BRIDGEKG-RAG: BRIDGING WORDING GAPS IN KNOWLEDGE-GRAPH RAG

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge-graph retrieval-augmented generation (KG-RAG) answers questions by linking phrases in the question to entities in a graph of document-grounded facts and retrieving evidence from their neighborhoods. This works when the question uses the same or similar names under which the relevant evidence is organized. However, users may instead mention a near-synonym, a loosely related term, or even a different kind of entity, while the evidence needed to answer is organized under an entity the question never names. We call this the *wording gap*. The entity mentioned in the question may be a valid graph entity, yet retrieval from its neighborhood can still miss the answer-bearing entity. Neither string similarity nor query rewriting necessarily resolves this mismatch, because the two entities may refer to different things, while traversal over factual edges cannot recover a connection that is absent from the graph. We propose BridgeKG-RAG, which augments the factual graph with typed *bridge* edges that capture wording relationships between entities. These bridges are used only to expand retrieval starting points and are never treated as answer evidence. Given a question, BridgeKG-RAG selects a small number of bridges to *traverse*, adds the reached entities to the retrieval set, and then retrieves document-grounded evidence from the factual graph. If the question already names the answer-bearing entity, retrieval remains unchanged. On GapQA, a wording-gap benchmark constructed from Doc2Dial, MultiDoc2Dial, and PrivacyQA, our proposed BridgeKG-RAG substantially improves evidence recall and answer quality.

1 INTRODUCTION

Large language models store substantial factual knowledge in their parameters (Petroni et al., 2019), but this knowledge can be incomplete, stale, or hallucinated on knowledge-intensive questions (Ji et al., 2023). Retrieval-augmented generation (RAG) addresses this limitation by grounding generation in evidence retrieved from an external corpus (Lewis et al., 2020). Most RAG systems retrieve passages using lexical or dense similarity to the question (Robertson & Zaragoza, 2009; Karpukhin et al., 2020). Knowledge-graph RAG (KG-RAG) instead links phrases in the question to entities in a graph of document-grounded facts and retrieves evidence from their neighborhoods (Pan et al., 2024). This structured representation makes entity relations explicit and supports more inspectable retrieval (Gao et al., 2023b; Han et al., 2026). However, KG-RAG still relies on the question naming, or closely matching, the entity under which the relevant evidence is organized.

Other methods improve retrieval or reasoning on top of this pipeline. Query rewriting and related retrieval methods reformulate the question or adjust the search process to better match relevant passages (Ma et al., 2023; Karpukhin et al., 2020; Gao et al., 2023a; Yan et al., 2024). Graph-based methods instead explore or plan paths over recorded relations, as in Think-on-Graph and Reasoning on Graphs (Sun et al., 2024; Luo et al., 2024). These approaches are effective when the answer-bearing neighborhood is already reachable, either because the wording can be matched to it or because a factual path connects it to the linked entities. They do not address the case where the question names a different but related entity and no recorded factual edge connects that entity to the one under which the answer is organized.

The methods above therefore share a challenge, which we call the *wording gap* and illustrate in Figure 1. Users do not always name the entity under which the answer is organized. Instead, they may ask vaguely through a related object, organization, form, product, or other concept associated

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

with what they actually need. The question and the documents can therefore concern the same situation while naming different entities (Furnas et al., 1987; Lakoff & Johnson, 1980; Nissim & Markert, 2003). Existing methods begin from the entity mentioned on the surface of the question and search for evidence in its neighborhood. If the documents never wrote the wording relationship down as a fact, the entity actually organizing the answer stays out of reach. The difficulty is therefore not simply a weak string match or an insufficient number of factual hops, but that the user's vague wording can place retrieval at a different, yet still plausible, starting entity.

We propose BridgeKG-RAG to close this gap. BridgeKG-RAG augments the original knowledge graph with typed *bridge* edges between entities that may be connected through plausible wording relationships. Unlike factual edges, bridges do not represent document-grounded claims and therefore never enter the evidence set. Given a question, the system selects a small number of bridges to *traverse*, adds the reached entities as additional retrieval starting points, and then retrieves document-grounded evidence from the expanded set. Following a bridge changes only where retrieval starts, not what counts as evidence. If the question already names the answer-bearing entity, no bridge is followed and retrieval remains unchanged. Together, the factual and bridge layers form an *extended knowledge graph*, allowing BridgeKG-RAG to reach answer-bearing neighborhoods that are inaccessible through factual edges alone while keeping the final answer grounded in document evidence.

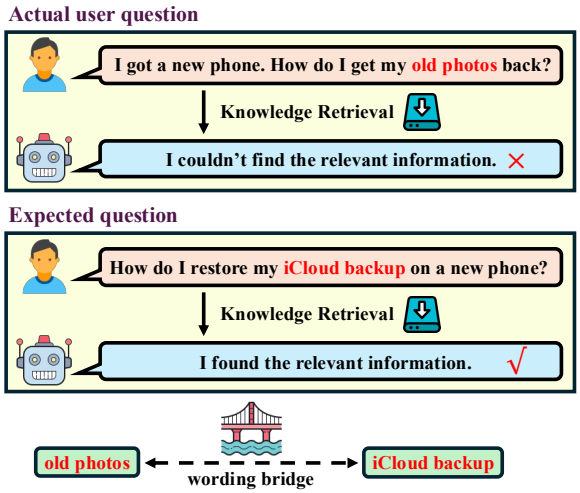

Figure 1: Illustration of the wording gap. When the question names the answer-bearing entity, retrieval finds the relevant information; a natural but vague reference to a related entity can instead lead retrieval to the wrong place. A wording bridge connects the user-mentioned entity to the entity under which the answer is organized.

Our contributions are:

- We identify the wording gap problem: the facts needed to answer can live under an entity the question never names, so methods that start from the surface name never reach them. This is distinct from finding a close paraphrase (Furnas et al., 1987) and from taking more hops over recorded facts (Sun et al., 2024; Luo et al., 2024).
- We introduce BridgeKG-RAG, which overlays typed bridge edges that encode wording relationships rather than facts, and a procedure that traverses selected bridges to add starting points before collecting evidence.
- We construct a vague question answering benchmark GapQA based on Doc2Dial, MultiDoc2Dial, and PrivacyQA (Feng et al., 2020; 2021; Ravichander et al., 2019). On this benchmark, traversal substantially improves evidence recall and answer quality over KG-RAG without a wording traversal and over retrieval and graph-exploration methods that assume the right neighborhood is already reachable.

2 Problem Formulation

Let $\mathcal{D}$ be a document corpus and $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ a knowledge graph extracted from it. Each node $v \in \mathcal{V}$ denotes an entity, and each edge $e \in \mathcal{E}$ is represented as a triplet $e = (h, r, t)$, where $h, t \in \mathcal{V}$ are the head and tail entities and r is the relation between them. All triplets in $\mathcal{E}$ are extracted from the source documents in $\mathcal{D}$, and we retain the corresponding source text from which each triplet is derived. Such a graph is designed to organize documented relations among entities, but is not necessarily optimized for information access when users refer to related but different entities.

