Prior Knowledge Makes It Possible: From Sublinear Graph Algorithms to LLM Test-Time Methods

Avrim Blum
Toyota Technological Institute at Chicago

Daniel Hsu
Columbia University

Cyrus Rashtchian
Google Research

Donya Saless
Toyota Technological Institute at Chicago

Abstract

Test-time augmentation, such as Retrieval-Augmented Generation (RAG) or tool use, critically depends on an interplay between a model's parametric knowledge and externally retrieved information. However, the theoretical underpinnings of this relationship remain poorly understood. Specifically, it is not clear how much pre-training knowledge is required to answer queries with a small number of augmentation steps, which is a desirable property in practice. To address this question, we formulate multi-step reasoning as an s-t connectivity problem on a knowledge graph. We represent a model's pre-training parametric knowledge as a partial, potentially noisy subgraph. We view augmentation as querying an oracle for true edges that augment the model's knowledge. Then, we characterize the necessary and sufficient number of augmentation steps for the model to generate an accurate answer given partial prior knowledge. One key result shows a phase transition: if the prior knowledge graph over n vertices is disconnected into small components, then finding a path via augmentation is inefficient and requires $\Omega(\sqrt{n})$ queries. On the other hand, once the density of correct knowledge surpasses a threshold, forming a giant component, we can find paths with an expected constant number of queries.

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

1 INTRODUCTION

Generating accurate and helpful answers with Large Language Models (LLMs) often involves a combination of *thinking* about the user query and *retrieving* relevant information from external sources. For reasoning problems, the LLM can produce a chain-of-thought, analyzing intermediate sub-problems before arriving at a final solution Comanici et al. (2025); Mirtaheri et al. (2025); Yang et al. (2025); DeepSeek-AI et al. (2025). For information seeking queries, the LLM can retrieve from databases or knowledge graphs to expand its grasp of new facts (Gutiérrez et al., 2025; Lewis et al., 2020; Min et al., 2019; Zhou et al., 2025; Vu et al., 2023). A common thread for these scenarios is that the LLM needs to augment its pre-training knowledge with additional information (either self-generated or external) before being able to adequately answer a question (Joren et al., 2024; Su et al., 2024; Wei et al., 2024a). However, this interplay between parametric and user-provided or externally-retrieved contextual knowledge remains poorly understood.

We develop a graph-theoretic framework to study the ability of LLMs to solve multi-step problems. Using our abstract model, we can shed light on some fundamental questions. For instance, we explore how properties of the pre-training knowledge can facilitate or impede the ability to solve a multi-step problem. Then, as one of our results we show that when prior quality is above a certain threshold, a constant expected number of augmentation steps suffices, and below this threshold, any strategy requires superconstantly many queries using external information. Of course, representing an LLM's vast, nuanced parametric knowledge as an unweighted subgraph is a significant simplification. Nonetheless, our model provides a formal lens on a key principle: we support the conjecture that efficient retrieval-augmented generation (RAG) requires a richness of parametric knowledge (Guu et al., 2020; Pan et al., 2024; Wei et al., 2024b; Xie et al., 2024; Liu et al., 2025). In other

words, if LLMs are to succeed in RAG tasks, then they need to both process language *and* have a sufficient density of world knowledge from their pre-training data. Similarly, we provide further evidence that the best reasoning models require a deep knowledge of mathematical facts (Ma et al., 2025).

1.1 Graph-theoretic Framework for Solving Multi-step Problems

Given that an LLM with test-time augmentation is a very complex system to study, we focus on a few key components. At a high level, we consider a knowledge graph G^* where nodes are entities in the world and edges represent relationships or facts that involve two entities. For example, consider a simple fact composition scenario in knowledge graphs: if we know "A is connected to B" and "B is connected to C," we can deduce "A is connected to C." Such chaining of local facts into a global conclusion lies at the heart of deductive reasoning. One interpretation is that "A is connected to B" corresponds to "A is equivalent to B" and we want to deduce other equivalence relations. Another is that a connection corresponds to a co-occurrence in a sentence (Yang et al., 2024). We provide further examples in Figure 1.

A core component of our framework is that we aim to formalize the difference between the partial knowledge that the model knows from pre-training and the potential facts that it could know through reasoning-based thinking or external retrieval mechanisms. We abstract this as designating a subgraph G of G^*. In graph terms, we frame "good" prior knowledge as exhibiting well-connectedness, expansion, and edge reliability. We use G to capture the fact that a pretained LLM will have only partial information about an unknown ground-truth graph G^*. Furthermore, each edge of G may or may not correspond to a true edge in G^*, and the number of edges in G will be a small fraction of the total in G^*. We want to find trade-offs where a model knows G but also requires some access to G^* to correctly answer a question. The model will access G^* using "oracles" that provide information. From a theoretical point of view, our work complements much literature on sub-linear time graph algorithms (Goldreich and Ron, 2011). In particular, the algorithm starts with the prior knowledge of G, rather than with no information about G^*, leading to a new twist on classical problems.

To study test-time augmentation, we define two query models: (i) given a vertex v, the *retrieval oracle* returns a random neighbor of v in G^* (and our lower bound also holds for a stronger oracle that given a vertex v, returns *all* neighbors in G^* of the connected component in G that contains v) and (ii) given a pair u, v, the *verifier oracle* returns whether $\{u, v\}$ is an edge in G^* or not. These oracles capture different aspects of test-time augmentations based on a knowledge graph. In a RAG setting, a retrieval mechanism provides information about the query. Often this is done through a similarity search (e.g., BM25 or embeddings) based on the entities in the query. However, we cannot guarantee exactly what information the retriever returns. Hence, in the retrieval oracle, we get a random neighbor entity of a vertex. The connected component oracle is stronger, returning many neighbors. The verifier oracle offers the option to query a specific edge, rather than a random one.

With these query models, we then study when it is possible to efficiently solve certain tasks. We are particularly interested in algorithms that use a constant number of queries, not scaling with graph size. We begin our analysis with the s–t path problem. For an input pair of vertices (s, t) in a hidden ground truth graph G^*, the goal is to output a path between s and t in G^* if there is one. The path finding task is a proxy for a fundamental aspect of deductive reasoning: composing local facts (edges) to infer a global conclusion (path connectivity). For example, each edge may represent a discrete factual or logical step (e.g., 'Alice likes Pizza' or 'Pizza is an Italian food'). Finding a path from 'Alice' to 'Italian food' means composing individual facts to infer a new relationship. We also consider a robust version, where we output a subgraph that connects s and t even after removing certain edges.

1.2 Main Results

We prove several new bounds, for multiple query models and algorithmic tasks. Table 1 summarizes our main results. Our contributions make progress toward the larger goal of characterizing when an AI system, along with test-time mechanisms, can solve reasoning tasks from partial, and possibly noisy, prior knowledge. We first introduce a graph-theoretic property, *Retrieval Friendliness* (Definition 2.3), that captures when a partial knowledge graph and its ground truth counterpart admit efficient reasoning about every s–t connectivity prompt using a constant expected number of retrieval queries. We then define a general property, admissible, that implies Retrieval Friendliness. Building on these concepts, we show that variations of a bidirectional search algorithm can identify different types of subgraphs in G^* while using few queries.

To instantiate our theory in a more concrete setting, we show that certain random graphs are admissible with high probability. The Erdős–Rényi graph model serves as a good testbed for reasoning from incomplete priors due to its homogeneity, connectedness, strong expansion, and small diameter. These are all char-

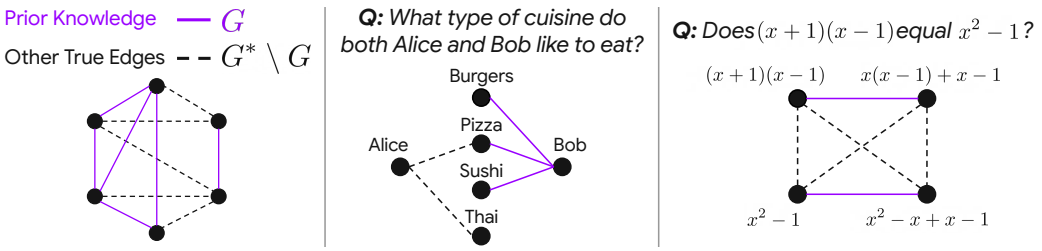

Figure 1: The left graph depicts that in our basic framework, we have a prior graph G, which is a subset of the true graph G^*. We illustrate two example knowledge graphs, where finding a path between nodes can answer the given question. Often the algorithm must query dotted edges from $G^* \setminus G$ that are outside of G. We also consider various ways to sample G and G^*, as well as cases where edges in G may be noisy and need verification.

Table 1: Algorithm Performance and Lower Bounds. Here a "✓" for *Grounded* means we provide a grounded algorithm (Definition 2.2), and an "×" means our lower bound holds for general algorithms. For brevity we only state results for the retrieval oracle (Definition 2.1) in this table, with results for other oracles in the paper.

Problem	**Prior G**	**True G^***	**Grounded**	**Results**	**Reference**
s–t path	Random dense subgraph	Erdős–Rényi	✓	$O(1)$	Thm. 4.3
	Random sparse subgraph	Erdős–Rényi	×	$\Omega(\sqrt{n})$	Thm. 3.4
s–t path	Double Star	+ Random Bridge	×	$\Theta(n)$	Prop. 3.1
s–t path	Empty	Complete graph	✓	$\Theta(\sqrt{n})$	Prop. 3.2
Int. K-connected	Random dense subgraph	Erdős–Rényi	✓	$O(\log K)$	Thm. 4.7

acteristics that intuitively facilitate finding paths. In other words, if a constant-query algorithm already fails here, then we would need stronger assumptions on the prior knowledge (e.g., more true information or a property where connectivity is tied to locality).

Our main technical results consist of new, nearly-tight lower bounds, which apply to multiple oracle models. First, we focus on the path problem with a random neighbor oracle, showing that can be hard to even find an s–t path without sufficient prior knowledge. We consider both worst-case and random graphs. Starting simple, we show that if G misses a single "bridge" edge, then we need $\Omega(n)$ queries. We next analyze Erdős–Rényi graphs, considering when the prior knowledge is *random* as opposed to structured, where we get an $\Omega(\sqrt{n})$ lower bound. Finally, we consider queries that return multiple neighbors, other tasks (e.g., multi-vertex connectivity), and robustness constraints. We analyze the reliability of prior knowledge, quantifying how many "false facts" (incorrect edges) we can tolerate while verifying paths. This highlights the importance of verifying the model's intermediate reasoning steps before relying on further retrieval.

One salient aspect of our new lower bounds is that algorithms need many queries *even when the expected path length is short*. It would be easy to prove that the number of queries grows with the path length. We go further, showing cases where the algorithm must explore many options, and the difficulty comes from *finding* a path. This is more interesting because LLMs often answer queries that only require a few hops.

1.3 Related Work

Retrieval-Augmented Generation (RAG) enhances LLMs by allowing them to access external knowledge bases. While RAG is effective in practice, the theoretical modeling of RAG is limited (Koga et al., 2025; Weller et al., 2025), and the interplay between a model's existing knowledge and the information it retrieves is not well understood. Classic RAG uses a single retrieve then generate step, which is often insufficient for multi-hop or evolving information needs. *Dynamic RAG* interleaves generation with retrieval, deciding both *when* and *what* to retrieve (Su et al., 2025; Asai et al., 2023; Gao et al., 2022). Corrective variants add evaluators to re-query when retrieval looks untrustworthy (Yan et al., 2024). Our model complements these systems by replacing heuristic trigger policies with *bounded query guarantees*: under structural conditions on the target knowledge graph, we characterize when constant expected retrieval suffice and when no bounded policy can succeed. Related instance-level criteria such as *sufficient context* (Joren et al., 2024) evaluate whether the retrieved snippets alone contain a solution; our concept of *retrieval friendliness* strengthens this by demanding

constant-query, zero-error guarantees while considering the effect of prior knowledge. Our lower bounds provide more evidence for the theoretical limitations of embedding-based retrieval (Weller et al., 2025).

