BAG-RAG: Bidirectional Retrieval-Augmented Generation Based on Multi-Layer Semantic Graphs for Budget Auditing QA

Gaofeng Xu[1,2], Runzhe Wang[1,2], Guilin Qi[1,2(✉)], Xiaolong Ye[3], Yongrui Chen[1,2], Yuxin Zhang[1,2], Xinbang Dai[1,2], Yuan Meng[1,2], and Shenwen Zhong[3]

[1] School of Computer Science and Engineering, Southeast University, Nanjing 211189, China
[2] Key Laboratory of New Generation Artificial Intelligence Technology and its Interdisplinary Applications (Southeast University), Ministry of Education, China
{230229361,220232335,gqi,yrchen,zzyx_cs,xbdai,yuan_meng}@seu.edu.cn
[3] China Mobile(Zhejiang) Research & Innovation Institute, China
{yexiaolong,zhongshenwen}@zj.chinamobile.com

Abstract. Government budget auditing is critical for ensuring the legality and rationality of resource allocation, but its hierarchical structure and extensive numerical data present significant challenges for question-answering(QA) systems. While large language models(LLMs) excel in general tasks, they struggle with in budget auditing QA task. **Retrieval-Augmented Generation(RAG)** offers a promising approach by integrating external knowledge, yet traditional RAG methods falter in handling complex hierarchical structure and aggregating semantic relationships from extensive audit materials. To address these limitations, we propose **BAG-RAG, a novel graph-based RAG framework tailored for government budget auditing**. Our approach introduces a **Multi-layered Semantic Graph Construction** method to capture hierarchical structures and establish clear semantic relationships by linking structural nodes and instance nodes. Additionally, we design a **Bidirectional Retrieval Strategy** that combines top-down retrieval of structural nodes for global context and bottom-up aggregation of instance nodes to ensure comprehensive and accurate answers. We validate our framework using a newly curated QA dataset specific to government budget auditing. Experimental results show that BAG-RAG effectively addresses hierarchical complexity and incomplete retrieval, delivering precise and comprehensive answers for budget auditing questions. [1]

Keywords: Budget Auditing QA.· RAG. · Multi-layer Semantic Graph.· Bidirectional Retrieval Strategy

[1] This research was funded by Southeast University-China Mobile Research Institute Joint Innovation Center

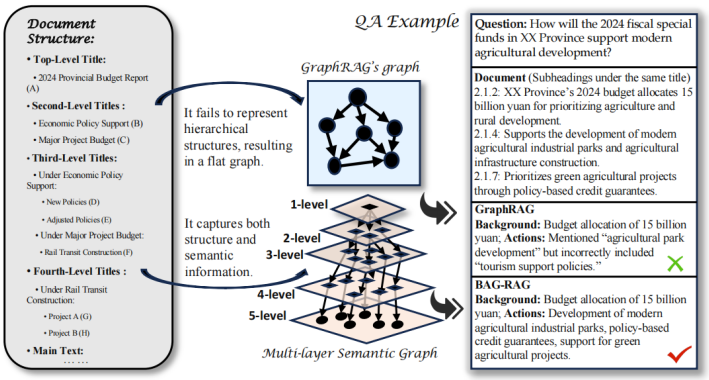

Fig. 1: The difference between the graph constructed by GraphRAG and the Multi-layer Semantic Graph.

1 Introduction

Budget auditing[1] is the cornerstone of government financial management[2], ensuring the legality and rationality of resource allocation. Question-answering tasks[3] in budget auditing significantly enhance audit efficiency by helping auditors quickly identify key information and handle complex numerical analyses. However, existing LLMs[4,5] are limited in this domain due to their lack of expertise. Therefore, incorporating extensive external data to support complex reasoning, enabling precise identification of key information and verification of intricate numerical details, is critical to addressing question-answering challenges in budget auditing.

