STRUCTSEEK: LEARNING TO ACQUIRE STRUCTURAL EVIDENCE FOR ZERO-SHOT GRAPH LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) have demonstrated strong semantic generalization, creating new opportunities for zero-shot graph learning. However, link prediction remains challenging because it depends critically on structural relations between node pairs. Existing Graph LLMs either require the LLM to recover such relations from serialized graph context or encode them implicitly in learned representations, while recent graph agents mainly retrieve graph observations rather than directly access task-relevant structural information. We propose **StructSeek**, an interactive graph agent that introduces a **structure-as-tool** paradigm for zero-shot graph learning. StructSeek actively seeks the semantic and structural evidence needed for each prediction through dedicated graph operators. The agent learns which operators to invoke, when to invoke them, and how to use the returned evidence for prediction. We train StructSeek with supervised fine-tuning on teacher trajectories followed by reinforcement learning. Experiments across multiple unseen graph domains show overall improvements on both node classification and link prediction. The gains are particularly pronounced for link prediction, where StructSeek-3B outperforms the strongest baseline by **12.9–25.8 percentage points** across four unseen graphs. These results demonstrate the benefit of directly accessible structural evidence for zero-shot graph learning.

1 INTRODUCTION

Large language models (LLMs) have shown promising cross-domain generalization for zero-shot graph learning, particularly for node classification, where their semantic knowledge supports predictions on unseen graphs without task-specific supervision (Jin et al., 2024a; Liu et al., 2024; Li et al., 2024; Wang et al., 2024a). However, this success has not translated equally well to link prediction (LP). Despite its binary output, LP requires a different type of information: whether two nodes are connected depends not only on their semantics, but also on their structural relationship. This raises a key challenge for zero-shot LP: *how can an LLM obtain the structural evidence needed for relational prediction on unseen graphs?*

Existing LLM-based graph methods do not provide a satisfactory answer. One common approach serializes neighborhoods, paths, or subgraphs as text (Zhao et al., 2023; Sun et al., 2025; Zhuang et al., 2026). Although this exposes graph structure to the LLM, structural features needed for LP must still be recovered from serialized descriptions. For example, identifying common neighbors requires comparing serialized neighborhoods and finding their overlap, a type of structural reasoning on which LLMs remain unreliable (Wang et al., 2023; Fatemi et al., 2024; Tang et al., 2025). Another approach encodes graph structure into learned representations and connects them to the LLM through alignment or projection (Chen et al., 2024; Tang et al., 2024; Kong et al., 2025). This avoids structural reasoning over text, but leaves structural features implicit in learned representations whose transfer to unseen graph domains depends on the encoder and alignment. In short, textual representations leave structural computation to the LLM, while learned representations make structural evidence implicit.

Yet many structural signals useful for graph prediction can be computed directly from the graph. Classical LP methods, for example, have long used neighborhood overlap, path-based measures, and local subgraph patterns to characterize node-pair relationships (Liben-Nowell & Kleinberg, 2007; Kovács et al., 2019; Zhang & Chen, 2018; Mao et al., 2024; Liang et al., 2026). When such structural

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

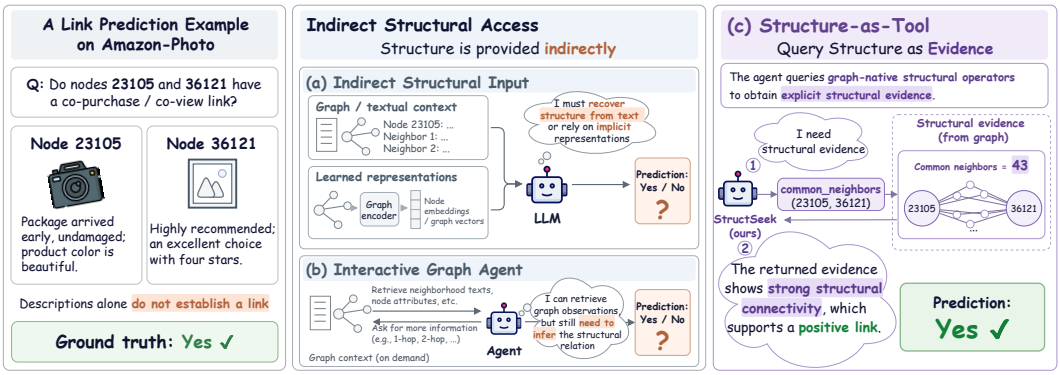

Figure 1: Different paradigms for zero-shot graph learning. Existing approaches provide graph structure through textual context or learned representations, while current graph agents primarily retrieve graph observations during reasoning. StructSeek directly queries task-relevant structural evidence through graph operators.

quantities can be computed directly, requiring the LLM to reconstruct them from text or relying on them to be captured implicitly by learned representations introduces an unnecessary intermediate step. This suggests a different division of labor: *the LLM decides what structural evidence is needed, while graph operators compute it directly.*

Recent agentic approaches provide a natural interface for this division of labor by allowing LLMs to interact with graphs during reasoning (Yao et al., 2023; Jin et al., 2024b; Sun et al., 2026). Existing graph agents, however, primarily use this interaction to retrieve graph context or semantic information, leaving structural evidence to be inferred from the retrieved observations. They improve how graph information is acquired, but provide limited support for directly computing task-relevant structural evidence.

We introduce a **structure-as-tool** paradigm (Figure 1) and propose **StructSeek**, an interactive agent for zero-shot graph learning. StructSeek equips the LLM with structural operators that compute graph properties and relations, together with semantic operators that retrieve node information. For each prediction, the agent decides what evidence is needed, invokes the corresponding operators, and reasons over the returned evidence. This separates structural computation from LLM reasoning: graph operators provide explicit structural evidence, while the LLM decides what to query and how to use the results. We train StructSeek on teacher trajectories using supervised fine-tuning followed by reinforcement learning, enabling it to learn effective operator use across graph domains and prediction tasks.

Our contributions are summarized as follows:

- We identify a key challenge in zero-shot graph learning that is particularly important for link prediction: LLMs need access to task-relevant structural evidence that cannot be reliably recovered from graph representations alone.
- We introduce the **structure-as-tool** paradigm and propose **StructSeek**, an interactive graph agent that directly queries semantic and structural evidence through dedicated graph operators.
- We construct teacher trajectories and train StructSeek with supervised fine-tuning and reinforcement learning, enabling effective operator use across unseen graph domains and prediction tasks.

2 METHOD

StructSeek realizes the **structure-as-tool** paradigm through two components: an evidence acquisition environment that provides executable graph operators and an acquisition policy that learns to selectively invoke these operators during prediction. Given a graph query, the policy iteratively decides what evidence to acquire, invokes the corresponding operator, and uses the returned evidence to determine subsequent actions. This process continues until the policy has acquired sufficient evidence to produce a prediction. Figure 2 provides an overview of the framework.

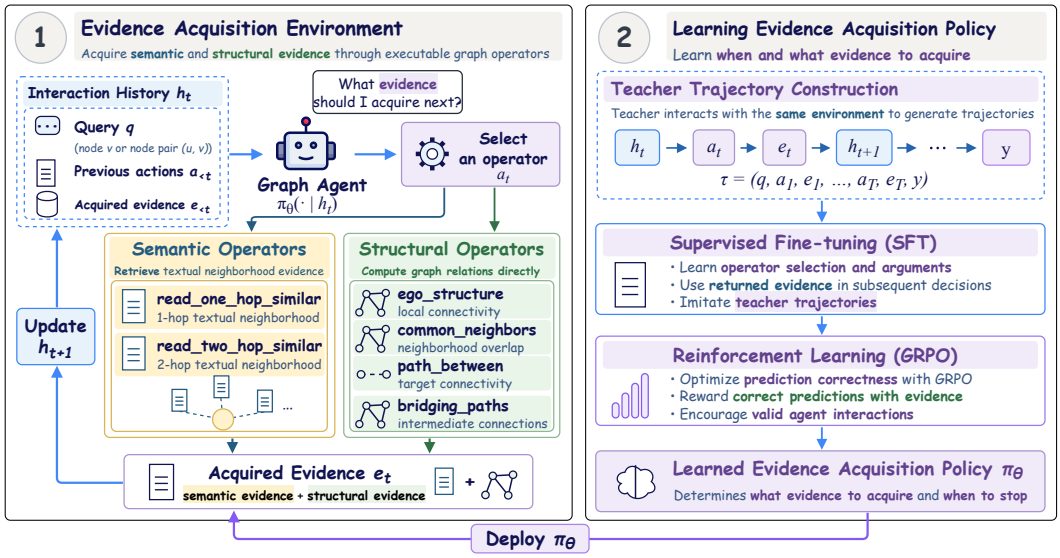

Figure 2: Overview of StructSeek. Given a graph prediction query, StructSeek sequentially acquires semantic and structural evidence through graph operators and uses the returned evidence to guide subsequent decisions and the final prediction. The acquisition policy is trained with supervised fine-tuning followed by reinforcement learning.