Some of our results are inspired by a process-based supervision model (Uesato et al., 2022; Lightman et al., 2023; Setlur et al., 2024; Rohatgi et al., 2025; Balcan et al., 2025). Unlike outcome-only feedback, which evaluates a complete solution, process-based supervision provides granular feedback on each intermediate step. For graph problems, this corresponds to validating edges. We model the step-level validation capability with a verifier oracle, which is a membership query (Angluin, 1988) on the edge set of G^*. Graph-structured training and tool use have improved relational reasoning with LLMs (Mirtaheri et al., 2025; Yao et al., 2023; Shalev-Shwartz and Shashua, 2025; Huang et al., 2022, 2023; Wu et al., 2024; Kim et al., 2025), and while synthetic continued pre-training (Yang et al., 2024) can strengthen the connectedness of parametric knowledge, our results clarify when prior knowledge can improve test-time retrieval efficiency.

Our work connects to a long line of literature on sublinear graph algorithms in query models; see Beame et al. (2020); Feige (2004); Feige and Ferster (2021); Rácz and Schiffer (2019); Rashtchian et al. (2020, 2021); Chen et al. (2020) and references therein. We utilize recent lower bounds on shortest paths in expanders and random graphs (Alon et al., 2023), where that research provides a foundation for studying path computations in large networks (Basu et al., 2025). Finally, our results imply lower bounds on CoT length in a reasoning-inspired model (Mirtaheri et al., 2025).

2 PRELIMINARIES

For integer $n \in \mathbb{N}$, define $[n] = \{1, \ldots, n\}$. Let $G^* = (V, E^*)$ be the ground truth graph on $V = [n]$. We use $N_G(u)$ for the neighbors of u in a graph G, and let $\deg_G(u) = |N_G(u)|$. An *s–t path* is a simple path $P = (v_0 = s, v_1, \ldots, v_k = t)$ consisting of distinct pairs $(v_i, v_{i+1}) \in E^*$. The algorithm has access to a prior graph $G = (V, E)$. To isolate the challenge of knowledge incompleteness, we begin by assuming the prior is reliable; that is, G is a subgraph of G^* with $E \subseteq E^*$. This 'clean prior' setting allows us to first study knowledge structure, before we later discuss extensions to unreliable or 'hallucinated' facts (incorrect edges). The algorithm can use one or more oracles to G^*, which serve as test-time augmentation methods. In addition to paths, we will also be interested in "robust" subgraphs. For $K \geq 1$, we say that P is an *internally K-connected* subgraph between s and t if s and t remain connected in P whenever fewer than K edges are removed from $P \cap G$.

2.1 Retrieving Relevant Knowledge

We start with our first query model. It is the most restrictive, but it will suffice for our algorithms. RAG systems provide relevant retrieval results through a variety of search methods. To abstract away their inner workings, we consider a basic *Retrieval Oracle* that, given a vertex u, returns one of its true neighbors in G^* chosen uniformly at random.

Definition 2.1 (Retrieval Oracle)**.** *Let $G^* = (V, E^*)$ be a ground-truth graph. The retrieval oracle $\mathcal{O}_{G^*} : V \to E^* \cup \{\perp\}$ is specified by the family $\{\mu_u^{G^*}\}_{u \in V}$ where each $\mu_u^{G^*}$ is the uniform probability distribution over neighbors of u in G^*. On query $u \in V$ the oracle returns*

$$\mathcal{O}_{G^*}(u) = \begin{cases} (u, v) & \text{with probability } \mu_u^{G^*}(v),\ v \in N_{G^*}(u), \\ \perp & \text{if } N_{G^*}(u) = \emptyset. \end{cases}$$

While real-world RAG systems use deterministic similarity searches, the retrieved results are imperfect proxies for true relevance. Modeling the output as stochastic is a tractable way to capture the uncertainty an algorithm faces, preventing it from exploiting an all-knowing retriever. Later, we prove a lower bound for a stronger oracle that returns many neighbors based on the connected component containing the query.

We are interested in the interplay between the pre-trained knowledge and the feasibility of outputting a correct answer. From an efficiency point of view, we also want to determine when a model can use a small number of retrieval queries in expectation. This is in contrast to cases where the model must make a number of queries that scales with the graph size, which would be infeasible in practice for large graphs. Our framework captures the fact that the the model often combines the knowledge learned through context with its own prior knowledge for answer generation.

We introduce the notions of *Grounded Algorithms* and *Retrieval Friendliness*. First, the distinction between grounded and general algorithms is crucial. A grounded algorithm should not "hallucinate" or guess connections; its reasoning is based on verified facts (e.g., via G or an oracle). This captures how RAG systems ideally work, where grounding refers to providing citations (Gao et al., 2023; Song et al., 2025). Our lower bounds that hold for general algorithms (marked with an "×" in Table 1) are stronger and apply to hypothetical algorithms with the ability to guess edges.

Definition 2.2 (Grounded Algorithm)**.** *An algorithm*

$\mathcal{A}$ is grounded *if it outputs only edges it has explicitly observed from oracle outputs or prior knowledge G.*

Combined with grounded algorithms, our next definition strengthens the notion of "sufficient context" of Joren et al. (2024). Sufficient context means that the LLM has enough information to answer a query. We go further, saying that the algorithm can make a conclusion after a constant number of queries in expectation, and it only uses explicitly observed edges.

Definition 2.3 (q-Retrieval Friendliness)**.** *The pair (G, G^*) is* q-retrieval friendly *for a grounded algorithm $\mathcal{A}$ if given s, t, access to G, and $\mathcal{O}_{G^*}$, the algorithm outputs: (a)* NO *when t is not reachable from s in G^*, and (b) when t is reachable, a simple s-t path P such that all edges of P are valid (they are also in G^*) by making at most q queries in expectation.*

Intuitively, Retrieval Friendliness implies that even though G may contain incomplete information about the true graph G^*, it is still possible to efficiently recover valid reachability information for every pair in G^* using only a constant number of retrieval queries in expectation.

2.2 Random Graphs & Asymptotic Notation

Our general framework applies to any way of constructing a pretrained knowledge graph G and a target graph G^*. In some cases, we analyze a standard random graph model. Let $\mathcal{G}(n, p)$ denote the Erdős–Rényi random graph with n vertices, where edges appear independently with probability p (which may depend on n, so we may write $p(n)$ for clarity). This model produces a high-entropy graph, with no correlation between edges, and provides a challenging regime for finding paths. All our asymptotics are as $n \to \infty$. An event B occurs *with high probability* if $\lim_{n\to\infty} \mathbb{P}[B] \to 1$. Unless stated otherwise, success probabilities are over randomness in G, G^*, the oracle, and any internal randomness in the algorithm.

3 LOWER BOUNDS

We establish limits of test-time augmentation by proving query complexity lower bounds. We begin with an adversarial "bridge" graph to show that even a single missing piece of information can be expensive to find. We then show that without any prior knowledge, grounded algorithms are inefficient even on well-connected graphs. Finally, we prove our main lower bound for the more complex setting of random graphs.

Bridge Graph Lower Bound. While dense priors can permit efficient retrieval, we now demonstrate that sheer volume of pretrained knowledge by itself is insufficient. To illustrate this, we construct a worst-case instance where the prior G is a subgraph of the target G^* containing all but a single edge forming an information bottleneck (see Figure 2). Formally, define the **double star with random bridge** on n vertices as a graph $G^* = (V, E^*)$ constructed as follows: V is partitioned into S and T of size $n/2$. Then, E^* contains the edges of two stars centered at designated vertices $c_s \in S$ and $c_t \in T$, plus a single bridge edge (u, v) where $u \in S \setminus \{c_s\}$ and $v \in T \setminus \{c_t\}$ are chosen uniformly at random. The learner knows the prior graph $G := G^* \setminus \{(u, v)\}$. Let (s, t) be a pair of vertices chosen uniformly at random from V. This example demonstrates a lower bound in an extreme case.

Figure 2: Double star with random bridge. The prior knowledge graph G consists of two disjoint star graphs on the left and right. The ground-truth graph G^* adds a single, hidden "bridge" edge between random leaves on the left and right. Any algorithm must query $\Omega(n)$ leaves on average to find this bottleneck edge.

Proposition 3.1. *Finding an s-t path in the double star with random bridge on n vertices requires $\Omega(n)$ retrieval queries to have success probability $\geq 2/3$.*

Intuitively, the algorithm must query vertices until it finds the bridge. However, each query reveals no extra information about the potential bridge endpoints, since the retrieval oracle returns a random neighbor. We provide the full proof in Appendix A.1. Note that the proof holds with access to a stronger retrieval oracle that treats every edge in the prior graph as already known and never repeats an edge it has either previously returned or that was present in the prior.

Complete Graph, Grounded Lower Bound. We next establish that in a favorable setting, constant query *grounded* retrieval is impossible without prior knowledge. We consider when the ground truth graph G^* is a unweighted complete graph, and the pretrained graph G has no edges. This is a worst case scenario for the learner in terms of prior information, but best-case in terms of the target graph's connectivity.

Proposition 3.2. *Let G^* be complete and G empty. For any nodes s and t, any grounded algorithm must make $\Omega(\sqrt{n})$ retrieval oracle (Definition 2.1) queries to find an s-t path with success probability at least $1/2$.*

The proof is an application of the birthday paradox and is stated in Appendix A.2. We note that the proof holds with access to a stronger retrieval oracle that never repeats an edge it has previously returned.

This lower bound holds for the fundamental problem of finding any path between two nodes, not merely a shortest path. The core of our argument is based on collision probability (i.e., the birthday paradox) and would apply to $G^* \sim \mathcal{G}(n,p)$ for any value of p.

General Lower Bound & Stronger Oracle. We now extend this result to the more general setting of Erdős–Rényi random graphs, even when the learner has access to a powerful retrieval oracle that given a vertex v returns *all* neighbors in G^* of the connected component in pretrained graph G that contains v.

Definition 3.3 (Connected Component Incident Retrieval Oracle)**.** *Let G^* be the ground truth graph and G the pretrained subgraph. For any vertex $v \in V$, let $C_G(v) \subseteq V$ denote the set of vertices in the connected component of G that contains v. The* CCI retrieval oracle *is a map $\mathcal{O}^{\text{cci}}_{G^*,G} : V \to 2^{E^*} \cup \{\perp\}$ defined by*

$$\mathcal{O}^{\text{cci}}_{G^*,G}(v) = \begin{cases} S_v, & S_v \neq \emptyset, \\ \perp, & \textit{otherwise,} \end{cases}$$

where $S_v := \{(u,w) \in E^* : u \in C_G(v),\ w \notin C_G(v)\}$.

Consider $G^* \sim \mathcal{G}(n,p)$ with $p \geq \frac{1.5 \log n}{n}$, which ensures G^* is connected with high probability. We are interested in the sparse but supercritical regime when p is above the connectivity threshold yet far from dense. For instance, when $p = 1$ then G^* is the complete graph and a single CCI query (Definition 3.3) trivializes path finding. Assume the learner also has access to pretrained knowledge G obtained by independently retaining each edge of G^* with probability η with $p \cdot \eta < \frac{1}{n}$, placing G in a regime that carries negligible *global* connectivity signal.

