Training Retrieval Augmented Generation for Text-Graph Question Answering with LLMs

Ryan Engel and Gilchan Park

Computational Science Initiative
Brookhaven National Laboratory

Abstract

Recent advancements have demonstrated that Large Language Models (LLMs) can effectively be used for text generation and question answering. Moreover, the development of Retrieval Augmented Generation (RAG) methods, which combine LLMs with information retrieval systems, has further enhanced the capabilities of these models. However, in the context of graph-based question answering, current RAG approaches primarily rely on fixed strategies for subgraph retrieval, often leading to the inclusion of irrelevant information or a loss of topological information. Furthermore, LLMs have been limited by their tendency to hallucinate, or generate inaccurate or nonsensical information. To address these limitations, we propose a RAG approach for text-graph question answering with a trainable subgraph retrieval mechanism using a graph convolutional neural network, called GCN-Retriever. To our knowledge, this approach is the first to effectively train the subgraph retrieval component for text-graph RAG. We evaluate the efficacy of our GCN-Retriever compared to other similar approaches and demonstrate comparable results.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in text generation and question answering tasks. The development of Retrieval Augmented Generation (RAG) methods has further enhanced these capabilities by combining LLMs with information retrieval systems [Béchard and Ayala, 2024]. However, in the context of graph-based question answering, current RAG approaches face limitations. They often rely on fixed strategies for subgraph retrieval, which can lead to the inclusion of irrelevant information or the loss of important topological data [Hu et al., 2024, Kang et al., 2023]. Additionally, LLMs are prone to hallucination, generating inaccurate or nonsensical information.

To address these challenges, we propose a novel RAG approach for text-graph question answering that incorporates a trainable subgraph retrieval mechanism using a graph convolutional neural network (GCN) [Kipf and Welling, 2017], which we call GCN-Retriever. To our knowledge, this is the first approach to effectively train the subgraph retrieval component for text-graph RAG. By training the entire RAG system end-to-end and utilizing LLM feedback, our model learns to retrieve information more relevant to the answer, potentially reducing hallucination and improving overall performance.

In this paper, we evaluate the efficacy of our GCN-Retriever compared to other similar approaches and demonstrate comparable results. This is the first approach to train the subgraph retrieval component for text-graph RAG. By training the entire RAG system end-to-end, and using the LLM feedback, the model learns to retrieve information more relevant to the answer. Our work contributes to the growing body of research on RAG systems and offers a new direction for improving graph-based question answering tasks.

2 Related Works

2.1 General Purpose Large Language Models

Large-scale language models like Llama [Touvron et al., 2023a], and Llama 2 [Touvron et al., 2023b], have highlighted the importance of data design and task-specific training in improving model performance across a variety of tasks. Additionally, the creation of the Mistral [Jiang et al., 2023] model helps to bring open-source LLMs to the forefront of scientific innovation. These strides in LLM research have brought significant advancements in other scientific disciplines like biology [Zhang et al., 2024].

2.2 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is a technique that combines the generative power of large language models with the precision of information retrieval systems, enabling the model to access and incorporate relevant external knowledge dynamically during the generation process. [Lewis et al., 2021] introduced the initial concept of RAG, demonstrating its effectiveness in tasks requiring extensive knowledge integration. Recent advancements have focused on optimizing the retrieval process and enhancing the integration between the retrieved information and the generative model. For example, [Izacard et al., 2022] proposed a few-shot learning approach for RAG, improving its adaptability to new tasks with minimal annotated data. Additionally, [Guu et al., 2020] explored pre-training strategies for retrieval-augmented language models, highlighting the benefits of incorporating retrieval mechanisms during the pre-training phase.

2.3 RAG for Graph-Based Question Answering

Much research has been focused on developing RAG for LLMs, with a particular emphasis on its application in graph question answering [Xu et al., 2024]. However, the effectiveness of these systems can be limited by the static nature of their retrieval components. Current graph-based RAG methods often rely on semantic similarity for subgraph retrieval, which can result in the propagation of irrelevant information. Examples of such approaches are discussed in the following papers.