Given a question q, let $A(q) \subseteq \mathcal{V}$ denote the entities mentioned in the question. We call an entity $v^\star \in \mathcal{V}$ an *answer-bearing entity* if the facts nearby $v^\star$ contain evidence needed to answer q. A

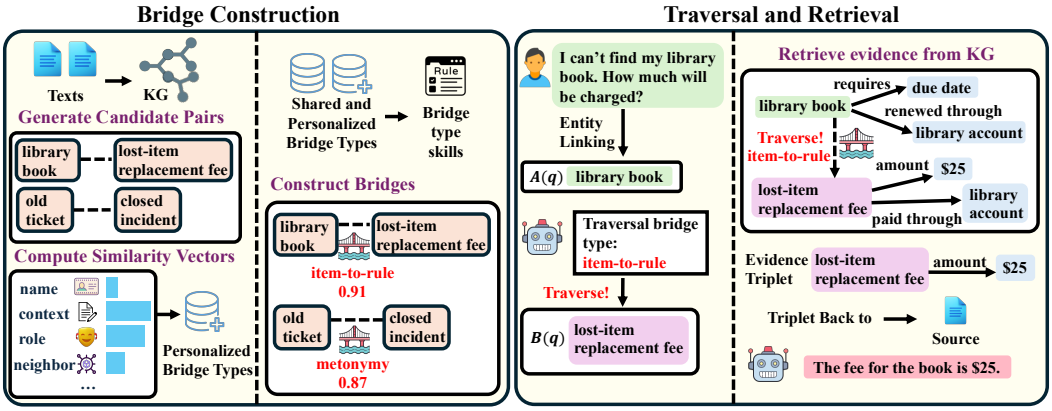

Figure 2: Overview of BridgeKG-RAG. **Bridge Construction** identifies and labels typed wording bridges using pairwise similarity patterns and corpus-adapted type information. **Traversal and Retrieval** selects useful bridges at query time to expand the retrieval starting set, while answer evidence is retrieved only from the original factual graph.

wording gap occurs when the entity named in the question is not the answer-bearing entity:

$$v^\star \notin A(q). \tag{1}$$

In the cases we study, the missing connection between $A(q)$ and $v^\star$ is a wording relationship rather than a document-grounded fact, so traversing factual edges alone does not recover the required neighborhood. The task is to answer q from document-grounded evidence while recovering the appropriate answer-bearing entity without treating the wording relationship itself as evidence.

3 Method

Given a question q, let $A(q) \subseteq \mathcal{V}$ be the entities mentioned in the question, as defined in section 2. BridgeKG-RAG augments the factual graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a second layer of typed *bridge* edges that connect question-mentioned entities to other entities that may organize the relevant evidence.

A bridge is a directed wording relationship

$$b = (u, \tau, v, c), \qquad c \in [0, 1], \tag{2}$$

written $u \xrightarrow{\tau, c} v$. The type τ describes how u can refer to v for retrieval purposes, such as an alternative name for the same entity or a related name used when the relevant knowledge is organized under v. The score c measures how strongly the bridge should be considered for traversal. Let $\mathcal{E}^{\mathrm{b}}$ denote the set of bridge edges. The resulting extended graph is

$$\mathcal{G}^{+} = (\mathcal{V},\, \mathcal{E} \cup \mathcal{E}^{\mathrm{b}}). \tag{3}$$

The two edge sets serve different purposes. $\mathcal{E}$ contains document-grounded facts and is the only source of answer evidence, whereas $\mathcal{E}^{\mathrm{b}}$ is used only to move between retrieval starting points. A bridge therefore changes which entities are searched, but never contributes evidence itself.

BridgeKG-RAG consists of two stages. Figure 2 summarizes the pipeline. **Bridge construction** constructs $\mathcal{E}^{\mathrm{b}}$ from $\mathcal{G}$ by generating candidate entity pairs, representing them with similarity features, estimating the wording types present in the corpus, and labeling bridge edges. **Traversal and Retrieval** uses the resulting bridge edges at inference time: starting from $A(q)$, it decides whether traversal is needed, follows a small number of selected bridges, and retrieves document-grounded facts from the resulting entity set for question answering.

3.1 Bridge Construction

The bridge set $\mathcal{E}^{\mathrm{b}}$ is not present in the document-grounded graph and must therefore be constructed from $\mathcal{G}$. We first generate a candidate set $\mathcal{C} \subseteq \mathcal{V} \times \mathcal{V}$ to avoid comparing all entity pairs. We then represent each candidate with a multi-dimensional similarity vector $\phi(u, v)$, and use the distribution of these vectors to estimate a corpus-adapted type inventory $\widetilde{\mathcal{T}}$ together with a skill $\mathcal{K}$ that describes

how those types tend to appear in ϕ-space. Finally, each candidate is labeled with $\phi(u, v)$, $\widetilde{\mathcal{T}}$, and $\mathcal{K}$ to create bridge edges. Clustering discovers recurring wording types at the corpus level; bridge creation remains a pair-level decision.

Candidate pairs and similarity vector. Most pairs in $\mathcal{V} \times \mathcal{V}$ are unrelated. We therefore construct a candidate pool $\mathcal{C}$ primarily from similar name strings and co-occurrence in the same document. When a corpus provides additional structure, optional blocking cues can be enabled (for example, conversation links across documents), but the core pool remains name- and document-based. Membership in $\mathcal{C}$ only means that a pair is worth comparing further; it does not mean the pair is a bridge. These cheap cues are coarse for the comparison because different wording relationships leave different signal patterns. We therefore represent each candidate by a multi-dimensional similarity vector

$$\phi(u, v) \in [0, 1]^d. \tag{4}$$

On our graphs, the coordinates are *name*, string similarity between entity labels; *context*, overlap between text surrounding their facts; *role_type*, overlap between extracted entity roles in text; *neighbor*, shared one-hop entities; *relation_sig*, overlap between incident relation names; *co_doc*, shared source documents; *co_section*, shared section headings; and *layout*, a corpus-structure signal that is high when a conversation links the entities' documents or when the same surface form is represented at different scopes. Keeping these coordinates separate lets later stages distinguish bridge types by their characteristic combinations: aliases tend to score high on *name*, while related-entity bridges may instead rely on *context* or *co_doc*. Domains with different document structure may use additional or alternative coordinates. How each coordinate is computed is given in Appendix B.

Corpus skill. Even with $\phi(u, v)$, we still need an inventory of plausible wording relationships and a description of how those relationships tend to look in the current corpus. A fixed inventory can capture general relations such as aliases and abbreviations, but may miss corpus-specific ways in which users refer to entities. Conversely, inventing an unconstrained relation name independently for every pair can produce unstable or fact-like labels.

We therefore begin with a shared catalog $\mathcal{T}^{\mathrm{s}}$ of wording types that are reusable across corpora, such as aliases and abbreviations, each with a short definition and a typical direction. We then adapt this inventory by looking for recurring patterns among the candidate vectors $\phi(u, v)$, allowing corpus-specific personalized types when the shared catalog is insufficient.

Specifically, we partition the candidates into K clusters:

$$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} \sum_{(u,v)\in C_k} \left\|\phi(u, v) - \mu_k\right\|_2^2, \qquad \mu_k = \frac{1}{|C_k|} \sum_{(u,v)\in C_k} \phi(u, v). \tag{5}$$

Here K sets the granularity of the discovered skill: smaller K merges distinct signal patterns, while larger K yields finer prototypes that may split one wording type across several clusters. Each centroid μ_k summarizes a recurring combination of similarity signals in the corpus. Given the cluster prototype μ_k, the meanings of the similarity dimensions, and the definitions of the existing types in $\mathcal{T}^{\mathrm{s}}$, an LLM interprets the recurring similarity pattern represented by the cluster and assigns it a wording-type name. If the pattern matches an existing type in $\mathcal{T}^{\mathrm{s}}$, the LLM reuses that type name; otherwise, it generates a short corpus-specific wording-type name that describes the pattern.