RAG[6,7,8,9] improves LLMs in question-answering task by dynamically integrating external information, enhancing accuracy and reliability. However, traditional RAG struggles with semantic aggregation when processing split chunks, limiting its performance in complex reasoning. GraphRAG[10,11,12,13] addresses this by combining knowledge graphs with RAG, dynamically building graphs to capture semantic relationships between chunks and identifying question-relevant nodes and paths during retrieval, thereby enhancing logical reasoning.

Although GraphRAG excels in various domains, it faces two critical challenges in government budget auditing. The first is the complexity of hierarchical graph construction, as budget materials with multi-level headings and entries are difficult to represent using GraphRAG's entity-relationship approach, causing semantic confusion and partial retrieval of irrelevant content, as illustrated in Fig. 1. The second is incomplete information retrieval, where GraphRAG focuses on local node associations and neglects global integration result in critical information omissions. This affects the accuracy of numerical calculations and the reliability of data analysis, ultimately limiting its effectiveness in budget auditing QA task.

To address the above challenges, we propose a novel graph-based RAG framework, called **BAG-RAG (Budget Auditing with Graph-based RAG)**, specifically designed for question-answering tasks in government budget auditing. To tackle the complexity of hierarchical graph construction, we introduce a **multi-layer semantic graph construction** method. This approach first extracts hierarchical structure information from documents to generate structural nodes, then splits chunks based on the structure to extract instance nodes, and finally associates relevant structural and instance nodes through a designed linking strategy, constructing a multi-layer semantic graph with clear hierarchy and precise semantics. To address the challenge of incomplete information retrieval, we proposes a **bidirectional retrieval strategy**. This strategy employs a top-down retrieval to sequentially explore hierarchical structural nodes, capturing global information relevant to the question, followed by bottom-up generation to recursively aggregate lower-level instance nodes along graph paths, achieving cross-level semantic integration for comprehensive and accurate answers. To validate the framework's effectiveness, we curated a real-world question-answer dataset from government budget auditing materials, creating a RAG question-answering dataset tailored to this domain. Experimental results demonstrate that BAG-RAG effectively resolves the challenges of complex hierarchical structure and incomplete retrieval, providing precise and comprehensive answers for budget auditing questions. In summary, the contributions of this paper include:

1. **Proposing the BAG-RAG Framework** This paper introduces BAG-RAG, a novel graph-based RAG framework specifically for budget auditing QA tasks, combining multi-layer semantic graph construction and bidirectional retrieval strategy to address the complexity of hierarchical graph construction and incomplete information retrieval.
2. **Design of Multi-layered Semantic Graph Construction and Bidirectional Retrieval Strategy** We propose a unique multi-layer semantic graph construction technique that links structural and instance nodes, creating a clear and precise hierarchy. Additionally, we introduce a bidirectional retrieval strategy that combines top-down filtering for global information with bottom-up aggregation for accurate and comprehensive answers.
3. **Constructing a High-Quality Dataset and Validating the Framework** To evaluate the performance of BAG-RAG, we built a high-quality QA dataset specifically for government budget auditing. Experiments on both public and custom datasets indicate that BAG-RAG outperforms existing methods across multiple metrics, achieving state-of-the-art performance.

2 Preliminaries

2.1 Multi-layer Semantic Graph

A multi-layered semantic graph G is a graph-structured dataset composed of structural nodes, instance nodes, and the relationships between these nodes. It

