G–RETRIEVER: RETRIEVAL–AUGMENTED GENERATION FOR TEXTUAL GRAPH UNDERSTANDING AND QUESTION ANSWERING

BY: HE ET AL.

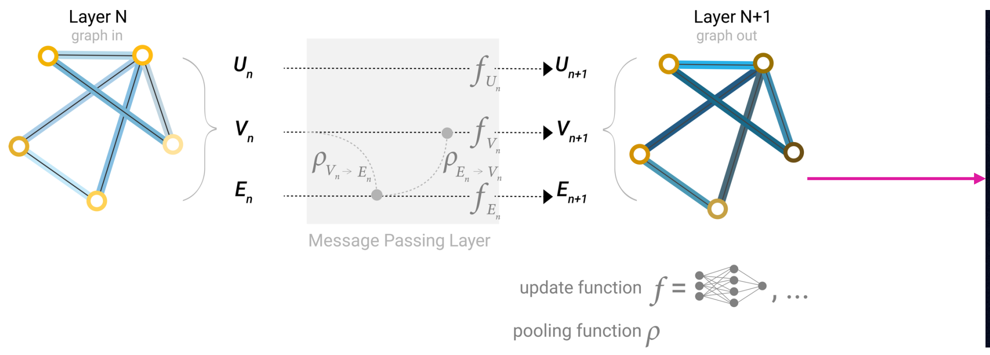
3/25/25

Presenter: Gurjot Singh

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Challenge

- Traditionally, graph neural networks (GNNs) have excelled in structural reasoning (e. g. , node classification or link prediction), while large language models (LLMs) have advanced natural language processing.
- The challenge, however, lies in building systems that can *“**chat with your graph**”.*

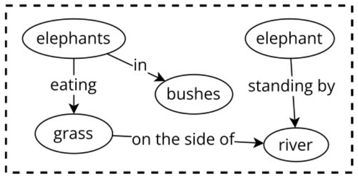


Challenge

Key Problem:

- **Graph Question Answering** (**GraphQA**): The goal is to answer complex questions over graphs with textual attributes, where the answer may require multi–hop reasoning and explicit references to parts of the graph.
- **Hallucination**: A major challenge with LLMs is their tendency to hallucinate details—i. e. , generating factually incorrect or invented elements. In the graph context, this can mean referencing nodes or edges that do not exist.
- **Scalability**: Converting an entire graph into a text sequence (for LLM consumption) is impractical when graphs are large, since LLMs have a limited context window. Thus, selective retrieval of the relevant subgraph is needed.

Challenge

User	 elephants in bushes eating grass on the side of river elephant standing by What kind of animal is in the bushes? Answer the question and list the nodes or edges from the graph that support your answer.
LLM w/ Graph Prompt Tuning	[Incorrect response with hallucinated nodes and edges] The animal in the bushes is a deer. Nodes: * Deer (node 1) * Bushes (node 2) Edges: * Deer $\rightarrow$ Bushes (edge 1) * Deer $\rightarrow$ Grass (edge 2) * Bushes $\rightarrow$ Grass (edge 3)
G-Retriever (Ours)	[Correct response with accurate node and edge references] Based on the information provided in the graph, the animal in the bushes is an elephant. The nodes that support this answer are: * Node 1: The elephant is mentioned in the first line of the graph. The edges that support this answer are: * The edge connects the elephant to the bushes, indicating that the elephant is in the bushes. Therefore, the answer to the question is: The animal in the bushes is an elephant.

Outline

- Introduction & Background
- Contributions
- Methodology
- Results & Analysis
- Limitations
- Conclusion

Key Contributions

- Developed a comprehensive GraphQA benchmark integrating diverse datasets to evaluate real–world graph question answering.
- Introduced G–Retriever, a novel retrieval–augmented generation architecture that effectively combines GNNs and LLMs.
- Employed a Prize–Collecting Steiner Tree method to extract relevant subgraphs, significantly reducing hallucination and enhancing scalability.

