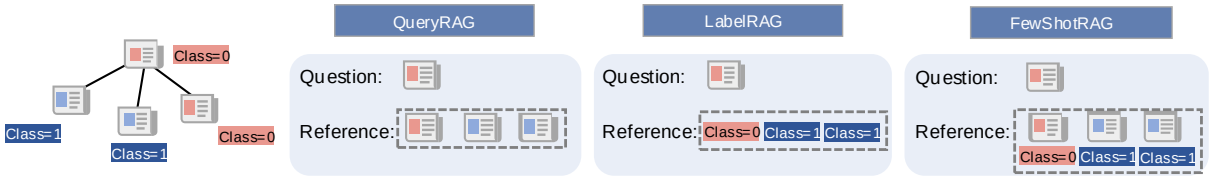


Figure 3: Illustration of the proposed QUERYRAG, LABELRAG, and FEWSHOTRAG frameworks. Specifically, QUERYRAG uses the retrieved queries as references, LABELRAG incorporates only the corresponding labels, and FEWSHOTRAG combines both query and label information as few-shot context.

LLMs' comprehension of the query. This mechanism aligns with the principles of label propagation in GNNs, where labels from neighboring nodes contribute to a node's representation. However, LABELRAG extends this idea to retrieval-augmented frameworks by embedding label context directly into the prompts. The augmented context, $\text{AUG}(\{y_j : v_j \in \mathcal{N}(v_i)\})$, provides semantically meaningful information that enriches the generative model's understanding of the query q_i. This enables the model to leverage both the semantic information from the query and the relational label context derived from the graph structure. This approach is particularly beneficial in scenarios where label correlations and dependencies play a crucial role, as it allows the model to capture structured relationships beyond the query's standalone textual features.

FEWSHOTRAG: few-shot learning on graphs. Both QUERYRAG and LABELRAG utilize external query and label information as contextual inputs to enhance the generation process, respectively. They follow similar principles to few-shot in-context learning, where K examples, consisting of queries and their corresponding responses, are provided to improve LLM comprehension and performance. Since incorporating query and label information as external context helps bridge the gap between the input and the desired output, a natural extension is to combine QUERYRAG and LABELRAG in a manner similar to few-shot learning:

$$r_i = \text{GEN}\left(\left\{q_i, \text{AUG}\left(\left\{(q_j, r_j) : v_j \in \mathcal{N}(v_i)\right\}\right)\right\}\right), \tag{9}$$

where the retrieved tuples (q_j, r_j) are used as context for the query q_i. Eq. (9) illustrates the concept of leveraging both query and label information from neighboring nodes as in-context examples, which we refer to as FEWSHOTRAG. FEWSHOTRAG enhances LLM reasoning in graph-based tasks by providing richer contextual guidance through both semantic query information and relational label context.

Remarks. QUERYRAG, LABELRAG, and FEWSHOTRAG are RAG architectures that integrate insights from in-context learning and message-passing GNNs in different ways. We provide a comparison of various in-context learning and message-passing approaches in Table 3. A given query q_i may represent the textual attributes of a node in a node classification task or a specific question in a question-answering task. Similarly, r_i can correspond to the ground-truth label of a node or the generated response. We use the one-hop neighbors of a node as the retrieved context for the query; however, the context can be extended to include multi-hop neighbors or defined based on proximity-based retrieval methods (Lewis et al., 2020). For comparison, we present the results on Cora and Pubmed in Figure 1. The promising results suggest that incorporating structured graph context provides valuable guidance for LLMs in reasoning over graph data.

5 Experiments

In this section, we present a rigorous evaluation of general-purpose LLMs augmented with our proposed in-context RAG frameworks. The experiments are divided into two main parts: a comparative analysis of QUERYRAG, LABELRAG, and FEWSHOTRAG against state-of-the-art GNNs and graph LLMs on the node and edge classification tasks, and an investigation of how different configurations impact the performance of general-purpose LLMs. Due to space limitations, we defer the detailed experimental settings to Appendix C and additional results to Appendix D. The source code, including the datasets and all necessary scripts for reproducing the results, is made available at `https://anonymous.4open.science/r/In-Context-RAG`.

	Cora	Pubmed	Fitness	History	Children	Photo	Computers	arxiv
🔥 MLP	$55.6_{\pm 0.1}$	$68.1_{\pm 0.5}$	$78.8_{\pm 0.3}$	$77.4_{\pm 0.3}$	$47.1_{\pm 0.4}$	$56.2_{\pm 0.2}$	$64.7_{\pm 0.4}$	$46.5_{\pm 0.2}$
❄ Label Propagation (Zhou et al., 2003)	$75.4_{\pm 0.0}$	$72.1_{\pm 0.0}$	$82.3_{\pm 0.0}$	$75.2_{\pm 0.0}$	$45.9_{\pm 0.0}$	$69.2_{\pm 0.0}$	$70.1_{\pm 0.0}$	$67.2_{\pm 0.0}$
🔥 GCN (Kipf and Welling, 2017)	$81.3_{\pm 0.7}$	$79.5_{\pm 0.3}$	$90.0_{\pm 0.2}$	$83.3_{\pm 0.8}$	$52.4_{\pm 1.1}$	$83.8_{\pm 0.2}$	$86.1_{\pm 0.4}$	$72.5_{\pm 0.2}$
🔥 GAT (Velickovic et al., 2018)	$82.1_{\pm 0.9}$	$79.1_{\pm 0.6}$	$91.5_{\pm 1.0}$	$83.4_{\pm 1.2}$	$53.1_{\pm 0.2}$	$83.3_{\pm 0.2}$	$86.6_{\pm 0.7}$	$72.7_{\pm 0.3}$
🔥 GraphSAGE (Hamilton et al., 2017)	$78.6_{\pm 0.5}$	$78.9_{\pm 0.6}$	$89.8_{\pm 0.1}$	$82.4_{\pm 0.8}$	$52.0_{\pm 1.0}$	$82.7_{\pm 0.7}$	$84.4_{\pm 0.5}$	$72.9_{\pm 0.5}$
🔥 RevGAT (Li et al., 2021)	$81.9_{\pm 0.2}$	$80.1_{\pm 0.9}$	**$91.6_{\pm 0.7}$**	$83.7_{\pm 0.7}$	$52.9_{\pm 0.6}$	$84.8_{\pm 0.5}$	$87.0_{\pm 1.2}$	**$74.5_{\pm 0.4}$**
🔥 GLEM (Zhao et al., 2023b)	82.9	85.7	88.9	84.0	52.1	79.8	80.2	74.7
🔥 TAPE (He et al., 2024a)	83.1	86.5	89.0	**85.8**	53.4	82.1	84.5	75.2
🔥 PRODIGY (Huang et al., 2023)	80.5	82.4	85.7	84.3	50.9	80.5	86.1	74.3
🔥 LLaGA (Chen et al., 2024)	83.4	**86.8**	90.0	**84.9**	**53.2**	83.1	88.0	75.2
❄ GraphCLIP (Zhu et al., 2025)	68.6	74.2	67.7	60.2	40.4	61.2	66.9	51.0
QWEN3-8B (small scale)								
❄ Zero-shot (Kojima et al., 2022)	59.7	68.6	45.1	39.2	24.6	51.2	54.1	39.4
❄ One-shot (Brown et al., 2020)	38.2	50.1	22.0	36.5	23.2	35.7	38.7	15.7
❄ Few-shot (Brown et al., 2020)	36.4	66.5	33.9	40.1	25.7	34.5	46.4	15.4
❄ RAG (Lewis et al., 2020)	58.5	67.4	51.4	41.3	22.0	48.5	55.2	40.9
❄ QUERYRAG	62.3	71.2	56.5	66.2	26.1	55.5	65.2	43.5
❄ LABELRAG	81.6	79.8	80.4	81.5	50.4	80.6	81.5	71.7
❄ FEWSHOTRAG	**83.9**	78.1	87.6	80.6	46.9	77.3	86.9	72.2
DEEPSEEK-V3 (large scale)								
❄ Zero-shot (Kojima et al., 2022)	68.5	70.1	46.2	58.5	25.0	50.7	55.4	42.1
❄ One-shot (Brown et al., 2020)	66.2	69.3	44.3	57.8	24.3	45.0	53.5	41.0
❄ Few-shot (Brown et al., 2020)	69.5	66.9	49.5	61.0	28.1	48.5	60.3	46.8
❄ RAG (Lewis et al., 2020)	69.4	71.9	57.8	70.2	32.5	52.4	60.7	57.2
❄ QUERYRAG	75.2	78.3	59.9	71.4	40.0	58.4	66.3	65.5
❄ LABELRAG	84.0	**88.5**	88.7	83.8	52.2	**84.5**	**87.4**	74.2
❄ FEWSHOTRAG	**86.5**	88.0	**90.2**	85.1	**54.0**	**85.7**	**88.6**	**75.8**

