Walk&Retrieve: Simple Yet Effective Zero-Shot Retrieval-Augmented Generation via Knowledge Graph Walks

Martin Böckling[1], Heiko Paulheim[1] and Andreea Iana[1,*]

[1]Data and Web Science Group, University of Mannheim, Germany

Abstract

Large Language Models (LLMs) have showcased impressive reasoning abilities, but often suffer from hallucinations or outdated knowledge. Knowledge Graph (KG)-based Retrieval-Augmented Generation (RAG) remedies these shortcomings by grounding LLM responses in structured external information from a knowledge base. However, many KG-based RAG approaches struggle with (i) aligning KG and textual representations, (ii) balancing retrieval accuracy and efficiency, and (iii) adapting to dynamically updated KGs. In this work, we introduce `Walk&Retrieve`, a simple yet effective KG-based framework that leverages walk-based graph traversal and knowledge verbalization for corpus generation for zero-shot RAG. Built around efficient KG walks, our method does not require fine-tuning on domain-specific data, enabling seamless adaptation to KG updates, reducing computational overhead, and allowing integration with any off-the-shelf backbone LLM. Despite its simplicity, `Walk&Retrieve` performs competitively, often outperforming existing RAG systems in response accuracy and hallucination reduction. Moreover, it demonstrates lower query latency and robust scalability to large KGs, highlighting the potential of lightweight retrieval strategies as strong baselines for future RAG research.

Keywords

Knowledge Graph Retrieval-Augmented Generation, Graph Walks, Zero-Shot Retrieval, Question Answering

1. Introduction

Large Language Models (LLMs) are pivotal to question answering (QA) due to their strong language understanding and text generation capabilities [1, 2, 3, 4, 5, 6]. However, LLMs often (i) struggle with outdated knowledge, (ii) lack interpretability due to their black-box nature [7], and (iii) can hallucinate convincingly yet factually inaccurate answers [8, 9, 10]. These issues are particularly pronounced in knowledge-intensive tasks [11], when dealing with domain-specific [12, 13] or rapidly changing knowledge [14]. Retrieval-augmented generation (RAG) mitigates these limitations by grounding responses in relevant external information [15, 16, 17]. Yet, text-based RAG primarily relies on semantic similarity search of textual content [17], which fails to capture the relational knowledge necessary to integrate passages with large semantic distance from the query in multi-step reasoning [18, 19, 20, 21, 22].

Consequently, several works leverage knowledge graphs (KGs) – structured knowledge bases representing real-world information as networks of entities and relations [23] – as external information sources to overcome standard RAG limitations [21]. Given a query, KG-based RAG systems retrieve relevant facts as nodes, triplets, paths, or subgraphs using graph search algorithms, or parametric retrievers based on graph neural networks or language models [21]. The retrieved graph data is then reformatted for the language model – via linearized triples [24], natural language descriptions [25, 26, 27, 28, 29], code-like forms [30], or node sequences [31, 32, 33] – and finally used by an LLM to generates the final response [21].

The existing body of work exhibits several drawbacks. First, augmenting a query with relevant KG triples [34, 35, 36] can lead to suboptimal retrieval performance due to the misalignment of structured

IR-RAG 2025: Information Retrieval's Role in RAG Systems, July 17, 2025, Padua, Italy

*Corresponding author.

martin.boeckling@uni-mannheim.de (M. Böckling); heiko.paulheim@uni-mannheim.de (H. Paulheim); andreea.iana@uni-mannheim.de (A. Iana)

0000-0002-1143-4686 (M. Böckling); 0000-0003-4386-8195 (H. Paulheim); 0000-0002-7248-7503 (A. Iana)

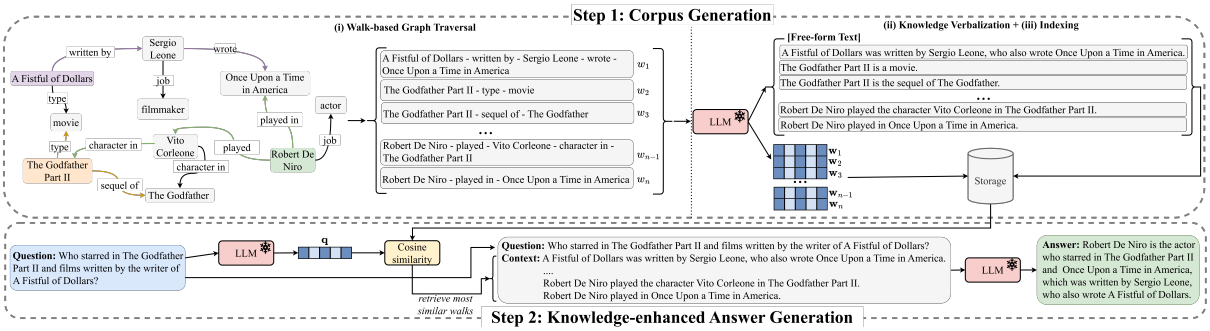

Figure 1: Overview of the `Walk&Retrieve` framework: (1) We combine walk-based graph traversal with knowledge verbalization for corpus generation; (2) The answer is generated with a prompt augmenting the query with the most similar verbalized walks.