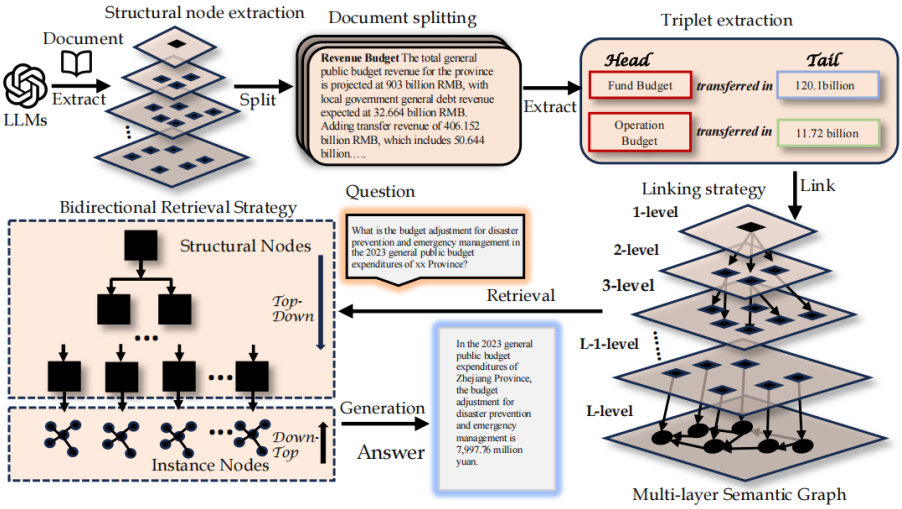

Fig. 2: The overall workflow of BAG-RAG.

is utilized to jointly represent both the structural and semantic information of documents. Formally defined as:

$$G = (V_s, V_f, R). \tag{1}$$

Here, $V_s = \{s_1, s_2, \ldots, s_i\}$ represents the set of structural nodes, with each node $s_i \in V_s$ denoting structural information extracted from document. $V_f = \{f_1, f_2, \ldots, f_i\}$ indicates the set of instance nodes, where each node $f_i \in V_f$ represents semantic information extracted from the document text. $R = \{r \mid r \in R, r : (x, y) \rightarrow, x, y \in V_s \cup V_f\}$ defines the set of relationships, used to describe directed relations between nodes x and y, with R as the set of relationship types and x, y belonging to the node set $V_s \cup V_f$.

2.2 Problem Formulation

Our objective is to retrieve and generate answers to user questions from the input documents. Specifically, given the input document $T = \{t_1, t_2, \ldots, t_n\}$, we construct a multi-layer semantic graph G_T for document T using the graph construction technique F, i.e., $G_T = F(T)$. On this basis, we utilize the bidirectional retrieval technique H, proposed in this paper, to search G_T based on the user question Q, and generate the final answer A_Q, such that $A_Q = H(Q, G_T)$.

3 Methodology

The overall workflow of BAG-RAG is illustrated in Fig. 2. Initially, structural nodes are extracted from the document's hierarchical structure. Subsequently,

split the document into chunks based on this hierarchical structure. The chunks undergo triple extraction, creating the corresponding instance nodes and relationships. Finally, structural nodes are linked to instance nodes using a linking strategy, resulting in a multi-layer semantic graph that is both hierarchically clear and semantically precise. Based on this semantic graph, we propose a bidirectional retrieval strategy that combines top-down global filtering with bottom-up instance node aggregation to generate comprehensive and accurate answers.

3.1 Multi-layer Semantic Graph Construction

Structural Node Extraction Government budget auditing documents typically include multi-tiered entries. To construct the structure of graphs with clear hierarchies, we propose a method that extracts documents headings and entries, generating corresponding structural nodes and relationships based on their hierarchical structure. Given an input document T, document structure processing tools extract a set of headings and entries $H = \{h_1, h_2, \ldots, h_m\}$, where h_k represents the k-th heading. Each heading is mapped to a corresponding structural node $s_i = \text{map}(h_i)$, where $\text{map}(*)$ denotes the mapping relationship between headings or entries and structural nodes. Structural nodes are categorized by their hierarchical relationships into different levels. Suppose there are L structural levels in document T; the set of structural nodes V_s is divided into levels as $V_s^l = \{s_k^l \mid \text{level}(h_k) = l,\ l = 1, 2, \ldots, L\}$. Here, V_s^l denotes the set of structural nodes at the l-th level, s_k^l represents a structural node at the l-th level, and level (h_k) indicates the hierarchical level of heading h_k.

Document Splitting We propose a hierarchy-based chunking method to mitigate context loss and the separation of entities and relationships, ensuring semantic integrity while enabling efficient construction of multi-layer semantic graphs. This method splits the document into chunks aligned with its hierarchical structure, adhering to the contextual limits of LLMs. Specifically, the document is first divided by its lowest-level headings, and if the resulting chunks exceed the model's maximum input length, further divided into smaller chunks.