Table 1: Node classification results (%) on eight text-attributed graphs. The top-3 best performances in each dataset are highlighted in **red**, blue, and **gray**. ❄ indicates training-free evaluation (no further training or fine-tuning); 🔥 indicates supervised training on the training set.

5.1 Main results

Node classification. We also present the results compared to various advanced baselines in Table 1. Several key observations emerge from the results. ❶ **Graph-guided retrieval significantly improves LLM performance:** All three RAG variants outperform standard zero-shot baselines (e.g., zero-shot, one-shot, and few-shot learning) on all datasets, demonstrating the effectiveness of graph-structured retrieval for augmenting LLM reasoning. ❷ **Off-the-shelf-LLMs can match the performance of supervised or fine-tuned methods:** With a sufficiently large LLM (e.g., DEEPSEEK-V3), the performance often matches or exceeds that of specialized graph LLMs like GLEM and TAPE. These results highlight the advantage of retrieval-augmented in-context learning, which effectively leverages graph structure as a source of context without requiring fine-tuning. ❸ **LABELRAG is particularly effective in facilitating in-context learning:** Simply incorporating label information from neighboring nodes (LABELRAG) results in a substantial accuracy gain compared to QUERYRAG, suggesting that labeled neighborhood context is highly beneficial for LLM-based graph learning. ❹ **FEWSHOTRAG shows mixed results compared to LABELRAG:** While FEWSHOTRAG integrates both query and label information, it does not consistently outperform LABELRAG. This could be due to the increased input length and complexity, which may hinder LLMs from efficiently processing relevant information. This aligns with prior findings that LLMs struggle with long-context reasoning (Liu et al., 2024b).

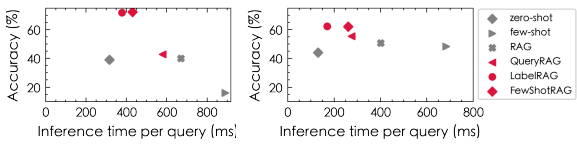
Edge classification. We present the edge classification results on temporal graph datasets in Table 2, comparing our proposed retrieval-augmented methods with several representative temporal GNNs. Since most existing graph LLMs are not originally designed for edge-level prediction, we exclude them from this comparison. From the results, we draw several observations. ❶ **Zero-/Few-shot LLMs without retrieval are insufficient:** Without retrieval, LLMs struggle on all datasets, with average performance far behind dedicated GNN models. This highlights the necessity of graph-contextual augmentation to unlock the reasoning capacity of LLMs in dynamic graph tasks. ❷

	Enron	Amazon movies	Yelp
MLP	$22.7_{\pm 0.3}$	$40.5_{\pm 1.1}$	$51.3_{\pm 1.8}$
TGAT (Xu et al., 2020)	$55.2_{\pm 0.3}$	$58.8_{\pm 0.5}$	$62.2_{\pm 1.1}$
JODIE (Kumar et al., 2019)	$64.8_{\pm 0.6}$	**$60.4_{\pm 0.7}$**	OOM
DyRep (Trivedi et al., 2019)	**$64.3_{\pm 0.6}$**	OOM	OOM
CAWN (Wang et al., 2021)	$56.9_{\pm 1.3}$	$60.5_{\pm 0.4}$	$60.8_{\pm 0.3}$
DyGFormer (Yu et al., 2023)	$55.7_{\pm 0.6}$	$59.7_{\pm 1.9}$	**$63.6_{\pm 0.8}$**
QWEN3-8B (small scale)			
Zero-shot (Kojima et al., 2022)	11.4	34.1	44.5
One-shot (Brown et al., 2020)	10.5	35.2	47.4
Few-shot (Brown et al., 2020)	11.2	37.6	49.1
RAG (Lewis et al., 2020)	12.8	33.4	51.2
QUERYRAG	15.4	42.2	55.7
LABELRAG	53.1	50.3	62.4
FEWSHOTRAG	58.2	53.2	62.0
DEEPSEEK-V3 (large scale)			
Zero-shot (Kojima et al., 2022)	13.7	37.2	50.7
One-shot (Brown et al., 2020)	12.3	38.9	55.0
Few-shot (Brown et al., 2020)	11.4	39.7	56.2
RAG (Lewis et al., 2020)	13.5	39.6	58.9
QUERYRAG	23.4	41.2	55.0
LABELRAG	61.2	61.6	67.0
FEWSHOTRAG	**65.3**	**63.3**	**69.7**

Table 2: Edge classification results (%) on three text-attributed graphs. We defer full results on six datasets to Table 6 in Appendix D.

Retrieval-augmented LLMs outperform classical temporal GNNs on large-scale datasets: On datasets like Yelp, where early methods such as JODIE and DyRep fail due to memory constraints, retrieval-augmented LLMs, especially DEEPSEEK-V3, achieve competitive or superior results without task-specific supervision. This demonstrates the scalability advantage of in-context learning with structured retrieval. ❸ **Label-aware RAG consistently boosts performance:** Across both LLM architectures, incorporating labeled neighbor context significantly improves edge classification accuracy compared to naive retrieval. ❹ **Large LLMs close the gap with state-of-the-art GNNs:** On datasets like Amazon movies and Yelp, the best LABELRAG or FEWSHOTRAG with DEEPSEEK-V3 match or surpass the performance of DyGFormer and CAWN, which are designed specifically for dynamic temporal graphs. These results indicate that large models with graph-guided prompting can serve as strong alternatives to fully supervised architectures.

Inference latency To better understand the trade-off between model performance and computational cost, we analyze the inference time and accuracy of different ICL methods on the arxiv and Yelp datasets, as shown in Figure 4. Each point corresponds to a specific ICL method, where the x-axis represents the average inference time per query, and the y-axis denotes the corresponding accuracy. Among all methods, LABELRAG achieves a significant improvement in accuracy with only a moderate increase in inference time, striking a favorable balance between performance and efficiency. FEWSHOTRAG further improves accuracy by incorporating both query and label information as context, with only a slight additional latency compared to LABELRAG. Importantly, the inference time for all proposed methods remains within a practically acceptable range. In contrast, although both FEWSHOTRAG and few-shot methods use a similar number of in-context examples, the latter retrieves examples without considering structural or semantic relevance to the query. This often leads to less informative context and increased input length, resulting in longer inference times and inferior performance. These findings highlight the advantage of graph-aware context engineering, where selecting structurally relevant context leads not only to better accuracy but also to more efficient inference.



(a) arxiv (b) Yelp

Figure 4: Speed (ms) versus accuracy of different ICL methods on the arxiv and Yelp datasets.