Theorem 3.4. *In the setup above, any algorithm for finding a path between given vertices s,t either makes $\Omega(\frac{1}{p \cdot \log^2 n \cdot \sqrt{n}})$ connected component incident retrieval (Definition 3.3) queries or finds an s–t path with probability at most $p \log^2 n + o(1)$.*

The proof is provided in Appendix A.3 and relies on a reduction that contracts $O(\log n)$-sized components into super-nodes, turning the problem into path finding in a meta-graph. It then uses a trace based analysis, that is, by fixing the algorithm's randomness, each execution is represented as a trace. Observe the number of edges discovered after $\Omega(\frac{1}{p \cdot \log^2 n \cdot \sqrt{n}})$ connected component incidence queries is $O(\sqrt{n})$ each call can reveal fewer than $p \cdot \log^2 n \cdot (n-1)$ incident edges. Having shown the $\Omega(\sqrt{n})$ lower bound for path recovery, we now present a nearly matching upper bound for Erdős-Rényi random graphs, which adapts a result of Alon et al. (2023) for regular expander graphs to random graphs that are only approximately regular.

Theorem 3.5. *Consider an Erdős-Rényi random graph $G^* \sim G(n,p)$, where $p(1-p) \geq C \cdot \frac{\log^4(n)}{n}$ and C is a sufficiently large constant. There exists an algorithm that, with high probability over the randomness of G^*, for every node s and every $\delta \in (0,1)$, finds an s–t path while visiting $O((\frac{n}{\delta})^{\frac{1}{2}+o(1)})$ vertices for all but a δ-fraction of targets t.*

The proof in Appendix A.4 uses tools from random matrix theory and is implementable with access to a retrieval oracle that never repeats an edge it has previously returned. Next, we explore the properties of the ground truth and prior knowledge graph that make retrieval friendliness (Definition 2.3) possible.

4 UPPER BOUNDS: EFFICIENT TEST-TIME AUGMENTATION ALGORITHMS

4.1 Reliable and Sufficient Prior for Efficient Retrieval

We start by stating a condition that makes retrieval friendliness possible by the strategy of decomposing the task of finding paths into *efficient sub-tasks*.

Intuitively, for path-finding, the model's prior knowledge G must connect disparate regions of the complete knowledge graph G^*. Even if we do not know how to globally connect our start s and end t nodes, we can find a subgraph that connects to both. Our formal condition, which we call admissibility, captures this idea: a well-connected component in the prior knowledge is "visible" from every node in the graph, covering a constant fraction of its local connections. We formalize this below and then show how a simple bidirectional search strategy can exploit it for efficient retrieval.

Definition 4.1 (γ-Admissible Pair)**.** *Let G^* be the ground-truth graph, G the pretrained subgraph, $\mathcal{O}_{G^*}$ the retrieval oracle, and $\gamma \in (0,1]$. For every vertex u, let $\mu_u^{G^*}$ be the uniform probability distribution over $N_{G^*}(u)$ given by G^*. We say that (G^*, G) is* γ-admissible *if there exists a connected component C of G such that, for every vertex u with $N_{G^*}(u) \neq \emptyset$,*

$$\mu_u^{G^*}\big(N_{G^*}(u) \cap V(C)\big) \geq \gamma.$$

for some constant γ.

Claim 4.2. *Any γ-admissible pair is $2/\gamma$-retrieval friendly for bidirectional-retrieval augmentation generation algorithm (Algorithm 1).*

The bidirectional strategy is given in Algorithm 1. After each pair of retrievals we *augment* the pretrained graph with the returned edges and attempt to generate

Algorithm 1 Bidirectional-Retrieval Augmentation Generation (BiRAG)

Require: γ-admissible pair (G^*, G), endpoints s, t, retrieval oracle $\mathcal{O}_{G^*}$.

$E_s \leftarrow \emptyset,\ E_t \leftarrow \emptyset$
repeat
  $e_s \leftarrow \mathcal{O}_{G^*}(s)$
  $e_t \leftarrow \mathcal{O}_{G^*}(t)$
  if $e_s = \perp$ **or** $e_t = \perp$ **then**
    return NO
  $E_s \leftarrow E_s \cup \{e_s\}$
  $E_t \leftarrow E_t \cup \{e_t\}$
  Augment: $\tilde{G} \leftarrow \textsc{Augment}(G, E_s, E_t)$
  Generate: $\Pi \leftarrow s$–t path from $\tilde{G}$
until $\Pi \neq \perp$
return Π

an s-t path in the augmented graph. Concretely, the Augment step forms $\tilde{G} = (V,\ E \cup E_s \cup E_t)$ by adding the edges in E_s and E_t to G. If the Generate step fails to find an s-t path in $\tilde{G}$, it returns $\Pi = \perp$ and the loop continues. Under the γ-admissibility assumption the loop in Algorithm 1 runs $1/\gamma$ times in expectation, issuing two retrieval calls per iteration. We provide the run time analysis and the full proof in Appendix A.5.

As an example, we now show that our condition for efficient retrieval holds with high probability in an Erdős-Rényi graph model G^*, provided it contains a sufficiently dense subgraph G.

Theorem 4.3. *Consider an Erdős–Rényi random graph $G^* \sim \mathcal{G}(n,p)$ with $p > C_0 \frac{\log n}{n}$ for a sufficiently large constant C_0. Let pretrained graph G be a subgraph formed by retaining each edge of G^* independently with probability $\eta \in (\frac{1}{\log n}, 1]$. Then, with high probability[1] over the randomness of G^* and G, the pair (G, G^*) is $\gamma/3$-admissible with $\gamma \in (0,1]$ being the unique solution to $\gamma = 1 - e^{-np\eta\gamma}$.*

As long as the edge probabilities p (for the true graph) and η (for the prior) are above the connectivity threshold, a giant component in the prior graph exists with high probability. Furthermore, the random nature of the remaining edges in $G^* \setminus G$ ensures that this component is distributed, making it 'visible' from all other nodes, thus satisfying our admissibility condition. Formally, using the equivalence that G^* can be obtained by adding independent edges to G, we can condition on the giant component of G to show every vertex connects into it with ratio at least $\gamma/3$. Note that since $np\eta > C_0$ in Theorem 4.3, γ is always at least some absolute positive constant. Full details are in Appendix A.6. Motivated by priors with

[1] with probability $1 - o(n^{-2})$

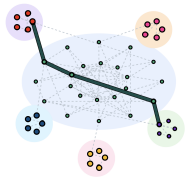

Figure 3: Illustrating γ-admissible & Algorithm 1.

community structure, we consider a partitioned revelation model that preserves intra-group edges while suppressing cross-group links.

Remark 4.4. *In Theorem 4.3, suppose the pretrained graph is obtained through a much harsher non-uniform revelation process. Let $V = \biguplus_{l=1}^{L} V_l$ be a fixed partitioning of the vertices for some constant L, and the pretrained graph G be a subgraph formed by retaining each* intra-group *edge of G^* independently with probability $\eta \in (\frac{1}{\log n} \cdot \frac{n}{\max_l |V_l|}, 1]$, and discarding all inter-group edges (i.e., $p_{i,j} = \eta \cdot \mathbf{1}\{\exists l : i,j \in V_l\}$). Then, with high probability over the randomness of G^* and G, the pair (G, G^*) is admissible.*

Beyond Paths: Finding Steiner Trees. One natural extension to finding a path between vertices is considering a set of M input vertices $(s_1, \ldots, s_M)$ and asking whether the learner can recover a Steiner tree connecting them all. This is a proxy for finding a set of facts that connect many entities, e.g., when an LLM must construct a coherent story involving multiple people. We can extend our bidirectional search to an M-directional algorithm, which works for admissible graphs. From each s_i, we connect it to the giant component and then stitch the paths together. That is, in γ-admissible pairs, we can find a Steiner tree containing the neighborhoods of $(s_1, \ldots, s_M)$ inside the giant component of G, and connect the s_i to it with M extra edges and by making M/γ queries in expectations.

4.2 Sufficient but Unreliable Prior

To establish a robust conclusion between two entities, one should not rely on a single line of reasoning. We therefore seek *many* candidate routes whose evidence is separated across the pretrained graph. Concretely, think of an K edge-coloring (or labeling) of E: each color (labeling) is a self-contained reasoning route. Concretely, we want K edge-disjoint segments in the pretrained graph. The benefit is robustness: even if we distrust up to $K-1$ edge types, a single remaining route of a trusted type still suffices to certify correctness. The next definition formalizes an structural property that satisfies this.

Definition 4.5 ((K,γ)-Robust-Admissible)**.** *Let G^**

*be the ground truth graph, G the pretrained subgraph, $K \in \mathbb{N}$ and $\gamma \in (0,1]$. For each vertex u, let $\mu_u^{G^*}$ be the uniform probability distribution over N_{G^*} given by G^*. We say (G^*, G) is (K,γ)-*robust-admissible *if there exist an edge-partition $E = \biguplus_{k=1}^{K} E_k$, with $G_k := (V, E_k)$, and for each $i \in [k]$, a connected component C_k of G_k such that $\forall u \in V$ with $N_{G^*}(u) \neq \emptyset$ and all $k \in [K]$*

$$\mu_u^{G^*}\big(N_{G^*}(u) \cap V(C_k)\big) \;\geq\; \gamma.$$

We now present an example of such a pair.

Corollary 4.6. *Consider $G^* \sim \mathcal{G}(n,p)$ with $p > C_1 \frac{\log n}{n}$, where $C_1 = KC_0$, and C_0 is the same constant as Theorem 4.3. Let the pretrained graph G be obtained by retaining each edge of G^* independently with probability $\eta \in (\frac{1}{\log n}, 1]$. Then, with high probability over the randomness of G^* and G, (G^*, G) is $(K, \gamma/3)$-robust-admissible with $\gamma \in (0,1]$ the unique constant solution to $\gamma = 1 - e^{-np\eta\gamma}$.*

Proof. Independently color each retained edge of G uniformly with one of K colors, yielding an edge-partition $E = \biguplus_{k=1}^{K} E_k$ and subgraphs $G_k := (V, E_k)$. For each k, an edge of G^* lands in E_k with probability η/K, so marginally $G_k \sim \mathcal{G}(n, \frac{\eta p}{K})$. As a direct corollary of Theorem 4.3, for each $k \in [K]$ with high probability over the randomness of G^*, G and G_k, the pair (G_k, G^*) is $\gamma/3$-admissible. The union bound gives that this holds for all $k \in [K]$ simultaneously with high probability. Therefore, it follows with high probability over the randomness of G^* and G, the pair is $(K, \gamma/3)$-robust-admissible. □

Theorem 4.7. *There exists an algorithm that for any (K,γ)-robust-admissible pair (G^*, G), for any two vertices $s, t \in V$ makes $O(\frac{\log K}{\gamma})$ calls to the retrieval oracle (Definition 2.1) in expectation and constructs internally K-connected subgraph between s and t.*

While the statement above is similar to a coupon collector argument we note that here every partition is hit with probability at least γ; hence, $O(\frac{\log K}{\gamma})$ queries suffice. The proof is provided in Appendix A.7.

Remark 4.8. *There exists an algorithm that for any (K,γ)-robust-admissible pair (G^*, G), for any two vertices $s, t \in V$ makes $O(1/\gamma)$ calls to the retrieval oracle (Definition 2.1) in expectation and constructs internally $K/2$-connected subgraph between s and t.*

From Robustness to Verification The use of large pretrained models as priors is promising, but their inherent unreliability creates a fundamental trade-off. In the discussed robustness guarantee above, we assumed E can be partitioned into k types and each type sees a constant fraction of true neighbors. Thus, trusting any one edge type yields a correct s,t path with efficient retrieval. When this assumption may fail we must switch to *verification*. That is, use the prior G to generate candidate s,t paths and let a verifier certify the first valid one. Concretely, a verifier for a graph $G^* = (V, E^*)$ is an oracle that given two vertices (u,v), returns YES if $(u,v) \in E^*$ and NO otherwise. This approach does not need a strong structural assumption and is appropriate when a verifier is available. To get a sense of how efficient the *generate then verify* approach can be, imagine that the prior graph is reasonably *grounded*: whenever it suggests a short path of length c, each edge in that path has at least probability r of being correct in the true graph. In other words, a whole c-hop path from the prior survives in the ground truth graph with probability at least r^c. Under this assumption, it is straightforward to see that we can find and certify a path using only about $O(c/r^c)$ verifier queries in expectation.