Further splitting presents two challenges: context loss and the separation of related entities and relationships. Context loss occurs when chunks lose meaning due to missing surrounding context, while entity and relationship separation happens when triplet components are split, causing incomplete semantics. To address these issues, we propose a splitting strategy that adds triplet information from the preceding chunk to maintain context and semantic completeness. To avoid entity or relation separation, we append the last few lines of the preceding chunk, $\text{sen}(i-1)$, to each current chunk to supplement any potential missing information. Ultimately, the chunk Chk_i input into the LLM is defined as follows:

$$Chk_i = \{\text{tri}(i-1), \text{text}(i) + \text{sen}(i-1)\} \tag{2}$$

where $\text{text}(i)$ represents the original content of the current splitting text, $\text{tri}(i-1)$ denotes the triplet information from the previous chunk, and $\text{sen}(i-1)$ signifies the last few lines of the preceding chunk.

Triple Extraction Upon acquiring input chunks Chk_i, we utilize LLMs to extract triplets[14,15] in two phases: entity extraction and relation extraction. This process furnishes detailed candidate instance nodes and their interrelations for the construction of a multi-layer semantic graph.

Entity extraction begins with identifying entity mentions within Chk_i, assuming the entities set is $E = \{e_1, e_2, \ldots, e_n\}$. During this phase, we require the LLM to accomplish the following tasks:

1. Extract entity mentions e_i in text(i) and obtain the set of sentences containing the entity $S(e_i)$, where $S(e_i) = \{s \mid s \in \text{text}(i), e_i \in s\}, \forall e_i \in E$.
2. For each entity e_i, provide a description $D(e_i)$, where $D(e_i) = \text{sum}(S(e_i))$, $\forall e_i \in E$, with $\text{sum}(S(e_i))$ being a summary of the sentence set $S(e_i)$.

During relation extraction, use the relevant sentences $S(e_i)$ marked during entity extraction as context, combined with entities E. For each relation r, the LLM generates a concise description $D(r)$ to elucidate the semantics of the relation, where $D(r) = \text{sum}(S(e_i, e_j))$, with $S(e_i, e_j) = \{s \mid s \in \text{text}(i), e_i, e_j \in s\}$. Results are represented as triplets $F = (e_i, r_k, e_j)$, where $e_i, e_j \in E$, $r_k \in R$.

Linking Strategy For the extracted structural nodes and the triplets, we have designed a linking strategy that effectively associates them, ultimately constructing a multi-layer semantic graph. The linking strategy consists of three steps: linking structure nodes, linking instance nodes, and linking structure nodes with instance nodes.

Linking Structure Nodes For any two structural nodes s_i and s_j, if a parent-child hierarchical relationship exists between them, a directed edge r_{ij} is defined to link the two nodes.

Linking Instance Nodes The linking of instance nodes primarily serves to connect triplets from different chunks. If two chunks Chk_i and Chk_j belong to the same lowest-level heading, merge identical entities and relationships. In the merged triplets, the entities represent instance nodes, and the relations denote the relationships between them. If the chunks Chk_i and Chk_j belong to different lowest-level headings, the triplets within each chunk are considered as instance nodes and their relationships. No links are established between the instance nodes of different chunks Chk_i and Chk_j, as content under different headings is typically semantically independent.

Linking Structure Nodes with Instance Nodes The connection between structural nodes and instance nodes is established based on the document's partitioning relationships. For each lowest-level heading h_k, the corresponding set of instance nodes V_f^k is associated with the corresponding structural node s_k. Specifically, an instance node $f_i \in V_f^k$ is linked with the structural node s_k.

3.2 Bidirectional Retrieval Strategy

After constructing the multi-layer semantic graph, we propose an innovative bidirectional retrieval strategy: top-down filtering of structural nodes captures global information related to the user's question, followed by bottom-up aggregation of relevant instance nodes for accurate, comprehensive answers.