Formulation

A *textual graph* is defined as

$$G = (V, E, \{x_n\}_{n \in V}, \{x_e\}_{e \in E}),$$

where:

- V and E denote the sets of nodes and edges, respectively.
- Each node $n \in V$ is associated with a text attribute $x_n \in D^{L_n}$.
- Each edge $e \in E$ is associated with a text attribute $x_e \in D^{L_e}$,

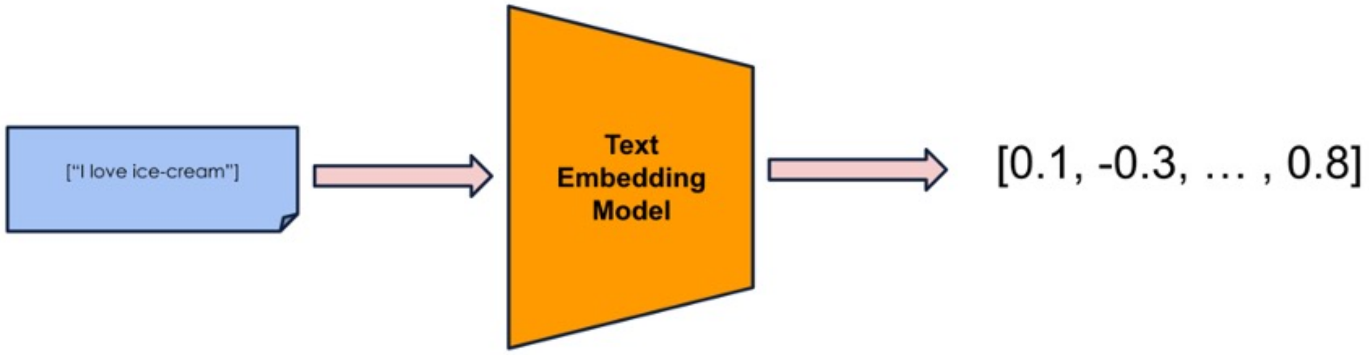
with D representing the vocabulary and L_n, L_e denoting the length of the respective texts.

Text Embeddings / Encoding

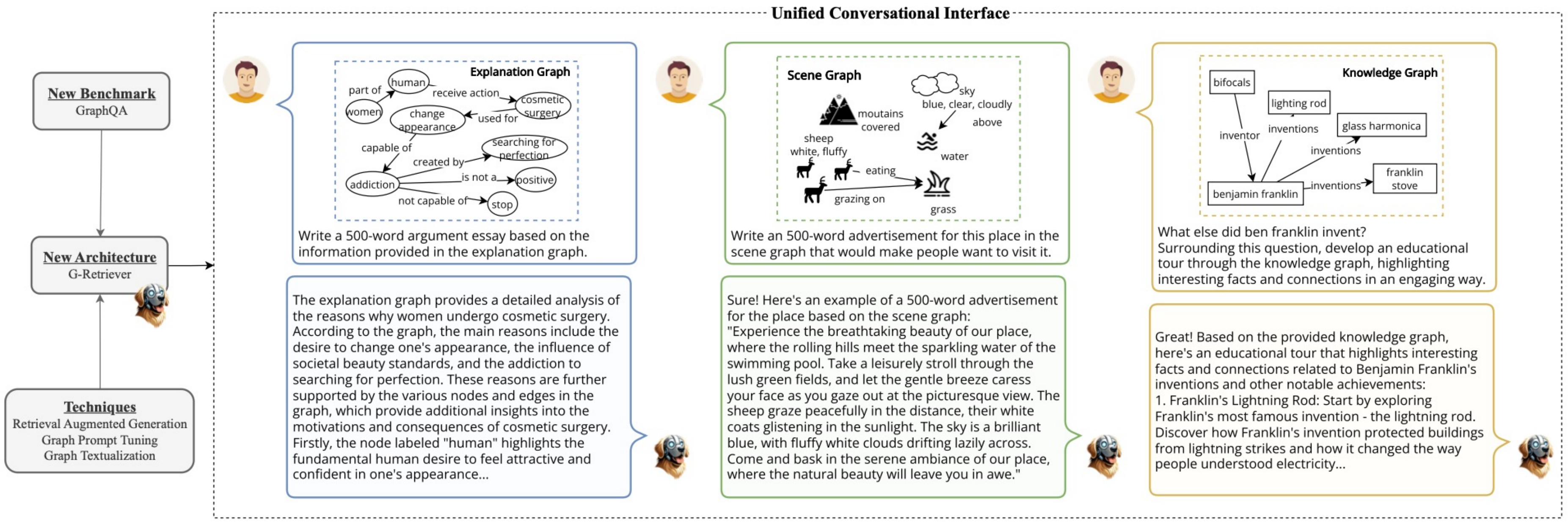
For a given node n with text attribute x_n, we use a language model (LM) to encode it into a vector representation:

$$z_n = \text{LM}(x_n) \in R^d,$$

where d is the dimension of the output embedding. An analogous encoding is applied to the edge attributes.