2.1 Graph Reasoning as Sequential Evidence Acquisition

Let $G = (V, E, X)$ denote an attributed graph, where V and E are the node and edge sets and X denotes node attributes, including textual information. We consider node classification (NC) and link prediction (LP) as representative graph prediction tasks. A query q specifies a target node for NC or a target node pair for LP. We focus on cross-domain zero-shot graph learning, where the model is evaluated on unseen graph domains without supervision or adaptation on the target domain.

Rather than providing the LLM with a fixed graph context, StructSeek formulates graph prediction as sequential evidence acquisition. At step t, the policy conditions on the query and its interaction history

$$h_t = (q, a_{<t}, e_{<t}), \tag{1}$$

where $a_{<t}$ denotes previous acquisition actions and $e_{<t}$ denotes the evidence returned by the environment. Each action $a_t = (o_t, z_t)$ specifies an operator o_t and its arguments z_t. The policy selects an action and receives the corresponding evidence:

$$a_t \sim \pi_\theta(\cdot \mid h_t), \qquad e_t = \mathrm{Env}(G, a_t). \tag{2}$$

The returned evidence is incorporated into the interaction history and conditions the next decision. The process continues until the policy produces the final prediction

$$\hat{y} \sim \pi_\theta(\cdot \mid h_{T+1}). \tag{3}$$

This formulation turns graph prediction from fixed-context inference into an active process in which the model determines what evidence to acquire and adapts subsequent decisions according to the returned evidence.

2.2 Evidence Acquisition Environment

The evidence acquisition environment provides two complementary forms of graph evidence. Semantic operators retrieve textual information from local neighborhoods, while structural operators compute graph properties and relations directly from the topology. Formally, the environment exposes

$$\mathcal{O} = \mathcal{O}_{\mathrm{sem}} \cup \mathcal{O}_{\mathrm{str}}, \tag{4}$$

where $\mathcal{O}_{\mathrm{sem}}$ and $\mathcal{O}_{\mathrm{str}}$ denote the semantic and structural operator sets, respectively.

Semantic operators. The operators `read_one_hop_similar` and `read_two_hop_similar` retrieve textual evidence from semantically relevant nodes within local neighborhoods. They allow the LLM to selectively access node information without serializing the entire neighborhood into its context.

Structural operators. Structural operators compute structural evidence directly from graph topology rather than requiring the LLM to infer it from retrieved observations. Specifically, `ego_structure` characterizes local connectivity, `common_neighbors` computes neighborhood overlap, `path_between` identifies connectivity between target nodes, and `bridging_paths` captures intermediate structural connections.

This operator interface creates a deliberate division of labor: semantic operators retrieve information for LLM interpretation, whereas structural operators perform structural computation directly on the graph. StructSeek can therefore access task-relevant structural evidence without reconstructing it from serialized graph context. Complete operator definitions, arguments, outputs, and execution constraints are provided in Appendix B.

2.3 Learning the Evidence Acquisition Policy

Access to graph operators alone does not determine how they should be used. The policy must learn which evidence is useful for a query, which operator to invoke, how returned evidence should affect subsequent decisions, and when sufficient evidence has been acquired. We train the acquisition policy in two stages: supervised fine-tuning (SFT) on teacher trajectories, followed by reinforcement learning with Group Relative Policy Optimization (GRPO) (Shao et al., 2024).

2.3.1 Teacher Trajectory Construction

Existing graph learning datasets typically provide task inputs and final labels but no supervision for intermediate evidence acquisition. We therefore use a teacher model to interact with the same environment and generate trajectories

$$\tau = (q, a_1, e_1, \ldots, a_T, e_T, y), \tag{5}$$

where a_t is the acquisition action at step t, e_t is the returned evidence, and y is the final answer. Each trajectory demonstrates how acquisition decisions adapt as new graph evidence becomes available. We retain trajectories that contain valid operator interactions and produce the correct prediction (Zelikman et al., 2022; Zeng et al., 2024), forming the supervised training set $\mathcal{D}$.

2.3.2 Supervised Fine-tuning

We first initialize the acquisition policy by imitating the filtered teacher trajectories. Unlike answer-only supervision, trajectory supervision teaches the model to select operators and generate their arguments conditioned on previously acquired evidence. We optimize

$$\mathcal{L}_{\mathrm{SFT}} = -\mathbb{E}_{\tau\sim\mathcal{D}}\left[\sum_{t=1}^{T} \log \pi_\theta(a_t \mid h_t) + \log \pi_\theta(y \mid h_{T+1})\right]. \tag{6}$$

The first term supervises evidence acquisition, while the second teaches the policy to integrate the acquired evidence into the final prediction. Because each h_t contains the outcomes of previous operator calls, SFT initializes a closed-loop policy that adapts its acquisition decisions to the evidence observed during interaction.

2.3.3 Reinforcement Learning

SFT imitates teacher trajectories but does not directly optimize the outcomes of the policy's own interactions. We therefore further optimize StructSeek with GRPO (Shao et al., 2024). For each sampled trajectory τ_i, the reward combines prediction correctness, evidence acquisition, and output validity:

$$R_i = R_{\mathrm{correct}} + R_{\mathrm{query}} + R_{\mathrm{format}}. \tag{7}$$

R_{correct} assigns 1 to a correct prediction and -1 otherwise. R_{query} assigns a bonus of 0.2 when a correct prediction is made after at least one operator call, a penalty of -0.2 when an incorrect

prediction is made without querying the environment, and 0 otherwise. R_{format} assigns 0.1 to a valid output and -0.3 otherwise. These components encourage correct prediction, effective use of graph evidence, and valid agent interactions.

For each query, GRPO samples K trajectories and computes the relative advantage

$$A_i = \frac{R_i - \mu_R}{\sigma_R + \epsilon}, \qquad \mu_R = \frac{1}{K}\sum_{j=1}^{K} R_j, \tag{8}$$

where σ_R is the standard deviation of rewards within the group and ϵ is a small constant for numerical stability. We then optimize the policy using the clipped GRPO objective with KL regularization to the reference policy (Schulman et al., 2017; Shao et al., 2024). Detailed training, reward, and optimization configurations are provided in Appendix C.

3 Experiments

We first introduce the datasets used to train and evaluate StructSeek, followed by the baselines and the experimental setup. We then conduct experiments to evaluate the zero-shot generalization of StructSeek and analyze how the learned policy acquires graph evidence during inference. Specifically, we focus on the following questions: **RQ1**: How does StructSeek compare with existing methods on zero-shot graph learning? **RQ2**: Does explicitly acquiring structural evidence improve graph reasoning? **RQ3**: Does StructSeek learn a task-adaptive evidence acquisition policy?

3.1 Experimental Setup

Datasets and tasks. Our task setup, dataset versions, and evaluation protocol follow Sun et al. (2026). The policy is trained on OGB-Arxiv and OGB-Products (Hu et al., 2020; He et al., 2024a) and evaluated zero-shot on four unseen graphs: PubMed (Sen et al., 2008), Amazon-Photo and Amazon-Computers (Yan et al., 2023), and Reddit (Yan et al., 2025). The six graphs cover three domains: paper citation, product co-purchase, and social networks. We evaluate both node classification and link prediction, sampling 1,000 instances for each task–dataset pair. Link prediction is formulated as a binary decision over balanced positive and negative pairs, with the queried edge masked before interaction for positive instances (Appendix B), and accuracy is used as the evaluation metric for both tasks.

Baselines. We compare StructSeek against 12 baselines from three families: GNN-based methods (GCN, GraphSAGE, RevGAT) (Kipf & Welling, 2017; Hamilton et al., 2017; Li et al., 2021); static-context LLMs, which reason over a graph context fixed before inference (GraphGPT, LLaGA, GraphICL, LinearRAG) (Tang et al., 2024; Chen et al., 2024; Sun et al., 2025; Zhuang et al., 2026); and interactive graph agents, which query the graph during reasoning (GraphCoT, Search-R1, Search-o1, AgentGL-R++, AgentGL-GRPO) (Jin et al., 2024b; 2025; Li et al., 2025b; Sun et al., 2026). The two LLM-based families differ in whether graph information is provided as a fixed context or acquired through interaction during inference. We report baseline results from Sun et al. (2026) and adopt the same Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct backbones and zero-shot protocol for StructSeek.

Implementation. StructSeek is initialized from Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct (Yang et al., 2024), which we refer to as the backbones. The model is first supervised fine-tuned on teacher trajectories generated by DeepSeek-V4-Flash (DeepSeek-AI, 2026), and is then further optimized with GRPO (Shao et al., 2024). Training details and evaluation settings are provided in Appendix C.