Top-Down Retrieval Strategy Given the multi-layer semantic graph G with L hierarchical levels, where structural nodes exist up to level $L-1$ and instance nodes occupy level L, we employ a top-down retrieval process to ensure comprehensiveness information retrieval. This involves filtering structural nodes V_s^l at each level $l = 1, 2, \ldots, L-1$, starting from the top and progressing downward to identify the structural nodes $S^{(L-1)}$ pertinent to the user's questions.

Specifically, starting from the first level, the LLM evaluates the relevance of structural nodes in this level to the question Q, selecting the relevant node set P^1, defined as $P^1 = \{s_i^1 | \text{LLM}(Q, s_i^1) = \text{True}, s_i^1 \in V_s^1\}$.

For the selected set P^1, we retrieve the connected structural nodes at the next level, $V_{s-\text{linked}}^2 = \{s_j^2 | (s_j^2, s_i^1) \in R, s_i^1 \in P^1\}$. From $V_{s-\text{linked}}^2$, nodes relevant to Q are further filtered to form P^2. To avoid omissions, we conduct a secondary screening on the remaining nodes $L^1 = \{s_k^2 | s_k^2 \in V_s^2 \wedge s_k^2 \notin P^2\}$. If any overlooked node s_m^2 is found relevant to Q, it is added to P^2, and the subsequent connected structural nodes are retrieved.

This process repeats until reaching the $L-1$ level, resulting in the retrieval of structural nodes $S^{(L-1)} = \bigcup_{k=1}^{L-1} P^k$. Based on $S^{(L-1)}$, the corresponding instance nodes set $F^i = \{f_i | (s_k^{(L-1)}, f_i) \in R, s_k^{(L-1)} \in P^{(L-1)}\}$ is obtained, thereby supporting the bottom-up generation phase. The mathematical expression for relevant structural nodes P^l at each layer is:

$$P^l = \begin{cases} \{\ s_i^1 \in V_s^1 | \mathsf{Rel}(s_i^1, \mathcal{Q}) > \tau\},\ l = 1 \\ \{\ s_i^l \in V_s^l | \mathsf{Rel}(s_i^l, \mathcal{Q}) > \tau, S_i^l \in V_{s-linked}^2\ \} \cup \mathsf{Miss}(L^l, \mathcal{Q}), 1 < l < L \end{cases} \tag{3}$$

where $\text{Rel}(s_i^l, Q)$ denotes the relevance score of structural node s_i^l to the user's question Q, with τ as the relevance threshold. $\text{Miss}(L^l, Q)$ indicates initially overlooked structural nodes in level l that are relevant to Q.

Bottom-Up Generation Strategy In the bottom-up generation phase, we draw inspiration from the GraphRAG framework, employing the hierarchical relationships of instance nodes to recursively aggregate information layer by layer, ultimately generating the final answer. Specifically, we first perform community detection on the instance node set F^i, resulting in a multi-layered node structure. Let the partitioning result be $\{L_1, L_2, \ldots, L_h\}$, where L_k represents the set of instance nodes at the k-th layer, and h denotes the total number of layers.

The generation process begins at the lowest layer L_h and recursively aggregates the information of instance nodes layer by layer. For each instance node

$f_j \in L_k$, its context is defined as $\text{Context}(f_j) = \bigcup_{f_i \in \text{linked}(f_j)} \text{Info}(f_i)$, where $\text{linked}(f_j)$ denotes the set of nodes directly connected to node f_j in layer L_k, and $\text{Info}(f_i)$ represents the semantic information contained within node f_i.

To ensure the relevance of the aggregated information, we introduce semantic similarity $\text{Rel}(f_j, Q)$, which measures the relevance of an instance node f_j to the user question Q. This is defined as $\text{Rel}(f_j, Q) = \text{Sim}(v(f_j), v(Q))$, where $\text{Sim}(.,.)$ represents a semantic similarity measure (such as cosine similarity), and $v(f_j)$ and $v(Q)$ are the semantic vector representations of the instance node and the user question, respectively. Only nodes with semantic similarity exceeding a threshold τ_Q are retained, forming the filtered context set: $\text{Relevant}(L_k) = \{f_j \in L_k \mid \text{Rel}(f_j, Q) \geq \tau_Q\}$.