The resulting personalized type set and corpus-adapted inventory are

$$\mathcal{T}^{\mathrm{p}} = \{\tau_k\}_{k=1}^{K} \setminus \mathcal{T}^{\mathrm{s}}, \qquad \widetilde{\mathcal{T}} = \mathcal{T}^{\mathrm{s}} \cup \mathcal{T}^{\mathrm{p}}. \tag{6}$$

We organize the information used for subsequent bridge labeling into a corpus *skill* $\mathcal{K}$. For shared types in $\mathcal{T}^{\mathrm{s}}$, the skill retains their predefined type descriptions. For personalized types discovered from the current corpus, it additionally records the cluster prototypes from which they are induced:

$$\mathcal{K} = \mathcal{K}^{\mathrm{s}} \cup \mathcal{K}^{\mathrm{p}}, \qquad \mathcal{K}^{\mathrm{p}} = \{(\tau_k, \mu_k) : \tau_k \in \mathcal{T}^{\mathrm{p}}\}. \tag{7}$$

Here, $\mathcal{K}^{\mathrm{s}}$ contains the semantic descriptions of the shared bridge types, whereas $\mathcal{K}^{\mathrm{p}}$ associates corpus-specific types with the characteristic similarity patterns from which they are derived. Thus, the skill combines reusable knowledge about general wording relationships with corpus-specific patterns discovered from $\mathcal{D}$.

This step only constructs the bridge type set and the labeling skill for the KG without labeling bridge types to any candidates.

Algorithm 1 Bridge construction

Require: graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, shared type catalog $\mathcal{T}^{\mathrm{s}}$
Ensure: bridge layer $\mathcal{E}^{\mathrm{b}}$, type inventory $\widetilde{\mathcal{T}}$, skill $\mathcal{K}$

construct candidate pool $\mathcal{C}$
compute $\phi(u, v)$ for each $(u, v) \in \mathcal{C}$
cluster $\{\phi(u, v) : (u, v) \in \mathcal{C}\}$
use an LLM to assign or generate a wording type τ_k for each cluster
$\mathcal{T}^{\mathrm{p}} \leftarrow \{\tau_k\}_{k=1}^{K} \setminus \mathcal{T}^{\mathrm{s}}$
$\widetilde{\mathcal{T}} \leftarrow \mathcal{T}^{\mathrm{s}} \cup \mathcal{T}^{\mathrm{p}}$
construct $\mathcal{K}^{\mathrm{s}}$ from the descriptions of types in $\mathcal{T}^{\mathrm{s}}$
$\mathcal{K}^{\mathrm{p}} \leftarrow \{(\tau_k, \mu_k) : \tau_k \in \mathcal{T}^{\mathrm{p}}\}$
$\mathcal{K} \leftarrow \mathcal{K}^{\mathrm{s}} \cup \mathcal{K}^{\mathrm{p}}$
$\mathcal{E}^{\mathrm{b}} \leftarrow \emptyset$
for each $(u, v) \in \mathcal{C}$ **do**
    $(\tau, \mathit{dir}, c) \leftarrow \textsc{LabelBridge}(u, v, \phi(u, v), \widetilde{\mathcal{T}}, \mathcal{K})$
    if $\tau \neq \perp$ **then**
        add the directed bridge specified by (τ, dir, c) to $\mathcal{E}^{\mathrm{b}}$
    end if
end for
return $\mathcal{E}^{\mathrm{b}}$, $\widetilde{\mathcal{T}}$, $\mathcal{K}$

Bridge labeling. With $\mathcal{C}$, ϕ, $\widetilde{\mathcal{T}}$, and $\mathcal{K}$ in place, we decide which candidates become bridges. For each $(u, v) \in \mathcal{C}$, the model receives the entity pair, $\phi(u, v)$, $\widetilde{\mathcal{T}}$, and $\mathcal{K}$. The skill is soft guidance only: it summarizes how wording types tend to look in this corpus, but does not hard-assign a label to the pair. The model judges the pair against $\phi(u, v)$ and $\widetilde{\mathcal{T}}$, and may override a nearby cluster suggestion when the pair does not support a useful wording bridge. It outputs a type τ, a direction, and a traversal score c, or $\tau = \perp$ when no sufficiently meaningful wording relationship is present. We further discard predictions whose score c falls below a confidence threshold (default 0.35), so only sufficiently supported wording relations enter $\mathcal{E}^{\mathrm{b}}$. If the direction is from u to v, we add

$$b = (u, \tau, v, c) \tag{8}$$

to $\mathcal{E}^{\mathrm{b}}$. Alias-like relations are stored in both directions, whereas asymmetric relations retain the predicted direction. The score c is later used to rank competing bridges during traversal.

Overall, bridge construction can be written as

$$(\mathcal{G}, \mathcal{C}, \phi, \widetilde{\mathcal{T}}, \mathcal{K}) \;\rightarrow\; \mathcal{E}^{\mathrm{b}}. \tag{9}$$

3.2 Traversal and retrieval

The bridge layer $\mathcal{E}^{\mathrm{b}}$ is used only to expand the set of retrieval starting points. All answer evidence is still drawn from the document-grounded factual edges in $\mathcal{E}$.

Given a question q, let $A(q)$ denote the entities mentioned in the question. On a wording-gap question, the answer-bearing entity may not belong to $A(q)$, so retrieval restricted to $\mathcal{N}(A(q))$ can miss the required evidence. BridgeKG-RAG therefore decides whether to traverse beyond $A(q)$, selects a small number of bridge edges when needed, and then performs standard KG-RAG from the expanded entity set.

Traversal policy. Not every question requires traversal. If the entities mentioned in q already include the entity under which the relevant evidence is organized, remaining at $A(q)$ is sufficient. Unnecessary traversal may instead introduce additional entities and off-topic facts.

BridgeKG-RAG therefore first uses LLMs to predict

$$\pi(q) = (\alpha_q, \mathcal{T}_q), \tag{10}$$

where $\alpha_q \in \{0, 1\}$ indicates whether traversal should be performed and $\mathcal{T}_q \subseteq \widetilde{\mathcal{T}}$ is a short list of bridge types that are plausible for the wording of q. This prediction uses only q and the type defini-

Algorithm 2 Traversal and retrieval

Require: q, $\mathcal{G}^+$, $\widetilde{\mathcal{T}}$, traversal budget k
Ensure: $\hat{y}$
$A(q) \leftarrow$ entities mentioned in q
$(\alpha_q, \mathcal{T}_q) \leftarrow \textsc{PredictTraversal}(q, \widetilde{\mathcal{T}})$
if $\alpha_q = 0$ **then**
    $B(q) \leftarrow \emptyset$
    $S(q) \leftarrow A(q)$
else
    rank bridges leaving $A(q)$, preferring types in $\mathcal{T}_q$ and larger c
    $B(q) \leftarrow$ top-k ranked bridges
    $S(q) \leftarrow A(q) \cup \{v : u \rightarrow v \in B(q)\}$
end if
retrieve query-relevant facts from $\mathcal{N}(S(q))$
set X to their source spans
$\hat{y} \leftarrow g(q, X, B(q))$
return $\hat{y}$

tions in $\widetilde{\mathcal{T}}$, without inspecting factual edges in $\mathcal{E}$, so that the traversal decision is made independently of the answer evidence.

If $\alpha_q = 0$, no bridge is followed:

$$B(q) = \emptyset, \tag{11}$$

and retrieval proceeds from $A(q)$ as in standard KG-RAG.