3.2 Main Results (RQ1)

Table 1 reports the zero-shot results on four unseen graphs. Overall, StructSeek achieves the best performance in 13 out of 16 evaluation settings. At the 3B scale, StructSeek obtains the best results in all eight settings, while at the 7B scale, it achieves the best results in five out of eight settings.

Table 1: Zero-shot results (%) on four unseen graphs for link prediction and node classification. Small colored badges report StructSeek minus each baseline at the matched backbone scale: red ↑ indicates that StructSeek performs better, while green ↓ indicates that it performs worse. For backbone-free GNNs, differences are computed using the mean of StructSeek-3B and StructSeek-7B. Best and second-best results within each backbone block are shown in **bold** and underlined, respectively.

Task	Link Prediction				Node Classification			
Method	PubMed	Amazon-Photo	Amazon-Computers	Reddit	PubMed	Amazon-Photo	Amazon-Computers	Reddit
GNN-based methods								
GCN	51.0 ↑39.3	49.8 ↑38.7	50.5 ↑43.4	53.2 ↑45.0	14.1 ↑70.5	8.0 ↑39.8	10.0 ↑53.5	7.1 ↑54.6
GraphSAGE	50.9 ↑39.4	51.6 ↑36.9	50.3 ↑43.6	57.6 ↑40.6	11.1 ↑73.5	8.2 ↑39.6	7.4 ↑56.1	3.1 ↑58.6
RevGAT	55.8 ↑34.6	59.4 ↑29.1	50.7 ↑43.2	61.8 ↑36.4	13.5 ↑71.1	12.1 ↑35.7	5.6 ↑57.9	8.2 ↑53.5
Qwen2.5-3B-Instruct								
Static-context LLMs								
GraphGPT	44.5 ↑43.9	54.5 ↑35.4	51.1 ↑43.4	49.8 ↑49.7	62.3 ↑17.7	3.0 ↑41.9	11.5 ↑51.4	9.5 ↑47.3
LLaGA	61.5 ↑26.9	51.3 ↑38.6	53.2 ↑41.3	43.7 ↑55.8	61.8 ↑18.2	4.2 ↑40.7	12.3 ↑50.6	10.1 ↑46.7
GraphICL	62.5 ↑25.9	54.8 ↑35.1	57.7 ↑36.8	62.0 ↑37.5	53.8 ↑26.2	35.2 ↑9.7	42.4 ↑20.5	37.1 ↑19.7
LinearRAG	46.0 ↑42.4	47.7 ↑42.2	45.7 ↑48.8	50.1 ↑49.4	64.2 ↑15.8	11.8 ↑33.1	20.1 ↑42.8	32.7 ↑24.1
Interactive graph agents								
GraphCoT	52.0 ↑36.4	50.7 ↑39.2	50.2 ↑44.3	50.3 ↑49.2	72.8 ↑7.2	38.0 ↑6.9	47.6 ↑15.3	37.2 ↑19.6
Search-R1	61.8 ↑26.6	54.6 ↑35.3	62.2 ↑32.3	69.0 ↑30.5	71.4 ↑8.6	37.8 ↑7.1	42.7 ↑20.2	38.7 ↑18.1
Search-o1	59.7 ↑28.7	54.9 ↑35.0	56.9 ↑37.6	54.5 ↑45.0	53.9 ↑26.1	34.6 ↑10.3	40.1 ↑22.8	38.2 ↑18.6
AgentGL-R++	72.3 ↑16.1	65.2 ↑24.7	68.7 ↑25.8	86.6 ↑12.9	73.6 ↑6.4	42.7 ↑2.2	49.8 ↑13.1	40.2 ↑16.6
AgentGL-GRPO	71.4 ↑17.0	66.4 ↑23.5	66.8 ↑27.7	79.7 ↑19.8	75.4 ↑4.6	42.6 ↑2.3	54.8 ↑8.1	39.2 ↑17.6
StructSeek-3B	**88.4**	**89.9**	**94.5**	**99.5**	**80.0**	**44.9**	**62.9**	**56.8**
Qwen2.5-7B-Instruct								
Static-context LLMs								
GraphGPT	48.7 ↑43.6	49.8 ↑37.3	44.4 ↑48.9	33.2 ↑63.6	79.2 ↑10.0	4.4 ↑46.3	12.3 ↑51.7	32.2 ↑34.4
LLaGA	67.7 ↑24.6	56.8 ↑30.3	60.2 ↑33.1	48.5 ↑48.3	78.6 ↑10.6	5.6 ↑45.1	14.6 ↑49.4	31.7 ↑34.9
GraphICL	72.3 ↑20.0	57.6 ↑29.5	61.8 ↑31.5	63.3 ↑33.5	70.4 ↑18.8	42.1 ↑8.6	49.0 ↑15.0	48.8 ↑17.8
LinearRAG	47.2 ↑45.1	46.3 ↑40.8	46.8 ↑46.5	49.8 ↑47.0	72.7 ↑16.5	47.6 ↑3.1	58.3 ↑5.7	51.4 ↑15.2
Interactive graph agents								
GraphCoT	50.5 ↑41.8	51.0 ↑36.1	50.2 ↑43.1	52.4 ↑44.4	81.9 ↑7.3	44.2 ↑6.5	54.4 ↑9.6	49.8 ↑16.8
Search-R1	72.3 ↑20.0	62.1 ↑25.0	70.1 ↑23.2	74.1 ↑22.7	81.6 ↑7.6	46.2 ↑4.5	54.6 ↑9.4	50.7 ↑15.9
Search-o1	65.8 ↑26.5	57.6 ↑29.5	60.6 ↑32.7	59.0 ↑37.8	69.5 ↑19.7	41.7 ↑9.0	49.3 ↑14.7	47.2 ↑19.4
AgentGL-R++	80.4 ↑11.9	77.0 ↑10.1	87.3 ↑6.0	**97.1** ↓0.3	82.6 ↑6.6	58.2 ↓7.5	67.5 ↓3.5	54.8 ↑11.8
AgentGL-GRPO	79.1 ↑13.2	69.4 ↑17.7	75.7 ↑17.6	88.5 ↑8.3	82.7 ↑6.5	**59.9** ↓9.2	**68.6** ↓4.6	54.1 ↑12.5
StructSeek-7B	**92.3**	**87.1**	**93.3**	96.8	**89.2**	50.7	64.0	**66.6**

These results demonstrate that the learned evidence acquisition policy generalizes effectively to unseen graphs across different domains and graph reasoning tasks.

The improvements are particularly notable for link prediction, where structural evidence plays a critical role in identifying relationships between nodes. At the 3B scale, StructSeek outperforms the strongest competing baseline on all four unseen graphs by 12.9–25.8 percentage points. At the 7B scale, it also achieves the best performance on PubMed, Amazon-Photo, and Amazon-Computers, with improvements of 6.0–11.9 percentage points over the strongest baselines. These gains at both model scales demonstrate the effectiveness of StructSeek for zero-shot link prediction. Notably, for link prediction, StructSeek-3B also outperforms its teacher model, DeepSeek-V4-Flash, on three of the four unseen graphs (Appendix C.2).

3.3 Effect of Structural Operators and Policy Training (RQ2)

To separate the contribution of structural operators from that of policy training, we compare three settings. The first evaluates the backbone with semantic operators only, the second equips the same backbone with structural operators, and the third applies the full SFT+RL training of StructSeek.

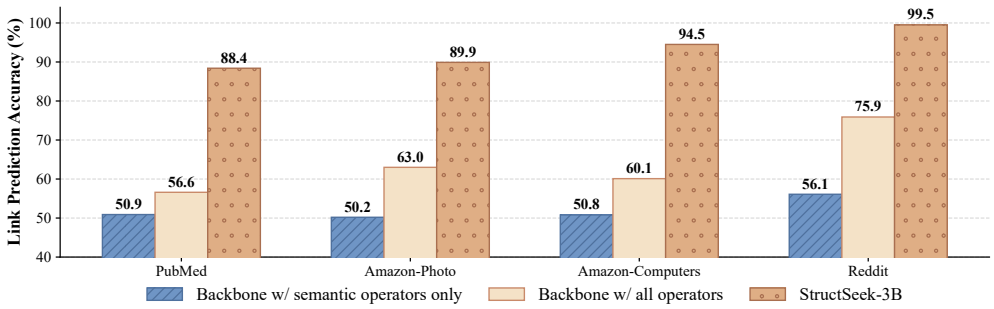

Figure 3: Effect of structural operators and policy training on zero-shot link prediction. The backbone (Qwen2.5-3B-Instruct) is evaluated with semantic operators only and with all operators, while StructSeek-3B uses the same full operator set after policy training.