graphs and the sequential token-based nature of the language model. Although converting KG data to a LLM-suitable tokenized format can help, naive triple linearization [37, 38], which directly converts KG triples into plain text without considering context, coherence, or structural nuances, often produces semantically incoherent descriptions [39].[1] Second, RAG systems that directly reason over KGs with LLMs perform a step-by-step graph traversal for fact retrieval [20, 32, 22]. This requires multiple LLM calls per query, significantly increasing complexity and latency. Third, KG-based RAG models often fine-tune retrievers [25, 40, 31] or generators [41, 37, 42, 31, 33] on task-specific data to better adapt to diverse KG structures and vocabularies. However, collecting high-quality instruction data is costly [43], and fine-tuning large models – even with parameter-efficient methods [44, 45, 46] – is expensive and limits generalization to dynamic KGs or unseen domains [36, 25].

Contributions. We propose `Walk&Retrieve`, a lightweight zero-shot KG-based RAG framework, designed as a simple yet competitive baseline to address these challenges. It combines efficient graph traversal, via random or breadth-first search walks, with verbalization of KG-derived information to build a contextual corpus of relevant facts for each KG entity. At inference, we retrieve the most similar nodes to the query, and their corresponding walks, respectively. We generate the final answer by prompting an LLM with the query, augmented with this relevant context. Unlike many existing KG-based RAG systems, `Walk&Retrieve`: **(1)** is *adaptable* to dynamic KGs – updates (e.g., node insertion or deletion) require no retraining, as new knowledge can be added by incrementally generating additional walks; **(2)** is more *efficient*, requiring no fine-tuning of the backbone LLM, and only a single LLM call per query; **(3)** enables *zero-shot RAG* with any *off-the-shelf LLM*. We show that `Walk&Retrieve` consistently generates accurate responses, while minimizing hallucinations. Our findings render walk-based corpus generation as a promising approach for scalable KG-based RAG, and establish `Walk&Retrieve` as a strong baseline for future research.

2. Methodology

Fig. 1 illustrates our proposed framework, comprising two stages: corpus generation and knowledge-enhanced answer generation.

2.1. Corpus Generation

In the first stage, we leverage the knowledge stored in KGs to construct a corpus of relevant facts. A Knowledge Graph is defined as $G = (V, E, R)$, where V denotes a set of nodes $v \in V$, and $E \subseteq V \times R \times V$ a set of directed edges labeled with relation types from the set R. For each node $v \in V$, we define its

[1] Given the triples: `A Fistful of Dollars` → `writtenBy` → `Sergio Leone`, and `The Godfather Part II` → `sequelOf` → `The Godfather`, a prompt based on naive linearization would be: *These facts might be relevant to answer the question: (A Fistful of Dollars, writtenBy, Sergio Leone), (The Godfather Part II, sequelOf, The Godfather)*[...].

System: Please provide me from an extracted triple set of a Knowledge Graph a sentence. The triple set consists of one extracted random walk. Therefore, a logical order of the shown triples is present. Please consider this fact when constructing the sentence. Prevent introduction words.
Human: Please return only the constructed sentence from the following set of node and edge labels extracted from the Knowledge Graph: {triples}.

(a) Knowledge verbalization.

System: You are provided with context information from a RAG retrieval, which gives you the top k context information. Please use the provided context information to answer the question. If you are not able to answer the question based on the context information, please return the following sentence: "I do not know the answer".
Human: Please answer the following question: {question}. Use the following context information to answer the question: {context}.

(b) Knowledge-enhanced answer generation.

Figure 2: Prompt templates used for knowledge verbalization and answer generation.

neighbor set as $N(v) := \{v' : \exists r \in R | (v, r, v') \in E\}$. Corpus generation consists of walk-based graph traversal, knowledge verbalization, and indexing.

Walk-based Graph Traversal. We extract relevant facts for all entities in the KG using two walk-based graph traversal approaches.

Random Walks (RW). In this method, we retrieve facts for a given vertex $v \in V$ by generating n_w graph walks $\mathscr{W}_l$ of length l rooted in v. A random walk is a stochastic process with variables $X_0, X_1, X_2, ..$, where each $X_t \in V$ denotes the vertex visited at time t [47]. At each step, when the random walker is at vertex v_i, it chooses the next node uniformly at random from one of its neighbors $v_j \in N(v_i)$ according to the following transition probability:

$$P(X_{t+1} = j | X_t = i) = \begin{cases} \frac{1}{|N(v_i)|} & \text{if}(v_i, r, v_j) \in E \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $|N(v_i)|$ denotes the neighborhood size of v_i. Finally, the graph corpus is obtained by aggregating n_w random walks $\mathscr{W}_l = (X_0, r_i, X_1, ..., r_k, X_l), r \in R$ per vertex, as $\mathscr{C}_{RW} = \bigcup_{i=1}^{|V|} \bigcup_{j=1}^{n_{walks}} \mathscr{W}_l$.