6 Conclusion

In this work, we establish a close connection between message-passing GNNs and RAG, offering a unified perspective on how GNNs utilize contextual information in graph-structured data. Building on this connection, we introduce three graph-guided in-context RAG frameworks—QUERYRAG, LABELRAG, and FEWSHOTRAG —that treat the graph structure as inherent context for in-context learning. These frameworks enable LLMs to perform graph-related tasks effectively without any fine-tuning or additional training. Extensive experiments across diverse datasets show that our RAG-based methods consistently enhance the performance of LLMs, in some cases even surpassing state-of-the-art supervised GNNs and specialized graph LLMs. Beyond empirical improvements, our work highlights a broader conceptual link between GNNs and RAG: by viewing message passing as a recursive retrieval step, we bridge structured and unstructured learning paradigms, demonstrating that LLMs can serve as in-context graph learners.

Limitations

Despite the promising results, our work has certain limitations. First, our approach assumes homophily in the graph structure, which may not generalize well to heterophilic graphs, where nodes with dissimilar attributes or labels are more likely to be connected. Second, our experiments primarily focus on static graphs, leaving the performance on dynamic or temporal graphs unexplored. Finally, while GNNs enhance the understanding of a query node by recursively aggregating information from its local neighborhood, RAG typically retrieves only one-hop context for LLMs, which may limit their ability to handle complex multi-hop reasoning tasks (Yang et al., 2018).

Ethics Statement

Our study uses only publicly available text-attributed graph benchmarks and does not involve human-subject experiments or the collection of new personal data. Since some benchmarks contain real-world, user-generated text (e.g., emails and reviews), we only report aggregate results and do not attempt to identify individuals or recover sensitive attributes. Our approach focuses on classification-style outputs and operates without task-specific supervision, which reduces risks associated with open-ended content generation. We use all benchmarks strictly for academic research/benchmarking per their original terms; we release only derived metrics and code to reproduce results, not redistributed raw data.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat S. Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 68 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *CoRR*, abs/2404.14219.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *NeurIPS*.

Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. 2023. Graphllm: Boosting graph reasoning ability of large language model. *CoRR*, abs/2310.05845.

Runjin Chen, Tong Zhao, Ajay Kumar Jaiswal, Neil Shah, and Zhangyang Wang. 2024. Llaga: Large language and graph assistant. In *ICML*. OpenReview.net.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and Jiliang Tang. 2023. Exploring the potential of large language models (llms)in learning on graphs. *SIGKDD Explor.*, 25(2):42–61.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Keyu Duan, Qian Liu, Tat-Seng Chua, Shuicheng Yan, Wei Tsang Ooi, Qizhe Xie, and Junxian He. 2023. Simteg: A frustratingly simple approach improves textual graph learning. *CoRR*, abs/2308.02565.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph RAG approach to query-focused summarization. *CoRR*, abs/2404.16130.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on RAG meeting llms: Towards retrieval-augmented large language models. In *KDD*, pages 6491–6501. ACM.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2024. Talk like a graph: Encoding graphs for large language models. In *ICLR*. OpenReview.net.

Yuyao Ge, Shenghua Liu, Baolong Bi, Yiwei Wang, Lingrui Mei, Wenjie Feng, Lizhe Chen, and Xueqi Cheng. 2025. Can graph descriptive order affect solving graph problems with llms? In *ACL (1)*, pages 6404–6420. Association for Computational Linguistics.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. *CoRR*, abs/2401.14196.

Jiayan Guo, Lun Du, and Hengyu Liu. 2023. Gpt4graph: Can large language models understand graph structured data ? an empirical evaluation and benchmarking. *CoRR*, abs/2305.15066.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. In *NeurIPS*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *ICML*. OpenReview.net.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *ICLR*. OpenReview.net.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2024a. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *ICLR*. OpenReview.net.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024b. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *NeurIPS*.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*.

Qian Huang, Horace He, Abhay Singh, Ser-Nam Lim, and Austin R. Benson. 2021. Combining label propagation and simple models out-performs graph neural networks. In *ICLR*. OpenReview.net.

Qian Huang, Hongyu Ren, Peng Chen, Gregor Krzmanc, Daniel Zeng, Percy Liang, and Jure Leskovec. 2023. PRODIGY: enabling in-context learning over graphs. In *NeurIPS*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR (Poster)*. OpenReview.net.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS*.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *KDD*, pages 1269–1278. ACM.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*.

Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. 2021. Training graph neural networks with 1000 layers. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 6437–6449. PMLR.

Jintang Li, Zheng Wei, Jiawang Dan, Jing Zhou, Yuchang Zhu, Ruofan Wu, Baokun Wang, Zhang Zhen, Changhua Meng, Hong Jin, Zibin Zheng, and Liang Chen. 2023. Hetero^{2}net: Heterophily-aware representation learning on heterogenerous graphs. *CoRR*, abs/2310.11664.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *DeeLIO@ACL*, pages 100–114. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Trans. Assoc. Comput. Linguistics*, 12:157–173.

Man Luo, Xin Xu, Yue Liu, Panupong Pasupat, and Mehran Kazemi. 2024. In-context learning with retrieved demonstrations for language models: A survey. *CoRR*, abs/2401.11624.

Lingrui Mei, Jiayu Yao, Yuyao Ge, Yiwei Wang, Baolong Bi, Yujun Cai, Jiazhi Liu, Mingyu Li, Zhong-Zhi Li, Duzhen Zhang, Chenlin Zhou, Jiayi Mao, Tianze Xia, Jiafeng Guo, and Shenghua Liu. 2025. A survey of context engineering for large language models. *Preprint*, arXiv:2507.13334.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. Metaicl: Learning to learn in context. In *NAACL-HLT*, pages 2791–2809. Association for Computational Linguistics.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Trans. Assoc. Comput. Linguistics*, 11:1316–1331.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *NAACL-HLT*, pages 2339–2352. Association for Computational Linguistics.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine*, 29(3):93–93.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. 2021. Masked label prediction: Unified message passing model for semi-supervised classification. In *IJCAI*, pages 1548–1554. ijcai.org.

Gemma Team. 2024. Gemma.

Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. 2019. Dyrep: Learning representations over dynamic graphs. In *ICLR*.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR (Poster)*. OpenReview.net.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023. Can language models solve graph problems in natural language? In *NeurIPS*.

Shijie Wang, Wenqi Fan, Yue Feng, Shanru Lin, Xinyu Ma, Shuaiqiang Wang, and Dawei Yin. 2025. Knowledge graph retrieval-augmented generation for llm-based recommendation. In *ACL (1)*, pages 27152–27168. Association for Computational Linguistics.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive representation learning in temporal networks via causal anonymous walks. In *ICLR*. OpenReview.net.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022a. Finetuned language models are zero-shot learners. In *ICLR*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying graph convolutional networks. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR.

Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. In *ICLR*. OpenReview.net.

Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, Weiwei Deng, Qi Zhang, Lichao Sun, Xing Xie, and Senzhang Wang. 2023. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. In *NeurIPS*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380. Association for Computational Linguistics.

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2024. Language is all a graph needs. In *EACL (Findings)*, pages 1955–1973. Association for Computational Linguistics.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. 2023. Towards better dynamic graph learning: New architecture and unified library. In *NeurIPS*.

Jiasheng Zhang, Jialin Chen, Menglin Yang, Aosong Feng, Shuang Liang, Jie Shao, and Rex Ying. 2024.

DTGB: A comprehensive benchmark for dynamic text-attributed graphs. In *NeurIPS*.

Haiteng Zhao, Shengchao Liu, Chang Ma, Hannan Xu, Jie Fu, Zhihong Deng, Lingpeng Kong, and Qi Liu. 2023a. GIMLET: A unified graph-text model for instruction-based molecule zero-shot learning. In *NeurIPS*.

Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023b. Learning on large-scale text-attributed graphs via variational inference. In *ICLR*. OpenReview.net.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. SWIFT:A scalable lightweight infrastructure for fine-tuning. *CoRR*, abs/2408.05517.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2003. Learning with local and global consistency. In *NeurIPS*, pages 321–328. MIT Press.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *ICLR*. OpenReview.net.