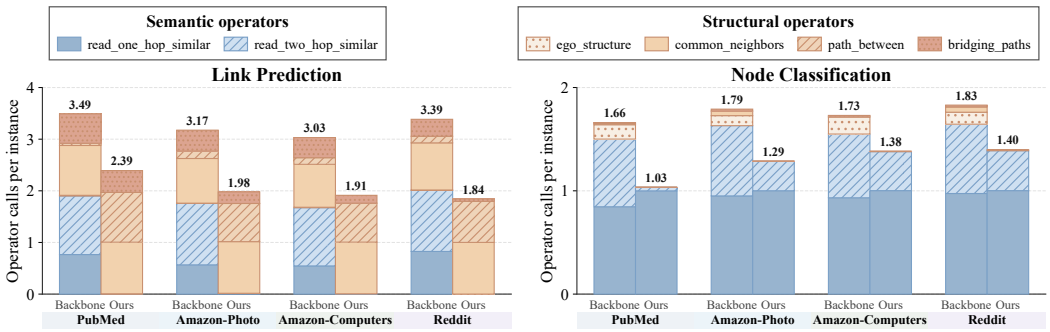

Figure 4: Evidence acquisition patterns of the backbone (Qwen2.5-3B-Instruct) and our method (StructSeek-3B) on zero-shot link prediction and node classification. Stacked bars show the average number of calls per instance to each semantic and structural operator.

As shown in Figure 3, providing structural operators already improves the backbone across all four unseen graphs, with gains of 5.7–19.8 percentage points. This demonstrates that structural evidence contains useful signals beyond node semantics.

After training, StructSeek-3B further improves over the backbone with structural operators by 23.6–34.4 percentage points. This indicates that the performance gain of StructSeek does not come only from accessing structural information, but also from learning an effective policy to acquire and exploit such evidence. Appendix C.1 further separates the two training stages: SFT accounts for most of this gain, and RL adds a further 2.4–5.8 percentage points on three of the four graphs.

3.4 Evidence Acquisition Policy Analysis (RQ3)

Figure 4 compares the operator usage behaviors of the backbone and StructSeek-3B on different graph reasoning tasks. Three main observations emerge.

First, the learned policy exhibits task-dependent evidence acquisition patterns. For link prediction, StructSeek primarily queries structural operators, especially common-neighbor and path-based operators, while for node classification it relies on semantic operators.

Second, training reshapes operator selection: the backbone spends more than half of its link prediction calls on semantic operators, whereas StructSeek-3B concentrates on structural operators.

Third, StructSeek reduces the average number of operator calls across all evaluated settings. Together with the improved prediction accuracy, this suggests that the learned policy acquires sufficient evidence with fewer operator calls.

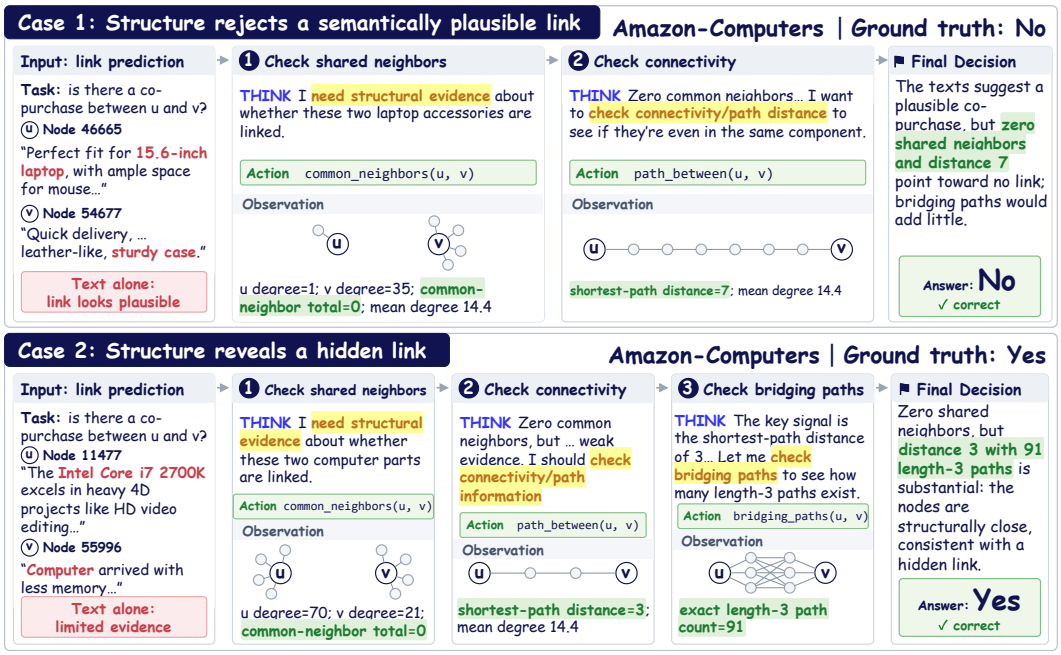

Figure 5: Representative evidence acquisition trajectories of StructSeek-3B for link prediction on Amazon-Computers. Reasoning is abridged.

3.5 Evidence Acquisition Case Study

To understand how StructSeek acquires and utilizes graph evidence, we visualize two representative link prediction trajectories in Figure 5, with full trajectories provided in Appendix D. The two cases demonstrate that structural evidence provides complementary information beyond node semantics. In the first case, although the node descriptions suggest a plausible product relationship, the agent retrieves negative structural evidence from the graph and correctly rejects the link. In the second case, the textual descriptions alone provide limited evidence, while the acquired structural information reveals a hidden relationship and enables correct prediction.

These examples show that StructSeek does not rely on textual similarity alone. Instead, it learns an adaptive evidence acquisition policy that determines when additional graph information is needed and how the acquired evidence should be used for prediction.

4 Related Work

4.1 Structural Evidence for Link Prediction

Link prediction differs from node-level prediction in that its target is defined over node pairs, making pair-specific structural evidence central to prediction. Prior studies identify local structural proximity, global structural proximity, and feature proximity as important factors underlying link formation (Mao et al., 2024). Analyses of message-passing GNNs further show that node-level representations do not necessarily preserve joint structural signals between candidate pairs, such as common-neighbor information (Zhang et al., 2021; Dong et al., 2024). Accordingly, modern LP methods explicitly model pairwise structure through enclosing subgraphs (Zhang & Chen, 2018), local structural sketches (Chamberlain et al., 2023), common-neighbor information (Wang et al., 2024b), pairwise structural encodings (Shomer et al., 2024), or multiple structural signals such as common neighbors and shortest paths (Ma et al., 2024). These studies establish that effective link prediction depends not only on node semantics but also on pair-specific structural evidence. Our work asks how an LLM can actively acquire task-relevant structural evidence for graph prediction under zero-shot transfer, with link prediction providing a particularly structure-dependent setting.

4.2 Static-Context Graph LLMs

Existing Graph LLMs make graph information accessible through language-model-compatible representations. GraphText and InstructGLM serialize graph structures into textual contexts (Zhao et al., 2023; Ye et al., 2024), while LLaGA, GraphGPT, and GOFA align graph representations with language models through embedding projection, graph instruction tuning, or joint graph–language modeling (Chen et al., 2024; Tang et al., 2024; Kong et al., 2025). More recent approaches strengthen structural encoding (Wang et al., 2024a; 2025), while retrieval-based methods such as G-Retriever and LinearRAG construct task-relevant graph contexts (He et al., 2024b; Zhuang et al., 2026). GraphICL and Graph-R1 further improve reasoning over supplied graph information through structured prompting or explicit reasoning (Sun et al., 2025; Wu et al., 2025). Despite these different designs, graph structure is primarily presented to the LLM as constructed, encoded, or retrieved context. Textual contexts require structural reasoning by the LLM, while learned representations make structural evidence implicit.

4.3 Interactive Graph Agents

Tool-using LLMs provide an alternative paradigm in which models interact with external environments and condition later decisions on previously acquired evidence. ReAct and Toolformer establish general frameworks for interleaving reasoning with actions or tool calls (Yao et al., 2023; Schick et al., 2023), while Search-R1 and Search-o1 study iterative information-seeking behavior (Jin et al., 2025; Li et al., 2025b). Recent work extends this paradigm to graphs: GraphCoT and AgentGL support dynamic graph exploration and retrieval (Jin et al., 2024b; Sun et al., 2026), while Graph-ToolFormer, GRRAF, and GraphChain expose graph APIs, executable graph queries, or specialized graph tools (Zhang, 2023; Li et al., 2025a; Wei et al., 2025). Agentic GraphRAG methods further learn multi-turn graph retrieval policies for knowledge-intensive question answering (Luo et al., 2026; Park et al., 2026). These methods focus primarily on graph exploration, retrieval, general graph operations, or question answering. StructSeek instead formulates interaction as task-driven evidence acquisition for zero-shot graph prediction.