Breadth-First Search (BFS) Walks. In this approach, we construct a spanning tree for each entity in G using the BFS algorithm. For a given root $v_r \in V$, we build walks by partitioning the reachable nodes v_j into layers L_i based on their shortest-path distance to the root [48]. Starting with $L_0 = \{v_r\}$, layers are recursively defined as

$$L_{i+1} = \{v_j \in V \setminus \cup_{k=0}^{i} L_k : \exists v_i \in L_i | (v_i, r, v_j) \in E\} \tag{2}$$

for $i \in [0, d]$, where d is the maximum depth (i.e., the maximum allowed shortest-path distance). This guarantees that each vertex is explored only once per search. Hence, the resulting corpus $\mathscr{C}_{BFS} = \bigcup_{i=1}^{|V|} L_i$ contains only non-duplicate walks for each vertex. We note that the maximum allowed shortest-distance path d of the BFS walks is equivalent to the length l of the randomly generated walks.

Knowledge Verbalization. As LLMs require textual inputs, we convert the extracted walks for each entity in G into free-form textual descriptions, to enable knowledge-enhanced reasoning for answer generation. In contrast to recent works that fine-tune an LLM on question-answer pairs to learn a graph-to-text transformation [25], we directly prompt the LLM – using the prompt template shown in Fig. 2a – to provide a natural language representation of the walks, obtaining the verbalized corpus. This approach aligns the KG-derived information with the LLM's representation space, while preserving the order of the nodes and edges in the walks. Moreover, by not fine-tuning the LLM, we (i) eliminate the need for labeled graph-text pairs, (ii) improve generalization to unseen KGs, and (iii) enable the usage of any LLM in the knowledge verbalization step.

Indexing. Lastly, we index the graph for efficient retrieval. After knowledge verbalization, each walk w_i^v of vertex v is converted into a vector $\mathbf{w}_i^v$. Moreover, we compute each node's global representation from

the concatenation of its respective walks. We store the embeddings of all nodes and corresponding walks to facilitate efficient retrieval during inference. Crucially, our walk-based corpus generation renders `Walk&Retrieve` highly adaptable to dynamic KGs: updates (deletions, modifications, or additions of nodes and edges) require recomputing only the walks involving the changed graph elements – a much smaller subset than the entire corpus.

2.2. Knowledge-enhanced Answer Generation

Given a query q, we encode it with the same LLM used for knowledge verbalization, so that the query and the retrieved facts share the same vector space. We then perform a k-nearest neighbor search to retrieve the k most similar nodes in G to q and, for each node, the k most relevant verbalized walks. Concretely, we define the sets of relevant nodes V_k and corresponding walks W_k based on the cosine similarity between the embeddings of the query $\mathbf{q}$ and each node $\mathbf{v}$ or walk $\mathbf{w}^v$, respectively. To this end, we compute:

$$\begin{aligned} V_k &= argtopk_{v \in V} cos(\mathbf{q}, \mathbf{v}) \\ W_k &= \bigcup_{v_k \in V_k} argtopk_{w^{v_k} \in \mathscr{C}} cos(\mathbf{q}, \mathbf{w}^{v_k}), \end{aligned} \tag{3}$$

where $\mathscr{C} \in \{\mathscr{C}_{RW}, \mathscr{C}_{BFS}\}$, and the *argtopk* operation retrieves the k nodes with the highest cosine similarity to the query. For zero-shot inference, we design a prompt that integrates the query q with the relevant context W_k, cf. template from Fig. 2b. Importantly, we instruct the LLM to refrain from responding if the context is insufficient, thereby grounding responses in the extracted structured knowledge, and reducing hallucinations. Finally, the prompt is fed into the previously used LLM to generate a response. By avoiding LLM fine-tuning, we reduce computational costs and eliminate the need for task-specific training data. Moreover, we reduce inference latency as `Walk&Retrieve` uses a single call to the LLM per query.[2]

3. Experimental Setup

Baselines. We compare `Walk&Retrieve` against three kinds of baselines: standard LLM, text-based RAG, and KG-based RAG. With *LLM only*, we test whether the LLM can answer questions without external data. For *Vanilla RAG*, following [34], we uniformly sample 5 triples from all 1-hop facts of the question entities. We consider two KG-RAG models. *SubgraphRAG* [36] retrieves subgraphs using a MLP and parallel-triple scoring; the LLM then reasons over the linearized triples of the subgraph to generate a response. *RetrieveRewriteAnswer* [25] uses constrained path search and relation path prediction for subgraph retrieval, which it then converts into free-form text to augment the prompt for response generation.

Data. We conduct experiments on MetaQA [49] and CRAG [50]. MetaQA [49] is a knowledge base QA benchmark, with over 400K questions (single- and multi-hop), and a KG containing 43K entities and 9 relation types. We use all its 1-hop, 2-hop, and 3-hop subsets with the "vanilla" question version. CRAG [50] is a factual QA benchmark for RAG, featuring over 4.4K question-answer pairs across five domains and eight question categories. It provides mock KGs with 2.6 million entries.[3] Table 1 summarizes their statistics.

Evaluation Metrics. We follow prior work [51, 52, 36] and use Hits@1 to measure if a response includes at least one correct entity. Additionally, we adopt the model-based evaluation setup of Yang et al. [50] to assess the quality of the generated answers using a three-way scoring system: *accurate* (1), *incorrect* (-1), or *missing* (0). Exact matches are labeled *accurate*; all others are evaluated with two LLMs, `gpt-4-0125-preview` [53] and `Llama-3.1-70B-instruct` [54], to mitigate self-preference [55].