Yun Zhu, Haizhou Shi, Xiaotang Wang, Yongchao Liu, Yaoke Wang, Boci Peng, Chuntao Hong, and Siliang Tang. 2025. Graphclip: Enhancing transferability in graph foundation models for text-attributed graphs. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, page 2183–2197. Association for Computing Machinery.

A Additional related work

We examine recent advancements in applying LLMs to graph data.

LLMs on graphs. The integration of graph learning and reasoning with LLMs is a rapidly growing area (Fatemi et al., 2024). One research direction involves representing graphs as textual data to leverage the reasoning capabilities of pre-trained LLMs. By encoding graph information in a textual format, LLMs can analyze graphs either on-the-fly (Chen et al., 2023; Guo et al., 2023; Wang et al., 2023; Ye et al., 2024) or through end-to-end fine-tuning (Ye et al., 2024; Chai et al., 2023; Zhao et al., 2023a). Real-world graphs are often complex, such as text-attributed graphs where nodes contain rich textual information. Another line of research focuses on utilizing the language understanding capabilities of LLMs to extract node features from raw data, which are then fed into GNNs for accurate predictions (He et al., 2024a; Zhao et al., 2023b). This approach bridges the strengths of LLMs in processing unstructured text with the structural learning capabilities of GNNs, improving performance on graph-based tasks. Note that our work differs from prior work in this family of models in two key aspects: (1) We use off-the-shelf LLMs for document reading without any further training; (2) We focus on how to choose documents for improved LLM performance on graphs (Ram et al., 2023).

B Discussions

B.1 Comparison between different ICL techniques

Table 3 compares different in-context learning techniques and message-passing approaches, illustrating their query inputs, contextual information, and retrieval conditions. Traditional in-context learning methods, such as zero-shot and few-shot learning, rely on either no context or a small set of demonstrations, while RAG retrieves external corpus elements based on a similarity metric. In contrast, GNNs and label propagation utilize graph structure to aggregate node features or labels from neighboring nodes. Our proposed frameworks, QUERYRAG, LABELRAG, and FEWSHOTRAG, bridge the gap between these paradigms by incorporating graph-guided retrieval as context for LLMs. Specifically, QUERYRAG retrieves query information from neighboring nodes, LABELRAG retrieves labels, and FEWSHOTRAG combines both, enhancing the in-context learning capabilities of LLMs. By leveraging both graph connectivity and semantic similarity, these frameworks provide a structured yet flexible approach to improving LLM performance on graph-based tasks.

B.2 Comparison with prior work

Our work is conceptually related to GraphRAG (Edge et al., 2024) in that both introduce *graph-structured context* to improve LLM reasoning. However, the two lines of research target different problem settings and make different design assumptions, which makes a direct head-to-head comparison potentially misleading.

First, most GraphRAG systems are designed for *knowledge-graph–grounded generation* (Edge et al., 2024; He et al., 2024b): they retrieve subgraphs (entities/relations) from curated or constructed knowledge graphs (KG) and feed them to an LLM to generate free-form answers (often for question answering, summarization, or agen-

Method	Query	Context	Conditions
Zero-shot	q_i	-	-
Few-shot	q_i	$\{(q_1, r_1), \ldots, (q_j, r_j)\}$	-
RAG	q_i	$\{c_1, \ldots, c_j\}$	$D(q_i, c_j) < \epsilon$
GNN	x_i	$\{x_1, \ldots, x_j\}$	$(v_i, v_j) \in \mathcal{E}$
Label Propagation	y_i	$\{y_1, \ldots, y_j\}$	$(v_i, v_j) \in \mathcal{E}$
QueryRAG	q_i	$\{q_1, \ldots, q_j\}$	$D(q_i, q_j) < \epsilon$ or $(v_i, v_j) \in \mathcal{E}$
LabelRAG	q_i	$\{r_1, \ldots, r_j\}$	$D(q_i, q_j) < \epsilon$ or $(v_i, v_j) \in \mathcal{E}$
FewshotRAG	q_i	$\{(q_1, r_1), \ldots, (q_j, r_j)\}$	$D(q_i, q_j) < \epsilon$ or $(v_i, v_j) \in \mathcal{E}$

Table 3: Comparison of context engineering techniques and message passing approaches. q: input query, r: output response, x: node representation, y: ground-truth label, c: external corpus. For conventional RAG, we use the distance metric $D(\cdot)$ and a sufficiently small constant ϵ to represent the retrieval step.

tic tasks). In contrast, our goal is *graph learning on text-attributed graphs*, where the evaluation is supervised classification over benchmark graphs rather than open-ended generation. The core challenge we address is how to elicit *message-passing–style aggregation* via in-context retrieval so that an LLM can act as a graph learner without any fine-tuning.

Second, GraphRAG typically assumes access to a relatively clean, explicitly relational graph (often a KG) with semantically meaningful nodes/edges, and its retrieval is primarily aimed at fetching *factual evidence* (Gutierrez et al., 2024; Gutiérrez et al., 2025). Our setting is broader and noisier: the "graph" is the learning object itself, and retrieval must support *neighborhood aggregation and relational generalization* rather than fact lookup. The mechanisms we propose—grounded in GNN principles such as locality, iterative aggregation, and robustness to outliers—are tailored to improve predictive performance on graph benchmarks under a strict *no-training* constraint.

Third, implementation details and compute budgets are not directly comparable. Many GraphRAG pipelines rely on additional components, such as KG construction or task-specific prompt engineering, for better generation quality. Our methods are deliberately lightweight and training-free, focusing on retrieval and aggregation strategies that emulate GNN behavior in-context.

For these reasons, we position our method as *orthogonal* to current GraphRAG research: GraphRAG enhances LLM reasoning *with* graphs as external knowledge, whereas we use graph learning principles to enhance RAG so that LLMs can better understand graph data.

B.3 Future work

For future work, we aim to investigate the in-context learning performance of LLMs in more challenging settings, where graphs may be dynamic, noisy, incomplete, or entirely unavailable. Additionally, we plan to extend our methods to larger and sparser graphs by exploring retrieval mechanisms that capture long-range node relationships beyond homophily. Beyond graph learning, we also plan to explore the broader applicability of our frameworks to natural language processing tasks. Specifically, we seek to apply our methods to structured knowledge retrieval, reasoning over knowledge graphs, and few-shot adaptation to domain-specific problems. These extensions would further demonstrate the versatility of RAG-based approaches and their ability to integrate structured and unstructured learning paradigms effectively.

C Experiment settings

C.1 Datasets

We conducted experiments on 14 text-attributed graph datasets: Cora, Pubmed (Sen et al., 2008), the CS-TAG (Yan et al., 2023) benchmarks, which include Sports-Fitness, Ele-History, Ele-Computers, Books-Children, Books-History, and Ogbn-Arxiv-TA, the DTGB (Zhang et al., 2024) benchmarks, including Enron, Stack elec, Stack ubuntu, Googlemap CT, Amazon movies, and Yelp. These datasets span diverse domains, such as citation networks and e-commerce, and vary in sparsity and size, ranging from small to large scales. The dataset statistics are provided in Table 4. Since the primary focus of our work is classification based on textual attributes, we also report the average number of input tokens for the textual attributes of nodes or edges in these graphs.