If $\alpha_q = 1$, BridgeKG-RAG considers bridges leaving entities in $A(q)$. Because a node may have many outgoing bridges, only a small number are retained. Bridges whose type lies in $\mathcal{T}_q$ are ranked first by their traversal score c. If the traversal budget is not exhausted, the remaining slots are filled by the highest-scoring outgoing bridges regardless of type. This keeps $\mathcal{T}_q$ as a preference rather than a hard filter, allowing useful bridges to remain available even when the predicted type set is incomplete.

Let $B(q)$ denote the retained bridges. The retrieval entity set is

$$S(q) = A \cup \{v : u \rightarrow v \in B(q)\}. \tag{12}$$

The bridge layer is consulted only in constructing $S(q)$. All subsequent retrieval uses the factual graph $\mathcal{E}$.

Retrieve and answer. Retrieval is then performed from $S(q)$ rather than from $A(q)$. We collect candidate facts from $\mathcal{N}(S(q))$, retain a short query-relevant subset predicted by LLMs, and recover the document spans associated with those facts. Let X denote the resulting evidence spans.

The final answer is generated as

$$\hat{y} = g(q, X, B(q)). \tag{13}$$

The traversed bridges indicate why entities outside $A(q)$ were included in retrieval, but they are not treated as factual evidence. Any factual content in $\hat{y}$ must be supported by the document spans in X.

Without traversal, the retriever observes only $\mathcal{N}(A)$. With traversal, it observes $\mathcal{N}(S(q))$, which may include the answer-bearing neighborhood that was unreachable from the question-mentioned entities alone.

4 Experiments

We evaluate BridgeKG-RAG on GapQA, a wording-gap benchmark derived from Doc2Dial, MultiDoc2Dial, and PrivacyQA, using evidence-span recall, token F1, and answer accuracy scored by an LLM (Rajpurkar et al., 2016; Zheng et al., 2023). We compare against passage-retrieval, query-rewriting, KG-RAG, and graph-exploration baselines, and ablate bridge types and traversal choices. A parameter study, case studies, human answer audit, inference time, and standard deviations appear in Appendix E; implementation details, similarity-coordinate definitions, and prompts are in Appendices B and D.

4.1 DATASETS

We evaluate on **GapQA**, a wording-gap benchmark derived from three public datasets: Doc2Dial, MultiDoc2Dial, and PrivacyQA (Feng et al., 2020; 2021; Ravichander et al., 2019). Doc2Dial and MultiDoc2Dial contain goal-oriented dialogues grounded in government help pages, while PrivacyQA contains question–answer pairs grounded in privacy policies. For each source corpus we keep answerable examples with their gold answers and supporting evidence spans, and extract a knowledge graph $\mathcal{G}$ whose nodes are entities and whose edges are document-grounded triplets with source spans. The documents themselves are never edited.

GapQA changes only the question wording on top of these examples. Starting from an original question that already names the answer-bearing entity, we replace that entity mention with a related but different entity that a confused user might say instead, and rephrase the sentence so that it remains fluent. The substitute is chosen to look like a plausible user mix-up rather than a near-paraphrase of the same name, so lexical matching alone is unlikely to recover the original entity. Gold answers and evidence spans stay fixed; only the question wording changes, which creates a wording gap by construction. The resulting benchmark has three splits, GapQA-Doc, GapQA-Multi, and GapQA-Privacy, with 500, 500, and 257 questions (Appendix Table 3). Appendix Table 4 lists the sizes of the document graphs and bridge indexes used for each source corpus. Additional details on GapQA construction are provided in Appendix C.

4.2 BASELINES

We compare against methods that answer from parameters, from document text, from a rewritten question, or from the factual graph without a wording layer. Closed-book answers from parameters only. Closed-book (SC) draws several such samples and synthesizes one answer (Wang et al., 2023b). Text RAG retrieves supporting spans by matching the question against document text, then conditions the answer on those spans (Lewis et al., 2020). Rewrite (closed) restates the question in clearer language and answers without retrieval (Ma et al., 2023). Rewrite + KG-RAG applies the same rewrite, then runs KG-RAG (no bridging). HyDE generates a hypothetical passage, retrieves with that passage as the query, and answers the original question from the retrieved spans (Gao et al., 2023a). CRAG quality-checks an initial passage set and, when it is judged weak, retrieves a corrective rewrite and merges the two (Yan et al., 2024). KG-RAG (no bridging) collects facts from the entities mentioned in the question, with no bridge layer (Pan et al., 2024). Dense Entity Expansion uses the same mentioned entities, then adds the nearest other graph entities by embedding similarity of entity names (Sentence-BERT; top-k with k=5) before the same retrieve as KG-RAG (no bridging); Dense Entity Expansion (ctx) instead embeds each name concatenated with a short neighboring context span (Reimers & Gurevych, 2019). ToG treats the model as an agent that iteratively explores entities and relations on $\mathcal{E}$, keeps a small beam of promising paths, and only then retrieves from the entities reached along those paths (Sun et al., 2024). RoG instead asks the model to plan a sequence of relations first, executes that plan as paths on $\mathcal{E}$, and retrieves from the entities those paths visit (Luo et al., 2024). HyDE and CRAG operate on document text; Dense Entity Expansion expands seeds by vector similarity rather than typed bridges; ToG and RoG explore only factual edges, so none of them follows a wording relationship that was never written as a fact. Unless noted, text and graph baselines use the same mentioned entities as the main run, and BridgeKG-RAG uses traversal budget k=5 (Appendix E.4). BridgeKG-RAG is the full method of section 3: traverse selected bridges, then retrieve from $\mathcal{E}$.

4.3 MAIN RESULTS

Table 1 reports every method on GapQA. BridgeKG-RAG improves evidence recall and accuracy over KG-RAG without traversal on all three splits. The gain is largest on the two dialogue splits, where the named entity and the answer-bearing entity are more often of different kinds. Token F1 moves less: gold answers are short dialogue or policy replies, so once the correct span is retrieved the token overlap is already partly fixed. On GapQA-Privacy, KG-RAG (no bridging) already finds more gold spans than on the dialogue splits, which is consistent with more document overlap between wordings; bridge traversal still improves recall and accuracy.

Text RAG, HyDE, and CRAG search from the wording of the question and therefore retrieve the wrong spans (Lewis et al., 2020; Gao et al., 2023a; Yan et al., 2024). HyDE is weakest on recall: a hypothetical document written from the question still uses the name the user used, not the entity that stores the answer. CRAG can reject a weak first pass, but the corrective query is still a rewording of the original question, so it does not reach a neighborhood that those words never name. Rewrite

Table 1: Main results on GapQA. Results are averaged over five runs. Best in each column in bold. N/A: no retrieval. Standard deviations are in Table 5.