[Hu et al., 2024] employs a strategy that compares all k-hop ego-graphs with the query vector to retrieve the top N similar subgraphs, then partially masks the irrelevant entities with a soft pruning module. The pruned ego-graphs are encoded into a soft graph token, and textual information is encoded into text tokens to be inputted to the language model for generation.

Similarly, the SURGE framework [Kang et al., 2023] includes a context-relevant subgraph retriever which retrieves subgraphs relevant to the given dialogue history from a knowledge graph. Specifically, they measure the similarity of a context and triplet embedding (node, edge, node) to compose the retrieval distribution. They then encode the subgraph with a GNN and use contrastive learning to ground the response to the graph's data.

[He et al., 2024] formulates the subgraph construction of their G-Retriever as a prize-collecting Steiner tree optimization, where the prize is maximizing semantic similarity of nodes to the query, while accounting for the size of the subgraph size. Then, an answer is generated using a 'graph prompt', a textualized graph, and the query.

While each of these approaches aim to solve the problem of retrieving irrelevant information, none of the retrieval mechanisms presented are trainable, utilizing the feedback from the generated answers themselves. The methods described primarily use fixed strategies to retrieve subgraphs based on pre-defined criteria such as semantic similarity and structural properties of the graph. These methods do not incorporate a feedback loop where the quality of the generated answers influences the retrieval process in a trainable manner.

For example, [Hu et al., 2024], use soft pruning to filter out irrelevant entities based on semantic similarity, but this process is not dynamically adjusted based on the performance of the generated answers. Similarly, [He et al., 2024] uses a prize-collecting Steiner tree optimization to construct subgraphs, and [Kang et al., 2023] employs a context-relevant subgraph retriever with contrastive learning, but neither approach adapts its retrieval strategy based on answer accuracy.

A trainable retrieval mechanism is necessary to dynamically learn the balance of semantic similarity with the relevance of node links to the correct answer. Incorporating a trainable retrieval mechanism allows the system to adjust its retrieval strategy based on feedback from the performance of the generated answers. This dynamic adjustment is essential for balancing semantic similarity with the relevance of node links, ensuring that the retrieved subgraphs are both contextually appropriate and accurate in terms of the information they contain.

Our proposed GCN-Retriever addresses this gap by introducing a trainable subgraph retrieval mechanism, potentially improving the overall performance of graph-based RAG systems for question answering tasks.

3 Problem Formalization

We propose a trainable subgraph retrieval mechanism using a graph convolutional network (GCN) [Kipf and Welling, 2017] to predict which nodes and edges are most relevant in answering the question.

The model is trained based on the quality of the generated answers by the LLM, so the system can learn to prioritize retrieving subgraphs that lead to more accurate and relevant responses. Using this GCN allows the system to dynamically build a subgraph based on the nodes and edges predicted by the model, which are learned through answering thousands

of questions. This approach addresses the limitations of static retrieval mechanisms and provides a more adaptive and effective solution for subgraph retrieval in graph-based RAG question answering systems.

3.1 Graph Convolutional Network

We define the GCN model as follows:

$$\begin{aligned} \mathbf{H}^{(1)} &= \mathrm{ReLU}(\mathbf{A}\mathbf{X}\mathbf{W}^{(0)}) \\ \mathbf{H}^{(2)} &= \mathbf{A}\mathbf{H}^{(1)}\mathbf{W}^{(1)} \end{aligned} \tag{1}$$

where:

- $\mathbf{A}$ is the adjacency matrix.
- $\mathbf{X}$ is the input feature matrix.
- $\mathbf{W}^{(0)}$ and $\mathbf{W}^{(1)}$ are learnable weight matrices.
- $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ are the hidden and output representations, respectively.