Dataset	#Nodes	#Edges	#Classes	Avg. token length	Homophily	Avg. degree
NODE CLASSIFICATION						
Cora	2,708	10,556	7	166	0.81	3.90
Pubmed	19,717	88,648	3	370	0.80	4.50
Books-Children	76,875	1,554,578	24	273	0.42	30.24
Books-History	41,551	358,574	12	297	0.66	12.11
Ele-Computers	87,229	721,081	10	113	0.83	14.41
Ele-Photo	48,362	500,928	12	183	0.75	18.07
Sports-Fitness	173,055	1,773,500	13	30	0.90	17.45
Ogbn-Arxiv-TA	169,343	1,166,243	40	222	0.66	13.67
EDGE CLASSIFICATION						
Enron	42,711	797,907	10	695	–	10.25
Stack elec	397,702	1,262,225	2	505	–	1.00
Stack ubuntu	674,248	1,497,066	2	651	–	1.02
Googlemap CT	111,168	1,380,623	5	183	–	1.15
Amazon movies	293,566	3,217,324	5	315	–	1.23
Yelp	2,138,242	6,,990,189	5	288	–	1.00

Table 4: Datasets statistics for node classification and edge classification tasks.

Node classification. We provide detailed descriptions of the datasets used in the main experiments as follows:

- **Cora and Pubmed** (Sen et al., 2008) are widely used citation graph datasets in graph learning research. We use the commonly adopted versions(Kipf and Welling, 2017) and collected the missing textual attributes for these datasets to ensure consistency with other datasets.
- **Ogbn-Arxiv-TA (arxiv)** (Yan et al., 2023) is derived from the citation dataset Ogbn-Arxiv(Hu et al., 2020). The task involves predicting the categories of the papers, formulated as a 40-class classification problem. The text attributes for each paper node are extracted from its title and abstract in Ogbn-Arxiv.
- **Books-Children and Books-History** (Yan et al., 2023) are extracted from the Amazon-Books dataset. Books-Children contains items with the second-level label Children', while Books-History contains items with the second-level label History'. The nodes represent books, and edges indicate that two books are frequently co-purchased or co-viewed. The labels correspond to the third-level categories of the books. The title and description of each book are used as the text attributes for the nodes.
- **Ele-Computers and Ele-Photo** (Yan et al., 2023) are extracted from the Amazon-Electronics dataset. Ele-Computers contains items with the second-level label Computers', and Ele-Photo contains items with the second-level label Photo'. Nodes represent electronics-related products, and an edge indicates frequent co-purchase or co-viewing between two products.
- **Sports-Fitness** (Yan et al., 2023) is extracted from the Amazon-Sports dataset. It consists of items with the second-level label 'Fitness'. Nodes represent fitness-related items, and an edge between two items indicates frequent co-purchase or co-viewing.

For the Cora and Pubmed datasets, we use the original splits from existing work (Kipf and Welling, 2017), which sample 20 nodes per class for the training set and 500/1000 nodes for validation/test, respectively. For the arxiv dataset, we adopt the public split provided in the benchmarks (Yan et al., 2023; Hu et al., 2020). For the remaining datasets, we use a random 10%/10%/80% split for train/validation/test to align with the strict settings typically encountered in real-world scenarios.

Edge classification. We also provide detailed descriptions of the datasets from the DTGB (Zhang et al., 2024), which are used for edge classification in the additional experiments:

- **Enron** [1] is derived from the internal email records of ENRON employees between 1999 and 2002. Each node denotes an employee,

[1] https://www.cs.cmu.edu/~enron

with optional textual descriptions derived from their department or job role. Edges are timestamped email messages, and their texts are cleaned and truncated versions of the original content. Emails are labeled into 10 semantic groups, such as scheduling, memos, or transaction discussions.

- **Stack elec** [2] is constructed from the Stack Exchange posts related to electronics. It forms a time-evolving bipartite graph where user nodes and question nodes are connected via answer and comment interactions. Node texts include self-introductions for users and full question content for questions. Edge texts are the actual replies, with binary labels indicating community feedback (useful or not).

- **Stack ubuntu** [3] follows the same construction as Stack elec but targets technical Q&A threads under the Ubuntu tag. Compared to Stack elec, its edge content more frequently mixes narrative explanation with executable code, posing additional challenges for semantic modeling.

- **Googlemap CT** [4] is a subset of Google Local Data covering businesses in Connecticut. The graph consists of users and business entities, connected via review edges. Only businesses have node-level descriptions, which include various metadata. Edge labels reflect 1-to-5 star ratings, and all reviews are chronologically organized after removing noisy tokens and symbols.

- **Amazon movies** [5] is derived from the Movies and TV section of the Amazon product review corpus. Products serve as nodes, annotated with metadata like description, category, and popularity ranking. Edges denote review texts written by users and are labeled according to the given star ratings. Temporal order is preserved based on review timestamps.

- **Yelp** [6] is sourced from the Yelp public dataset and focuses on user feedback about local businesses. Nodes include customers and service providers, with textual attributes such as user registration metadata and business details. Reviews form the edge content, and scores from 1 to 5 are used as edge categories. Edge sequence reflects the time of review submission.

We follow the standard setting in edge classification (Zhang et al., 2024), which chronologically split each dataset into train/validation/test sets by 7:1.5:1.5.

C.2 Baselines

In our experimental evaluation, we include a diverse range of state-of-the-art methods to ensure a comprehensive assessment: (i) supervised baseline without graph structure: MLP; (ii) supervised message-passing GNNs for node classification, including GCN (Kipf and Welling, 2017), GAT (Velickovic et al., 2018), GraphSAGE (Hamilton et al., 2017), and RevGAT (Li et al., 2021); supervised temporal GNNs for edge classification, including TGAT (Xu et al., 2020), JODIE (Kumar et al., 2019), DyRep (Trivedi et al., 2019), CAWN (Wang et al., 2021), and DyGFormer (Yu et al., 2023). (iii) specialized graph LLMs, including GLEM (Zhao et al., 2023b), TAPE (He et al., 2024a), PRODIGY (Huang et al., 2023), and LLaGA (Chen et al., 2024); and (iv) in-context baselines, including zero-shot, few-shot (Brown et al., 2020), and RAG (Lewis et al., 2020). We provide detailed descriptions of the baselines used in the main experiments as follows:

(i) Supervised (temporal) GNNs:

- **GCN** (Kipf and Welling, 2017): GCN is a foundational GNN model that applies convolution operations on graph-structured data, aggregating information from neighboring nodes.

- **GAT** (Velickovic et al., 2018): GAT introduces attention mechanisms to learn the importance of neighboring nodes, enabling more expressive and context-sensitive node feature aggregation.

- **GraphSAGE** (Hamilton et al., 2017): GraphSAGE uses a sampling-based neighborhood aggregation approach to efficiently learn node embeddings, particularly for large-scale graphs.

- **RevGAT** (Li et al., 2021): RevGAT combines reversible connectivity with a deep network

[2] https://archive.org/details/stackexchange
[3] https://archive.org/details/stackexchange
[4] https://datarepo.eng.ucsd.edu/mcauley_group/gdrive/googlelocal
[5] https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2
[6] https://www.yelp.com/dataset