[2]Note that the preprocessing step's computational overhead is a *one-time* cost, as subsequent graph changes require only incremental, inexpensive updates to the corpus.

[3]In our experiments, we use the public test set of CRAG.

Table 1
Statistics of MetaQA [49] and CRAG [50] test sets.

	MetaQA			CRAG
	1-hop	2-hop	3-hop	
# Question types	13	21	15	8
# Questions	9,947	14,872	14,274	1,335

Table 2
Question-answering performance. We report numbers in percentage, and the query runtime in seconds. For MetaQA, we average results over its k-hop subsets. The best results per column are highlighted in bold, the second best underlined.

		MetaQA					CRAG				
Baseline Type	**Model**	Hits@1 ↑	Accuracy ↑	Hallucination ↓	Missing ↓	Time (s) ↓	Hits@1 ↑	Accuracy ↑	Hallucination ↓	Missing ↓	Time (s) ↓
LLM only	Direct	30.37	31.79	18.86	61.89	**13.03**	11.05	9.31	23.95	67.49	**14.14**
Text-based RAG	Vanilla RAG	25.08	14.73	13.52	65.70	22.11	15.21	16.94	**19.53**	**51.39**	26.01
KG-based RAG	SubgraphRAG	43.88	41.17	18.08	32.53	23.12	–	–	–	–	–
	RetrieveRewriteAnswer	47.49	34.01	22.92	32.12	22.37	–	–	–	–	–
	`Walk&Retrieve-RW`	55.60	41.11	15.31	37.13	22.12	19.31	19.40	19.64	51.94	22.15
	`Walk&Retrieve-BFS`	**67.99**	**57.08**	**12.74**	**28.27**	21.31	**21.31**	**21.53**	23.01	53.40	23.34

We report averages of *accurate*, *hallucinated*, and *missing* responses, and the overall *truthfulness* (i.e., accuracy minus hallucination) from the LLM evaluators.

Implementation Details. We retrieve $k = 3$ similar nodes and walks, respectively, for answer generation.[4] Our main experiments use `Llama-3.1-70B-instruct` [54] with temperature $t = 0$ and speculative decoding for all models. We perform 60 walks for random-walk corpus generation. For both `Walk&Retrieve` model variants, we use walks of depth 4 on MetaQA and 3 on CRAG. We train and evaluate the baselines using their official implementations, and conduct all experiments on two NVIDIA A6000 48 GB GPUs.[5]

4. Results and Discussion

Table 2 summarizes the QA performance of `Walk&Retrieve` and the baselines with `Llama-3.1`. On MetaQA, `Walk&Retrieve-BFS` consistently outperforms all other models in answer accuracy and Hits@1, achieving a relative improvement of 38.64% over the best baseline (SubgraphRAG). While other KG-based RAG systems yield high accuracy, they tend to hallucinate more than the simpler LLM-only and Vanilla RAG systems, which often produce no answer rather than an incorrect one. In contrast, `Walk&Retrieve-BFS` minimizes both hallucinations and missing responses. Although LLM-only has the lowest query latency due to the absence of a retrieval step, `Walk&Retrieve` achieves the fastest inference time per query among all RAG approaches, underscoring its efficiency. Fig. 3 breaks down MetaQA performance by number of hops. LLM-only and Vanilla RAG fail to answer over 60% of 2- and 3-hop questions. Both SubgraphRAG and RetrieveRewriteAnswer lower the missing rate below 35% across hops, although truthfulness remains under 25%. Conversely, `Walk&Retrieve-BFS` better trades off accuracy and hallucination (55%+ truthfulness for 1-hop and 37%+ for 2- and 3-hop questions), while greatly reducing non-responses.

On CRAG, both `Walk&Retrieve` variants outperform LLM-only and Vanilla RAG in answer accuracy, while matching them in hallucination and missing rates. Note that, SubgraphRAG and RetrieveRewriteAnswer could not be evaluated on CRAG due to scalability and computational constraints.[6] These results highlight the scalability of our walk-based corpus generation approach, which limits traversal to small-hop neighborhoods rather than the full graph. While performance drops on CRAG,

[4] In preliminary experiments with $k \in [1, 5]$, we found $k = 3$ to be the optimal value that balances accuracy and hallucination.
[5] Code available at https://github.com/MartinBoeckling/KGRag
[6] SubgraphRAG fails to scale to CRAG's KG (over 1 million edges), and RetrieveRewriteAnswer requires fine-tuning the backbone LLM beyond our available resources.