5 Conclusion

We presented StructSeek, an interactive graph agent that instantiates a **structure-as-tool** paradigm for zero-shot graph learning. Rather than requiring LLMs to recover graph relations from serialized graph contexts, StructSeek directly acquires task-relevant semantic and structural evidence through dedicated graph operators. We further construct teacher trajectories and train the acquisition policy with supervised fine-tuning and reinforcement learning, enabling the model to learn what evidence to acquire and when sufficient evidence has been collected. Experiments on node classification and link prediction show that StructSeek achieves the best results in most settings, with particularly strong gains on link prediction, supporting the importance of direct structural evidence for relational prediction. A remaining limitation is that the learned acquisition policy may transfer imperfectly across graphs with substantially different structural distributions, where the same structural evidence can have different predictive significance. Future work could investigate graph-aware calibration or training objectives that better adapt evidence acquisition to unseen graph domains.

Reproducibility Statement

The datasets, query construction, and zero-shot evaluation protocol are described in Section 3.1 and Appendix A. Appendix B specifies the evidence acquisition environment, including the semantic and structural operators, working-graph construction, interaction protocol, and prompts used by the agent. Appendix C provides details of teacher trajectory construction and filtering, as well as the SFT, RL, and inference configurations. Appendix D provides the complete trajectories used in the case studies. Code is available in our anonymous repository at `https://anonymous.4open.science/r/StructSeek-36D4`.

Ethics Statement

This work uses existing public benchmark graphs of scientific papers, product co-purchases, and public forum posts, and introduces no new data collection involving human participants. Node texts are used as distributed with the original benchmarks. Teacher trajectories were generated through a commercial LLM API under its terms of service using benchmark-provided node texts, the prompts described in Appendix B, and observations computed from the corresponding graphs.

AI Use Statement

Generative AI tools were used to aid and polish writing, assist with retrieval and discovery of related work, support research execution through code implementation and debugging, and generate synthetic training data in the form of teacher trajectories used for supervised fine-tuning. Generative AI tools were not used to draft sections of the paper or prove mathematical claims. All AI-assisted work was reviewed and verified by the authors, who take responsibility for the final content of this paper.

References

Benjamin Paul Chamberlain, Sergey Shirobokov, Emanuele Rossi, Fabrizio Frasca, Thomas Markovich, Nils Yannick Hammerla, Michael M. Bronstein, and Max Hansmire. Graph neural networks for link prediction with subgraph sketching. In *International Conference on Learning Representations*, 2023.

Runjin Chen, Tong Zhao, Ajay Kumar Jaiswal, Neil Shah, and Zhangyang Wang. LLaGA: Large language and graph assistant. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 7809–7823, 2024.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Kaiwen Dong, Zhichun Guo, and Nitesh V. Chawla. Pure message passing can estimate common neighbor for link prediction. *Advances in Neural Information Processing Systems*, 37:73000–73035, 2024.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large language models. In *International Conference on Learning Representations*, 2024.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. Harnessing explanations: LLM-to-LM interpreter for enhanced text-attributed graph representation learning. In *International Conference on Learning Representations*, 2024a.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907, 2024b. doi: 10.52202/079017-4224.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open Graph Benchmark: Datasets for machine learning on graphs. *Advances in Neural Information Processing Systems*, 33:22118–22133, 2020.

Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. Large language models on graphs: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 36 (12):8622–8642, 2024a. doi: 10.1109/TKDE.2024.3469578.

Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, and Jiawei Han. Graph Chain-of-Thought: Augmenting large language models by reasoning on graphs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 163–184, 2024b. doi: 10.18653/v1/2024.findings-acl.11.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O. Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Proceedings of the 2nd Conference on Language Modeling*, 2025.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Lecheng Kong, Jiarui Feng, Hao Liu, Chengsong Huang, Jiaxin Huang, Yixin Chen, and Muhan Zhang. GOFA: A generative one-for-all model for joint graph language modeling. In *International Conference on Learning Representations*, 2025.

István A. Kovács, Katja Luck, Kerstin Spirohn, Yang Wang, Carl Pollis, Sadie Schlabach, Wenting Bian, Dae-Kyum Kim, Nishka Kishore, Tong Hao, Michael A. Calderwood, Marc Vidal, and Albert-László Barabási. Network-based prediction of protein interactions. *Nature Communications*, 10:1240, 2019. doi: 10.1038/s41467-019-09177-y.

Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. Training graph neural networks with 1000 layers. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6437–6449, 2021.

Hanqing Li, Sharika Mahadevan, Kiran Jyothi Sheena, Henry Liang, and Diego Klabjan. Zero-shot graph reasoning via retrieval augmented framework with LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 17041–17054, 2025a. doi: 10.18653/v1/2025.findings-emnlp.924.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, 2025b. doi: 10.18653/v1/2025.emnlp-main.276.

Yuhan Li, Peisong Wang, Zhixun Li, Jeffrey Xu Yu, and Jia Li. ZeroG: Investigating cross-dataset zero-shot transferability in graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1725–1735, 2024. doi: 10.1145/3637528.3671982.

Shuming Liang, Yu Ding, Zhidong Li, Bin Liang, Siqi Zhang, Yang Wang, and Fang Chen. Can GNNs learn link heuristics? A concise review and evaluation of link prediction methods. *IEEE Transactions on Big Data*, 12(1):1–14, 2026. doi: 10.1109/TBDATA.2025.3600031.

David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007. doi: 10.1002/asi.20591.

Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for All: Towards training one graph model for all classification tasks. In *International Conference on Learning Representations*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Haoran Luo, Haihong E, Guanting Chen, Qika Lin, Yikai Guo, Fangzhi Xu, Zemin Kuang, Meina Song, Xiaobao Wu, Yifan Zhu, and Anh Tuan Luu. Graph-R1: Towards agentic GraphRAG framework via end-to-end reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Li Ma, Haoyu Han, Juanhui Li, Harry Shomer, Hui Liu, Xiaofeng Gao, and Jiliang Tang. Mixture of link predictors on graphs. *Advances in Neural Information Processing Systems*, 37:16043–16070, 2024.

Haitao Mao, Juanhui Li, Harry Shomer, Bingheng Li, Wenqi Fan, Yao Ma, Tong Zhao, Neil Shah, and Jiliang Tang. Revisiting link prediction: A data perspective. In *International Conference on Learning Representations*, 2024.

Jinyoung Park, Sanghyeok Lee, Omar Zia Khan, Hyunwoo J. Kim, and Joo-Kyung Kim. HyperGraphPro: Progress-aware reinforcement learning for structure-guided hypergraph RAG. *arXiv preprint arXiv:2601.17755*, 2026.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008. doi: 10.1609/aimag.v29i3.2157.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Yang Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Harry Shomer, Yao Ma, Haitao Mao, Juanhui Li, Bo Wu, and Jiliang Tang. LPFormer: An adaptive graph transformer for link prediction. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2686–2698, 2024. doi: 10.1145/3637528.3672025.

Yuanfu Sun, Zhengnan Ma, Yi Fang, Jing Ma, and Qiaoyu Tan. GraphICL: Unlocking graph learning potential in LLMs through structured prompt design. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2440–2459, 2025. doi: 10.18653/v1/2025.findings-naacl.131.

Yuanfu Sun, Kang Li, Dongzhe Fan, Jiajin Liu, and Qiaoyu Tan. AgentGL: Towards agentic graph learning with LLMs via reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25313–25335, 2026. doi: 10.18653/v1/2026.acl-long.1161.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. GraphGPT: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 491–500, 2024. doi: 10.1145/3626772.3657775.

Jianheng Tang, Qifan Zhang, Yuhan Li, Nuo Chen, and Jia Li. GraphArena: Evaluating and exploring large language models on graph computation. In *International Conference on Learning Representations*, 2025.

Duo Wang, Yuan Zuo, Fengzhi Li, and Junjie Wu. LLMs as zero-shot graph learners: Alignment of GNN representations with LLM token embeddings. *Advances in Neural Information Processing Systems*, 37:5950–5973, 2024a. doi: 10.52202/079017-0193.

Duo Wang, Yuan Zuo, Guangyue Lu, and Junjie Wu. UniGTE: Unified graph–text encoding for zero-shot generalization across graph tasks and domains. *Advances in Neural Information Processing Systems*, 38:92095–92120, 2025. doi: 10.52202/085713-3081.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36:30840–30861, 2023.

Xiyuan Wang, Haotong Yang, and Muhan Zhang. Neural common neighbor with completion for link prediction. In *International Conference on Learning Representations*, 2024b.

Chunyu Wei, Wenji Hu, Xingjia Hao, Xin Wang, Yifan Yang, Yunhai Wang, Yang Tian, and Yueguo Chen. GraphChain: Large language models for large-scale graph analysis via tool chaining. *Advances in Neural Information Processing Systems*, 38:7402–7436, 2025. doi: 10.52202/085713-0253.