Prompt Name	Prompt Content
Zero-shot	**Task:** Classify the following text into one of the predefined categories: ⟨list of categories⟩. Make your decision based on the main topic and overall content of the text. If the text is ambiguous or does not clearly fit into any category, choose the closest match. Provide only the category name as the output. \n **Text:** ⟨textual content⟩ \n **Answer:**
Few-shot	**Task:** Classify the above text into one of the predefined categories: ⟨list of categories⟩. Use the provided few-shot examples to enhance your understanding of the topic and context for accurate classification. ...category name as the output. \n **Examples:** \n **Text**: ⟨few-shot text⟩ \n **Answer**: ⟨few-shot label⟩ \n . . . (more randomly sampled examples). . . \n **Text**: ⟨textual content⟩ \n **Answer:**
RAG	**Task:** Classify the above text into one of the predefined categories: ⟨list of categories⟩. Use the provided references to enhance your understanding of the topic and context for accurate classification. ...category name as the output. \n **Reference:** \n **Text**: ⟨retrieved text⟩ \n . . . (more retrieved texts). . . \n **Text**: ⟨textual content⟩ \n **Answer:**
QueryRAG	**Task:** Classify the above text into one of the predefined categories: ⟨list of categories⟩. Use the provided references to enhance your understanding of the topic and context for accurate classification. ...category name as the output. \n **Reference:** \n **Text**: ⟨text of retrieved nodes⟩ \n . . . (more graph retrieved texts). . . \n **Text**: ⟨textual content⟩ \n **Answer:**
LabelRAG	**Task:** Classify the above text into one of the predefined categories: ⟨list of categories⟩. Use the provided category of reference texts to enhance your understanding of the topic and context for accurate classification. ...category name as the output. \n **Reference:** \n **Answer**: ⟨category of retrieved nodes⟩ \n . . . (more graph retrieved labels). . . \n **Text**: ⟨textual content⟩ \n **Answer:**
FewshotRAG	**Task:** Classify the above text into one of the predefined categories: ⟨list of categories⟩. Use the provided references and corresponding categories to enhance your understanding of the topic and context for accurate classification. ...category name as the output. \n **Reference:** \n **Text**: ⟨few-shot text⟩ \n **Answer**: ⟨few-shot label⟩ \n . . . (more graph-guided few-shot examples). . . \n **Text**: ⟨textual content⟩ \n **Answer:**

Table 5: An illustration of prompts we use for in-context node classification on text-attributed graphs.

architecture, enabling memory-efficient and scalable training of GNNs with deeper layers.

- **TGAT** (Xu et al., 2020): TGAT uses the self-attention mechanism as the basic block to model the temporal information of graph topology as well as A functional time encoding technique to capture temporal patterns.

- **JODIE** (Kumar et al., 2019): JODIE is a sequence-based model, which adopts recurrent neural networks to learn and update the trajectory embeddings between temporal interactions.

- **CAWN** (Wang et al., 2021): CAWN is a random-walk based model, which adopts an anonymization strategy of random walks to explore the causality of network dynamics and learn inductive node representations.

- **DyRep** (Trivedi et al., 2019): DyRep models the intertwined dynamics of temporal interactions using a dual time-scale temporal point process, where the event intensity is parameterized by a temporal attention network that incorporates evolving structural context into node embeddings.

- **DyGFormer** (Yu et al., 2023): DyGFormer is a Transformer-based model designed for dynamic graph learning, which captures temporal dependencies and node correlations in dynamic graphs through neighbor co-occurrence encoding and patching technique.

(ii) Graph LLMs:

- **GLEM** (Zhao et al., 2023b): GLEM is a variational Expectation Maximization framework that trains LLMs and GNNs jointly on a graph. We adopt DeBERTa (He et al., 2021) as the language model for fine-tuning, following the original implementation.

- **TAPE** (He et al., 2024a): TAPE is an LLM-enhanced framework that leverages textual explanations generated by LLMs to enhance the performance of downstream GNNs. In our experiments, we use the original explanations provided by the authors for the Cora, Pubmed, and arXiv datasets, and we generate explanations for the remaining datasets using Qwen3-8B. The downstream GNN used is a two-layer GCN, and the encoding model for the textual explanations is DeBERTa (He et al., 2021).

- **PRODIGY** (Huang et al., 2023): PRODIGY

is a pretraining framework that formulates in-context learning over graphs with a novel prompt graph representation. Following the original implementation, we pretrain PRODIGY on arXiv and evaluate its in-context performance on other datasets.

- **LLaGA** (Chen et al., 2024): LLaGA is an LLM-based graph learning framework that introduces a versatile linear projector to seamlessly bridge graph structures with the token space understood by LLMs. We implement LLaGA using Vicuna-7B-v1.5-16K (Chiang et al., 2023) as the foundational base model and SimTeg (Duan et al., 2023) as the default text-encoding model.

(iii) In-context baselines:

- **Few-shot** (Brown et al., 2020): Few-shot learning provides a small number of labeled examples as context for the language model to perform downstream tasks without fine-tuning. In our experiments, we randomly draw $K = \deg(v_i)$ samples from the graph as in-context examples for node v_i.

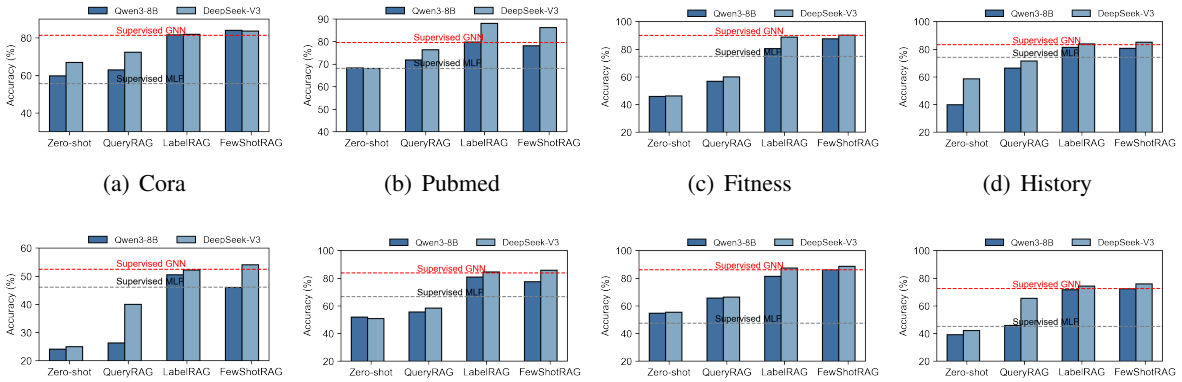
- **One-shot** (Brown et al., 2020): One-shot learning provides a single labeled example to guide the language model in solving tasks with minimal supervision, equivalent to $K = 1$ in few-shot learning.

- **Zero-shot** (Kojima et al., 2022): Zero-shot learning evaluates the language model using only task instructions without any labeled examples or demonstrations. This setting reflects the basic capability of LLMs to solve graph learning tasks.

- **RAG** (Lewis et al., 2020): RAG combines pretrained language models with a retriever mechanism to fetch relevant context for in-context learning and improve task performance. We use BERT (Devlin et al., 2019) as the embedding model and the FAISS library (Johnson et al., 2019) for efficient similarity search.

(iv) Our goal is to explore the in-context learning capabilities of LLMs. To this end, we evaluate a range of advanced language models, including QWEN3-8B (Yang et al., 2025), QWEN2.5-7B-INSTRUCT (Yang et al., 2024), GEMMA-2-9B (Team, 2024), PHI-3.5-MINI-INSTRUCT (Abdin et al., 2024), and LLAMA3.1-8B-INSTRUCT (Dubey et al., 2024). Additionally, we conduct experiments with DEEPSEEK-V3 (Liu et al., 2024a) to assess its performance on the node classification task under in-context learning settings. We use pretrained models under their respective licenses and terms as provided by the model publishers/platforms. Due to budget constraints, we did not conduct comprehensive experiments with other advanced models, such as GPT-5. We provide the download links for the LLM on Hugging Face and ModelScope below:

- QWEN3-8B:
  - Hugging Face: `https://huggingface.co/Qwen/Qwen3-8B`
  - ModelScope: `https://modelscope.cn/models/Qwen/Qwen3-8B`
- LLAMA3.1-8B-INSTRUCT:
  - Hugging Face: `https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct`
  - ModelScope: `https://www.modelscope.cn/LLM-Research/Meta-Llama-3.1-8B-Instruct`
- QWEN2.5-7B-INSTRUCT:
  - Hugging Face: `https://huggingface.co/Qwen/Qwen2.5-7B-Instruct`
  - ModelScope: `https://www.modelscope.cn/qwen/Qwen2.5-7B-Instruct`
- GEMMA-2-9B-IT:
  - Hugging Face: `https://huggingface.co/google/gemma-2-9b-it`
  - ModelScope: `https://www.modelscope.cn/LLM-Research/gemma-2-9b-it`
- PHI-3.5-MINI-INSTRUCT:
  - Hugging Face: `https://huggingface.co/microsoft/Phi-3.5-mini-instruct`
  - ModelScope: `https://www.modelscope.cn/LLM-Research/Phi-3.5-mini-instruct`
- QWEN2.5-72B-INSTRUCT:



(a) Cora (b) Pubmed (c) Fitness (d) History

(e) Children (f) Photo (g) Computers (h) arxiv

Figure 5: In-context performance of QUERYRAG, LABELRAG, and FEWSHOTRAG compared to the zero-shot performance of QWEN3-8B and DEEPSEEK-V3.