Methodology



Write a 500-word argument essay based on the information provided in the explanation graph.

The explanation graph provides a detailed analysis of the reasons why women undergo cosmetic surgery. According to the graph, the main reasons include the desire to change one's appearance, the influence of societal beauty standards, and the addiction to searching for perfection. These reasons are further supported by the various nodes and edges in the graph, which provide additional insights into the motivations and consequences of cosmetic surgery. Firstly, the node labeled "human" highlights the fundamental human desire to feel attractive and confident in one's appearance...

Write an 500-word advertisement for this place in the scene graph that would make people want to visit it.

Sure! Here's an example of a 500-word advertisement for the place based on the scene graph: "Experience the breathtaking beauty of our place, where the rolling hills meet the sparkling water of the swimming pool. Take a leisurely stroll through the lush green fields, and let the gentle breeze caress your face as you gaze out at the picturesque view. The sheep graze peacefully in the distance, their white coats glistening in the sunlight. The sky is a brilliant blue, with fluffy white clouds drifting lazily across. Come and bask in the serene ambiance of our place, where the natural beauty will leave you in awe."

What else did ben franklin invent?
Surrounding this question, develop an educational tour through the knowledge graph, highlighting interesting facts and connections in an engaging way.

Great! Based on the provided knowledge graph, here's an educational tour that highlights interesting facts and connections related to Benjamin Franklin's inventions and other notable achievements:
1. Franklin's Lightning Rod: Start by exploring Franklin's most famous invention - the lightning rod. Discover how Franklin's invention protected buildings from lightning strikes and how it changed the way people understood electricity...

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Graph QA Benchmark

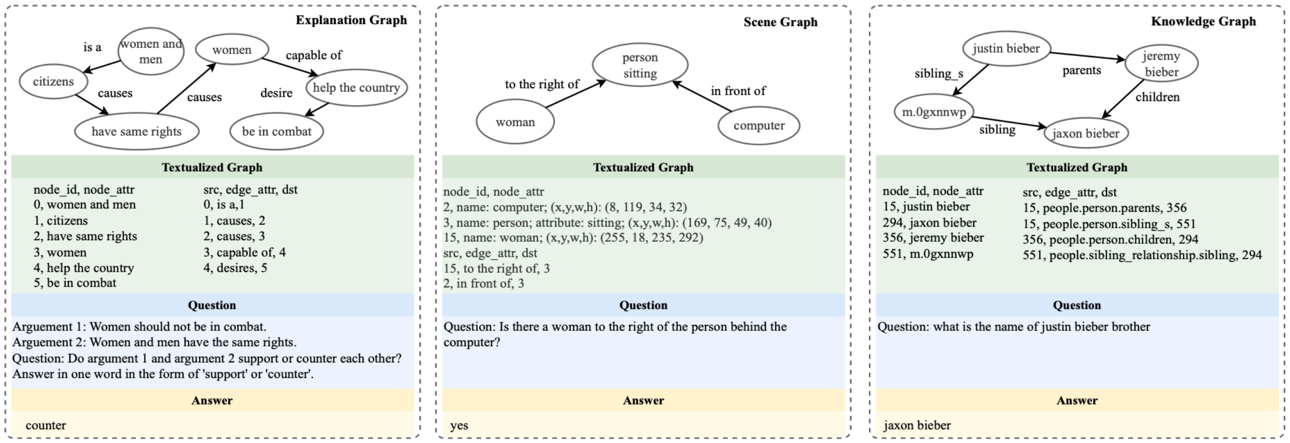
Data Format

Each instance in the GraphQA benchmark consists of:

- A **textual graph**, where nodes and edges are represented in a natural language format (e.g., CSV-like format with columns `node_id`, `node_attr` for nodes and `src`, `edge_attr`, `dst` for edges).
- A **question** related to the graph.
- One or more **answers** extracted from the graph's attributes.