Yicong Wu, Guangyue Lu, Yuan Zuo, Huarong Zhang, and Junjie Wu. Graph-R1: Incentivizing the zero-shot graph learning capability in LLMs via explicit reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 23909–23927, 2025. doi: 10.18653/v1/2025.emnlp-main.1220.

Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, et al. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. *Advances in Neural Information Processing Systems*, 36:17238–17264, 2023.

Hao Yan, Chaozhuo Li, Jun Yin, Zhigang Yu, Weihao Han, Mingzheng Li, Zhengxin Zeng, Hao Sun, and Senzhang Wang. When graph meets multimodal: Benchmarking and meditating on multimodal attributed graph learning. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5842–5853, 2025. doi: 10.1145/3711896.3737404.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 1955–1973, 2024. doi: 10.18653/v1/2024.findings-eacl.132.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077, 2024. doi: 10.18653/v1/2024.findings-acl.181.

Jiawei Zhang. Graph-ToolFormer: To empower LLMs with graph reasoning ability via prompt augmented by ChatGPT. *arXiv preprint arXiv:2304.11116*, 2023.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. *Advances in Neural Information Processing Systems*, 34:9061–9073, 2021.

Jianan Zhao, Le Zhuo, Yikang Shen, Meng Qu, Kai Liu, Michael Bronstein, Zhaocheng Zhu, and Jian Tang. GraphText: Graph reasoning in text space. *arXiv preprint arXiv:2310.01089*, 2023.

Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. LinearRAG: Linear graph retrieval augmented generation on large-scale corpora. In *International Conference on Learning Representations*, 2026.

A Dataset and Evaluation Details

Graphs and statistics. We adopt the benchmark setting of AgentGL (Sun et al., 2026): six text-attributed graphs from three domains, released with node texts, edge lists, class labels, and node splits. The policy is trained on the two training graphs, OGB-Arxiv and OGB-Products (Hu et al., 2020; He et al., 2024a), and evaluated on four graphs that are not used during training: PubMed (Sen et al., 2008), Amazon-Photo and Amazon-Computers (Yan et al., 2023), and Reddit (Yan et al., 2025). Table 2 summarizes the statistics of all graphs. Node and edge counts are as reported by

AgentGL. Operators use the released edge lists as undirected simple graphs, merging reciprocal entries and dropping self-loops, so graph-level context in operator observations (e.g., mean degree) is computed on this representation. In the citation graphs, nodes represent papers and edges represent citation relationships; in the product graphs, nodes represent products and edges represent co-purchase relationships; in Reddit, nodes represent posts and edges represent co-post relationships. Node texts follow the released benchmark preprocessing, including paper titles and abstracts, product reviews, and user posts.

Table 2: Statistics of the six graphs, following AgentGL: node and edge counts of the released graphs and the size of the node-classification label space.

Dataset	Domain	Nodes	Edges	Classes	Role
OGB-Arxiv	citation	169,343	1,166,245	40	training
OGB-Products (subset)	product co-purchase	54,025	74,420	47	training
PubMed	citation	19,717	44,338	3	zero-shot transfer
Amazon-Photo	product co-purchase	48,362	500,939	12	zero-shot transfer
Amazon-Computers	product co-purchase	87,229	721,107	10	zero-shot transfer
Reddit	social	15,894	566,160	20	zero-shot transfer

Training instances are constructed only from the two training graphs. A candidate pool is sampled from the standard training split for node classification and from existing and non-existing node pairs for link prediction. The teacher corpus and reinforcement-learning instances are constructed from this pool.

Task instances. We evaluate StructSeek on two graph reasoning tasks: node classification and link prediction. For node classification, each instance provides a target node and requires the model to predict its category from the predefined label space. For link prediction, each instance provides a pair of nodes and requires the model to determine whether an edge exists between them.

For each graph, we sample 1,000 node-classification instances from the standard test split. For link prediction, we construct 1,000 balanced instances, including 500 positive pairs sampled from existing edges and 500 negative pairs sampled from non-connected node pairs. For positive pairs, the target edge is masked before interaction to prevent direct access to the answer through the operators.

Zero-shot protocol. The policy is trained once on the two training graphs and evaluated without adaptation on the four transfer graphs. These graphs, including their structures, node texts, and label spaces, are not observed during training. For each task, the same prompt configuration, operator set, edge-masking protocol, and interaction budget are used across all graphs.

Baselines. The baseline results in Table 1 are taken from AgentGL and obtained under its zero-shot protocol (Sun et al., 2026). We compare StructSeek with representative methods from three families: GNN-based methods, static-context LLMs, and interactive graph agents. The compared methods use the same benchmark graphs and task settings for evaluation.

B Agent Environment and Prompts

This appendix describes the interaction protocol, operator interface, and prompt configuration used during training and evaluation. The same interaction environment, operator set, and edge-masking protocol are used throughout training and evaluation.

Interaction protocol. Each episode starts from a task instance and proceeds through a sequence of interaction turns. At each turn, the policy either requests information from the graph environment through an available operator or produces the final prediction. After an operator call, the environment executes the corresponding operator and returns the resulting observation. The observation is appended to the interaction history and becomes available in subsequent turns.

History and budget. During interaction, the policy receives the task instance together with the history generated by previous turns. The history records the previous actions and returned observations. Each episode is limited to 11 turns, allowing at most 10 operator calls before the final answer.

The same interaction budget is used during teacher trajectory collection, supervised fine-tuning, reinforcement learning, and evaluation.

Link-prediction edge masking. For link-prediction instances, the queried edge is masked before interaction when the node pair corresponds to an existing edge. This prevents graph operators from directly revealing the target label. For node-classification instances, no edge masking is applied. The same masking protocol is used during training and evaluation.

Operators. The six operators available to the agent are summarized in Table 3. Semantic operators provide bounded textual views of node neighborhoods, while structural operators return deterministic graph measurements.

Table 3: The six evidence operators used by the agent. Semantic operators return at most five node texts, while structural measurements are exact.

Operator	Type	Arguments	Returns	Bound
`read_one_hop_similar`	semantic	`node_id`	Textual view of the one-hop neighborhood with selected text-bearing neighbors	At most 5 nodes
`read_two_hop_similar`	semantic	`node_id`	Textual view of the two-hop neighborhood with selected text-bearing nodes	At most 5 nodes
`ego_structure`	structural	`node_id`	Node degree, local clustering coefficient, k-core number, and mean and median neighbor degree	Exact measurement
`common_neighbors`	structural	`u,v`	Number of common neighbors and representative examples	Exact measurement
`path_between`	structural	`u,v`	Shortest-path distance between two nodes	Exact measurement
`bridging_paths`	structural	`u,v`	Number of length-three paths between two nodes	Exact measurement

Structural-query ablation. The structural ablation in Section 3.3 removes the four structural operators and retains only the two semantic operators. The interaction budget and edge-masking protocol remain unchanged.

Prompts. The system prompt consists of a shared protocol and a task-specific prompt for node classification or link prediction, shared across all graphs. Both are summarized below. At each turn, the user message contains the query and interaction history. The model responds with a reasoning block in `<think>` tags followed by exactly one operator call in `<tool_call>` tags or a final answer in `<answer>` tags. Format violations receive the penalty described in Appendix C.

Shared protocol

Role

You are a graph-task agent. You solve graph reasoning tasks by deciding whether to obtain additional graph information or provide the final answer.

Available Information

You receive:
- a target node or node pair;
- available text descriptions;
- a fixed set of answer choices;
- optional graph operators.

Shared protocol (continued)

The returned observations come from the current working graph.

Task

At each interaction step, decide whether the current information is sufficient for prediction or whether additional graph information is required.

When necessary, request information through available graph operators and use the returned observations together with the current trajectory to reason about the task.

Interaction Rules

- The trajectory contains the complete history of previous reasoning steps, actions, and observations.
- The trajectory is read-only and must not be modified or extended.
- Do not assume the result of an operator before it appears in the trajectory.
- Do not repeat an identical operator call with the same arguments.

Output Requirements

Each response must contain exactly one action:

1. Tool action:
Request one graph operator.

2. Answer action:
Return the final answer using exactly one of the provided answer choices.

No additional text is allowed outside the required format.

Constraints

- Use only information provided in the task description and observed trajectory.
- Follow the task-specific instructions and output format.
- Complete the task within the allowed interaction budget.

Task-specific prompt: node classification

Task

Node classification: assign the target node to one category from the provided label space.

The final answer must be exactly one valid category.

Available Information

The task provides:
- the target node description;
- available answer choices;
- textual and structural information obtained through operators.

Available Operators

Semantic operators provide textual views of graph neighborhoods.