The GCN model consists of two layers, each performing a convolution operation followed by a ReLU activation function. Through multiple epochs, the model learns to predict the importance of the nodes and edges in answering the query based on the following loss function:

3.2 Loss Calculation

The loss for the GCN model is calculated using the negative log-likelihood of the node and edge probabilities, weighted by the loss from the LLM:

$$\mathcal{L}_{\text{gcn}} = -\sum_{i=1}^{N} \left(\sum_{j=1}^{V_i} \log(p_{v_{ij}} + \epsilon) + \sum_{k=1}^{E_i} \log(p_{e_{ik}} + \epsilon) \right) \cdot \mathcal{L}_{\text{llm}} \tag{2}$$

where:

- N is the number of graphs.
- V_i and E_i are the number of nodes and edges in graph i.
- $p_{v_{ij}}$ and $p_{e_{ik}}$ are the probabilities of node j and edge k in graph i.
- ϵ is a small constant to prevent log of zero.
- $\mathcal{L}_{\text{llm}}$ is the loss from the LLM.

The node and edge probabilities calculated in the subgraph generation function, along with the LLM loss, are used to calculate the loss for the GCN. The probabilities are calculated from a probability score, outputted from the GCN, which represent the likelihood for each node and query to be useful in generating the correct answer.

This approach is effective because when we backpropagate this loss through the GCN, its goal is to minimize this loss, thus making node importance predictions that result in a lower LLM loss. Generating a subgraph that results in lower LLM loss is the goal because a lower LLM loss means a more accurate answer.

3.3 Subgraph Generation

The subgraph generation process involves calculating node importance using the GCN model, followed by selecting top-k nodes and their corresponding edges:

$$\mathbf{S}_i = \text{softmax}(\mathbf{C}_i \odot \mathbf{H}_i) \tag{3}$$

where:

- $\mathbf{C}_i$ is the cosine similarity between nodes and query embeddings.
- $\mathbf{H}_i$ is the node importance predicted by the GCN.
- $\odot$ denotes element-wise multiplication.
- $\mathbf{S}_i$ is the combined importance score for graph i.

The top-k nodes are selected based on $\mathbf{S}_i$, and the edges connecting these nodes are included in the subgraph. This subgraph is then fed into the LLM along with the query during inference, and the feedback is generated to help train the retriever.

3.4 Training and Evaluation

The training process involves optimizing the GCN model using the Adam optimizer:

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} - \eta \nabla \mathcal{L}_{\text{gcn}} \tag{4}$$

where η is the learning rate.
During evaluation, the best model is selected based on the validation loss:

$$\mathcal{L}_{\text{val}} = \frac{1}{|V|} \sum_{i \in V} \mathcal{L}_{\text{gcn},i} \tag{5}$$

where V is the validation set.

This approach leverages the strengths of both GCNs and LLMs for improved question answering over large graphs. The GCN learns to identify relevant portions of the graph by considering both structural properties and semantic relevance to the query. This selective mechanism allows the LLM to focus on the most pertinent information, allowing us to directly optimize the retrieval step.

The loss function is particularly noteworthy. By using the LLM's loss to guide the GCN's learning, we create a training signal. When the LLM demonstrates strong performance, indicated by a low loss, it reinforces the GCN to continue selecting similar subgraphs in subsequent iterations. Conversely, when the LLM exhibits poor performance, resulting in higher loss, the GCN undergoes more substantial updates, thereby encouraging the exploration of alternative subgraph selections.

This joint learning approach allows the retrieval mechanism (GCN) to adapt to the needs of the reasoning component (LLM), leading to more effective and efficient question answering systems for large-scale graph-structured data.