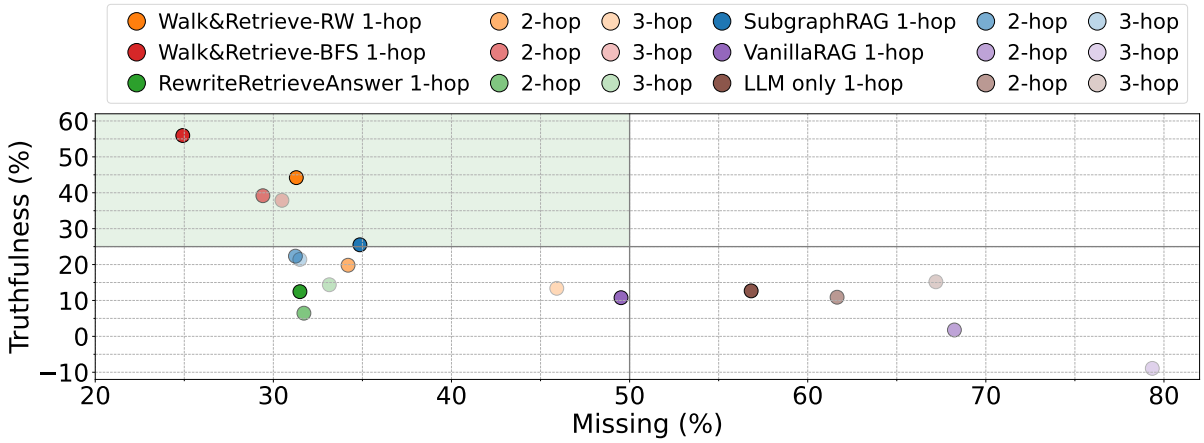

Figure 3: Missing vs. truthfulness rates over MetaQA subsets.

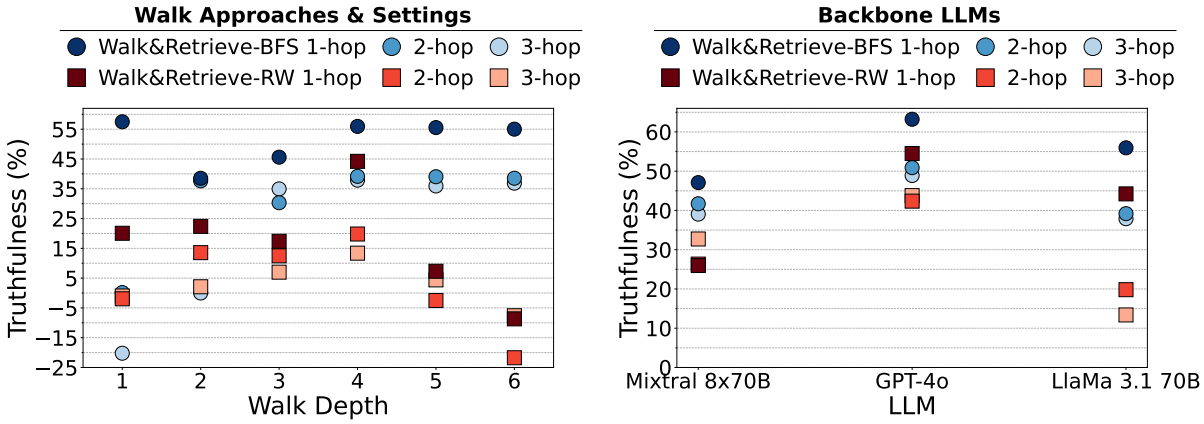

Figure 4: Truthfulness rates for different (i) walk approaches and (ii) backbone LLMs, over the MetaQA subsets.

likely due to its greater complexity (i.e., MetaQA expects only entity answers) and focus on holistic RAG performance, `Walk&Retrieve` remains robust. Even though the findings are promising, we plan to further evaluate `Walk&Retrieve` on larger KGs and other challenging benchmarks (e.g., WebQSP [56], CWQ [57]) to fully showcase its capabilities.

Ablation of Walk Approach. The graph traversal strategy and its hyperparameters define a node's relevant context, directly impacting corpus quality and, consequently, retrieval accuracy in RAG systems. The left graph in Fig. 4 shows MetaQA results for `Walk&Retrieve` with walk depths ranging from 1 to 6.[7] We find that a walk depth of 4 offers the best trade-off between answer accuracy and hallucination. Notably, regardless of walk length, `Walk&Retrieve-BFS` consistently yields higher truthfulness than `Walk&Retrieve-RW`, likely due to is systematic graph exploration, which avoids duplicate walks (cf. §2). In contrast, random walks tend to produce noisier context and fewer unique paths, thus capturing less relevant information from the KG.[8] While they may be more efficient on large-scale KGs, as they do not compute full neighborhoods, this efficiency comes at the cost of increased noise.[9]

Robustness to Backbone LLMs. Lastly, we evaluate model robustness using different LLMs (see right graph of Fig. 4), including `Mixtral-8x7B-Instruct` [58] and `GPT-4o` [59]. `Mixtral` improves answer

[7]For `Walk&Retrieve-RW`, we also ablate $n_w \in [10, 100]$ (step of 10); for brevity, we report results for $n_w = 60$, as other values perform comparably.

[8]On average, each node yields 60 duplicated and 8.74 unique random walks, whereas BFS generates 9.41 unique walks. Although RW could be modified to avoid duplicates, our current setup spans the full spectrum from randomness (RW) to structure (BFS).

[9]The time complexity of BFS is $\mathcal{O}(|V| + |E|)$, whereas that of RW varies between $\mathcal{O}(|V| \log |V|)$ and $\mathcal{O}(|V|^3)$.

truthfulness over `Llama-3.1` on 2- and 3-hop questions, while `GPT-4o` yields the highest truthfulness across all types of questions. The RW approach exhibits considerably higher variance across LLMs compared to the BFS-based model, which we attribute to the noisier and less relevant information in its generated corpus.