Graph QA Benchmark

Dataset	ExplaGraphs	SceneGraphs	WebQSP
#Graphs	2,766	100,000	4,737
Avg. #Nodes	5.17	19.13	1370.89
Avg. #Edges	4.25	68.44	4252.37
Node Attribute	Commonsense concepts	Object attributes (*e.g.,* color, shape)	Entities in Freebase
Edge Attribute	Commonsense relations	Relations (*e.g.,* actions, spatial relations)	Relations in Freebase
Task	Common sense reasoning	Scene graph question answering	Knowledge based question answering
Evaluation Matrix	Accuracy	Accuracy	Hit@1



Textualized Graph

```
node_id, node_attr          src, edge_attr, dst
0, women and men            0, is a,1
1, citizens                 1, causes, 2
2, have same rights         2, causes, 3
3, women                    3, capable of, 4
4, help the country         4, desires, 5
5, be in combat
```

Question

Arguement 1: Women should not be in combat.
Arguement 2: Women and men have the same rights.
Question: Do argument 1 and argument 2 support or counter each other? Answer in one word in the form of 'support' or 'counter'.

Answer

counter

Textualized Graph

```
node_id, node_attr
2, name: computer; (x,y,w,h): (8, 119, 34, 32)
3, name: person; attribute: sitting; (x,y,w,h): (169, 75, 49, 40)
15, name: woman; (x,y,w,h): (255, 18, 235, 292)
src, edge_attr, dst
15, to the right of, 3
2, in front of, 3
```

Question

Question: Is there a woman to the right of the person behind the computer?

Answer

yes

Textualized Graph

```
node_id, node_attr     src, edge_attr, dst
15, justin bieber      15, people.person.parents, 356
294, jaxon bieber      15, people.person.sibling_s, 551
356, jeremy bieber     356, people.person.children, 294
551, m.0gxnnwp         551, people.sibling_relationship.sibling, 294
```

Question

Question: what is the name of justin bieber brother

Answer

jaxon bieber

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Graph QA Benchmark

Task and Evaluation Metrics

The primary task in GraphQA is to answer questions that require multi-hop reasoning over a graph's nodes and edges. To handle potential ambiguities from multiple valid answers, the evaluation uses:

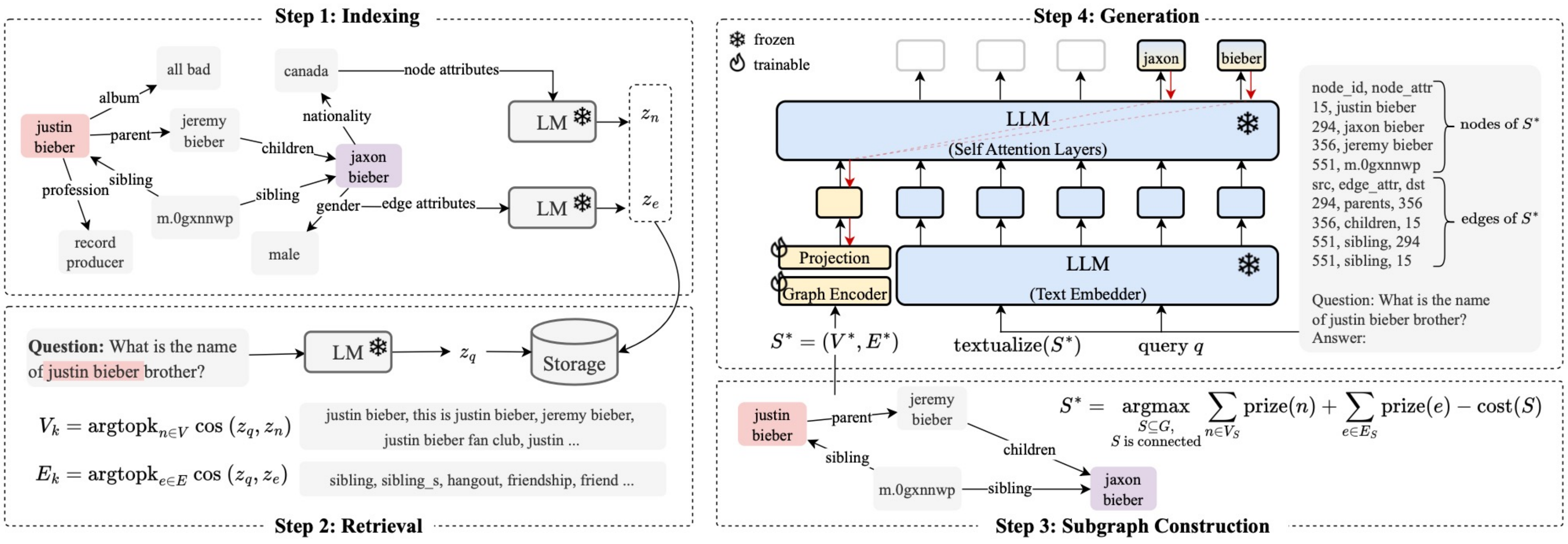
- **Accuracy**: For tasks like commonsense reasoning and scene understanding.
- **Hit@1**: Especially for knowledge-based question answering where the precision of the top answer is crucial.