Through layer-by-layer filtering and recursive aggregation, the complete semantic context $\text{Context}(Q)$ is ultimately constructed:

$$\text{Context}(\mathcal{Q}) = \bigcup_{k=1}^{h} \bigcup_{\text{Relevant}(L_k)} \text{Info}(f_j) \tag{4}$$

Finally, a response A to the question Q is generated using the semantic context assembled by the ensemble of LLMs:

$$A = \text{LLM}(\mathcal{Q}, \text{Context}(\mathcal{Q})) \tag{5}$$

4 Experiments

4.1 Dataset

To evaluate the proposed BAG-RAG framework, we utilized the ChatGLM LLM FinTech dataset[2], designed for RAG tasks in the FinTech domain, which includes annual reports of publicly listed companies from 2019 to 2021. Additionally, we extracted 7,844 real-world QA pairs from stock evaluation websites to form the FinTec-QA test set, comprising 6,745 numerical and 1,099 non-numerical QA pairs. Meanwhile, we constructed a RAG dataset based on government budget auditing materials, covering budget revenues, expenditures, debt management, and investment project budgets across various departments. In collaboration with budget auditing experts, we manually compiled the Gba-QA dataset, consisting of 186 numerical and 27 non-numerical QA pairs, to evaluate BAG-RAG's performance in government budget auditing QA tasks.

4.2 Experimental Setting

Baseline We evaluated various RAG methods across three LLMs: Llama3 (70B), Gemini-pro, and GPT-4. Our approach was compared with standard RAG implemented by LangChain and GraphRAG by Microsoft Azure, using the same RAG data and test sets to ensure fairness.

[2] `https://modelscope.cn/datasets/modelscope/chatglm_llm_fintech_raw_dataset`

Table 1: Accuracy (%) of LLMs using different retrieval methods. The best results are highlighted in bold.

Technology	Dataset	Model	Acc_{num}	Acc_{sem}	Acc_{total}
RAG	FinTec-QA	Llama3-70B	70.1	83.7	72.0
		Gemini-pro	64.8	79.4	66.8
		GPT-4	72.4	85.8	74.3
	Gba-QA	Llama3-70B	55.6	72.2	57.7
		Gemini-pro	53.2	66.7	54.9
		GPT-4	60.5	72.6	62.0
GraphRAG	FinTec-QA	Llama3-70B	71.5	82.3	73.0
		Gemini-pro	66.9	78.9	68.6
		GPT-4	73.3	87.4	75.3
	Gba-QA	Llama3-70B	64.0	73.5	65.2
		Gemini-pro	60.8	71.6	62.2
		GPT-4	69.7	78.5	70.8
BAG-RAG	FinTec-QA	Llama3-70B	78.7	88.9	80.1
		Gemini-pro	73.8	79.5	74.6
		GPT-4	**78.2**	**91.0**	**80.0**
	Gba-QA	Llama3-70B	77.4	75.8	77.2
		Gemini-pro	72.6	72.0	72.5
		GPT-4	**78.1**	**83.4**	**78.8**

Metrics Model performance was assessed using **Accuracy**. QA pairs were divided into numerical and non-numerical categories based on whether the answers included numerical values. Semantic similarity between reference and generated answers was measured with BERTScore, using a threshold of 0.85. For numerical QA pairs, correctness required both accurate numerical values and semantic similarity; for non-numerical QA pairs, only semantic similarity was required. Accuracy was computed separately for each category, and a weighted average served as the comprehensive evaluation metric, as defined by formula 6.

$$Acc_{total} = w_{num} \cdot Acc_{num} + w_{sem} \cdot Acc_{sem} \tag{6}$$

where $w_{num} = \frac{|Q_{num}|}{|Q_{total}|}, w_{sem} = \frac{|Q_{sem}|}{|Q_{total}|}$. Acc_{num} is the accuracy for numerical QA pairs, and Acc_{sem} is the accuracy for non-numerical QA pairs. The symbol $|\cdot|$ represents the count. Specifically, $|Q_{num}|$ denotes the number of numerical QA pairs, $|Q_{sem}|$ denotes the number of non-numerical QA pairs, and $|Q_{total}|$ represents the total number of QA pairs.