5 CONCLUSION

We provide the first theoretical framework of test-time augmentation with multiple types of query models. We analyze the s–t path finding problem serving as a basic testbed for reasoning as well as the internally K-connected problem, a generalization of it. We study the interplay between the test time query complexity of solving these problems and prior knowledge in various types of graphs, providing upper and lower bounds. Depending on properties of prior knowledge, our bounds delineate "easy" regimes where we can find a correct s–t path with few retrieval query calls as well as "hard" regimes where any algorithm requires $\Omega(\sqrt{n})$ or even $\Omega(n)$ queries. Overall, our results provide evidence that density and the structure of the pretrained knowledge is critical for efficient RAG or tool use. We list several problems in Appendix B.

Limitations. One shortcoming is that we only study a few oracle models, and there may be different trade-offs for other test-time augmentation methods. For example, it would be ideal to more closely align with similarity-based retrieval methods in real RAG systems. Another limitation is that our asymptotic analysis may not be precise enough to explain the nuanced trade-offs in real-world systems. As another aspect, our upper-bound results address the existence of certain subgraphs rather than the optimal versions (e.g., shortest path or minimum spanning tree), which we view as an important direction for future work. Additionally, while we studied multiple, distinct graph families, we did not fully characterize all ways to generate the prior knowledge graph G and the target graph G^*. Finally, retrieval friendliness is a broad concept, ex-

tending well beyond the path-finding problem. Characterizing the algorithms and conditions that enable it across problems remains a compelling direction for theory and practice.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under grants CCF-2212968, DMS-2502259, and ECCS-2216899, by the Simons Foundation under the Simons Collaboration on the Theory of Algorithmic Fairness, and by the Office of Naval Research under MURI Grant N000142412742 and Grant N000142412700. We would like to thank Parsa Mirtaheri, Enric Boix-Adserà, Andrew Tomkins, and Rocco Servedio for helpful discussions.

References

Alon, N., Grønlund, A., Jørgensen, S. F., and Larsen, K. G. (2023). Sublinear time shortest path in expander graphs. *arXiv preprint arXiv:2307.06113*.

Angluin, D. (1988). Queries and concept learning. *Mach. Learn.*, 2(4):319–342.

Asai, A., Wu, Z., Wang, Y., Sil, A., and Hajishirzi, H. (2023). Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Balcan, M.-F., Blum, A., Li, Z., and Sharma, D. (2025). On learning verifiers and implications to chain-of-thought reasoning. In *NeurIPS*.

Basu, S., Kōshima, N., Eden, T., Ben-Eliezer, O., and Seshadhri, C. (2025). A sublinear algorithm for approximate shortest paths in large networks. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, WSDM '25, page 20–29, New York, NY, USA. Association for Computing Machinery.

Beame, P., Har-Peled, S., Ramamoorthy, S. N., Rashtchian, C., and Sinha, M. (2020). Edge estimation with independent set oracles. *ACM Transactions on Algorithms (TALG)*, 16(4):1–27.

Chen, X., Levi, A., and Waingarten, E. (2020). Nearly optimal edge estimation with independent set queries. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2916–2935. SIAM.

Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., et al. (2025). Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

Feige, U. (2004). On sums of independent random variables with unbounded variance, and estimating the average degree in a graph. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 594–603.

Feige, U. and Ferster, T. (2021). A tight bound for the clique query problem in two rounds. *arXiv preprint arXiv:2112.06072*.

Frieze, A. and Karoński, M. (2015). *Introduction to Random Graphs*. Cambridge University Press.

Gao, L., Dai, Z., Pasupat, P., Chen, A., Chaganty, A. T., Fan, Y., Zhao, V. Y., Lao, N., Lee, H., Juan, D.-C., et al. (2022). Rarr: Researching and revising what language models say, using language models. *arXiv preprint arXiv:2210.08726*.

Gao, T., Yen, H., Yu, J., and Chen, D. (2023). Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488.

Goldreich, O. and Ron, D. (2011). Algorithmic aspects of property testing in the dense graphs model. *SIAM Journal on Computing*, 40(2):376–445.

Gutiérrez, B. J., Shu, Y., Qi, W., Zhou, S., and Su, Y. (2025). From rag to memory: Non-parametric continual learning for large language models. *arXiv preprint arXiv:2502.14802*.

Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M. (2020). Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Huang, Q., Ren, H., Chen, P., Kržmanc, G., Zeng, D., Liang, P., and Leskovec, J. (2023). Prodigy: Enabling in-context learning over graphs.

Huang, Q., Ren, H., and Leskovec, J. (2022). Few-shot relational reasoning via connection subgraph pretraining.

Joren, H., Zhang, J., Ferng, C.-S., Juan, D.-C., Taly, A., and Rashtchian, C. (2024). Sufficient context: A new lens on retrieval augmented generation systems. *arXiv preprint arXiv:2411.06037*.

Kim, J., Wu, D., Lee, J., and Suzuki, T. (2025). Metastable dynamics of chain-of-thought reasoning: Provable benefits of search, rl and distillation.

Koga, T., Wu, R., and Chaudhuri, K. (2025). Privacy-preserving retrieval-augmented generation with differential privacy.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. (2023). Let's verify step by step.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W. S., and Lin, M. (2025). Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Ma, Q., Wu, Y., Zheng, X., and Ji, R. (2025). Benchmarking abstract and reasoning abilities through a theoretical perspective. *arXiv preprint arXiv:2505.23833*.

Min, S., Chen, D., Zettlemoyer, L., and Hajishirzi, H. (2019). Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Mirtaheri, P., Edelman, E., Jelassi, S., Malach, E., and Boix-Adsera, E. (2025). Let me think! a long chain-of-thought can be worth exponentially many short ones. *arXiv preprint arXiv:2505.21825*.

Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., and Wu, X. (2024). Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Rácz, M. Z. and Schiffer, B. (2019). Finding a planted clique by adaptive probing. *arXiv preprint arXiv:1903.12050*.

Rashtchian, C., Woodruff, D., Ye, P., and Zhu, H. (2021). Average-case communication complexity of statistical problems. In *Conference on Learning Theory*, pages 3859–3886. PMLR.

Rashtchian, C., Woodruff, D. P., and Zhu, H. (2020). Vector-matrix-vector queries for solving linear algebra, statistics, and graph problems. *arXiv preprint arXiv:2006.14015*.

Rohatgi, D., Shetty, A., Saless, D., Li, Y., Moitra, A., Risteski, A., and Foster, D. J. (2025). Taming imperfect process verifiers: A sampling perspective on backtracking.

Setlur, A., Nagpal, C., Fisch, A., Geng, X., Eisenstein, J., Agarwal, R., Agarwal, A., Berant, J., and Kumar, A. (2024). Rewarding progress: Scaling automated process verifiers for llm reasoning.

Shalev-Shwartz, S. and Shashua, A. (2025). From reasoning to super-intelligence: A search-theoretic perspective.

Song, M., Sim, S. H., Bhardwaj, R., Chieu, H. L., Majumder, N., and Poria, S. (2025). Measuring and enhancing trustworthiness of llms in rag through grounded attributions and learning to refuse. In *The Thirteenth International Conference on Learning Representations*.

Su, H., Yen, H., Xia, M., Shi, W., Muennighoff, N., Wang, H.-y., Liu, H., Shi, Q., Siegel, Z. S., Tang, M., et al. (2024). Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. *arXiv preprint arXiv:2407.12883*.

Su, W., Ai, Q., Zhan, J., Dong, Q., and Liu, Y. (2025). Dynamic and parametric retrieval-augmented generation.

Uesato, J., Kushman, N., Kumar, R., Song, F., Siegel, N., Wang, L., Creswell, A., Irving, G., and Higgins, I. (2022). Solving math word problems with process- and outcome-based feedback.

Vu, T., Iyyer, M., Wang, X., Constant, N., Wei, J., Wei, J., Tar, C., Sung, Y.-H., Zhou, D., Le, Q., et al. (2023). Freshllms: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214*.

Vu, V. H. (2007). Spectral norm of random matrices. *Combinatorica*, 27(6):721–736.

Wei, J., Karina, N., Chung, H. W., Jiao, Y. J., Papay, S., Glaese, A., Schulman, J., and Fedus, W. (2024a). Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*.

Wei, J., Yang, C., Song, X., Lu, Y., Hu, N., Huang, J., Tran, D., Peng, D., Liu, R., Huang, D., et al. (2024b). Long-form factuality in large language models. *Advances in Neural Information Processing Systems*, 37:80756–80827.

Weller, O., Boratko, M., Naim, I., and Lee, J. (2025). On the theoretical limitations of embedding-based retrieval. *arXiv preprint arXiv:2508.21038*.

Wu, S., Zhao, S., Huang, Q., Huang, K., Yasunaga, M., Cao, K., Ioannidis, V. N., Subbian, K., Leskovec, J., and Zou, J. (2024). Avatar: Optimizing llm agents for tool usage via contrastive reasoning.

Xie, J., Zhang, K., Chen, J., Lou, R., and Su, Y. (2024). Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *International Conference on Learning Representations (ICLR)*.

Yan, S.-Q., Gu, J.-C., Zhu, Y., and Ling, Z.-H. (2024). Corrective retrieval augmented generation.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. (2025). Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, Z., Band, N., Li, S., Candes, E., and Hashimoto, T. (2024). Synthetic continued pretraining. *arXiv preprint arXiv:2409.07431*.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models.

Zhou, Y., Su, Y., Sun, Y., Wang, S., Wang, T., He, R., Zhang, Y., Liang, S., Liu, X., Ma, Y., et al. (2025). In-depth analysis of graph-based rag in a unified framework. *arXiv preprint arXiv:2503.04338*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. [Yes/No/Not Applicable]

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A Appendix

Definition A.1 (Prior-Aware Retrieval Oracle with Memory)**.** *Let $G^* = (V, E^*)$ be the ground-truth graph and $G = (V, E)$ a pretrained subgraph. The* retrieval oracle *is specified by the family $\{\mu_u^{G^*}\}_{u \in V}$ where each $\mu_u^{G^*}$ is the uniform probability distribution over neighbors of u in G^*.*

The oracle never repeats an edge it has already revealed or one present in the prior graph. It maintains a set of seen edges $E_{\text{seen}} \subseteq E^*$, initialized as $E_{\text{seen}} := E$. For each vertex $u \in V$, let

$$N_{G^*}^{\text{unseen}}(u) = \{v \in N_{G^*}(u) : (u, v) \notin E_{\text{seen}}\},$$

and let $\pi_u^{G^*,\text{unseen}}$ denote the restriction of $\mu_u^{G^*}$ to this set (renormalized). On query $u \in V$, the oracle returns

$$\mathcal{O}_{G^*}(u) = \begin{cases} (u, v) & \text{with probability } \pi_u^{G^*,\text{unseen}}(v), \\ \perp & \text{if } N_{G^*}^{\text{unseen}}(u) = \emptyset. \end{cases}$$

After returning (u, v), the oracle updates E_{seen} and its dependent sets accordingly.

A.1 Proof of Proposition 3.1

Proof. By Yao's Minimax Principle, it suffices to prove a lower bound for the best deterministic algorithm against an input distribution that is hard on average. In the distribution from the proposition the bridge endpoints (u, v) are chosen uniformly and independently. A pair of vertices (s, t) chosen uniformly at random falls into one of following cases. Either both s and t lie in the same partition which occurs with probability approaching $1/2$ and a path already exists within the prior and no retrieval queries is needed. Or, s and t lie in different partitions, and any path from s to t must traverse the hidden bridge (u, v). Thus, to achieve an overall success probability of at least $2/3$ on a uniformly random pair, an algorithm must have at least a constant success probability on the inter-star instances. We establish a lower bound for this sub-problem which reduces to identifying the bridge. For the sake of proving a lower bound suppose access to a the prior aware retrieval oracle with memory (Definition A.1).