The model architecture is shown below:

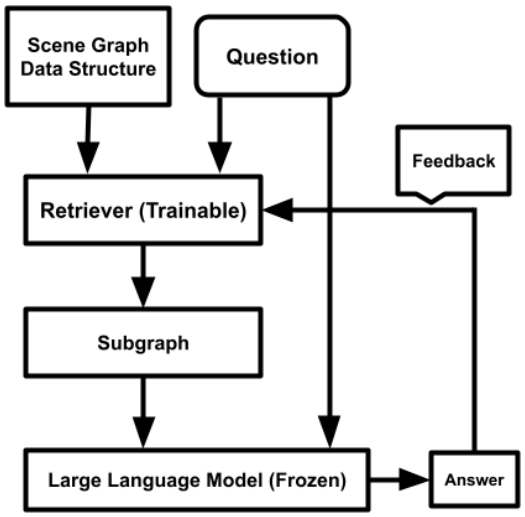

Figure 1: Retriever Model Architecture

4 Datasets

The dataset used is the scene graphs dataset used by [He et al., 2024] as part of the GraphQA Benchmark. This dataset contains 100,000 graphs, with an average of 19.13 nodes and 68.44 edges per graph. The node attributes include text which describes object attributes (e.g., color, shape), the edge attributes include text which describe relations (e.g., actions, spatial relations), and the given task is to answer questions based on the given graph, with accuracy as the metric. A visual description of the dataset is shown in figure 2, illustrating the details of the data structure.

5 Experiments

We evaluate our proposed algorithm against the scene graphs dataset used by the G-Retriever algorithm [He et al., 2024]. First, we compare our algorithm's performance against the G-Retriever's frozen LLM. In this experiment we keep the LLM frozen and only train the GCN model. Next, we evaluate our algorithm against the G-Retriever's prompt-tuned LLM [Lester et al., 2021], and in doing so we train our GCN and the prompt-tuning in an end-to-end system. The results demonstrate that our algorithm shows comparable results with the other algorithms, indicating that this approach could be adapted for further performance. All experiments were evaluated using the Llama2-7B LLM [Touvron et al., 2023b], utilizing 2 x NVIDIA H100 80GB HBM3 GPUs. The results of the experiments can be found in Table 1.

6 Results

In running these experiments, we demonstrate that the text-graph RAG can be trained effectively. Although our GCN-Retriever algorithm doesn't outperform the G-Retriever, it is

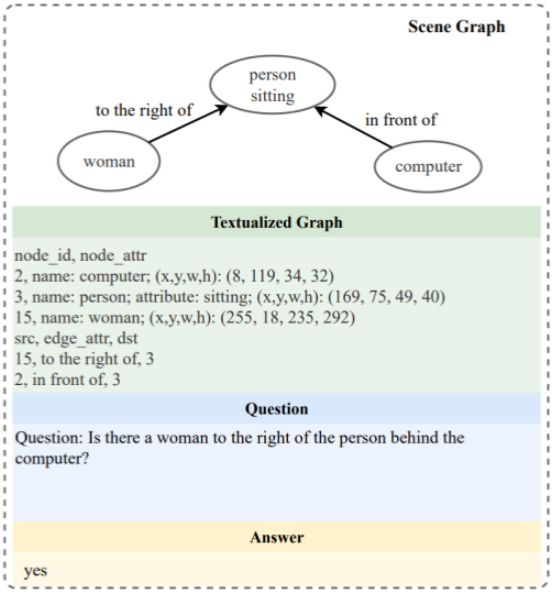

Figure 2: Scene Graphs Dataset [He et al., 2024]

Algorithm	Zero-shot	Prompt Tuning
GCN-Retriever	0.3261	0.5541
G-Retriever	0.3974	0.6341
GraphToken	—	0.4903

Table 1: Comparison of Algorithms on Scene Graphs Dataset

more accurate than the GraphToken [Perozzi et al., 2024] in the prompt tuning experiment. The performance differences observed between the GCN-Retriever and G-Retriever is likely due to the fundamental difference in subgraph retrieval methodologies.