Detail The Llama3(70B) model was used to construct the multi-layer semantic graph. In baseline methods RAG and GraphRAG, documents were split into chunks with a maximum length of 128 to comply with LLM input constraints. RAG retrieved the top three relevant documents for each question as evidence for inference, while GraphRAG selected the top three relevant nodes and their neighboring nodes. All models used a unified prompt template, with a sampling temperature of 0.7 to balance diversity and stability.

4.3 Results

The experimental results in Table 1 evaluate the performance of RAG, GraphRAG and the proposed BAG-RAG framework on FinTec-QA and Gba-QA datasets across various LLMs, measured by Acc_{num}, Acc_{sem}, and Acc_{total}. BAG-RAG consistently outperforms RAG and GraphRAG. On FinTec-QA, BAG-RAG achieves the highest overall accuracy with GPT-4 scoring 80.0%, surpassing GraphRAG 75.3% and RAG 74.3%. On Gba-QA, BAG-RAG achieves 78.8% with GPT-4, exceeding GraphRAG 70.8% and RAG 62.0%. The significant improvements on Gba-QA highlight BAG-RAG's superior integration of numerical precision and semantic understanding, particularly in addressing complex cross-layer reasoning tasks with GPT-4.

4.4 Human Evaluation of QA Tasks in Budget Auditing

Based on human evaluations of the Gba-QA dataset, we compared the QA performance of BAG-RAG, RAG, and GraphRAG. Seven audit experts scored the models on relevance (Pert.), understandability (Und.), and correctness (Cor.) using a 100-point scale. As shown in Table 2, BAG-RAG outperformed RAG and GraphRAG across all metrics, particularly in numerical tasks. These results highlight BAG-RAG's strength in hierarchical reasoning, enabling precise numerical and semantic understanding in budget audit tasks.

Table 2: Human evaluation on Gba-QA dataset.

Model	Category	*Pert.*	*Und.*	*Cor.*
RAG	numerical	70	65	68
	non-numerical	73	68	70
GraphRAG	numerical	78	74	76
	non-numerical	80	76	78
BAG-RAG	numerical	88	85	86
	non-numerical	90	86	88

4.5 Ablation Results

We conducted an ablation study on the Gba-QA dataset using GPT-4 as the base model to analyze the contributions of BAG-RAG's components (Table 3). The full BAG-RAG framework (Model_1) achieved the best performance. Removing Document Splitting (Model_2) led to a noticeable drop, especially in *Accnum* (-3.6%), highlighting its role in numerical precision. Removing Bidirectional Retrieval (Model_3) caused further declines, emphasizing its importance for capturing global and local reasoning. Replacing the Multi-layer Semantic Graph with GraphRAG (Model_4) resulted in the lowest scores, underscoring its critical role in representing hierarchical relationships. These results confirm all components

Table 3: Ablation study of BAG-RAG evaluated by accuracy (%).

Name	Detail	Acc_{num}	Acc_{sem}	Acc_{total}
$Model_1$	**BAG-RAG**	**78.1**	**83.4**	**78.8**
$Model_2$	-Document Splitting	74.5	81.3	75.4
$Model_3$	-Bidirectional Retrieval	72.2	80.7	73.3
$Model_4$	-Multi-layer Semantic Graph	69.7	78.5	70.8

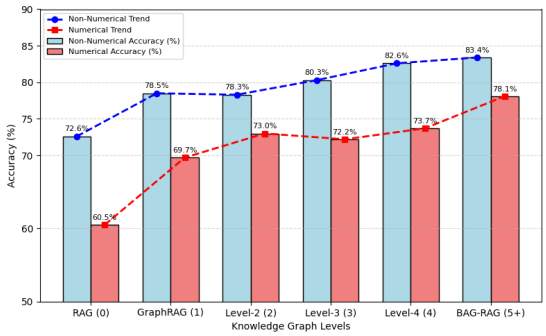

Fig. 3: Impact of Knowledge Graph Levels on RAG, GraphRAG, and BAG-RAG .

are essential, with the Multi-layer Semantic Graph and Bidirectional Retrieval being particularly critical for complex reasoning in budget auditing.