We first demonstrate that an optimal algorithm will focus on identifying one of the bridge's endpoints. That is, the learner queries the leaves of one star sequentially in order to find the target hidden leaf $u \in S \setminus \{c_s\}$ among $\frac{n}{2} - 1$ such candidates. To see why this strategy is optimal, suppose the algorithm instead made q_1 queries to the leaves of the star centered at c_s and $q - q_1$ queries to the leaves of the star centered at c_t. The probability of missing the target leaf on the star centered at c_s is $1 - \frac{q_1}{n/2-1}$ and the probability of missing the special leaf on the star centered at c_t is $1 - \frac{q-q_1}{n/2-1}$. Hence the success probability is

$$1 - (1 - \frac{q - q_1}{n/2 - 1}) \cdot (1 - \frac{q_1}{n/2 - 1}) = \frac{q}{n/2 - 1} - \frac{q_1(q - q_1)}{(n/2 - 1)^2}.$$

Now, consider a learner that focuses on the leaves of one star instead and uses q retrieval queries. Since the retriever is prior aware with memory, this process is equivalent to choosing q leaves all at once from a set of $n/2 - 1$ leaves to find the target leaf.

$$\Pr[\text{bridge discovered within } q \text{ queries}] = \frac{q}{n/2 - 1},$$

Demonstrating the latter strategy is optimal since $\frac{q}{n/2-1} - \frac{q_1(q-q_1)}{(n/2-1)^2}$ is strictly smaller than $\frac{q}{n/2-1}$.

Therefore, for an algorithm to succeed on the inter star instances with a constant probability, it must make $q = \Omega(n)$ queries (it is easy to see that an adaptive learner has the same query complexity as well). The overall expected query complexity is the average over both cases; thus, any algorithm that succeeds with an overall probability of at least $2/3$ must perform $\Omega(n)$ queries. □

A.2 Proof of Proposition 3.2

Proof. Let k_s, k_t be the number of queries at s and t, and $Q = q_s + q_t$. The probability that the direct edge (s, t) is revealed is at most $\frac{q_s}{n-1} + \frac{q_t}{n-1} \leq \frac{Q}{n-1}$. To succeed with probability at least $1/2$ we need $Q = \Omega(n)$; thus, the learner needs to target finding a path of length at least two.

Next, the proof establishes the lower bound by first considering the specific problem of finding a length-two path, whose structure motivates a more general argument. A query to a vertex $v \notin \{s,t\}$ finds a path only if the returned neighbor is s (and v is known to be a neighbor of t) or t (and v is known to be a neighbor of s); therefore, without loss of generality assume that $\mathcal{A}$ queries s and t, and $Q = q_s + q_t$. After q_s and q_t queries, the discovered neighbor sets $S_Q, T_Q \subseteq V \setminus \{s,t\}$ satisfy $|S_Q| \le q_s$, $|T_Q| \le q_t$. A length-2 path is found iff $S \cap T \neq \emptyset$, that is we need $\Pr(\text{success}) = \Pr(|S_Q \cap T_Q| \ge 1) \le \mathbb{E}[|S \cap T|] \le \frac{(n-2)q_s q_t}{(n-1)^2}$ to be at least $1/2$. For fixed total $Q = q_s + q_t$, the product $q_s q_t$ is maximized at $q_s = q_t = Q/2$; thus, for a success probability greater than half we need $Q = \Omega(\sqrt{n})$. We note that this search for a common element between two incrementally revealed sets has the structure of the birthday paradox; we now generalize this to paths of longer lengths. The argument rests on reducing the path-finding problem to that of inducing a collision, defined as any event where a query returns a vertex that has already been discovered. Label $V \setminus \{s,t\} = \{1,\dots,n-2\}$, and in each round i the algorithm chooses a vertex $u_i \in V$ and the oracle returns a random neighbor $v_i \in N_{G^*}(u_i)$. At each round, place a ball in the bin of the queried vertex and in the bin of the returned neighbor. Let a *connection collision* be the event that the *returned neighbor* v_i falls into a bin that already contains a ball from some earlier round. Note that finding a connection collision is a lower bound on finding a connected path, and by birthday paradox, any algorithm needs $\Omega(\sqrt{n})$ rounds in expectation before the first connection collision, and hence $\Omega(\sqrt{n})$ retrieval queries in expectation before it can find an s-t path. Therefore, the minimum expected number of queries required by is $\Omega(\sqrt{n})$, completing the proof. □

Lemma A.2 (K-Birthday Paradox)**.** *Throw m balls independently and uniformly into n bins, and let C be the number of bins with at least 2 balls. Then, $m = o(\sqrt{Kn})$ implies $C < K$ with high probability.*

Proof. Let X_i be the number of balls in bin i. The event of interest is $\sum_{i=1}^n \mathbf{1}\{X_i \ge 2\} \ge K$ which is monotone in m. By standard Poissonization,

$$\Pr\Big(\sum_{i=1}^n \mathbf{1}\{X_i \ge 2\} \ge K\Big) \le 2\Pr\Big(\sum_{i=1}^n \mathbf{1}\{Y_i \ge 2\} \ge K\Big),$$

where $Y_1,\dots,Y_n$ are i.i.d. Poi(λ) with $\lambda = m/n$. Let $Z_i = \mathbf{1}\{Y_i \ge 2\}$ and $\mu = \mathbb{E}[Z_i] = \Pr(Y_i \ge 2)$. Observe that

$$\mu = 1 - e^{-\lambda}(1+\lambda) \le 1 - (1-\lambda)(1+\lambda) = \lambda^2 = (\frac{m}{n})^2$$

and we have $\mathbb{E}[\sum_{i=1}^n Z_i] = n\mu \le \frac{m^2}{n}$. Therefore, for any $\epsilon > 0$, if $m \le \sqrt{\frac{Kn}{1+\epsilon}}$ then $K \ge (1+\epsilon)\,\mathbb{E}[\sum_{i=1}^n Z_i]$. Thus, by Chernoff bound

$$\Pr\Big(\sum_{i=1}^n Z_i \ge K\Big) \le \Pr\Big(\sum_{i=1}^n Z_i \ge (1+\gamma)n\mu\Big) \le \exp(-n\mu\epsilon^2/3),$$

completing the proof. □

Remark A.3. *In the setting of Proposition 3.2, any algorithm that finds K edge disjoint s–t paths with constant probability requires $\Omega(\sqrt{Kn})$ retrieval queries. First, note that there can be* at most one *length one path; therefore, for $K \ge 2$ at least $K-1$ of the paths must have length more than one. Moreover, finding the direct edge path requires $\Omega(n)$ queries as stated before. For length two paths, let q_s, q_t be the numbers of queries at s and t, and set $Q := q_s + q_t$. As in the proof above,*

$$\mathbb{E}\big[\,|S_Q \cap T_Q|\,\big] = \sum_v \Pr(v \in S_Q \cap T_Q) \le \frac{(n-2)\,q_s q_t}{(n-1)^2} \le \frac{(n-2)\,Q^2}{4(n-1)^2}.$$

By Markov's inequality,

$$\Pr\big(|S_Q \cap T_Q| \ge K\big) \;\le\; \frac{\mathbb{E}[|S_Q \cap T_Q|]}{K} \;\le\; \frac{(n-2)Q^2}{4K(n-1)^2}.$$

Thus achieving constant success probability for K edge disjoint s-t paths requires $Q = \Omega(\sqrt{Kn})$.

For longer paths, we can again use the balls and bins argument and reduce it to K-birthday paradox (see Lemma A.2). Each edge disjoint path needs at least one distinct intermediate vertex, so we need at least K connection collisions, that is, at least K bins with at least two balls which requires $Q = \Omega(\sqrt{Kn})$ as shown in lemma A.2

A.3 Proof of Theorem 3.4

Proof. Our proof adapts Theorem 4 of Alon et al. (2023). Given each edge from G^* is retained in the pretrained graph G with probability η, the pretrained graph G itself is an Erdős–Rényi random graph $G \sim \mathcal{G}(n, p_{\text{partial}})$ with $p_{\text{partial}} = p \cdot \eta < \frac{1}{n}$. It is known that in this subcritical regime an Erdős–Rényi random graph $G \sim G(n,p)$ with $p < 1/n$, the largest connected component has size $O(\log n)$ with high probability.

We note that the generation process is equivalent to first sampling $G \sim \mathcal{G}(n, p\eta)$ and then, to form G^*, adding each edge not present in G independently with probability $q = \frac{p-p\eta}{1-p\eta}$. This ensures G^* is a valid $\mathcal{G}(n,p)$ graph. This allows us to first condition on the realization of G (and thus the partition of the vertices and connected components) and then analyze the properties of G^* and the meta graph we introduce in what follows.

For the lower bound, take the extremal case where all components have size $C \log n$ for a fixed absolute constant $C \geq 1$. Contracting components yields $n' = \frac{n}{C \log n}$ *super-nodes*. Let G' be the meta-graph on the super nodes. Two super nodes are adjacent in G' if there exists at least one cross edge in G^* between their underlying components. By a union bound over at most $(C \log n)^2$ potential cross-edges between two components,

$$\Pr[\text{edge in } G'] \leq (C \log n)^2 p.$$

With $p = \frac{\log n}{n}$ and $n' = \frac{n}{C \log n}$, this simplifies to

$$\Pr[\text{edge in } G'] \leq \frac{C^2 \log^3 n}{n} = \frac{C \log^2 n}{n'} \leq \frac{4C \log^2 n'}{n'}.$$

Therefore G' is no denser than $\mathcal{G}\left(n', \frac{4C \log^2 n'}{n'}\right)$ for large n'.

Let $\mathcal{A}^*$ be a (possibly randomized) algorithm for computing an s-t path in G'. Without loss of generality, we label vertices so that $s = 1$ and $t = n'$ as it does not change the success probability. Let α^* denote the probability that $\mathcal{A}^*$ outputs a valid s-t, that is, the path also exists in G^*. Suppose $\mathcal{A}^*$ has access to a node incident retrieval oracle. This oracle for a queried vertex u returns the entire set of edges incident to u in G^* or $\perp$ if u is isolated. Observe that node incident retrieval queries in G' are equivalent to connected component incident retrieval (Definition 3.3) queries in G'. Let q be the worst case number of node incident retrieval queries made by $\mathcal{A}^*$.

Note that for $\mathcal{A}^*$ making an expected q queries, one can make it worst case $O(q)$ queries by decreasing α^* by a small additive constant. Here the probability is over both the random choices of algorithm $\mathcal{A}^*$ and the randomness of graph G'. By linearity of expectation, we may fix the random choices of $\mathcal{A}^*$ to obtain a deterministic algorithm $\mathcal{A}$ that outputs a valid s-t path with probability $\alpha \geq \alpha^*$. It thus suffices to prove an upper bound on α for such deterministic $\mathcal{A}$.

For the graph G', let $\pi(\mathcal{A}, G')$ denote the *trace* of running the deterministic $\mathcal{A}$ on G'. If $i_1(G'), \ldots, i_q(G')$ denotes the sequence of edges queried by $\mathcal{A}$ on G, and $\mathcal{N}_1(G'), \ldots, \mathcal{N}_q(G')$ denotes the returned sets of edges, then

$$\pi(\mathcal{A}, G') = \langle i_1(G'), \mathcal{N}_1(G'), i_2(G'), \ldots, i_q(G'), \mathcal{N}_q(G') \rangle.$$

If we condition on a particular trace $\tau = (i_1, N_1, i_2, \ldots, i_q, N_q)$, the distribution of G' conditioned on $\pi(\mathcal{A}, G') = \tau$ is the same as if we condition on the set of edges incident to $i_1, \ldots, i_q$ being $N_1, \ldots, N_q$. This is because the algorithm $\mathcal{A}$ is deterministic and the execution of $\mathcal{A}$ is the same for all graphs G' with the same such sets of edges incident to $i_1, \ldots, i_q$. Furthermore, no graph G' with a different set of incident edges for $i_1, \ldots, i_q$ will result in the trace τ.