Benchmark Significance

The GraphQA benchmark is significant because it:

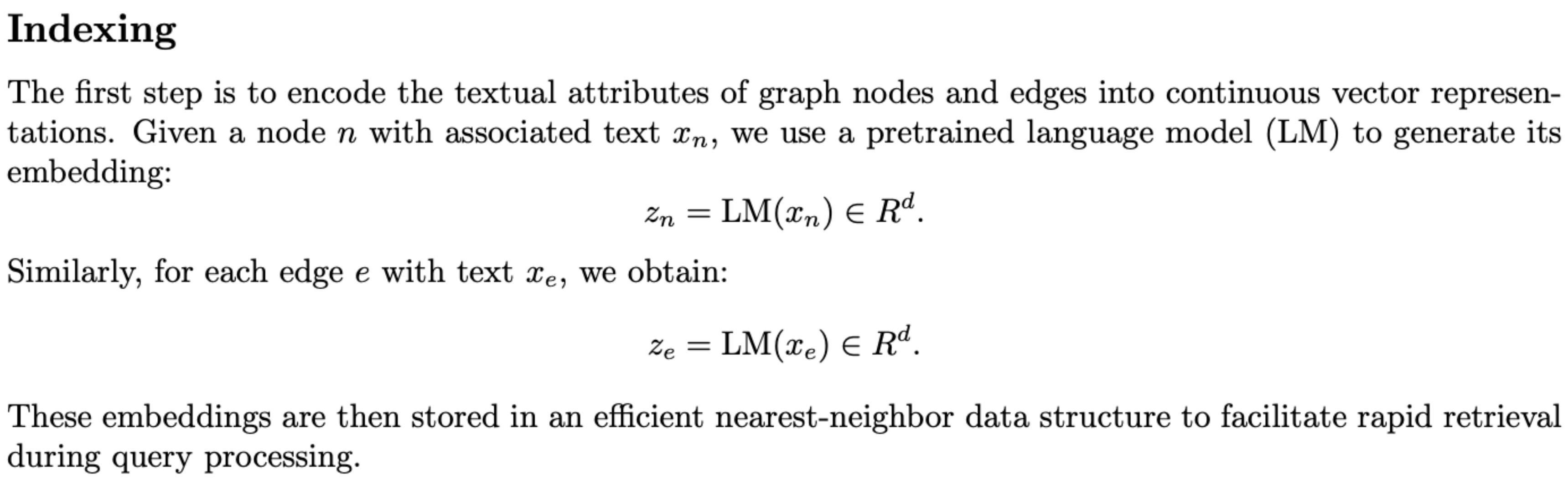
- Bridges diverse applications, from commonsense reasoning to scene and knowledge graph understanding, into a unified evaluation framework.
- Provides a standardized data format that facilitates fair comparisons across different graph reasoning models.
- Encourages the development of systems that can interact with graph-structured data through natural language, pushing the boundaries of graph-augmented language models.

G-Retriever



UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

G–Retriever



Indexing

The first step is to encode the textual attributes of graph nodes and edges into continuous vector representations. Given a node n with associated text x_n, we use a pretrained language model (LM) to generate its embedding:

$$z_n = \mathrm{LM}(x_n) \in R^d.$$

Similarly, for each edge e with text x_e, we obtain:

$$z_e = \mathrm{LM}(x_e) \in R^d.$$

These embeddings are then stored in an efficient nearest-neighbor data structure to facilitate rapid retrieval during query processing.