4.6 Case Study

To analyze QA accuracy with increasing knowledge graph levels, we evaluated Gba-QA using GPT-4, as shown in Fig. 3. The study compares numerical and non-numerical accuracy from RAG (level 0) to BAG-RAG (level 5+). Results show a general upward trend, with BAG-RAG achieving the best performance. Non-numerical accuracy improves steadily, while numerical accuracy fluctuates at intermediate levels (3 and 4), highlighting the importance of multi-layer node construction. These findings demonstrate the effectiveness of structured nodes in enhancing QA performance, especially for numerical reasoning in budget auditing.

5 Conclusion

This paper proposes the BAG-RAG framework for government budget auditing QA task, integrating multi-layer semantic graph construction and bidirectional retrieval strategies to effectively represent hierarchical relationships and achieve precise answer generation. Experiments on real-world and public datasets demonstrate that BAG-RAG significantly outperforms baseline methods, particularly in handling complex numerical computations.

References

1. Laurence Ferry, Henry Midgley, and Pasquale Ruggiero. Regulatory space in local government audit: An international comparative study of 20 countries. *Public Money & Management*, 43(3):233–241, 2023.
2. Tobias Polzer, Isabella M Nolte, and Johann Seiwald. Gender budgeting in public financial management: a literature review and research agenda. *International Review of Administrative Sciences*, 89(2):450–466, 2023.
3. Ziyuan Zhuang, Zhiyang Zhang, Sitao Cheng, Fangkai Yang, Jia Liu, Shujian Huang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. Efficientrag: Efficient retriever for multi-hop question answering. pages 3392–3411, 2024.
4. Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. *World Wide Web*, 27(5):58, 2024.
5. Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, 36:50117–50143, 2023.
6. Patrick S. H. 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. 2020.
7. Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on RAG meeting llms: Towards retrieval-augmented large language models. pages 6491–6501, 2024.
8. Zhangchi Feng, Dongdong Kuang, Zhongyuan Wang, Zhijie Nie, Yaowei Zheng, and Richong Zhang. Easyrag: Efficient retrieval-augmented generation framework for automated network operations. *CoRR*, abs/2410.10315, 2024.
9. Weijian Xie, Xuefeng Liang, Yuhui Liu, Kaihua Ni, Hong Cheng, and Zetian Hu. Weknow-rag: An adaptive approach for retrieval-augmented generation integrating web search and knowledge graphs. *CoRR*, abs/2408.07611, 2024.
10. Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.
11. Xishi Zhu, Xiaoming Guo, Shengting Cao, Shenglin Li, and Jiaqi Gong. Structugraphrag: Structured document-informed knowledge graphs for retrieval-augmented generation. 4(1):242–251, 2024.
12. Junde Wu, Jiayuan Zhu, Yunli Qi, Jingkun Chen, Min Xu, Filippo Menolascina, and Vicente Grau. Medical graph rag: Towards safe medical large language model via graph retrieval-augmented generation. *arXiv preprint arXiv:2408.04187*, 2024.
13. Dawei Li, Shu Yang, Zhen Tan, Jae Young Baik, Sukwon Yun, Joseph Lee, Aaron Chacko, Bojian Hou, Duy Duong-Tran, Ying Ding, Huan Liu, Li Shen, and Tianlong Chen. DALK: dynamic co-augmentation of llms and KG to answer alzheimer's disease questions with scientific literature. pages 2187–2205, 2024.
14. Bowen Zhang and Harold Soh. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. pages 9820–9836, 2024.
15. Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. Llms for knowledge graph construction and reasoning: recent capabilities and future opportunities. *World Wide Web (WWW)*, 27(5):58, 2024.