For a trace $\tau = (i_1, N_1, \ldots, i_q, N_q)$, call the trace *connected* if there is a path from s to t using the discovered edges

$$\bigcup_{j=1}^{q} N_j.$$

Otherwise, call it *disconnected*. Intuitively, if a trace is disconnected, then it is unlikely that $\mathcal{A}$ will succeed in outputting a valid path connecting s and t, as it has to guess some of the edges along such a path. Furthermore, if $\mathcal{A}$ makes too few queries, then it is unlikely that the trace is connected. Letting $\mathcal{A}(G')$ denote the output of $\mathcal{A}$ on the graph G', we have for a random graph G' that

$$\alpha = \Pr[\mathcal{A}(G') \text{ is valid}] \leq \Pr[\pi(\mathcal{A}, G') \text{ is connected}] + \Pr[\mathcal{A}(G') \text{ is valid} \mid \pi(\mathcal{A}, G') \text{ is disconnected}].$$

We first bound $\Pr[\mathcal{A}(G') \text{ is valid} \mid \pi(\mathcal{A}, G') \text{ is disconnected}]$. For this, let $\tau = (i_1, N_1, \ldots, i_q, N_q)$ be an arbitrary disconnected trace in the support of $\pi(\mathcal{A}, G')$ when G' is an Erdős–Rényi random graph, where each edge is present with probability $p' \geq \frac{4C \log^2 n'}{n'}$. Observe that the output of $\mathcal{A}$ is determined from τ. Since τ is disconnected, the path reported by $\mathcal{A}$ on τ must contain at least one edge (u, v) where neither u nor v is among $\cup_j \{i_j\}$, or otherwise the output path is valid with probability 0 conditioned on τ. But conditioned on the trace τ, every edge that is not connected to $\{i_1, \ldots, i_q\}$ is present independently with probability p'. We thus conclude:

$$\Pr[\mathcal{A}(G') \text{ is valid} \mid \pi(\mathcal{A}, G') = \tau] \leq p'.$$

Since this holds for every disconnected τ, we conclude:

$$\Pr[\mathcal{A}(G') \text{ is valid} \mid \pi(\mathcal{A}, G') \text{ is disconnected}] \leq p'.$$

Next we bound the probability that $\pi(\mathcal{A}, G')$ is connected. For this, define for $1 \leq k \leq q$:

$$\pi_k(G') = \langle i_1(G'), \mathcal{N}_1(G'), i_2(G'), \ldots, i_k(G'), \mathcal{N}_k(G') \rangle.$$

as the trace of $\mathcal{A}$ on G' after the first k queries. As for $\pi_k(G')$, we say that $\pi_k(G')$ is connected if there is a path from s to t using the discovered edges

$$E(\pi_k(G')) = \bigcup_{j=1}^{k} \mathcal{N}_j(G')$$

and that it is disconnected otherwise. We further say that $\pi_k(G')$ is *useless* if it is both disconnected and $|E(\pi_k(G'))| \leq 2p'n'k$. Since

$$\Pr[\pi_k(G') \text{ is disconnected}] \geq \Pr[\pi_k(G') \text{ is useless}],$$

we prove that $\Pr[\pi_k(G') \text{ is useless}]$ is large. Therefore, we lower bound

$$\Pr[\pi_k(G') \text{ is useless} \mid \pi_{k-1}(G') \text{ is useless}].$$

Note that the base case $\pi_0(G')$ is defined to be useless as s and t are not connected when no queries have been asked and also $|E(\pi_0(G'))| = 0 \leq 2p'n' \cdot 0 = 0$. Let $\tau_{k-1} = (i_1, N_1, \ldots, i_{k-1}, N_{k-1})$ be any useless trace. The query $i_k = i_k(G')$ is uniquely determined when conditioning on $\pi_{k-1}(G') = \tau_{k-1}$, and so is the edge set $E_{k-1} = E(\pi_{k-1}(G'))$. Furthermore, we know that $|E_{k-1}| \leq 2p'n'(k-1)$. We now bound the probability that the query discovers more than $2p'n'$ new edges. If i_k has already been queried, no new edges are discovered and the probability is 0. So assume $i_k \notin \{i_1, \ldots, i_{k-1}\}$. Now observe that conditioned on $\pi_{k-1}(G') = \tau_{k-1}$, the edges (i_k, i) where $i \notin \{i_1, \ldots, i_{k-1}\}$ are independently included in G' with probability p' each. The number of new edges discovered is thus a sum of $m \leq n'$ independent Bernoulli's $X_1, \ldots, X_m$ with success probability p'. A Chernoff bound implies

$$\Pr\left[\sum_i X_i > 2n'p'\right] < (e/4)^{n'p'} < e^{-n'p'/3}.$$

Since we assume $p' \geq \frac{4C \log^2 n'}{n'}$, this is at most $n'^{-4C \log n'/3}$. We now bound the probability that the discovered edges $\mathcal{N}_k(G')$ makes s and t connected in $E(\pi_k(G'))$. For this, let V_s denote the nodes in the connected component of s in the subgraph induced by the edges E_{k-1}. Define V_t similarly. We split the analysis into three cases. First, if $i_k \in V_s$, then $\mathcal{N}_k(G')$ connects s and t if and only if one of the edges $\{i_k, v\}$ with $v \in V_t$ is in G'. Conditioned on $\pi_{k-1}(G') = \tau_{k-1}$, each such edge is in G' independently either with probability 0, or with probability p' (depending on whether one of the end points is in $\{i_1, \ldots, i_{k-1}\}$). A union bound implies that s and t are connected in $E(\pi_k(G'))$ with probability at most $p'|V_t|$. A symmetric argument upper bounds the probability by $p'|V_s|$ in case $i_k \in V_t$. Finally, if i_k is in neither of V_s and V_t, it must have an edge to both a node in V_s and in V_t to connect s and t. By independence, this happens with probability at most $p'^2|V_s||V_t|$. We thus conclude that

$$\Pr[\pi_k(G') \text{ is connected} \mid \pi_{k-1}(G') = \tau_{k-1}] \leq p' \max\{|V_s|, |V_t|\} \leq p'(|E_{k-1}| + 1) \leq 2p'n'k.$$

By union bound

$$\Pr[\pi_k(G') \text{ is useless} \mid \pi_{k-1}(G') \text{ is useless}] \geq 1 - 2p'^2 n' k - \frac{1}{n'^{4C\log n'/3}}.$$

Thus

$$\begin{aligned}
\Pr[\pi_q(G') \text{ is useless}] &= \prod_{k=1}^{q} \Pr[\pi_k(G') \text{ is useless} \mid \pi_{k-1}(G') \text{ is useless}] \\
&\geq \prod_{k=1}^{q} \left(1 - 2p'^2 n' k - \frac{1}{n'^{4C\log n'/3}}\right) \\
&\geq 1 - \sum_{k=1}^{q} \left(2p'^2 n' k + \frac{1}{n'^{4C\log n'/3}}\right) \\
&\geq 1 - p'^2 n' q(q+1) - \frac{q}{n'^{4C\log n'/3}}.
\end{aligned}$$

It follows

$$\Pr[\pi(G') \text{ is connected}] = 1 - \Pr[\pi(G') \text{ is disconnected}] \leq 1 - \Pr[\pi(G') \text{ is useless}] \leq p'^2 n'(q+1)^2 + \frac{q}{n'^{4C\log n'/3}}.$$

For $q = o\left(\frac{1}{p'\sqrt{n'}}\right)$ and $p' \geq \frac{4C\log^2 n'}{n'}$ node-incident queries in the meta-graph G', the success probability remains $o(1)$. Therefore, $q = \Omega\left(\frac{1}{p\cdot\log^2 n\cdot\sqrt{n}}\right)$ connected component incident retrieval (Definition 3.3) queries in given G are necessary. □

A.4 Proof of Theorem 3.5

Proof. We consider an Erdős-Rényi random graph $G \sim G(n,p)$, where p satisfies

$$p(1-p) \geq C \cdot \frac{\log^4(n)}{n}. \tag{1}$$

and, $C > 0$ is taken to be a sufficiently large constant.

Vertex degrees of the random graph. Let $\deg(i)$ denote the degree of vertex i in G. By Bernstein's inequality and a union bound, we have the following with probability at least $1 - 2/n$:

$$(1-\delta_0)pn \leq \deg(i) \leq (1+\delta_0)pn \quad \text{for all vertices } i \text{ in } G \tag{2}$$

where

$$\delta_0 = \delta_0(p,n) := \frac{1}{n} + 2\sqrt{\frac{(1-p)\ln(n)}{pn}} + \frac{2\ln(n)}{3pn}.$$

The assumption in Equation (1) implies that $\delta_0 = O(1/\log^{3/2}(n))$.

Deviation of the random adjacency matrix from its expectation. Let the $n \times n$ random matrix A denote the adjacency matrix of G, so

$$A_{i,j} = \begin{cases} 1 & \text{if } i \neq j \text{ and } \{i,j\} \text{ is an edge in } G; \\ 0 & \text{otherwise.} \end{cases}$$

Let $X = A - \mathbb{E}(A)$, so we have the following:

$$\begin{aligned}
|X_{i,j}| &\leq 1 && \text{for all } 1 \leq i \leq j \leq n; \\
\mathbb{E}(X_{i,j}) &= 0 && \text{for all } 1 \leq i \leq j \leq n; \\
\operatorname{var}(X_{i,j}) &= p(1-p) && \text{for all } 1 \leq i < j \leq n.
\end{aligned}$$

Then, using the assumption in Equation (1) and the above properties of the random matrix X, Theorem 1.4 of Vu (2007) implies that, there is a constant $C' > 0$ such that with probability at least $1 - o(1)$,

$$\|X\|_2 \leq 2\sqrt{p(1-p)n} + C'(p(1-p)n)^{1/4}\log(n).$$

Here, the norm on X is the spectral norm (i.e., largest singular value). For any $\varepsilon > 0$, there is a large enough C in Equation (1) such that the above inequality implies

$$\|A - \mathbb{E}(A)\|_2 \leq (2+\varepsilon)\sqrt{p(1-p)n}. \tag{3}$$

We henceforth condition on the event that both Equation (2) and Equation (3) hold.

Eigenvalues and eigenvectors of the expected adjacency matrix. For a symmetric matrix M, let $\lambda_k(M)$ denote its k-th largest eigenvalue. The matrix $\mathbb{E}(A)$ can be written as

$$\mathbb{E}(A) = pnuu^\intercal - pI_n,$$

where $u := n^{-1/2}1_n$, $1_n := (1, 1, \ldots, 1)$ is the all-1s vector in $\mathbb{R}^n$, and I_n is the $n \times n$ identity matrix. Therefore, the largest eigenvalue of $\mathbb{E}(A)$ is $\lambda_1(\mathbb{E}(A)) = p(n-1)$, and u is a corresponding (unit length) eigenvector. All other eigenvalues of $\mathbb{E}(A)$ are $\lambda_k(\mathbb{E}(A)) = -p$, for $k \neq 1$, and the corresponding eigenvectors $u_\perp$ satisfy $1_n^\intercal u_\perp = 0$.