G–Retriever

Retrieval

Given a user query x_q, we first encode it into a query embedding:

$$z_q = \mathrm{LM}(x_q) \in R^d.$$

Using cosine similarity as a measure, we perform a k-nearest neighbors search over the precomputed node and edge embeddings. This yields the most relevant nodes and edges:

$$V_k = \mathrm{argtopk}_{n \in V} \cos(z_q, z_n) \quad \text{and} \quad E_k = \mathrm{argtopk}_{e \in E} \cos(z_q, z_e).$$

These sets V_k and E_k represent the preliminary selection of graph components that are semantically related to the query.

G–Retriever

Subgraph Construction

Since the entire graph may be too large to process directly, we extract a connected subgraph that is both relevant and succinct. This is achieved by formulating subgraph extraction as a Prize-Collecting Steiner Tree (PCST) problem. In our setting, we assign prizes to nodes and edges based on their relevance scores. For a node $n \in V_k$, the prize is given by:

$$\text{prize}(n) = \begin{cases} k - i + 1, & \text{if } n \text{ is the } i\text{-th most relevant node,} \\ 0, & \text{otherwise.} \end{cases}$$

Edges are assigned prizes in a similar fashion. The goal is then to find a connected subgraph $S^* = (V_S, E_S)$ that maximizes the total prize while minimizing the cost:

$$S^* = \text{argmax}_{S \subseteq G,\, S \text{ connected}} \left(\sum_{n \in V_S} \text{prize}(n) + \sum_{e \in E_S} \text{prize}(e) - \text{cost}(S) \right),$$

where the cost is defined as:

$$\text{cost}(S) = |E_S| \cdot C_e,$$

with C_e being a predefined cost per edge. To handle cases where an edge's prize exceeds its cost, we introduce a virtual node that preserves the connectivity while avoiding negative costs.

G–Retriever

Answer Generation

Once the relevant subgraph S^* is constructed, it is processed to generate an answer. This step involves two components: graph encoding and text generation.

Graph Encoding

The subgraph S^* is first encoded using a graph encoder such as a Graph Attention Network (GAT) or Graph Transformer. The encoder aggregates the information of the nodes and edges:

$$h_g = \mathrm{POOL}(\mathrm{GNN}(S^*)) \in R^{d_g},$$

where `POOL` represents a mean or sum pooling operation over the node representations, and d_g is the graph encoder's output dimension. This graph-level representation is then projected into the LLM's embedding space using a multilayer perceptron (MLP):

$$\hat{h}_g = \mathrm{MLP}(h_g) \in R^{d_l},$$

where d_l is the hidden dimension of the LLM.

G–Retriever

Textualization and Fusion

The retrieved subgraph is also converted into a natural language description using a *textualization* function:

$$\text{textualize}(S^*),$$

which flattens node and edge information into a readable format. This textualized graph is concatenated with the original query:

$$\text{Input} = [\text{textualize}(S^*); x_q].$$

The concatenated input is then embedded using the LLM's text embedder:

$$h_t = \text{TextEmbedder}([\text{textualize}(S^*); x_q]) \in R^{L \times d_l},$$

where L is the number of tokens.

LLM Generation with Graph Prompt Tuning

Finally, the graph token $\hat{h}_g$ is used as a soft prompt. It is concatenated with the text embedder output h_t and fed into the frozen LLM:

$$p(Y \mid S^*, x_q) = \prod_{i=1}^{r} p\Big(y_i \mid y_{<i}, [\hat{h}_g; h_t]\Big),$$

where $Y = \{y_1, y_2, \ldots, y_r\}$ is the generated answer sequence. Although the LLM parameters remain frozen, the soft prompt (i.e., $\hat{h}_g$) is updated through backpropagation to optimize the performance on graph-based question answering.

Results

Model Configurations: Three main settings are examined:

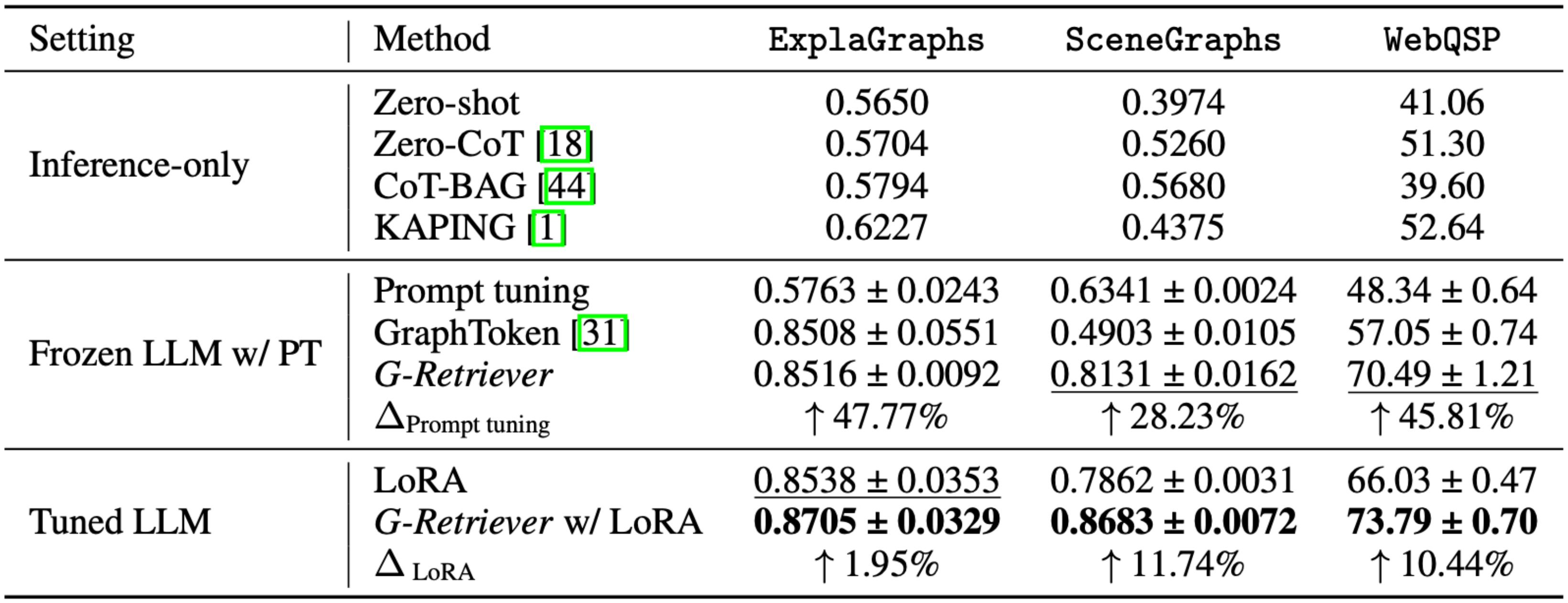
- *Inference-only:* The frozen LLM is directly used for question answering with a textual graph input.
- *Frozen LLM with Prompt Tuning (PT):* Here, only the prompt (soft prompt derived from the graph encoder) is trained while keeping the LLM frozen.
- *Tuned LLM:* The LLM is fine-tuned using LoRA, allowing for more adaptation to the graph-based tasks.

Metrics:

- Accuracy is used for commonsense and scene graph tasks.
- Hit@1 is used for knowledge graph tasks where the top answer's correctness is crucial.

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Results



Setting	Method	ExplaGraphs	SceneGraphs	WebQSP
Inference-only	Zero-shot	0.5650	0.3974	41.06
	Zero-CoT [18]	0.5704	0.5260	51.30
	CoT-BAG [44]	0.5794	0.5680	39.60
	KAPING [1]	0.6227	0.4375	52.64
Frozen LLM w/ PT	Prompt tuning	0.5763 ± 0.0243	0.6341 ± 0.0024	48.34 ± 0.64
	GraphToken [31]	0.8508 ± 0.0551	0.4903 ± 0.0105	57.05 ± 0.74
	G-Retriever	0.8516 ± 0.0092	0.8131 ± 0.0162	70.49 ± 1.21
	$\Delta_{\text{Prompt tuning}}$	$\uparrow$ 47.77%	$\uparrow$ 28.23%	$\uparrow$ 45.81%
Tuned LLM	LoRA	0.8538 ± 0.0353	0.7862 ± 0.0031	66.03 ± 0.47
	G-Retriever w/ LoRA	**0.8705 ± 0.0329**	**0.8683 ± 0.0072**	**73.79 ± 0.70**
	Δ_{LoRA}	$\uparrow$ 1.95%	$\uparrow$ 11.74%	$\uparrow$ 10.44%

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Results

Dataset	Before Retrieval (Avg.)			After Retrieval (Avg.)		
	# Tokens	# Nodes	Min/Epoch	# Tokens	# Nodes	Min/Epoch
`SceneGraphs`	1,396	19	123.1	235 (↓83%)	5 (↓74%)	86.8 (↓29%)
`WebQSP`	100,627	1,371	18.7	610 (↓99%)	18 (↓99%)	6.2(↓67%)

By retrieving only the most relevant nodes and edges, the approach dramatically reduces the number of tokens and nodes processed. This leads to reduced training time – for example, token counts dropped by up to 99% and training time by up to 67% on large datasets.