5. Conclusion

Current KG-based RAG faces challenges in aligning structured and textual representations, balancing accuracy with efficiency, and adapting to dynamic KGs. We proposed `Walk&Retrieve`, a simple yet effective KG-based framework for zero-shot RAG. It leverages walk-based graph traversal and LLM-driven knowledge verbalization for corpus generation. At inference time, the LLM is prompted with the query augmented by relevant verbalized walks for enhanced reasoning. Its efficient retrieval mechanism supports seamless adaptation to evolving KGs through incremental generation of new walks. `Walk&Retrieve` is compatible with any off-the-shelf LLM, and reduces computational overhead by avoiding fine-tuning of the backbone LLM. Despite its simplicity, `Walk&Retrieve` outperforms existing RAG approaches in answer accuracy and in the reduction of hallucinated or missing responses, while maintaining low query latency. Our results highlight walk-based corpus generation as a promising strategy for scaling to large-size KGs. These findings establish `Walk&Retrieve` as a simple, yet strong baseline for KG-based RAG, and we hope they inspire further research into adaptable and scalable RAG systems.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-4o in order to perform grammar and spelling check, paraphrase and reword. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language Models are Few-shot Learners, Advances in Neural Information Processing Systems 33 (2020) 1877–1901.

[2] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al., Training Language Models to Follow Instructions with Human Feedback, Advances in Neural Information Processing Systems 35 (2022) 27730–27744.

[3] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al., A Survey of Large Language Models, arXiv preprint arXiv:2303.18223 (2023). doi:https://doi.org/10.48550/arXiv.2303.18223.

[4] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, G. Neubig, Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing, ACM Computing Surveys 55 (2023) 1–35. doi:https://doi.org/10.1145/3560815.

[5] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., Llama 2: Open Foundation and Fine-tuned Chat Models, arXiv preprint arXiv:2307.09288 (2023). doi:https://doi.org/10.48550/arXiv.2307.09288.

[6] V. Liévin, C. E. Hother, A. G. Motzfeldt, O. Winther, Can Large Language Models Reason about Medical Questions?, Patterns 5 (2024).

[7] M. Danilevsky, K. Qian, R. Aharonov, Y. Katsis, B. Kawas, P. Sen, A Survey of the State of Explainable AI for Natural Language Processing, in: Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, 2020, pp. 447–459. doi:https://doi.org/10.18653/v1/2020.aacl-main.46.

[8] V. Rawte, S. Chakraborty, A. Pathak, A. Sarkar, S. T. I. Tonmoy, A. Chadha, A. Sheth, A. Das, The Troubling Emergence of Hallucination in Large Language Models-An Extensive Definition, Quantification, and Prescriptive Remediations, in: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 2023, pp. 2541–2573. doi:https://doi.org/10.18653/v1/2023.emnlp-main.155.

[9] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, P. Fung, Survey of Hallucination in Natural Language Generation, ACM Computing Surveys 55 (2023) 1–38. doi:10.1145/3571730.

[10] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, et al., A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions, ACM Transactions on Information Systems (2024). doi:https://doi.org/10.1145/3703155.

[11] A. Mallen, A. Asai, V. Zhong, R. Das, D. Khashabi, H. Hajishirzi, When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023, pp. 9802–9822. doi:https://doi.org/10.18653/v1/2023.acl-long.546.

[12] S. Tonmoy, S. Zaman, V. Jain, A. Rani, V. Rawte, A. Chadha, A. Das, A Comprehensive Survey of Hallucination Mitigation Techniques in Large Language Models, arXiv preprint arXiv:2401.01313 (2024). doi:https://doi.org/10.48550/arXiv.2401.01313.

[13] K. Sun, Y. Xu, H. Zha, Y. Liu, X. L. Dong, Head-to-Tail: How Knowledgeable are Large Language Models (LLMs)? AKA Will LLMs Replace Knowledge Graphs?, in: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), 2024, pp. 311–325. doi:https://doi.org/10.18653/v1/2024.naacl-long.18.

[14] T. Vu, M. Iyyer, X. Wang, N. Constant, J. Wei, J. Wei, C. Tar, Y.-H. Sung, D. Zhou, Q. Le, T. Luong, FreshLLMs: Refreshing Large Language Models with Search Engine Augmentation, in: L.-W. Ku, A. Martins, V. Srikumar (Eds.), Findings of the Association for Computational Linguistics: ACL 2024, Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 13697–13720. URL: https://aclanthology.org/2024.findings-acl.813/. doi:10.18653/v1/2024.findings-acl.813.

[15] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, Advances in Neural Information Processing Systems 33 (2020) 9459–9474.

[16] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, Retrieval-Augmented Generation for Large Language models: A Survey, arXiv preprint arXiv:2312.10997 (2023). doi:https://doi.org/10.48550/arXiv.2312.10997.

[17] W. Fan, Y. Ding, L. Ning, S. Wang, H. Li, D. Yin, T.-S. Chua, Q. Li, A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models, in: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2024, pp. 6491–6501. doi:https://doi.org/10.1145/3637528.3671470.

[18] J. Larson, S. Truitt, GraphRAG: Unlocking LLM Discovery on Narrative Private Data, 2024. URL: https://www.microsoft.com/en-us/research/blog/graphrag-unlocking-llm-discovery-on-narrative-private-data/.