Structural operators provide graph measurements computed from the current working graph.

Task-specific prompt: node classification (continued)

Returned structural measurements may include:
- neighborhood statistics;
- connectivity information;
- path-based information;
- node-level structural properties.

The agent should interpret observations according to the graph context and the node classification task.

Reasoning Requirements

Structural and textual observations provide evidence for prediction but do not directly determine the answer.

Decide independently:
- whether additional information is needed;
- which operator to use;
- when sufficient evidence has been collected.

No fixed operator order is required.

Output Requirements

Return exactly one category from the provided answer choices.

Task-specific prompt: link prediction

Task

Link prediction: determine whether the queried node pair corresponds to an edge.

The final answer must be exactly one provided answer choice.

Working Graph Protocol

For a queried node pair:
- if the target edge exists, it is removed before interaction;
- all operator results are computed from the resulting working graph.

The hidden edge label is never directly revealed through graph operators.

Available Information

The task provides:
- the target node pair;
- node descriptions;
- answer choices;
- textual and structural information obtained through operators.

Available Operators

Semantic operators provide textual views of graph neighborhoods.

Structural operators provide graph measurements computed from the current working graph.

Returned structural measurements may include:
- neighborhood statistics;
- connectivity information;

Task-specific prompt: link prediction (continued)

- path-based information;
- node-level structural properties.

The agent should interpret observations according to the graph context and the link prediction task.

Reasoning Requirements

Structural and textual observations provide evidence for prediction but do not directly determine the answer.

Decide:
- whether additional information is needed;
- which operator to call;
- when enough evidence has been obtained.

Output Requirements

Return exactly one answer choice.

C TRAINING DETAILS

Training pipeline. We train StructSeek through three stages: teacher trajectory collection, supervised fine-tuning, and reinforcement learning. First, we collect teacher trajectories on the training graphs described in Appendix A. During collection, the teacher interacts with the graph environment using the same operator interface and edge-masking protocol as the final agent. Each trajectory contains the interaction history, operator calls, observations, and the final prediction.

Before supervised fine-tuning, we retain only trajectories with correct final predictions that are not produced by forced termination and are not marked as overlength or discarded. Trajectories with teacher reasoning exceeding 2,400 characters or containing meta-level contamination are also removed, and decision steps marked as ineligible for SFT are excluded. Each remaining trajectory is then split at its decision points into training examples, where each example consists of the available interaction history and the next action selected by the teacher. These examples are used to supervise the acquisition policy.

Supervised fine-tuning and reinforcement learning. The supervised fine-tuning stage optimizes the policy to imitate the teacher's acquisition decisions. After SFT, we further optimize the policy with reinforcement learning in the graph environment. During RL training, the policy generates complete episodes by interacting with the graph and receiving operator observations.

The episode reward is defined as $R = R_{\text{correct}} + R_{\text{query}} + R_{\text{format}}$. The correctness term is $+1.0$ for a correct prediction and -1.0 otherwise. The query term is $+0.2$ when a correct prediction is made after at least one operator call, -0.2 when an incorrect prediction is made without using any operator, and 0 otherwise. The format term is $+0.1$ for a valid response containing exactly one `<answer>` block and -0.3 for malformed responses, including missing or duplicated answer blocks or protocol tags appearing inside the answer block.

We use GRPO for policy optimization following the formulation described in Section 2. All training instances are sampled from the training graphs only, and the zero-shot transfer graphs are never used during optimization.

Training configuration. We train StructSeek-3B and StructSeek-7B from Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct (Yang et al., 2024). The same configuration is applied to both backbones. Teacher trajectories, supervised fine-tuning, reinforcement learning, and inference use the interaction constraints described in Appendix B. The main training hyperparameters are summarized in Table 4.

Table 4: Training and inference configurations.

Stage	Setting	Value
Teacher collection	Teacher model	DeepSeek-V4-Flash (snapshot 0731)
	Decoding	Greedy decoding
	Trajectories / decision-step examples	4,168 / 11,338
Supervised fine-tuning	Backbone	Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct
	Optimizer	AdamW (Loshchilov & Hutter, 2019)
	Learning rate	2×10^{-5}
	Epochs	2
	Batch size	32
Reinforcement learning	Algorithm	GRPO
	Training instances	3,000
	Group size	8
	Training batch size	16
	Rollout temperature	1.0
	Learning rate	10^{-6}
	Epochs	2
Inference	Decoding	Greedy decoding
	Maximum model length	6,144 tokens
	Maximum generation length	1,792 tokens
	Maximum turns	11
	Maximum operator calls	10

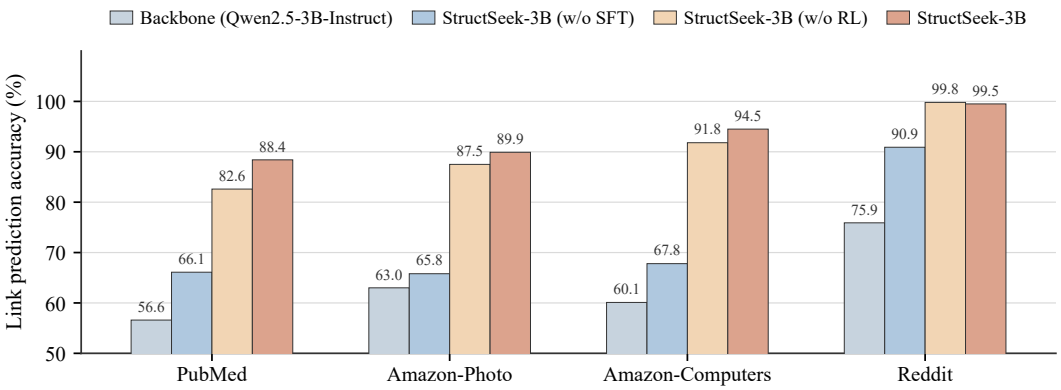

Figure 6: Training component analysis for zero-shot link prediction on the four unseen graphs. All variants use the same Qwen2.5-3B-Instruct backbone, operator set, prompts, and evaluation protocol. We compare the backbone, StructSeek-3B without SFT, StructSeek-3B without RL, and the full StructSeek-3B trained with both SFT and RL.

C.1 Analysis of Training Components

To analyze how different training components contribute to the learned evidence acquisition policy, we conduct ablation experiments on supervised fine-tuning (SFT) and reinforcement learning (RL). We focus on link prediction because it requires explicit structural evidence between node pairs, providing a clearer view of how training affects evidence acquisition. All variants use the same backbone, operator set, prompts, and evaluation protocol, differing only in their training strategy. Specifically, we compare the backbone, StructSeek-3B without SFT, StructSeek-3B without RL, and the full StructSeek-3B.

As shown in Figure 6, SFT contributes the largest performance gain. Compared with the backbone, the variant without RL improves link prediction accuracy by 26.0, 24.5, 31.7, and 23.9 points on the four unseen graphs, respectively. In contrast, training without SFT yields smaller gains of 9.5, 2.8, 7.7, and 15.0 points. Starting from the SFT-trained policy, RL further improves accuracy by 5.8, 2.4, and 2.7 points on PubMed, Amazon-Photo, and Amazon-Computers, while performance on Reddit remains nearly unchanged (-0.3 points). These results show that SFT establishes the

Table 5: A large reasoning model and the 3B policy under the same interface, the same six operators, working-graph masking, and turn budget, on the four zero-shot transfer graphs. DeepSeek-V4-Flash generated the teacher trajectories. Bold marks the best entry per column and underline the second best.

	Link prediction				**Node classification**			
Model	PubMed	Amazon-Photo	Amazon-Computers	Reddit	PubMed	Amazon-Photo	Amazon-Computers	Reddit
Backbone (Qwen2.5-3B-Instruct)	56.6	63.0	60.1	75.9	64.6	30.7	46.2	44.7
StructSeek-3B	**88.4**	<u>89.9</u>	**94.5**	**99.5**	<u>80.0</u>	<u>44.9</u>	<u>62.9</u>	<u>56.8</u>
DeepSeek-V4-Flash (snapshot 0731)	<u>75.0</u>	**92.5**	<u>92.3</u>	<u>97.7</u>	**91.5**	**70.4**	**74.2**	**84.7**

primary evidence acquisition capability, while RL provides additional refinement of the learned policy.

C.2 Comparison with the Teacher Model

To further examine the effectiveness of trajectory-based training, we compare the trained 3B policy with the large reasoning model used to generate the teacher trajectories. Both models use the same operator interface, turn budget, and evaluation protocol, with results reported in Table 5.

On link prediction, the 3B policy outperforms the teacher model on three of the four unseen graphs, improving by 13.4 points on PubMed and around 2 points on Amazon-Computers and Reddit, while trailing by 2.6 points on Amazon-Photo. These results demonstrate that the training process can transfer useful evidence acquisition behaviors from the teacher model to a much smaller backbone.