Method	GapQA-Doc			GapQA-Multi			GapQA-Privacy		
	Recall	F1	Acc.	Recall	F1	Acc.	Recall	F1	Acc.
Closed-book	N/A	0.3280	0.4540	N/A	0.3282	0.4740	N/A	0.2649	0.1673
Closed-book (SC)	N/A	0.3078	0.4420	N/A	0.3107	0.4800	N/A	0.2840	0.1984
Text RAG	0.1520	0.3635	0.3800	0.1440	0.3600	0.3900	0.1556	0.3019	0.2490
Rewrite (closed)	N/A	0.3004	0.3900	N/A	0.3031	0.4220	N/A	0.2696	0.2023
Rewrite + KG-RAG	0.3540	0.3954	0.5020	0.3040	0.3940	0.5420	0.5720	0.3527	0.3774
HyDE	0.0660	0.3424	0.3740	0.0740	0.3409	0.4220	0.1284	0.3328	0.3852
CRAG	0.1840	0.3687	0.4860	0.1800	0.3636	0.5060	0.2023	0.3227	0.4008
KG-RAG (no bridging)	0.3680	0.3975	0.5040	0.3380	0.3954	0.5580	0.5798	0.3442	0.3813
Dense Entity Expansion	0.4120	0.4028	0.5340	0.3860	0.4002	0.5680	0.6109	0.3516	0.4047
Dense Entity Expansion (ctx)	0.4280	0.4056	0.5480	0.4020	0.4038	0.5800	0.6265	0.3552	0.4163
ToG	0.3800	0.3964	0.5460	0.3720	0.4016	0.5820	0.5720	0.3489	0.4241
RoG	0.3640	0.3954	0.5620	0.3460	0.4012	0.5540	0.5642	0.3479	0.4163
BridgeKG-RAG	**0.6640**	**0.4309**	**0.6740**	**0.6460**	**0.4264**	**0.6600**	**0.7471**	**0.3831**	**0.4864**

without retrieval cannot see the documents (Ma et al., 2023), and the rewritten question answered from parameters is no more grounded than closed-book (Petroni et al., 2019). Rewrite followed by KG-RAG tracks KG-RAG (no bridging), so restating the question does not create a path that $\mathcal{E}$ does not contain. Dense Entity Expansion improves only slightly over KG-RAG (no bridging), and Dense Entity Expansion (ctx) helps only a little more: nearest neighbors remain mostly near-synonyms or same-type names, whereas GapQA frequently pairs entities of different kinds. ToG and RoG remain close to KG-RAG (no bridging): whether the model explores interactively or follows a planned relation sequence, it still walks only recorded facts and cannot cross a wording relationship that was never written as a relation (Sun et al., 2024; Luo et al., 2024). Closed-book answers can be fluent and sometimes judged plausible, yet they are not grounded, and their recall is undefined. Self-consistency among closed-book samples does not close the gap (Wang et al., 2023b), which indicates that the missing evidence is not sitting in the model's parameters. Taken together, the comparisons indicate that the failure is reachability of the answer-bearing neighborhood, not a missing paraphrase, not dense neighborhood search over entity embeddings, and not an extra hop on $\mathcal{E}$. Standard deviations for Table 1 are in Table 5 (Appendix E).

4.4 Ablations

Table 2 removes one design choice at a time from BridgeKG-RAG. *no bridges* is standard KG-RAG from the mentioned entities only. *no type preference* still traverses bridges, but samples neighbors without using $\mathcal{T}_q$; it beats *no bridges*, so some wording links help even without type guidance, yet it stays well below the full method. *no personalized bridges* keeps only shared types in $\mathcal{T}^{\text{s}}$ (alias, abbreviation, hierarchy, and other relations reused across corpora); the drop shows that local wording types, not only identity-like aliases, are needed for reachability. *no passages* reuses the same retrieved facts as BridgeKG-RAG but answers from triplets alone, so recall matches by construction. Accuracy and F1 fall, especially on GapQA-Privacy, which indicates that document spans attached to the facts are needed to write the answer, and that bridges are not a substitute for that text. Further analyses of the traversal budget k, qualitative cases, human answer audit, inference time, and evaluation on original (non-rewritten) questions are deferred to Appendix E.

5 Related Work

Passage retrieval and query rewriting. Retrieval-augmented generation answers knowledge-intensive questions by fetching external text and conditioning generation on the retrieved evidence (Lewis et al., 2020; Guu et al., 2020; Gao et al., 2023c). Open-domain systems retrieve passages from Wikipedia or domain corpora using lexical or dense similarity (Chen et al., 2017; Lee et al., 2019; Izacard & Grave, 2021; Robertson & Zaragoza, 2009; Karpukhin et al., 2020; Khattab & Zaharia, 2020; Xiong et al., 2021; Reimers & Gurevych, 2019; Ni et al., 2022). Query rewriting and

Table 2: Ablations on GapQA: each row removes one component from BridgeKG-RAG. Averaged over five runs. Best in each column in bold.

Method	GapQA-Doc			GapQA-Multi			GapQA-Privacy		
	Recall	F1	Acc.	Recall	F1	Acc.	Recall	F1	Acc.
no bridges	0.3680	0.3975	0.5040	0.3380	0.3954	0.5580	0.5798	0.3442	0.3813
no type preference	0.5020	0.4172	0.6180	0.4820	0.4114	0.6120	0.6148	0.3585	0.4241
no personalized bridges	0.4820	0.4057	0.5520	0.4700	0.4102	0.6160	0.6148	0.3718	0.4319
no passages	0.6640	0.4088	0.5360	0.6460	0.4078	0.6040	0.7471	0.2985	0.3619
BridgeKG-RAG	**0.6640**	**0.4309**	**0.6740**	**0.6460**	**0.4264**	**0.6600**	**0.7471**	**0.3831**	**0.4864**

expansion restate the question before retrieval (Ma et al., 2023; Wang et al., 2023a), while HyDE and related methods generate hypothetical text to serve as a retrieval query (Gao et al., 2023a; Yu et al., 2023). Adaptive and corrective pipelines decide when to retrieve, critique retrieved evidence, or issue follow-up searches (Asai et al., 2024; Yan et al., 2024; Jeong et al., 2024); multi-step systems interleave retrieval with reasoning (Wei et al., 2022; Trivedi et al., 2023; Press et al., 2023), and hierarchical indexes retrieve at different levels of abstraction (Sarthi et al., 2024). These approaches improve how text is matched or searched, but they still rely on the query wording, or a transformation of it, to lead retrieval toward the answer-bearing evidence. When the relevant evidence is organized under a different entity, better matching alone may not recover the required neighborhood.

Knowledge-graph RAG and graph exploration. KG-augmented answering links mentions in a question to graph entities and retrieves neighboring facts as evidence (Pan et al., 2024; Baek et al., 2023; Jiang et al., 2023a; Wu et al., 2023). Classical KGQA and multi-hop systems retrieve subgraphs or follow relation paths over large knowledge bases (Yih et al., 2015; Bordes et al., 2015; Talmor & Berant, 2018; Sun et al., 2018; 2019; Saxena et al., 2020; Jiang et al., 2023b; Lan et al., 2021; Yang et al., 2018). Recent LLM-based methods explore graph neighborhoods interactively (Sun et al., 2024; Ma et al., 2025), plan relation paths before execution (Luo et al., 2024; Chen et al., 2024), or construct query-relevant subgraphs for generation (He et al., 2024). Other approaches combine graph structured prompts with graph-based retrieval and reasoning (Yasunaga et al., 2021; Wang et al., 2024; He et al., 2024; Edge et al., 2024). These methods are effective when the answer-bearing entity is reachable through recorded graph relations. BridgeKG-RAG addresses a different case: the entity named in the question may be valid, yet the entity organizing the answer is connected to it only by a wording relationship that is absent from the factual graph.

Wording mismatch and grounding. Users and documents often refer to the same situation using different vocabulary (Furnas et al., 1987). Such mismatch is not limited to synonymy: related expressions, including metonymic references, can point to a different entity that organizes the relevant information (Lakoff & Johnson, 1980; Nissim & Markert, 2003). Entity linking and disambiguation may therefore be correct in identifying the entity explicitly mentioned by the user (Hoffart et al., 2011; Kolitsas et al., 2018; Wu et al., 2020), while retrieval from that entity still misses the answer-bearing neighborhood. We treat this mismatch as a reachability problem. Typed bridges encode wording relationships between entities and are used only to expand retrieval starting points; answer generation remains grounded in document-supported facts from the original graph.