- Hugging Face: `https://huggingface.co/Qwen/Qwen2.5-72B-Instruct`
- ModelScope: `https://www.modelscope.cn/qwen/Qwen2.5-72B-Instruct`

C.3 Evaluation

For the supervised baselines, we train the models using the standard training splits provided for each dataset and evaluate their performance on the corresponding test splits. Accuracy is used as the performance metric, and we report the averaged scores and standard deviations across ten runs for MLP and GNNs. For LLM-based baselines, due to computational resource limitations, all results are averaged over three runs to account for variability in the few-shot sampling process.

C.4 Implementation

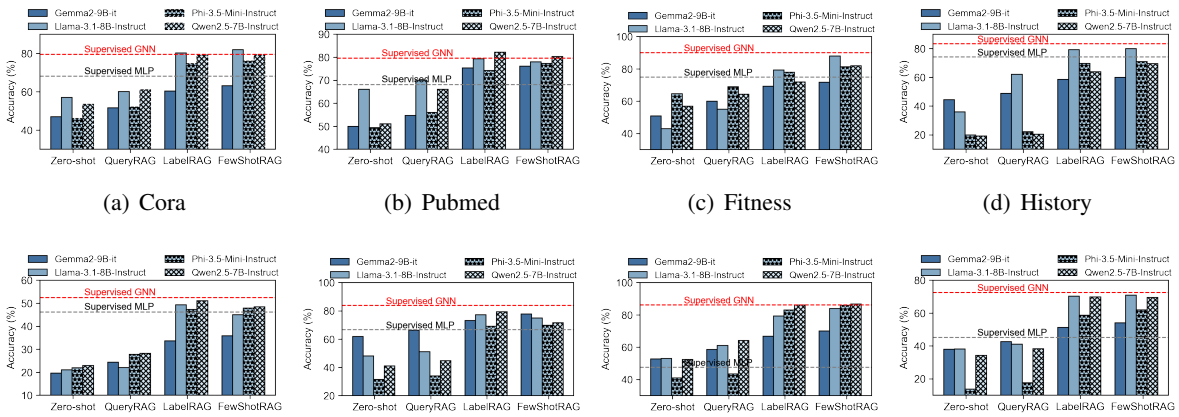
For few-shot learning, we evaluate each example in the test set by randomly selecting K examples from the task's training set as conditioning examples. In the zero-shot setting, we use a natural language prompt, and for $K = 0$, no demonstrations are provided. The same evaluation protocol is applied across all LLM architectures to ensure a fair comparison. For QUERYRAG, LABELRAG, and FEWSHOTRAG, we evaluate them in few-shot in-context learning setups without additional fine-tuning of the LLMs. In these setups, textual context and/or label information obtained from the *one-hop* neighbors of each node are incorporated into the prompts. Our implementation is based on the ms-swift (Zhao et al., 2024) and transformers (Wolf et al., 2019) libraries. The experiments were conducted on a Linux system with 8 NVIDIA 4090 GPUs, each with 24GB of memory. We present the prompts used for in-context node classification on text-attributed graphs in Table 5. The prompt for the edge classification tasks can be similarly designed by adapting the input format to include a pair of nodes (source and target) and their associated temporal context.

D Additional experiment results

In-context performance. We investigate how our proposed in-context RAG frameworks enhance the in-context learning capabilities of LLMs. Figure 5 presents the performance of QUERYRAG, LABELRAG, and FEWSHOTRAG with QWEN3-8B and DEEPSEEK-V3 across several text-attributed graph benchmarks. The results show that LLMs generally benefit from the additional contextual information provided by our RAG frameworks. In particular, LLMs with LABELRAG and FEWSHOTRAG achieve significant performance improvements, outperforming supervised MLPs and, in some cases, matching or even surpassing supervised LLMs. Although GNNs still perform slightly better in certain cases, such as on the History dataset, the performance gap is small. Overall, these findings indicate that our proposed RAG frameworks, especially FEWSHOTRAG, serve as strong alternatives for graph-related tasks, enabling effective in-context learning without requiring any modifications to the LLMs.

D.1 Different LLM architectures

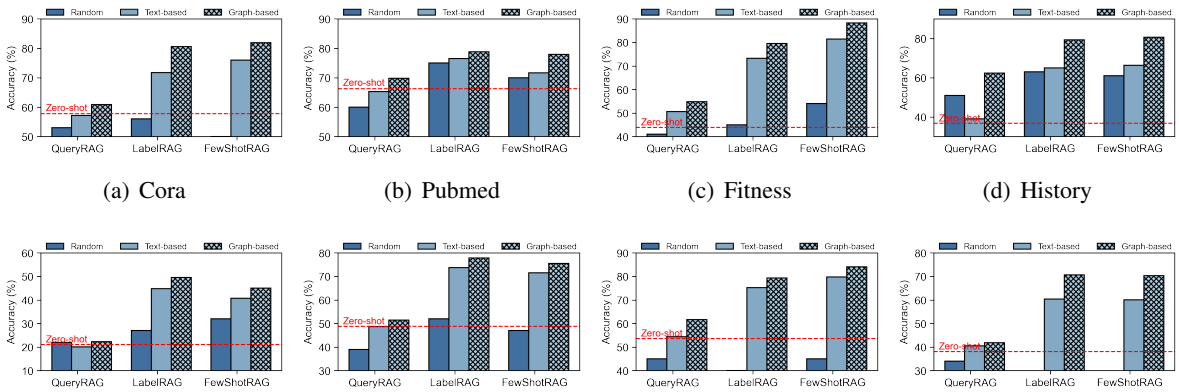
To validate the generality of our proposed RAG frameworks, we evaluate them using additional



(a) Cora (b) Pubmed (c) Fitness (d) History

(e) Children (f) Photo (g) Computers (h) Arxiv

Figure 6: Ablation results on the use of different LLMs on the node classification task.



(a) Cora (b) Pubmed (c) Fitness (d) History

(e) Children (f) Photo (g) Computers (h) arxiv

Figure 7: Ablation results on different retrieval mechanisms.

LLMs, including QWEN2.5-7B-INSTRUCT(Yang et al., 2024), GEMMA-2-9B(Team, 2024), PHI-3.5-MINI-INSTRUCT(Abdin et al., 2024), and LLAMA3.1-8B-INSTRUCT(Dubey et al., 2024). The experimental results on the node classification task are presented in Figure 6. Across all eight datasets, our in-context RAG frameworks (QUERYRAG, LABELRAG, and FEWSHOTRAG) consistently improve performance across different LLM architectures. Among them, FEWSHOTRAG achieves the highest accuracy in all settings, demonstrating the benefit of incorporating both query and label information into the in-context learning process.

In addition to small-scale LLMs, we further evaluate our methods on QWEN2.5-72B-INSTRUCT to investigate whether medium-sized models can match the performance of supervised GNNs. As shown in Table 7, QWEN2.5-72B-INSTRUCT is able to match or even surpass GCN in most cases. These results confirm that our RAG frameworks are robust and adaptable across a wide range of LLM architectures, offering a scalable and effective solution for graph-related tasks by leveraging the capabilities of diverse pre-trained models.