Eigenvalues of the random adjacency matrix. By Weyl's inequality,

$$|\lambda_k(A) - \lambda_k(\mathbb{E}(A))| \leq \|A - \mathbb{E}(A)\|_2$$

for all k. Therefore, using Equation (3), we find that

$$(1-\delta_1)pn \leq \lambda_1(A) \leq (1+\delta_1)pn \tag{4}$$

where

$$\delta_1 = \delta_1(p,n) := \frac{1}{n} + (2+\varepsilon)\sqrt{\frac{1-p}{pn}}.$$

Furthermore, $\lambda(A) := \max\{\lambda_2(A), |\lambda_n(A)|\}$ satisfies

$$\lambda(A) \leq (2+\varepsilon+\delta_2)\sqrt{p(1-p)n} \tag{5}$$

where

$$\delta_2 = \delta_2(p,n) := \sqrt{\frac{p}{(1-p)n}}.$$

The assumption in Equation (1) implies that $\delta_1 = O(1/\log^2(n))$ and $\delta_2 = O(1/\log^2(n))$.

Leading eigenvector of the random adjacency matrix. Let v_1 be any unit length eigenvector corresponding to the largest eigenvalue $\lambda_1(A)$ of A. Recall that $u = n^{-1/2}1_n$ is a unit length eigenvector corresponding to the largest eigenvalue of $\mathbb{E}(A)$. We show that v_1 (or $-v_1$) is close to u in terms of both the Euclidean norm as well as the l^∞ norm.

The closeness of v_1 to u in Euclidean norm follows from the Davis-Kahan $\sin(\Theta)$ theorem, but here we give a direct argument. We can write

$$u = c_1 v_1 + c_2 v_\perp$$

for some unit vector $v_\perp$ orthogonal to v_1, and some coefficients $c_1 = u^\intercal v_1$ and c_2 satisfying $c_1^2 + c_2^2 = 1$. Then

$$\begin{aligned}
(A - \mathbb{E}(A))v_1 &= (A - pnuu^\intercal - pI_n)v_1 \\
&= \lambda_1(A)v_1 - c_1 pnu - pv_1 \\
&= \lambda_1(A)v_1 - c_1 pn(c_1 v + c_2 v_\perp) - pv_1 \\
&= (\lambda_1(A) - p - c_1^2 pn)v_1 - c_1 c_2 pn v_\perp.
\end{aligned}$$

Since v_1 and $v_\perp$ are orthogonal, the Pythagorean theorem implies

$$\|(A - \mathbb{E}(A))v_1\|_2 \geq |\lambda_1(A) - p - c_1^2 pn|.$$

On the other hand, by Equation (3), we have

$$\|(A - \mathbb{E}(A))v_1\|_2 \leq (2+\varepsilon)\sqrt{p(1-p)n}.$$

Therefore,

$$\lambda_1(A) - p - c_1^2 pn \leq (2+\varepsilon)\sqrt{p(1-p)n},$$

which, together with Equation (4) and Equation (1), implies

$$(u^\intercal v_1)^2 = c_1^2 \geq \frac{\lambda_1(A)}{pn} - \frac{1}{n} - (2+\varepsilon)\sqrt{\frac{1-p}{pn}} = 1 - 2\delta_1.$$

In particular, this implies $\|\mathrm{sign}(c_1)v_1 - u\|_2 = \sqrt{2(1-|c_1|)} \leq \sqrt{2(1-\sqrt{1-2\delta_1})} = O(\sqrt{\delta_1})$.

Now we show closeness of v_1 to u in l^∞ norm. For any non-negative integer k, define the vector

$$u^{(k)} = (u_1^{(k)}, \ldots, u_n^{(k)}) := \frac{1}{\lambda_1(A)^k} A^k u.$$

Also define

$$\delta_0' = \delta_0'(p,n) := \frac{1+\delta_0}{1-\delta_1} - 1.$$

The assumption in 1 implies that $\delta_0' = O(\delta_0) = O(1/\log^{3/2}(n))$. We show, by induction, that for all $k \leq 1 + 1/(2\delta_0')$,

$$u_i^{(k)} \in \left[\frac{1-2k\delta_0'}{\sqrt{n}}, \frac{1+2k\delta_0'}{\sqrt{n}}\right] \quad \text{for all vertices } i \text{ in } G.$$

The base case $k = 0$ clearly holds by definition of $u^{(0)} = u$. So assume the claim holds for $k-1$. Then, for any vertex i,

$$\begin{aligned}
u_i^{(k)} = \frac{1}{\lambda_1(A)} \sum_{j=1}^n A_{i,j} u_j^{(k-1)} &\leq \frac{\deg(i)}{\lambda_1(A)} \cdot \frac{1+2(k-1)\delta_0'}{\sqrt{n}} \quad \text{(inductive hypothesis)} \\
&\leq \frac{1+\delta_0}{1-\delta_1} \cdot \frac{1+2(k-1)\delta_0'}{\sqrt{n}} \quad \text{(by Equations (2) and (4))} \\
&= \frac{(1+\delta_0')(1+2(k-1)\delta_0')}{\sqrt{n}} \\
&\leq \frac{1+2k\delta_0'}{\sqrt{n}} \quad \text{(by the upper-bound on } k\text{)}.
\end{aligned}$$

Similarly,

$$\begin{aligned}
u_i^{(k)} &\geq \frac{\deg(i)}{\lambda_1(A)} \cdot \frac{1-2(k-1)\delta_0'}{\sqrt{n}} \quad \text{(inductive hypothesis)} \\
&\geq \frac{1-\delta_0}{1+\delta_1} \cdot \frac{1-2(k-1)\delta_0'}{\sqrt{n}} \quad \text{(by Equations (2) and (4))} \\
&\geq \frac{(1-\delta_0')(1-2(k-1)\delta_0')}{\sqrt{n}} \quad \text{(by comparison with } \delta_0'\text{)} \\
&\geq \frac{1-2k\delta_0'}{\sqrt{n}}.
\end{aligned}$$

Therefore the claim holds for all $k \leq 1 + 1/(2\delta_0')$.

Since $\delta_0' = O(1/\log^{3/2}(n))$, we can choose (with foresight)

$$k \asymp \frac{\log(n)}{\log(1/\rho(A))},$$

where

$$\rho(A) := \frac{\lambda(A)}{\lambda_1(A)} \leq \frac{2+\varepsilon+\delta_2}{1-\delta_1}\sqrt{\frac{1-p}{pn}} = O(\delta_1).$$

This choice of k satisfies $k \leq 1 + 1/(2\delta_0')$. Observe that

$$\begin{aligned} u^{(k)} = \frac{1}{\lambda_1(A)^k}A^k u &= \frac{1}{\lambda_1(A)^k}A^k(c_1 v_1 + c_2 v_\perp) \\ &= c_1 v_1 + c_2 \frac{1}{\lambda_1(A)^k}A^k v_\perp. \end{aligned}$$

Therefore

$$\begin{aligned} \left\| \frac{1}{c_1}u^{(k)} - v_1 \right\|_\infty &= \left| \frac{c_2}{c_1 \lambda_1(A)^k} \right| \left\| A^k v_\perp \right\|_\infty \\ &\leq \left| \frac{c_2}{c_1 \lambda_1(A)^k} \right| \left\| A^k v_\perp \right\|_2 \\ &\leq \left| \frac{c_2}{c_1} \right| \left(\frac{\lambda(A)}{\lambda_1(A)} \right)^k = \left| \frac{c_2}{c_1} \right| \rho(A)^k. \end{aligned}$$

We also have

$$\|u^{(k)} - u\|_\infty \leq \frac{2k\delta_0'}{\sqrt{n}} \quad \text{and} \quad \left\| \frac{1}{|c_1|}u - u \right\|_\infty = \left(\frac{1}{|c_1|} - 1 \right)\frac{1}{\sqrt{n}}.$$

By the triangle inequality,

$$\|\mathrm{sign}(c_1)v_1 - u\|_\infty \leq \left(\frac{1}{|c_1|} - 1 \right)\frac{1}{\sqrt{n}} + \frac{2k\delta_0'}{|c_1|\sqrt{n}} + \left| \frac{c_2}{c_1} \right| \rho(A)^k.$$

Now we use the specific choice of k to conclude

$$\|\mathrm{sign}(c_1)v_1 - u\|_\infty \leq \frac{\epsilon_0}{\sqrt{n}} \tag{6}$$

where

$$\epsilon_0 = O\left(\frac{\delta_0 \log(n)}{\log(1/\rho(A))} \right).$$

The assumption in Equation (1) implies that $\epsilon_0 = O(1/(\sqrt{\log(n)}\log\log(n)))$.

We can now prove the main result, using mostly the same argument as in the proof of Lemma 1 from Alon et al. (2023). Without loss of generality, we assume $\mathrm{sign}(c_1) = 1$ (else we replace v_1 with $-v_1$). Fix any vertex i in G, any $\delta \in (0,1)$, and any distance d. Let Z denote the subset of vertices j such that the (i,j)-th entry of A^d is zero. In other words, there are no length d paths from i to vertices in Z. Let 1_Z be the $\{0,1\}$-characteristic vector for Z, i.e., the j-th component of 1_Z is 1 if and only if $j \in Z$. We can write

$$1_Z = b_1 v_1 + b_2 v_\perp$$

for some unit vector $v_\perp$ orthogonal to v_1, and some coefficients $b_1 = 1_Z^\intercal v_1$ and b_2 satisfying $b_1^2 + b_2^2 = \|1_Z\|_2^2 = |Z|$.

Let e_j denote the j-th coordinate basis vector. Note that by Equation (6), we have

$$e_j^\intercal v_1 \geq \frac{1-\epsilon_0}{\sqrt{n}} \quad \text{and} \quad b_1 = 1_Z^\intercal v_1 = \sum_{j \in Z} e_j^\intercal v_1 \geq \frac{|Z|(1-\epsilon_0)}{\sqrt{n}}.$$

Therefore

$$\begin{aligned} e_i^\intercal A^d 1_Z &= e_i^\intercal A^d (b_1 v_1 + b_2 v_\perp) \\ &= b_1 \lambda_1(A)^d e_i^\intercal v_1 + b_2 e_i^\intercal A^d v_\perp \\ &\geq \lambda_1(A)^d |Z| \frac{(1-\epsilon_0)^2}{n} - |b_2|\lambda(A)^d \\ &\geq \lambda_1(A)^d |Z| \frac{(1-\epsilon_0)^2}{n} - \sqrt{|Z|}\lambda(A)^d. \end{aligned}$$

On the other hand, we have $e_i^\intercal A^d 1_Z = 0$ since the (i,j)-th entry of A^d is zero for all $j \in Z$. Combining with the above inequality, we have

$$\lambda_1(A)^d |Z| \frac{(1-\epsilon_0)^2}{n} - \sqrt{|Z|}\lambda(A)^d \leq 0,$$

which rearranges to

$$\frac{|Z|}{n} \leq \frac{n}{(1-\epsilon_0)^4}\rho(A)^{2d}.$$

The right-hand side is at most δ provided that

$$d \geq \frac{1}{2} \cdot \frac{\log\left(\frac{n}{\delta(1-\epsilon_0)^4}\right)}{\log(1/\rho(A))}.$$

We conclude that there are at most δn vertices with distance from i more than

$$\frac{1}{2} \cdot \frac{\log\left(\frac{n}{\delta(1-\epsilon_0)^4(1-\rho(A)^2)}\right)}{\log(1/\rho(A))}.$$

This implies that for any vertex i, for at least $(1-\delta)n$ other vertices j, the number of nodes visited by the double BFS algorithm is at most

$$\begin{aligned}\left(\max_i \deg(i)\right)^{\left\lceil \frac{1}{4} \cdot \frac{\log\left(\frac{n}{\delta(1-\epsilon_0)^4(1-\rho(A)^2)}\right)}{\log(1/\rho(A))} \right\rceil} &\leq ((1+\delta_0)pn)^{\left\lceil \frac{1}{4} \cdot \frac{\log\left(\frac{n}{\delta(1-\epsilon_0)^4(1-O(\delta_1^2))}\right)}{\log(1/\rho(A))} \right\rceil} \\ &= O(n/\delta)^{\frac{1}{4} \cdot \frac{\log((1+\delta_0)pn)}{\log(1/\rho(A))}}.\end{aligned}$$