6 Conclusion

We study the wording gap as a retrieval reachability problem: the evidence needed to answer a question may be organized under an entity that the question does not mention and that cannot be reached through recorded factual relations alone. BridgeKG-RAG addresses this problem by augmenting the factual graph with typed bridge edges that capture wording relationships between entities. These bridges are used only to expand retrieval starting points; the final answer remains grounded in document-supported facts from $\mathcal{E}$. On GapQA, constructed from Doc2Dial, MultiDoc2Dial, and PrivacyQA, BridgeKG-RAG consistently improves evidence recall and answer quality over other methods. Our ablations further show that corpus-adapted bridge types and type-aware traversal both contribute to the gains, while document spans remain necessary for grounded answer generation.

7 AI USE STATEMENT

The method, implementation, and experiments reported in this paper were designed and carried out by the authors. Generative AI tools were used only to polish the wording of the manuscript. We take responsibility for the final content of this work.

8 REPRODUCIBILITY STATEMENT

We release our codes and benchmarks. Appendix B lists model, decoding, run settings, and similarity-coordinate definitions; Appendix D lists the method prompts. Code and data will be available at: `https://anonymous.4open.science/r/BridgeKG-RAG`.

9 ETHICS STATEMENT

BridgeKG-RAG is intended to help retrieval systems reach the right document-grounded evidence when users and documents use different names. Like other retrieval-augmented systems, it may still return incomplete or misleading answers if the underlying corpus is biased, outdated, or wrongly interpreted. We therefore view the method as an aid to information access rather than a substitute for human judgment in high-stakes decisions. Deployments that serve end users should keep answers inspectable against source documents and retain appropriate human oversight.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. In *Proceedings of the First Workshop on Matching From Unstructured and Structured Data (MATCHING)*, 2023.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2017.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Advances in Neural Information Processing Systems*, 2024.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. Doc2Dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2020.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. MultiDoc2Dial: Modeling dialogues grounded in multiple documents. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2021.

G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 1987.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2023a.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2023b.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023c.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. REALM: Retrieval-augmented language model pre-training. In *Proceedings of the International Conference on Machine Learning*, 2020.

Kaiqiao Han, LuAn Tang, Renliang Sun, Peng Yuan, Wei Cheng, Haoyu Wang, Wei Wang, Yizhou Sun, and Haifeng Chen. Structure-aware RAG: Structured retrieval augmented generation from noisy data for conversational agents. *arXiv preprint arXiv:2605.24366*, 2026.

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

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2011.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, 2021.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*, 2024.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 2023.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. StructGPT: A general framework for large language model to reason over structured data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2023a.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *International Conference on Learning Representations*, 2023b.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2020.

Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. In *Proceedings of the Conference on Computational Natural Language Learning*, 2018.

George Lakoff and Mark Johnson. *Metaphors We Live By*. University of Chicago Press, 1980.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2021.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2019.

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

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations*, 2024.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. In *International Conference on Learning Representations*, 2025.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting in retrieval-augmented large language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2023.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics*, 2022.

Malvina Nissim and Katja Markert. Syntactic features and word similarity for supervised metonymy resolution. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2003.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2016.

Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. Question answering for privacy policies: Combining computational and legal perspectives. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2019.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 2009.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations*, 2024.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2020.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *International Conference on Learning Representations*, 2024.

Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2023.

Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2023a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023b.

Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. Knowledge graph prompting for multi-document question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2020.

Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. Retrieve-rewrite-answer: A KG-to-text enhanced LLMs framework for knowledge graph question answering. *arXiv preprint arXiv:2309.11206*, 2023.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Javad Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2015.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators. In *International Conference on Learning Representations*, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, 2023.

A LIMITATIONS

Our public evaluation uses English dialogue and privacy-policy corpora. We have not measured how the same bridge types transfer to other languages or to domains whose layout cues differ sharply from these sources. Building $\mathcal{E}^{\mathrm{b}}$ also incurs a one-time labeling cost over the candidate pool; that cost scales with the number of candidates retained for a corpus, though it is paid before any question is answered.

B IMPLEMENTATION DETAILS

Model and decoding. All LLM calls in BridgeKG-RAG use **GPT-4o** through a chat-completions API. This includes bridge-type discovery, pair labeling, traversal preference, answer generation, and answer scoring. Self-consistency draws several closed-book samples and synthesizes one answer.

Runs and metrics. Every number in Tables 1 and 2, and in the appendix plots and tables below, is the mean over **five independent runs** with different random seeds. We report three metrics: evidence-span recall (whether any retrieved span overlaps a gold supporting span), token-level F1 against the gold answer, and binary accuracy (an LLM scores the prediction on a 1 to 5 rubric relative to the gold answer; we count a score of at least 4 as correct). A human audit of accuracy is reported in Appendix E.2. Standard deviations for the main comparison are isolated in Table 5.

Graph and retrieval settings. Document-grounded graphs $\mathcal{G}$ are built once per source corpus from supporting documents. Table 4 reports corpus and index sizes for the graphs used in our runs. At query time we freeze the mentioned-entity set $A(q)$ across BridgeKG-RAG and graph baselines so differences come from traversal or exploration rather than linker variance. Default BridgeKG-RAG uses traversal budget $k{=}5$, prefers bridges whose types lie in $\mathcal{T}_q$, and fills remaining slots by traversal score c. Observed-edge retrieval takes a short lexical shortlist of facts per seed entity, optionally one factual hop, then an LLM filter before recovering source spans. Answers are generated from those spans. Traversed bridges may be listed as wording hypotheses, but they are never treated as facts.

Table 3: GapQA splits. Questions replace the answer-bearing entity with a related mention and are rephrased for fluency; gold answers and evidence spans are unchanged.

Split	**Source corpus**	**Size**	**Domain**
GapQA-Doc	Doc2Dial	500	Goal-oriented dialogue
GapQA-Multi	MultiDoc2Dial	500	Multi-document dialogue
GapQA-Privacy	PrivacyQA	257	Privacy-policy QA

Table 4: Graph and index statistics for the three source corpora. $|\mathcal{V}|$ and $|\mathcal{E}|$ are entities and factual edges; $|\mathcal{C}|$ is the bridge-candidate pool; $|\mathcal{E}^{\mathrm{b}}|$ is the retained bridge layer; $|\widetilde{\mathcal{T}}|$ is the number of corpus-adapted bridge types. The entity linker indexes all entity names in $\mathcal{V}$.

Corpus	**Docs**	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{E}^{\mathrm{b}}\rvert$	$\lvert\widetilde{\mathcal{T}}\rvert$
Doc2Dial	473	3,916	11,482	26,740	3,854	22
MultiDoc2Dial	480	4,508	13,965	31,128	4,672	25
PrivacyQA	35	1,847	5,216	9,640	1,928	17

Similarity coordinates. Each candidate pair (u, v) is represented by $\phi(u, v) \in [0, 1]^8$. We first build a per-entity profile from $\mathcal{G}$: the entity label; soft role types from extraction metadata; incident neighbors and relation names; source document IDs; section headings; and context tokens drawn from the text attached to the entity's facts. Coordinates are then computed as follows. *name* is

the maximum of (i) SequenceMatcher ratio on lowercased labels, (ii) Jaccard overlap of character trigrams, (iii) Jaccard overlap of whitespace tokens, and (iv) a containment bonus when one label string contains the other. *context*, *role_type*, *neighbor*, and *relation_sig* are Jaccard overlaps of context-token sets, soft role-type sets, one-hop neighbor sets, and incident relation-name sets, respectively. *co_doc* is $|\mathrm{docs}(u) \cap \mathrm{docs}(v)| / \min(|\mathrm{docs}(u)|, |\mathrm{docs}(v)|)$ when both entities have documents, and 0 otherwise. *co_section* is 1 if the entities share a section heading, else the Jaccard overlap of section-heading tokens. *layout* (called `bridge` in the implementation) is high when a dialogue links the entities' documents across turns, or when one entity is a scoped instance and the other a shared surface form of the same concept (matched by surface key, weak name similarity, or weak context overlap).