Results

The method also substantially mitigates hallucination. The retrieved subgraph, being directly grounded in the actual graph, leads to much higher validity of nodes and edges referenced in the answer.

Table 5: Quantitative comparison of hallucination on the SceneGraphs dataset.

	Baseline	*G-Retriever*
Valid Nodes	31%	77%
Valid Edges	12%	76%
Fully Valid Graphs	8%	62%

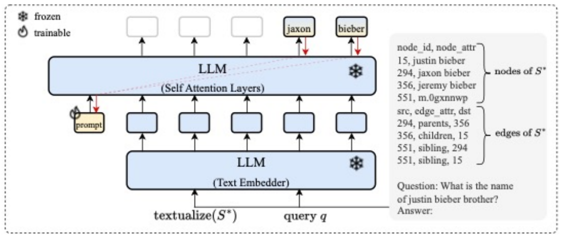
Graph Encoder	WebQSP	ExplaGraphs
GCN [17]	70.70	0.8394
GAT [43]	70.27	0.8430
Graph Transformer [37]	70.49	0.8516

Results

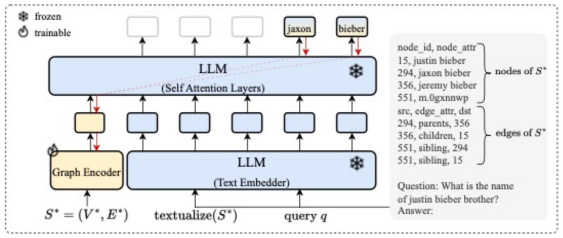
Detailed studies reveal that each component (graph encoder, projection layer, textualized graph, and retrieval mechanism) plays a critical role. Removing any one of them causes a significant drop in performance, emphasizing their complementary contributions.

Table 6: Ablation study on the `WebQSP` dataset.

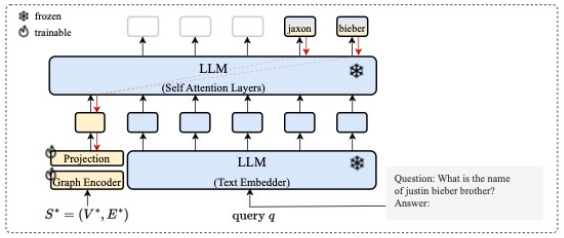
Method	Hit@1	$\Delta_{\textit{G-Retriever}}$
w/o Graph Encoder	54.62 ± 0.78	↓ 22.51%
w/o Projection Layer	69.70 ± 0.68	↓ 1.11%
w/o Textualized Graph	56.96 ± 1.83	↓ 19.19%
w/o Retrieval	63.84 ± 0.41	↓ 9.43%



(a) w/o GraphEncoder



(b) w/o Projection Layer



(c) w/o Textualized Graph

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Limitation

1. The retrieval component in G–Retriever is static, meaning it isn't trainable end–to–end with the rest of the model. This limits the system's ability to adaptively refine retrieval strategies during learning.

2. The reliance on textual representation of graph components may lead to a loss of nuanced structural information, potentially affecting performance on tasks that require deep understanding of complex graph structures.

Conclusion

- **Effective Integration**:
  Combines GNNs and LLMs using a retrieval–augmented framework that grounds answers in relevant subgraphs, thereby reducing hallucination.

- **Comprehensive Benchmark**:
  Introduces a unified GraphQA benchmark covering diverse real–world graph applications, enabling robust evaluation across multiple domains.

- **Scalable Efficiency**:
  Demonstrates significant efficiency gains via targeted subgraph extraction and prompt tuning, ensuring scalability to large textual graphs.

UNIVERSITY OF WATERLOO

FACULTY OF MATHEMATICS

Thank you!