We can simplify the exponent in the final expression:

$$\begin{aligned}\frac{1}{4} \cdot \frac{\log((1+\delta_0)pn)}{\log(1/\rho(A))} &\leq \frac{1}{4} \cdot \frac{\log\left(\frac{1+\delta_0}{1-\delta_1}\lambda_1(A)\right)}{\log(1/\rho(A))} \\ &= \frac{1}{4} \cdot \left(\frac{\log\left(\frac{1+\delta_0}{1-\delta_1}\right)}{\log(1/\rho(A))} + \frac{\log(\lambda_1(A))}{\log(\lambda_1(A)) - \log(\lambda(A))} \right) \\ &\leq \frac{1}{4} \cdot \left(\frac{\frac{\delta_0+\delta_1}{1-\delta_1}}{\log(1/\delta_1)} + \frac{1}{1 - \frac{\log(\lambda(A))}{\log(\lambda_1(A))}} \right) \\ &\leq \frac{1}{4} \cdot \left(\frac{\frac{\delta_0+\delta_1}{1-\delta_1}}{\log(1/\delta_1)} + \frac{1}{1 - \frac{\log((2+\varepsilon+\delta_2)\sqrt{pn})}{\log((1-\delta_1)pn)}} \right) \\ &= \frac{1}{4} \cdot \left(\frac{\frac{\delta_0+\delta_1}{1-\delta_1}}{\log(1/\delta_1)} + \frac{1}{1 - \frac{\log(2+\varepsilon+\delta_2) + \frac{1}{2}\log(pn)}{\log(1-\delta_1)+\log(pn)}} \right) \\ &= \frac{1}{4} \cdot \left(o(1) + \frac{1}{1 - \frac{1/2+o(1)}{1-o(1)}} \right) \\ &= \frac{1}{4} \cdot (2 + o(1)) = \frac{1}{2} + o(1).\end{aligned}$$

Therefore the number of nodes visited is

$$O(n/\delta)^{\frac{1}{2}+o(1)}.$$

□

A.5 Proof of Claim 4.2

Proof. If $\mathrm{comp}_G(s) = \mathrm{comp}_G(t)$, the algorithm returns a simple s-t path $\Pi \subseteq E(G)$ upon the first run of generation phase. Note that $E(G) \subseteq E(G^*)$ implies $\Pi \subseteq E(G^*)$.

Otherwise, given that (G, G^*) is an admissible pair, by definition, there exists a connected component C of G such that, for every vertex $u \in V$ with $N_{G^*}(u) \neq \emptyset$,

$$\mu_u^{G^*}\big(N_{G^*}(u) \cap V(C)\big) \geq \gamma.$$

The algorithm repeatedly queries $\mathcal{O}_{G^*}(s)$ and $\mathcal{O}_{G^*}(s)$ until it finds a neighbor of both s and t denoted by v_s and v_t in C. Since C is a connected component of G, in the generation phase, a BFS in G yields a simple path $\Pi_C \subseteq E(G)$ from v_s to v_t. Note that $E(G) \subseteq E(G^*)$ implies $\Pi_C \subseteq E(G^*)$. Moreover, both of the (s, v_s) and (v_t, t) edges are returned by the retrieval oracle on G^*, and therefore, lie in $E(G^*)$. Thus, a path will be found during the generation phase. Note that if the oracle returns $\perp$ either s or t, then there can be no s-t path in G^*, so the algorithm outputs NO.

It remains to bound the number Q of retrieval calls. For any endpoint x with $N_{G^*}(x) \neq \emptyset$, γ-admissibility implies that each query to $\mathcal{O}_{G^*}(x)$ hits C with probability at least γ, independently of past failures. Therefore the expected number of query calls for each end point is bounded by γ, that is, $\mathbb{E}[Q_x] \leq 1/\gamma$. For $Q := Q_s + Q_t$ by linearity of expectation $\mathbb{E}[Q] \leq 2/\gamma$ when a path exists; thus, the pair (G, G^*) is $2/\gamma$-retrieval friendly. □

A.6 Proof of Theorem 4.3

Proof. Since $G \sim \mathcal{G}(n, p\eta)$ and $np\eta > 1$, the standard Erdős–Rényi giant-component theorem Frieze and Karoński (2015) implies that with high probability there is a unique giant C with $|C| = (\gamma \pm o(1))n$ with $\gamma = 1 - e^{-np\eta\gamma}$, and all other components are $O(\log n)$. Similarly, with high probability G^* is connected. The generation process is equivalent to first sampling $G \sim \mathcal{G}(n, p\eta)$ and then, to form G^*, adding each edge not present in G independently with probability $q = \frac{p - p\eta}{1 - p\eta}$. This ensures G^* is a valid $\mathcal{G}(n, p)$ graph. This allows us to first condition on the realization of G (and thus its giant component C) and then analyze the properties of G^*. Note that in Erdős–Rényi model for all u, π_u is uniform over $N_{G^*}(u)$, and admissibility reduces to

$$\frac{|N_{G^*}(u) \cap V(C)|}{|N_{G^*}(u)|} \geq \gamma.$$

Fix constant $\alpha \in (0, 1/5]$. For any vertex $u \in V$, its degree $|N_{G^*}(u)|$ follows a binomial distribution $\mathrm{Bin}(n-1, p)$. Since $p(n-1) \geq \log n$, Chernoff bounds give, for each fixed u

$$\Pr\big(|N_{G^*}(u)| > (1+\alpha)(n-1)p\big) \leq \exp\left(-\frac{\alpha^2}{3}(n-1)p\right)$$

Similarly, conditioned on G, for each vertex not in the giant component the number of its neighbors within the giant component, $|N_{G^*}(u) \cap V(C)|$, follows $\mathrm{Bin}(|C|, p)$

$$\Pr\big(|N_{G^*}(u) \cap V(C)| < (1-\alpha)|C|p \mid G\big) \leq \exp\left(-\frac{\alpha^2}{3}|C|p\right)$$

To ensure these bounds hold for all vertices simultaneously, we apply a union bound. The probability of the first event failing for at least one vertex is at most

$$\Pr\Big(\exists u :\ |N_{G^*}(u)| > (1+\alpha)(n-1)p\Big) \leq n \cdot \exp\left(-\frac{\alpha^2 C_0}{3}\frac{(n-1)\log n}{n}\right)$$

$$\Pr\Big(\exists u :\ |N_{G^*}(u) \cap V(C)| < (1-\alpha)|C|p \;\Big|\; G\Big) \leq n \cdot \exp\left(-\frac{\alpha^2 C_0}{3}\frac{|C|\log n}{n}\right)$$

For sufficiently large n, $|C| \geq \frac{n\gamma}{2}$. For $C_0 > \frac{18}{\gamma\alpha^2}$ probabilities above are $o(n^{-2})$, and the lower bound ratio for any vertex u:

$$\frac{|N_{G^*}(u) \cap V(C)|}{|N_{G^*}(u)|} \geq \frac{(1-\alpha)|C|p}{(1+\alpha)(n-1)p} = \frac{1-\alpha}{1+\alpha} \cdot \frac{|C|}{n-1} \geq \gamma/3.$$

□

A.7 Proof of Theorem 4.7

Proof. Fix an endpoint $x \in \{s, t\}$. Consider K bins, one per partition $i \in [K]$, such that bin i corresponds to C_i, and throw one ball per querying the retriever $\mathcal{O}_{G^*}(x)$. A ball occupies bin i if the returned neighbor lies in $V(C_i)$. By Definition 4.5, for every i,

$$\Pr[\text{ball occupies bin } i] = \mu_u^{G^*}\big(N_{G^*}(x) \cap V(C_k)\big) \geq \gamma$$

Note that one ball may occupy multiple bins, that is, the same vertex can lie in many C_i which only helps cover faster. After t throws, for any fixed i the probability bin i is still empty is at most $(1-\gamma)^t \leq e^{-\gamma t}$. By a union bound over the K bins,

$$\Pr[\exists \text{ empty bin after } t \text{ balls}] \;\leq\; Ke^{-\gamma t}.$$

Doing this for both endpoints yields an expected total of at most $O(\frac{\log K}{\gamma})$ queries. Then, for both endpoint $\{s, t\}$ bin i is occupied by anchor $v_s^{(i)}, v_t^{(i)} \in V(C_i)$. Since C_i is a connected component of $G_i = (V, E_i)$, a BFS yields a simple path $\Pi_{C_i} \subseteq E(G)$ from v_s to v_t. Note that $E(G) \subseteq E(G^*)$ implies $\Pi_{C_i} \subseteq E(G^*)$. Moreover, every $(s, v_s^{(i)})$ and $(v_t^{(i)}, t)$ edges are returned by the retrieval oracle on G^*, and therefore, lie in $E(G^*)$. Thus, every $(s, v_s^{(i)}) \circ \Pi_{C_i} \circ (v_t^{(i)}, t)$ is also in G^*. Moreover, since Π_{C_i} uses only edges of E_i, and $\{E_i\}_{i=1}^K$ partitions E, the internal segments $\{\Pi_{C_i}\}_{i=1}^K$ are pairwise edge disjoint in G and G^*. □

B Future Directions

B.1 Empirical Directions

Another extension is to provide empirical evidence that matches our theory. An easy route is to implement synthetic experiments showing that the model accuracy has an inflection point: low with to few queries and high with enough queries. However, this does not shed light on the impact of parametric knowledge in real LLMs. Also, we already have much evidence that RAG and tool use work well with modern LLMs.

To validate the necessity of dense parametric knowledge, it would be ideal to train models on multiple mixtures of pre-training corpora, crafted to have different proportions of a target domain. For example, one could train on a mix of general purpose web data and selectively chosen data in a niche domain, like medical or law documents.

B.2 Theoretical Directions

In this work, we focused mainly on finding a path between two vertices s, t and our examples on an Erdős–Rényi random graph with η retention threshold on the prior. It is natural to seek query-complexity thresholds for other graph-theoretic tasks under a partially observed prior. The relevant phenomenon is a prior sensitive phase transition, that is, a critical retention level $\eta(P)$ at which a task P switches from requiring $\omega(1)$ queries to allowing $O(1)$ expected queries. In general, there are many sub-linear graph and matrix questions that we can study with prior knowledge. For example, see Beame et al. (2020); Feige (2004); Feige and Ferster (2021); Rácz and Schiffer (2019); Rashtchian et al. (2020, 2021); Chen et al. (2020) and references therein. Importantly, our work opens up new questions, where we can study how the query complexity changes based on the knowledge G instead of starting with no information about G^*. This includes problems with more global dependencies, such as Minimum Spanning Tree recovery. Note that a natural extension to finding a (shortest) path between two vertices is to consider a set of M input vertices $(s_1, \ldots, s_M)$ and ask whether the learner can efficiently recover a (minimum) spanning tree connecting them all.

What is the tightest possible lower bound on the query complexity for finding a tree spanning input vertices $(s_1, \ldots, s_M)$? This bound should be characterized as a function of the structural properties and densities of both the pretrained graph G and the ground truth graph G^*? What structural properties beyond admissibility

of (G, G^*) guarantee a constant upper bound on the expected number of retrieval queries for recovering a tree spanning $(s_1, \ldots, s_M)$?

Moreover, problems concerning local structure, like triangle detection and counting are interesting to explore. Another interesting future direction is the observation model that generates G and G^*. In this work we use i.i.d. edge retention, but other realistic mechanisms include radius-dependent thinning in random geometric k-NN graphs, which models conserving local edges while suppressing long edges, and adversarial deletions. Each induces a different critical $\eta(P)$ and poses open problems at the interface of random graph theory and query complexity.