The static retrieval method used by G-Retriever relies on pre-defined subgraph structures that do not change during training. This approach can be beneficial for stability and speed, as the retrieval process is consistent and computationally efficient. However, it lacks the flexibility to adapt to new information or patterns that may emerge during training, limiting its ability to optimize retrieval.

In contrast, our dynamic retrieval component allows the model to learn and adapt the subgraph structures based on feedback from the LLM. This means that our GCN-Retriever can identify and incorporate new, relevant connections in the graph data during the training process. This adaptability can lead to more robust optimization, as the model can refine its retrieval strategy to better suit the specific task at hand. However, this leads to increased computational overhead during training, which contributes to the observed lag in performance compared to the more straightforward static retrieval approach.

Despite these performance differences, the dynamic retrieval methodology offers significant

long-term advantages. By allowing the model to adapt and learn from the graph data in real-time, we can achieve a more nuanced and accurate understanding of the underlying relationships within the data. This capability is particularly valuable in applications where the model must specialize its retrieval for a given task. By allowing the direct optimization of the retrieval component, we can effectively train the RAG framework based on the LLMs performance on the task. A dynamic retrieval component in RAG is ideal for optimizing the retrieval step and enabling end-to-end training for more robust optimization, addressing gaps in current research.

7 Conclusions & Future Work

Overall, we demonstrate that the text-graph RAG can be trained effectively. Although GCN-Retriever doesn't outperform the G-Retriever, it is more accurate than the GraphToken. Thus, our algorithm performs comparably to the other algorithms. This is significant because it allows RAG to learn from the graph data dynamically, based on the LLM's feedback, unlike other static retrieval components. A dynamic retrieval component in RAG is ideal for optimizing the retrieval step and enabling end-to-end training for more robust optimization, addressing gaps in current research.

Future work could focus on expanding this idea and developing more complicated and robust learning algorithms for the retrieval components. Additionally, future work could be using this technology in applications to other scientific disciplines, such as biology, chemistry, and physics.

References

Patrice Béchard and Orlando Marquez Ayala. Reducing hallucination in structured outputs via retrieval-augmented generation, 2024. URL https://arxiv.org/abs/2404.08189.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/guu20a.html.

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, 2024. URL https://arxiv.org/abs/2402.07630.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. Grag: Graph retrieval-augmented generation, 2024. URL https://arxiv.org/abs/2405.16506.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models, 2022. URL https://arxiv.org/abs/2208.03299.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Minki Kang, Jin Myung Kwak, Jinheon Baek, and Sung Ju Hwang. Knowledge graph-augmented language models for knowledge-grounded dialogue generation, 2023. URL https://arxiv.org/abs/2305.18846.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017. URL https://arxiv.org/abs/1609.02907.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021. URL https://arxiv.org/abs/2104.08691.

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, 2021. URL https://arxiv.org/abs/2005.11401.

Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan Halcrow. Let your graph do the talking: Encoding structured data for llms, 2024. URL https://arxiv.org/abs/2402.05862.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023a. URL https://arxiv.org/abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b. URL https://arxiv.org/abs/2307.09288.

Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR 2024. ACM, July 2024. doi: 10.1145/3626772.3661370. URL http://dx.doi.org/10.1145/3626772.3661370.

Qiang Zhang, Keyang Ding, Tianwen Lyv, Xinda Wang, Qingyu Yin, Yiwen Zhang, Jing Yu, Yuhao Wang, Xiaotong Li, Zhuoyi Xiang, Kehua Feng, Xiang Zhuang, Zeyuan Wang, Ming Qin, Mengyao Zhang, Jinlu Zhang, Jiyu Cui, Tao Huang, Pengju Yan, Renjun Xu, Hongyang Chen, Xiaolin Li, Xiaohui Fan, Huabin Xing, and Huajun Chen. Scientific large language models: A survey on biological & chemical domains, 2024. URL `https://arxiv.org/abs/2401.14656`.