[19] J. Chen, H. Lin, X. Han, L. Sun, Benchmarking Large Language Models in Retrieval-Augmented Generation, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, 2024, pp. 17754–17762. doi:https://doi.org/10.1609/aaai.v38i16.29728.

[20] B. Jin, C. Xie, J. Zhang, K. K. Roy, Y. Zhang, Z. Li, R. Li, X. Tang, S. Wang, Y. Meng, et al., Graph Chain-of-Thought: Augmenting Large Language Models by Reasoning on Graphs, arXiv preprint arXiv:2404.07103 (2024). doi:https://doi.org/10.48550/arXiv.2404.07103.

[21] B. Peng, Y. Zhu, Y. Liu, X. Bo, H. Shi, C. Hong, Y. Zhang, S. Tang, Graph Retrieval-Augmented Generation: A Survey, arXiv preprint arXiv:2408.08921 (2024).

[22] S. Ma, C. Xu, X. Jiang, M. Li, H. Qu, C. Yang, J. Mao, J. Guo, Think-on-Graph 2.0: Deep and Faithful Large Language Model Reasoning with Knowledge-guided Retrieval Augmented Generation, arXiv

preprint arXiv:2407.10805 (2024). doi:https://doi.org/10.48550/arXiv.2407.10805.
[23] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, et al., Knowledge Graphs, ACM Computing Surveys (Csur) 54 (2021) 1–37.
[24] J. Kim, Y. Kwon, Y. Jo, E. Choi, KG-GPT: A General Framework for Reasoning on Knowledge Graphs Using Large Language Models, in: Findings of the Association for Computational Linguistics: EMNLP 2023, 2023, pp. 9410–9421. doi:https://doi.org/10.18653/v1/2023.findings-emnlp.631.
[25] Y. Wu, N. Hu, S. Bi, G. Qi, J. Ren, A. Xie, W. Song, Retrieve-Rewrite-Answer: A KG-to-Text Enhanced LLMs Framework for Knowledge Graph Question Answering, arXiv preprint arXiv:2309.11206 (2023). doi:https://doi.org/10.48550/arXiv.2309.11206.
[26] S. Li, Y. Gao, H. Jiang, Q. Yin, Z. Li, X. Yan, C. Zhang, B. Yin, Graph Reasoning for Question Answering with Triplet Retrieval, in: Findings of the Association for Computational Linguistics: ACL 2023, 2023, pp. 3366–3375. doi:https://doi.org/10.18653/v1/2023.findings-acl.208.
[27] Y. Wen, Z. Wang, J. Sun, MindMap: Knowledge Graph Prompting Sparks Graph of Thoughts in Large Language Models, arXiv preprint arXiv:2308.09729 (2023). doi:https://doi.org/10.48550/arXiv.2308.09729.
[28] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, J. Larson, From Local to Global: A Graph RAG Approach to Query-Focused Summarization, arXiv preprint arXiv:2404.16130 (2024). doi:https://doi.org/10.48550/arXiv.2404.16130.
[29] B. Fatemi, J. Halcrow, B. Perozzi, Talk like a Graph: Encoding Graphs for Large Language Models, in: The Twelfth International Conference on Learning Representations, 2024. URL: https://openreview.net/forum?id=IuXR1CCrSi.
[30] J. Guo, L. Du, H. Liu, M. Zhou, X. He, S. Han, GPT4Graph: Can Large Language Models Understand Graph Structured Data ? An Empirical Evaluation and Benchmarking, arXiv preprint arXiv:2305.15066 (2023). doi:https://doi.org/10.48550/arXiv.2305.15066.
[31] L. Luo, Y.-F. Li, R. Haf, S. Pan, Reasoning on Graphs: Faithful and Interpretable Large Language Model Reasoning, in: The Twelfth International Conference on Learning Representations, 2024.
[32] J. Sun, C. Xu, L. Tang, S. Wang, C. Lin, Y. Gong, L. Ni, H.-Y. Shum, J. Guo, Think-on-Graph: Deep and Responsible Reasoning of Large Language Model on Knowledge Graph, in: The Twelfth International Conference on Learning Representations, 2024.
[33] C. Mavromatis, G. Karypis, GNN-RAG: Graph Neural Retrieval for Large Language Model Reasoning, arXiv preprint arXiv:2405.20139 (2024). doi:https://doi.org/10.48550/arXiv.2405.20139.
[34] P. Sen, S. Mavadia, A. Saffari, Knowledge Graph-augmented Language Models for Complex Question Answering, in: Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE), 2023, pp. 1–8. doi:https://doi.org/10.18653/v1/2023.nlrse-1.1.
[35] A. O. M. Saleh, G. Tur, Y. Saygin, SG-RAG: Multi-Hop Question Answering With Large Language Models Through Knowledge Graphs, in: M. Abbas, A. A. Freihat (Eds.), Proceedings of the 7th International Conference on Natural Language and Speech Processing (ICNLSP 2024), Association for Computational Linguistics, Trento, 2024, pp. 439–448. URL: https://aclanthology.org/2024.icnlsp-1.45/.
[36] M. Li, S. Miao, P. Li, Simple Is Effective: The Roles of Graphs and Large Language Models in Knowledge-Graph-Based Retrieval-Augmented Generation, in: International Conference on Learning Representations, 2025. URL: https://openreview.net/pdf?id=JvkuZZ04O7.
[37] X. He, Y. Tian, Y. Sun, N. V. Chawla, T. Laurent, Y. LeCun, X. Bresson, B. Hooi, G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering, arXiv preprint arXiv:2402.07630 (2024). doi:https://doi.org/10.48550/arXiv.2402.07630.
[38] J. Baek, A. F. Aji, A. Saffari, Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering, in: Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE), 2023, pp. 78–106. doi:https://doi.org/10.18653/v1/2023.nlrse-1.7.