D.2 Retriever

QUERYRAG, LABELRAG, and FEWSHOTRAG were initially proposed as RAG frameworks that utilize graph structure as inherent context, eliminating the need for additional retrieval efforts. Since *context* is crucial for the in-context learning capabilities of LLMs, we perform ablation experiments on the retriever in our proposed RAG frameworks, comparing random-based, text-based, and graph-based retrieval methods. For text-based retrieval, we follow the standard RAG protocol, retrieving textual neighbors based on node attributes rather than relying on graph structure. In this setup, the retriever leverages dense text embed-

	Enron	Googlemap CT	Stack elec	Stack ubuntu	Amazon movies	Yelp
🔥 MLP	$22.7_{\pm 0.3}$	$52.4_{\pm 0.8}$	$39.5_{\pm 1.4}$	$47.1_{\pm 0.5}$	$40.5_{\pm 1.1}$	$51.3_{\pm 1.8}$
🔥 TGAT (Xu et al., 2020)	$55.2_{\pm 0.3}$	$61.3_{\pm 0.2}$	**$63.6_{\pm 0.3}$**	$70.5_{\pm 0.1}$	$58.8_{\pm 0.5}$	$62.2_{\pm 1.1}$
🔥 JODIE (Kumar et al., 2019)	$64.8_{\pm 0.6}$	$61.8_{\pm 0.1}$	OOM	OOM	**$60.4_{\pm 0.7}$**	OOM
🔥 DyRep (Trivedi et al., 2019)	**$64.3_{\pm 0.6}$**	$61.3_{\pm 0.1}$	OOM	OOM	OOM	OOM
🔥 CAWN (Wang et al., 2021)	$56.9_{\pm 1.3}$	$61.7_{\pm 0.1}$	**$61.1_{\pm 1.8}$**	$60.0_{\pm 2.5}$	$60.5_{\pm 0.4}$	$60.8_{\pm 0.3}$
🔥 DyGFormer (Yu et al., 2023)	$55.7_{\pm 0.6}$	$61.2_{\pm 0.1}$	$48.0_{\pm 4.3}$	**$69.8_{\pm 2.5}$**	$59.7_{\pm 1.9}$	**$63.6_{\pm 0.8}$**
QWEN3-8B (small scale)						
❄ Zero-shot (Kojima et al., 2022)	11.4	51.5	27.5	32.6	34.1	44.5
❄ One-shot (Brown et al., 2020)	10.5	50.2	26.4	32.2	35.2	47.4
❄ Few-shot (Brown et al., 2020)	11.2	48.7	23.1	33.4	37.6	49.1
❄ RAG (Lewis et al., 2020)	12.8	54.4	28.6	34.5	33.4	51.2
❄ QUERYRAG	15.4	59.1	32.7	34.6	42.2	55.7
❄ LABELRAG	53.1	**65.2**	49.2	55.9	50.3	62.4
❄ FEWSHOTRAG	58.2	65.0	46.9	53.8	53.2	62.0
DEEPSEEK-V3 (large scale)						
❄ Zero-shot (Kojima et al., 2022)	13.7	55.2	38.3	36.7	37.2	50.7
❄ One-shot (Brown et al., 2020)	12.3	53.9	34.4	35.2	38.9	55.0
❄ Few-shot (Brown et al., 2020)	11.4	51.5	32.6	36.5	39.7	56.2
❄ RAG (Lewis et al., 2020)	13.5	56.8	39.7	37.1	39.6	58.9
❄ QUERYRAG	23.4	59.4	40.5	37.3	41.2	55.0
❄ LABELRAG	61.2	**69.7**	62.1	**70.7**	61.6	67.0
❄ FEWSHOTRAG	**65.3**	68.2	60.9	69.3	**63.3**	**69.7**

Table 6: Edge classification results (%) on six text-attributed graphs. The top-3 best performances in each dataset are highlighted in **red**, blue, and **gray**. OOM means out-of-memory.

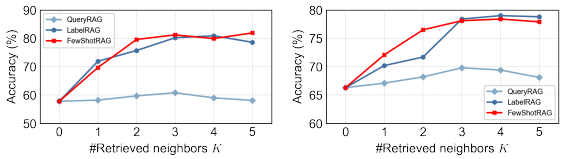
	Cora	Pubmed	Fitness	History	Children	Photo	Computers	arxiv
MLP	$55.6_{\pm 0.1}$	$68.1_{\pm 0.5}$	$78.8_{\pm 0.3}$	$77.4_{\pm 0.3}$	$47.1_{\pm 0.4}$	$56.2_{\pm 0.2}$	$64.7_{\pm 0.4}$	$46.5_{\pm 0.2}$
GCN (Kipf and Welling, 2017)	$81.3_{\pm 0.7}$	$79.5_{\pm 0.3}$	**$90.0_{\pm 0.2}$**	**$83.3_{\pm 0.8}$**	$52.4_{\pm 1.1}$	**$83.8_{\pm 0.2}$**	$86.1_{\pm 0.4}$	$72.5_{\pm 0.2}$
QWEN2.5-72B-INSTRUCT								
Zero-shot (Kojima et al., 2022)	66.5	71.7	57.4	60.3	30.1	56.3	60.7	64.5
One-shot (Brown et al., 2020)	66.3	77.6	55.5	63.2	29.7	53.0	59.4	63.5
Few-shot (Brown et al., 2020)	66.9	85.1	57.2	65.9	30.3	55.0	61.1	62.9
RAG (Lewis et al., 2020)	68.7	74.3	57.2	65.9	31.1	55.8	61.5	65.8
QUERYRAG	69.3	74.0	59.1	65.6	31.0	56.7	70.9	66.0
LABELRAG	77.5	88.0	78.8	74.5	51.7	80.1	84.1	73.7
FEWSHOTRAG	82.1	87.7	88.1	79.4	53.8	82.8	86.5	74.5

Table 7: Node classification results (%) with QWEN2.5-72B-INSTRUCT.

dings to locate relevant documents (Karpukhin et al., 2020). These embeddings are generated using a pre-trained BERT model (Devlin et al., 2019), which captures semantic similarity between nodes' textual attributes. The retrieved context is then incorporated into input prompts for LLMs. To ensure consistency, the number of retrieved neighbors matches the number of graph-based neighbors for each node. The ablation results, shown in Figure 7, evaluate the impact of replacing graph structure with random-based and text-based retrieval on all datasets. We use QWEN3-8B as the LLM model. As observed, graph-based retrieval consistently outperforms both text-based and random-based retrieval across QUERYRAG, LABELRAG, and FEWSHOTRAG, aligning with findings from GraphRAG (Edge et al., 2024). These results highlight the importance of graph structure as a source of inherent context and its critical role in enhancing the in-context learning capabilities of LLMs.

D.3 Number of retrieved neighbors

To investigate the effect of the number of retrieved neighbors K on the performance of our proposed RAG frameworks (QUERYRAG, LABELRAG, and FEWSHOTRAG), we conduct ablation experiments by varying the number of neighbors retrieved for each node. Specifically, we evaluate performance with 0, 1, 2, 3, 4, and 5 retrieved neighbors on the



(a) Cora (b) Pubmed

Figure 8: Node classification results on Cora and Pubmed datasets with proposed QUERYRAG, LABELRAG, and FEWSHOTRAG.

Cora and Pubmed datasets, where $K = 0$ corresponds to the zero-shot setting. We use QWEN3-8B as the LLM model. The results, shown in Figure 8, indicate that increasing the number of retrieved neighbors initially improves performance, as LLMs gain access to more contextual information. However, performance plateaus or even slightly declines beyond a certain point (e.g., more than 3 neighbors), likely due to the introduction of less relevant or redundant information. While retrieving more neighbors can enhance performance by providing richer context, excessively large retrieval sets may dilute the relevance of critical information, highlighting the need for efficient context selection in RAG frameworks.