Baselines. Text RAG, HyDE, and CRAG retrieve document passages rather than graph facts. Dense Entity Expansion expands KG seeds by embedding similarity of entity names, then uses the same retrieve as KG-RAG (no bridging); Dense Entity Expansion (ctx) embeds names concatenated with a short neighboring context span. Rewrite (closed) and Rewrite+KG-RAG share the same rewrite model; the latter then runs KG-RAG (no bridging). ToG and RoG explore only $\mathcal{E}$. Entities they reach are fed into the same retrieve-and-answer stack used by KG-RAG (no bridging), so the comparison isolates exploration of facts versus traversal of bridges.

C GapQA Construction

GapQA is constructed from Doc2Dial, MultiDoc2Dial, and PrivacyQA. We begin with examples for which the original dataset provides a valid answer and supporting evidence. The underlying documents, gold answers, and supporting evidence are kept unchanged throughout construction; only the question is modified.

Candidate construction. For each retained example, we identify the answer-bearing entity explicitly expressed in the original question and construct a wording-gap variant around a related but distinct entity. The substitute entity is chosen to reflect a plausible way in which a user might refer to the same underlying information need without naming the entity under which the answer is organized. Candidate questions are then rewritten so that the substitute entity is expressed naturally in context while the original answer-bearing entity is no longer explicitly mentioned. Candidate rewrites are produced through a controlled model-assisted rewriting procedure using the original question, answer-bearing entity, and supporting context.

Acceptance criteria. A candidate is retained only when it satisfies all of the following conditions:

1. **Information-need preservation:** the rewritten question expresses the same underlying information need as the original question;
2. **Entity displacement:** the original answer-bearing entity is no longer explicitly named in the rewritten question;
3. **Plausible wording gap:** the substitute entity is a plausible user expression associated with the intended information need rather than an arbitrary perturbation;
4. **Answer preservation:** the original gold answer remains valid;
5. **Evidence preservation:** the original supporting evidence remains sufficient to answer the rewritten question.

We discard candidates that reduce to aliases, trivial renamings, or near-paraphrases of the original answer-bearing entity, as these cases can typically be resolved through surface-form matching alone.

GapQA is constructed independently of the bridge layer learned by BridgeKG-RAG. In particular, predicted bridge edges, bridge confidence scores, learned corpus skills, and query-time traversal decisions are not used to generate or filter benchmark examples. The knowledge graph used by the evaluated systems is constructed only after the benchmark questions have been fixed. This separation prevents benchmark membership from being determined by the bridge structures produced by the proposed method and avoids circular evaluation.

Dataset statistics. The final GapQA benchmark contains 1,257 examples: 500 derived from Doc2Dial, 500 from MultiDoc2Dial, and 257 from PrivacyQA.

D PROMPT TEMPLATES

We list the prompts that implement the method stages in section 3. Placeholders such as {taxonomy} and {skill_block} are filled at runtime with the shared type catalog, corpus skill summaries, or available bridge types. Prompts return strict JSON, except answer generation, which returns plain text. Long schema lines are broken for display; the model still receives a single JSON object.

Corpus skill (cluster to type)

Task. Discover likely bridge-type patterns for dataset={dataset}.

Scope. This step is exploratory only. Clustering and the assignment suggest which wording classes may appear. They do not label individual pairs; final pair decisions happen later.

Constraint. Bridge edges are query-time wording or organization links. They are not factual KG triplets.

Inputs. Shared type catalog in {taxonomy}.

Per cluster. (1) Prefer a catalog type whose typical high dimensions match the cluster mean scores. (2) Multiple clusters may share one type. (3) If the catalog is insufficient, propose a short reusable snake_case type and a one-sentence definition.

Output JSON schema.

```
{"assignments":[{"cluster_id":"c0",
  "type":"alias_of","is_novel":false,
  "definition":"...","confidence":0.0,
  "rationale":"..."}]}
```

Bridge labeling (pair level)

Task. Label bridge edges for dataset={dataset}. This step is authoritative for each candidate pair. The skill or cluster card is only soft guidance.

Goal. Find substitutable wordings for user questions: related, metonymic, or organizational phrases a user might say instead of the precise term. These edges create solvable vagueness at query time. They are not factual KG triplets. Reject weak co-occurrence that would not help rewrite or traverse a question.

Inputs. Type catalog in {taxonomy}; soft skill guidance in {skill_block} (override freely per pair).

Per pair. Prefer a catalog type when it fits; otherwise invent a short reusable type or return none. Direction is undirected, a_to_b, or b_to_a. Confidence is in $[0, 1]$.

Output JSON schema.

```
{"relations":[{"pair_index":1,
  "relation":"none|type","is_novel":false,
  "direction":"undirected","confidence":0.0,
  "rationale":"..."}]}
```

Traversal preference $\pi(q)$

Task. Decide whether a user question needs a bridge traverse before KG retrieval.

Meaning of a bridge. A bridge is a query-time wording or organization link between entities, not a factual triplet. Traversing means: if the question mentions a surface entity, follow one bridge hop to a more answer-bearing related entity, then retrieve factual edges from the expanded set.

Visibility. The model does not see KG triplets here. It sees only the question and bridge-type information.

Inputs. Available types in {available}; type notes in {type_notes}.

Decide. needs_traverse (true/false); ordered preferred_bridge_types (at most {top_k}); short query_surface_cues.

Output JSON schema.

```
{"needs_traverse":true,
```

Table 5: Standard deviations (std only) across five runs for the means in Table 1. N/A: no retrieval.

Method	GapQA-Doc			GapQA-Multi			GapQA-Privacy		
	Recall	F1	Acc.	Recall	F1	Acc.	Recall	F1	Acc.
Closed-book	N/A	0.008	0.010	N/A	0.009	0.012	N/A	0.008	0.013
Closed-book (SC)	N/A	0.014	0.018	N/A	0.016	0.023	N/A	0.018	0.022
Text RAG	0.004	0.005	0.010	0.004	0.006	0.010	0.005	0.007	0.013
Rewrite (closed)	N/A	0.007	0.013	N/A	0.009	0.012	N/A	0.010	0.018
Rewrite + KG-RAG	0.006	0.005	0.010	0.007	0.005	0.009	0.009	0.007	0.013
HyDE	0.004	0.006	0.015	0.005	0.007	0.014	0.006	0.008	0.018
CRAG	0.008	0.006	0.012	0.009	0.006	0.011	0.009	0.008	0.016
KG-RAG (no bridging)	0.003	0.003	0.009	0.003	0.004	0.010	0.004	0.005	0.012
Dense Entity Expansion	0.005	0.004	0.008	0.006	0.004	0.009	0.006	0.004	0.010
Dense Entity Expansion (ctx)	0.005	0.004	0.009	0.006	0.004	0.009	0.006	0.005	0.011
ToG	0.013	0.007	0.013	0.013	0.007	0.015	0.012	0.009	0.021
RoG	0.009	0.007	0.015	0.013	0.007	0.015	0.012	0.007	0.017
BridgeKG-RAG	**0.007**	**0.006**	**0.009**	**0.007**	**0.006**	**0.011**	**0.009**	**0.006**	**0.013**

```
"preferred_bridge_types":["..."],
"query_surface_cues":["..."],
"rationale":"..."}
```

Answer generation

Task. Answer as a helpful agent using only the supplied evidence passages.