[39] Y. Wu, Y. Huang, N. Hu, Y. Hua, G. Qi, J. Chen, J. Pan, CoTKR: Chain-of-Thought Enhanced Knowledge Rewriting for Complex Knowledge Graph Question Answering, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, 2024, pp. 3501–3520. doi:https://doi.org/10.18653/v1/2024.emnlp-main.205.

[40] T. Guo, Q. Yang, C. Wang, Y. Liu, P. Li, J. Tang, D. Li, Y. Wen, KnowledgeNavigator: Leveraging Large Language Models for Enhanced Reasoning over Knowledge Graph, Complex & Intelligent Systems 10 (2024) 7063–7076.

[41] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, J. Leskovec, QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2021, pp. 535–546. doi:https://doi.org/10.18653/v1/2021.naacl-main.45.

[42] Y. Hu, Z. Lei, Z. Zhang, B. Pan, C. Ling, L. Zhao, GRAG: Graph Retrieval-Augmented Generation, arXiv preprint arXiv:2405.16506 (2024). doi:https://doi.org/10.48550/arXiv.2405.16506.

[43] Y. Cao, Y. Kang, C. Wang, L. Sun, Instruction Mining: Instruction Data Selection for Tuning Large Language Models, arXiv preprint arXiv:2307.06290 (2023). doi:https://doi.org/10.48550/arXiv.2307.06290.

[44] E. J. Hu, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al., LoRA: Low-Rank Adaptation of Large Language Models, in: International Conference on Learning Representations, 2022.

[45] Z. Chai, T. Zhang, L. Wu, K. Han, X. Hu, X. Huang, Y. Yang, GraphLLM: Boosting Graph Reasoning Ability of Large Language Model, arXiv preprint arXiv:2310.05845 (2023). doi:https://doi.org/10.48550/arXiv.2310.05845.

[46] B. Perozzi, B. Fatemi, D. Zelle, A. Tsitsulin, M. Kazemi, R. Al-Rfou, J. Halcrow, Let Your Graph Do the Talking: Encoding Structured Data for LLMs, arXiv preprint arXiv:2402.05862 (2024).

[47] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online Learning of Social Representations, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710. doi:https://doi.org/10.1145/2623330.2623732.

[48] P. Ristoski, H. Paulheim, Rdf2vec: RDF Graph Embeddings for Data Mining, in: International semantic web conference, Springer, 2016, pp. 498–514.

[49] Y. Zhang, H. Dai, Z. Kozareva, A. Smola, L. Song, Variational Reasoning for Question Answering with Knowledge Graph, in: Proceedings of the AAAI conference on artificial intelligence, volume 32, 2018.

[50] X. Yang, K. Sun, H. Xin, Y. Sun, N. Bhalla, X. Chen, S. Choudhary, R. D. Gui, Z. W. Jiang, Z. Jiang, et al., CRAG–Comprehensive RAG Benchmark, 38th Conference on Neural Information Processing Systems (NeurIPS 2024), Track on Datasets and Benchmarks (2024).

[51] A. Saxena, A. Kochsiek, R. Gemulla, Sequence-to-Sequence Knowledge Graph Completion and Question Answering, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 2814–2828. doi:https://doi.org/10.18653/v1/2022.acl-long.201.

[52] P. Sen, A. F. Aji, A. Saffari, Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering, in: Proceedings of the 29th International Conference on Computational Linguistics, 2022, pp. 1604–1619.

[53] OpenAI, ChatGPT, 2023. URL: https://openai.com/index/chatgpt/.

[54] AI@Meta, Llama 3 Model Card (2024). URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[55] A. Panickssery, S. R. Bowman, S. Feng, LLM Evaluators Recognize and Favor Their Own Generations, arXiv preprint arXiv:2404.13076 (2024). doi:https://doi.org/10.48550/arXiv.2404.13076.

[56] W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang, J. Suh, The Value of Semantic Parse Labeling for Knowledge Base Question Answering, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2016, pp. 201–206. doi:https://doi.org/10.18653/v1/P16-2033.

[57] A. Talmor, J. Berant, The Web as a Knowledge-Base for Answering Complex Questions, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 641–651. doi:https://doi.org/10.18653/v1/N18-1059.

[58] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al., Mistral 7B, arXiv preprint arXiv:2310.06825 (2023). doi:https://doi.org/10.48550/arXiv.2310.06825.

[59] OpenAI, GPT-4 Technical Report, arXiv preprint arXiv:2303.08774 (2023). doi:https://doi.org/10.48550/arXiv.2303.08774.