D Case Studies

This appendix presents four complete trajectories of StructSeek-3B on the zero-shot transfer graphs. Cases 1 and 2 are the link prediction trajectories abridged in Figure 5; Cases 3 and 4 are node classification trajectories on Amazon-Photo and Reddit. Each step reports the model's reasoning (Think) followed by either an operator call (Action) with the returned observation (Observation) or the final answer, all reproduced from the evaluation records. Operator calls are written in function notation, and per-neighbor similarity scores are omitted from the observations of semantic operators.

Case 1: link prediction on Amazon-Computers (zero-shot transfer)

Task Link prediction **Dataset** Amazon-Computers **Query** Node 46665 ↔ Node 54677
Ground truth No, they do not have co-purchased or co-viewed relationships
Prediction No, they do not have co-purchased or co-viewed relationships (correct)

Target node descriptions
Node 46665: "Perfect fit for 15.6-inch laptop, with ample space for mouse and cord."
Node 54677: "Quick delivery, vibrant colors, easy assembly, leather-like, sturdy case."
Answer choices
Yes, they have co-purchased or co-viewed relationships; No, they do not have co-purchased or co-viewed relationships.

Step 1
Think I need structural evidence about whether these two laptop accessories are linked. I'll start by checking their shared neighbors in the working graph.
Action `common_neighbors(u=46665, v=54677)`

Working graph: Node 46665 degree=1; Node 54677 degree=35; exact common-neighbor total=0.
Graph context: 87229 nodes, mean degree 14.4.

Case 1: link prediction on Amazon-Computers (zero-shot transfer) (continued)

Step 2
Think Zero common neighbors despite node 46665 having degree 1 and node 54677 degree 35. I want to check connectivity/path distance to see if they're even in the same component, which gives more evidence.
Action `path_between(u=46665, v=54677)`

> Working graph path between Node 46665 and Node 54677: shortest-path distance=7. Graph context: 87229 nodes, mean degree 14.4. For held-out link-prediction queries, the queried direct edge is absent from this working graph under either label; the measurement alone does not reveal it.

Step 3
Think The two products are both laptop accessories (a laptop sleeve/case and a laptop stand/mousepad), suggesting a plausible co-purchase relationship. However, the working graph shows zero common neighbors, shortest-path distance 7, and node 46665 has degree 1. This sparse connectivity suggests no strong structural proximity. I could check bridging paths for length-3 connections, but the evidence already points toward no link. Given the low degree of 46665 and the long distance, the absence of shared neighbors and the weak connectivity make a co-purchase/co-view link unlikely. I'll answer no.
Answer No, they do not have co-purchased or co-viewed relationships

Case 2: link prediction on Amazon-Computers (zero-shot transfer)

Task Link prediction **Dataset** Amazon-Computers **Query** Node 11477 ↔ Node 55996
Ground truth Yes, they have co-purchased or co-viewed relationships
Prediction Yes, they have co-purchased or co-viewed relationships (correct)

Target node descriptions
Node 11477: "The Intel Core i7 2700K excels in heavy 4D projects like HD video editing, offering better performance and load tolerance than the i5."
Node 55996: "Computer arrived with less memory, faced repeated boot failures due to memory configuration issues despite fixes."
Answer choices
Yes, they have co-purchased or co-viewed relationships; No, they do not have co-purchased or co-viewed relationships.

Step 1
Think I need structural evidence about whether these two computer parts are linked. I'll start by checking their shared neighbors in the working graph.
Action `common_neighbors(u=11477, v=55996)`

> Working graph: Node 11477 degree=70; Node 55996 degree=21; exact common-neighbor total=0. Graph context: 87229 nodes, mean degree 14.4.

Step 2
Think Zero common neighbors, but the graph is large and sparse (mean degree 14.4), so zero shared neighbors is weak evidence. I should check connectivity/path information to see if there is any structural signal linking these two nodes.
Action `path_between(u=11477, v=55996)`

> Working graph path between Node 11477 and Node 55996: shortest-path distance=3. Graph context: 87229 nodes, mean degree 14.4. For held-out link-prediction queries, the queried direct edge is absent from this working graph under either label; the measurement alone does not reveal it.

Step 3
Think The two nodes share zero common neighbors, but the graph is very large (87229 nodes, mean degree 14.4), so zero shared neighbors is weak evidence. The key signal is the shortest-path distance of 3, meaning there is a length-3 path connecting them. In a co-purchase/co-view graph, a short path (distance 3) is consistent with a hidden edge. Let me check bridging paths to see how many length-3 paths exist, which would strengthen the evidence.
Action `bridging_paths(u=11477, v=55996)`

> Working graph: exact simple length-3 path count between Node 11477 and Node 55996=91. For held-out link-prediction queries, the queried direct edge is hidden before this measurement.

Case 2: link prediction on Amazon-Computers (zero-shot transfer) (continued)

Step 4
Think Evidence: zero common neighbors, shortest-path distance 3, and 91 length-3 bridging paths. In a graph of 87229 nodes with mean degree 14.4, a distance-3 connection with 91 distinct length-3 paths is substantial. This indicates the two nodes are structurally close, consistent with a hidden co-purchase/co-view relationship. Additional calls would add little; the evidence supports a positive link.
Answer Yes, they have co-purchased or co-viewed relationships

Case 3: node classification on Amazon-Photo (zero-shot transfer)

Task Node classification **Dataset** Amazon-Photo **Query** Node 43529
Ground truth Accessories
Prediction Accessories (correct)

Target node description
Node 43529: "Product performs excellently, meeting all expectations."
Answer choices
Accessories; Bags & Cases; Binoculars & Scopes; Digital Cameras; Film Photography; Flashes; Lenses; Lighting & Studio; Tripods & Monopods; Underwater Photography; Video; Video Surveillance.

Step 1
Think We need to classify node 43529. We should inspect its neighborhood to infer category. Let's start with `read_one_hop_similar` on 43529.
Action `read_one_hop_similar(node_id=43529)`

> Node 43529 1-hop view: exact total=7; text-bearing=7; showing=5; selection=target-text TF-IDF cosine descending.
> - Node 11209: Substantial strap, easy-to-use lens cap fits perfectly, quick to atta...
> - Node 15421: Sony-branded 49mm lens cap for Fuji X100, highly durable, textured, s...
> - Node 27633: Perfect fit for Sony a6000, strong springs, secure attachment, good s...
> - Node 28037: Sony A6000 cap fits perfectly; purchased lens cap holder to prevent l...
> - Node 31497: Case lacked button markings, interfered with tripod use, obstructed b...

Step 2
Think The one-hop neighbors are all camera accessories (lens caps, lens cap holders, tripod cases). The target node 43529 is described as performing excellently, meeting all expectations. These neighbors strongly suggest a camera accessory category. Among the choices, "Accessories" fits best. I could verify with structural profile, but the textual evidence is already clear. Answer now.
Answer Accessories

Case 4: node classification on Reddit (zero-shot transfer)

Task Node classification **Dataset** Reddit **Query** Node 9403
Ground truth pottery
Prediction pottery (correct)

Target node description
Node 9403: "practicing rib throwing with a heat gun to get my walls extra thin"
Answer choices
The 20 subreddit names of the graph, including pottery, woodcarving, gardening, and designmyroom.

Step 1
Think The target node describes practicing rib throwing with a heat gun to get walls extra thin. This sounds like a DIY/rib-stretching technique, likely related to woodworking or furniture making. I should inspect the node's neighborhood to gather evidence before answering. Let me start with the one-hop similar view.
Action `read_one_hop_similar(node_id=9403)`

Case 4: node classification on Reddit (zero-shot transfer) (continued)

Node 9403 1-hop view: exact total=8; text-bearing=8; showing=5; selection=target-text TF-IDF cosine descending.
- Node 9476: a piece from my first raku firing. it cracked a little bit but still...
- Node 9538: first glaze test was a successful firing but with lackluster results....
- Node 9577: a fun round one 14"x14" burnished black b mix on top of soldate 60
- Node 9640: large serving bowl 15"x5" cone 10 reduction :)
- Node 9850: i call this one “old timey bicycle horn”

Step 2
Think The one-hop neighbors are all about ceramics/raku firing/glaze testing, which strongly suggests the user is in the pottery/sculpture domain. The target description about “rib throwing with a heat gun to get walls extra thin” could be a ceramic technique (ribbing) or possibly woodburning/decorating. The neighbor texts are clearly pottery-related. I could check the ego structure profile or two-hop view for more confirmation, but the textual evidence is already quite strong. Let me consider whether a structural profile adds useful info. The neighbors are all ceramics, so the category is likely ‘pottery’. I’ll answer now.
Answer pottery