Wording. The question may be vague or use related wording. When bridge mappings are listed, each is a candidate reading from a surface mention to a related entity. These are hypotheses, not facts.

Procedure. Form a small set of plausible intents consistent with the question and the bridges. Prefer the intent that the evidence best supports. Do not invent policy facts beyond the evidence.

Output. Plain answer text only, in one to four sentences.

E Additional experimental results

E.1 Standard deviations

Table 1 reports five-run means. Table 5 reports the corresponding sample standard deviations across the five runs, for recall, token F1, and accuracy. N/A marks methods that do not retrieve evidence, so recall is undefined.

E.2 Human answer audit

We also run a small human audit on the same model answers used in Table 1. For each GapQA split we sample 100 questions uniformly at random and compare only *KG-RAG (no bridging)* against *BridgeKG-RAG*. Annotators see the question, the gold answer, and one system answer at a time (method identity hidden), and mark the answer as correct if it is factually consistent with the gold answer and would satisfy the information need; otherwise incorrect. Table 6 reports human accuracy (percent correct). The ranking and margins match the automatic accuracy in Table 1: BridgeKG-RAG remains clearly ahead of no bridging on all three splits.

Table 6: Human accuracy (%) on 100 sampled questions per split. Annotators judge each system answer against the gold answer with method identity hidden.

Method	GapQA-Doc	GapQA-Multi	GapQA-Privacy
KG-RAG (no bridging)	50	52	38
BridgeKG-RAG	**66**	**68**	**47**

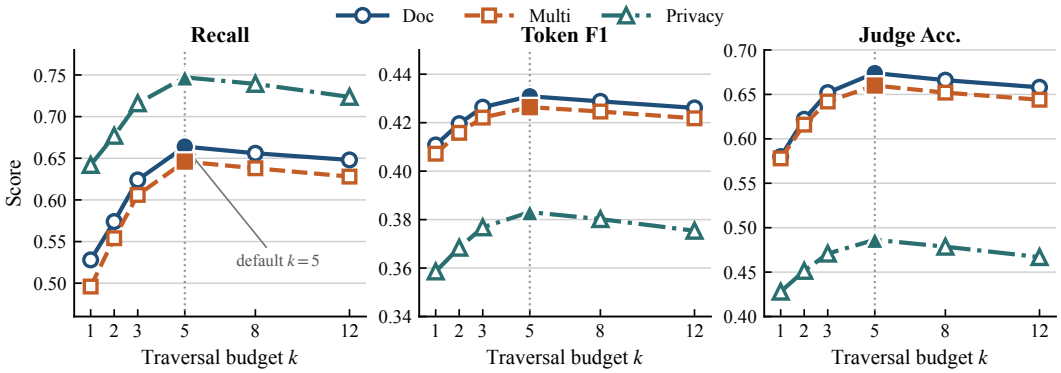

Figure 3: Effect of traversal budget k. Each panel is one metric (recall, token F1, accuracy). Curves are GapQA-Doc / Multi / Privacy. Filled markers and the dotted line mark the default k=5, which is the shared peak.

E.3 Inference time

Under the default evaluation settings (GPT-4o API, k=5), answering one GapQA question with BridgeKG-RAG takes roughly 7–9 seconds from question input to the generated answer. Almost all of this cost comes from sequential LLM calls rather than from local graph operations. The dominant cost is retrieve-and-answer: collecting candidate facts from $\mathcal{N}(S(q))$, filtering them with an LLM, and generating $\hat{y}$ from the recovered spans.

E.4 Traversal budget k

We vary the maximum number of bridges followed per question while keeping type preference and retrieval fixed. Figure 3 plots each metric against $k \in \{1, 2, 3, 5, 8, 12\}$. The figure has three panels for recall, token F1, and accuracy. Inside each panel, the three curves are the GapQA splits (Doc, Multi, Privacy), so domain trends are comparable on a shared vertical scale. Even k=1 already exceeds KG-RAG (no bridging) on every split. Scores rise through k=5 as more answer-bearing entities enter $S(q)$, then dip slightly at k=8 and k=12 as extra endpoints add off-topic facts.

E.5 Case studies

Figure 4 walks through GapQA examples with the original system answers (not paraphrases). Each box states the wording gap explicitly: which entity the question names, which entity holds the gold evidence, and that $\mathcal{E}$ has no factual edge between them. Without traversal, retrieval stays at the named entity and either abstains or answers from the wrong neighborhood. With traversal, one bridge reaches the answer-bearing entity, and the reply is written from that entity's source spans. The bridge is used only to choose where to start retrieval. It never appears as evidence.

E.6 Original questions

Table 7 compares KG-RAG (no bridging) and BridgeKG-RAG on the original questions in Doc2Dial, MultiDoc2Dial and PrivacyQA, before rewriting into GapQA. Here a wording gap is

Case 1 (GapQA-Doc)

Question. *I need to know how to pay the search fee.*

Wording gap. Named entity: `search fee`. Answer-bearing entity: `DIAL-IN Search Account fees`. The deposit-and-balance payment rule sits under the account-fee entity; the graph has no factual edge from the bare fee mention to that account.

Bridge used. `search fee` $\xrightarrow{documented_under}$ `DIAL-IN Search Account fees`

With traverse. To pay the search fee, you need to include a deposit with your application for a search account. Your deposit will be added to your account balance, and the search fee will be subtracted from this balance when you make a search. Ensure your first deposit is enough to cover at least two months of searches. (correct: deposit covering at least two months of searches)

Without traverse. To pay the search fee, you can use your account balance for mail-order and online searches. Alternatively, you can pay by personal check or money order made payable to the "Commissioner of Motor Vehicles." Please note that payment by credit card is not available at this time. (incorrect: fluent but wrong payment path from the wrong neighborhood)

Case 2 (GapQA-Multi)

Question. *Great. And for the Child Disability Report I need an appointment, correct?*

Wording gap. Named entity: `Child Disability Report`. Answer-bearing entity: `SSI application`. The appointment rule is written under the SSI application document, and $\mathcal{E}$ has no factual link from the report name to that application.

Bridge used. `Child Disability Report` $\xrightarrow{cross_document_context}$ `SSI application`

With traverse. Yes, you need an appointment to complete the SSI application, which includes the Child Disability Report. You can schedule this by phone or in person. (correct: yes, grounded in the SSI application neighborhood)

Without traverse. No, an appointment is not required to complete the Child Disability Report. However, scheduling one may reduce your waiting time if you choose to visit a Social Security office or speak to a representative. (incorrect: yes/no flip; answers from the report neighborhood alone)

Figure 4: GapQA cases. Each box names the wording gap, the bridge that closes it, and the original with/without-bridging answers.

Table 7: Comparison on the original questions (samples not rewritten into GapQA).

Source	KG-RAG (no bridging)			BridgeKG-RAG		
	Recall	F1	Acc.	Recall	F1	Acc.
Doc2Dial	0.6380	0.3284	0.6320	0.6520	0.3312	0.6400
MultiDoc2Dial	0.6960	0.3300	0.6540	0.7080	0.3331	0.6620
PrivacyQA	0.3463	0.2334	0.3346	0.3611	0.2386	0.3424

not forced: the question may already name the answer-bearing entity, so traversal is optional. BridgeKG-RAG is still slightly better on every source corpus and every metric. The gains are small compared with GapQA, which fits the claim that traversal mainly helps when the named entity is not already the answer-bearing one, while remaining harmless when the question is well aligned.