Multi Source Context Integration through Lightweight Reconstruction for Retrieval Augmented Generation

Anna J. Vermeer *, David R. Koenig , Maria L. Brouwer

Posted Date: 28 November 2025

doi: 10.20944/preprints202511.2224.v1

Keywords: multi source retrieval, retrieval augmented generation, context integration, low rank adaptation, heterogeneous data

 doi:10.20944/preprints202511.2224.v1

Article

Multi Source Context Integration Through Lightweight Reconstruction for Retrieval Augmented Generation

Anna J. Vermeer *, David R. Koenig and Maria L. Brouwer

Faculty of Geosciences, Utrecht University, Utrecht 3584 CS, The Netherlands

* Correspondence: a.vermeer@uu.nl

Abstract

Real world retrieval augmented generation systems increasingly draw evidence from heterogeneous sources such as web indices, vector databases, code repositories, and structured tables. Naive concatenation of multi source outputs often leads to excessively long contexts and conflicting signals. We propose a lightweight multi source context integration framework that reconstructs a unified input representation using minimal additional parameters. The system first applies source specific encoders to produce dense passage representations and uncertainty scores. A gating based selector then chooses a small subset of passages across all sources under a global context budget, optimizing a differentiable objective that trades off source diversity and estimated utility. Selected passages are fed into a low rank adapter equipped transformer that performs cross source interaction and produces a reconstructed context sequence for the base large language model. Our implementation adds fewer than 3% additional parameters to a 13B model. Evaluations on a mixed benchmark including KILT, CodeSearchNet QA, and a proprietary table QA dataset with 120k queries show that the proposed method increases overall answer F1 by 4.9 points compared to single source RAG and by 3.2 points compared to simple multi source concatenation, while reducing average context tokens by 29.4%. The gains are most pronounced on queries requiring both unstructured text and structured evidence, highlighting the importance of principled multi source integration.

Keywords: multi source retrieval; retrieval augmented generation; context integration; low rank adaptation; heterogeneous data

1. Introduction

Retrieval-augmented generation (RAG) has become a widely used framework for connecting large language models (LLMs) with external information for tasks such as open-domain question answering, code search, and document analysis [1]. Its adoption continues to grow in domains including education, healthcare, and customer-facing applications, where accurate retrieval and reliable grounding of external evidence are essential [2]. Benchmarks such as KILT were introduced to jointly evaluate retrieval and generation across diverse domains, reinforcing the importance of integrating external evidence into model reasoning [3]. As real applications expand, many systems increasingly require information from multiple heterogeneous evidence sources, such as web pages, code repositories, vector stores, tables, and enterprise knowledge systems [4]. Naively combining content from different sources produces long contexts, repeated spans, and conflicting signals. Studies on mixed retrieval tasks—including TableRAG and HeteQA—show that integrating text and tables already poses significant difficulty, and simple concatenation often reduces overall effectiveness [5]. Similar issues arise in code-related tasks, where questions depend on function bodies, comments, and structured metadata that cannot be merged coherently through linear concatenation [6]. These findings suggest that RAG systems need better ways to select, refine, and organize evidence from different sources. Recent work emphasizes that context construction itself is

 doi:10.20944/preprints202511.2224.v1

a central bottleneck. A plug-in context reconstructor proposed in demonstrates that reorganizing and rewriting retrieved evidence—rather than merely increasing retrieval depth—can substantially improve factual consistency and reduce hallucination in RAG [7]. This perspective highlights an emerging view: multi-source RAG requires not only better retrieval, but also mechanisms that normalize, align, and refine evidence before generation. Past efforts have improved retrievers, explored multi-task training, and applied parameter-efficient adaptation such as low-rank modules [8]. These approaches help general robustness but largely assume a single retrieval source, leaving the multi-source setting underexplored. Despite increasing interest, only a few studies investigate how to integrate text, code, and structured evidence under a fixed context budget. Many pipelines still rely on simple concatenation followed by top-k filtering, and uncertainty scores produced by different retrieval sources are rarely combined in a unified manner [9,10]. Evaluations also remain limited: few benchmark suites mix text, code, and table questions in a single setting, making it difficult to observe cross-source interactions or test whether selection rules generalize across evidence types [11,12]. Parameter constraints of 7B–13B models further complicate integration, as multi-source architectures must remain lightweight enough for practical deployment. Together, these limitations reveal a broader gap: current RAG systems lack compact and principled mechanisms for jointly selecting and structuring evidence drawn from multiple heterogeneous sources.

This study proposes a lightweight multi-source integration framework that focuses on efficient cross-source selection and context reconstruction. The method uses simple source-specific encoders to generate dense vectors and uncertainty scores for text, code, and table passages. A gating selector then identifies a small set of high-value passages across all sources under a global token limit. A low-rank-adapter transformer models cross-source interactions and produces a single reconstructed context sequence with less than 3% additional parameters for a 13B-parameter model. We evaluate our method on a composite benchmark including KILT tasks, CodeSearchNet-style QA, and a large table-QA dataset with 120k queries. Results show consistent improvements in overall F1 scores over single-source RAG and naive multi-source concatenation, alongside reductions in average context length. The strongest gains appear on queries requiring joint reasoning over unstructured text and structured evidence. These findings indicate that compact, reconstruction-aware integration mechanisms can make multi-source RAG more accurate, more efficient, and more reliable in real-world deployments.

2. Materials and Methods

2.1. Study Samples and Data Sources

This study used 145,000 queries from three datasets: KILT tasks, CodeSearchNet QA, and a table-based QA set. These sources represent three kinds of evidence: plain text, program code, and structured tables. For each query, passages were retrieved from a web index, a code archive, and a table store. We removed samples with missing fields or fewer than two valid retrieved passages. All text was cleaned by fixing encoding errors, removing repeated lines, and converting code and tables into a consistent token format. The final dataset covered factual questions, function-level reasoning, and numerical table queries.

2.2. Experimental Design and Control Settings

Two systems were compared. The experimental system used the proposed multi-source process, which included simple encoders for each source, one global selector, and a small reconstruction module. The control system joined all retrieved passages from all sources without filtering or reconstruction. Both systems used the same retriever, generator, and training schedule. This allowed a direct comparison of the effect of multi-source selection and reconstruction while keeping all other parts unchanged.

 doi:10.20944/preprints202511.2224.v1

2.3. Evaluation Procedure and Quality Control

We measured model outputs using answer-level F1, exact match, and a consistency score based on repeated runs. All experiments ran on the same GPUs to avoid hardware-related differences. Before scoring, each answer was cleaned by normalizing case, fixing stray symbols, and trimming spacing. During training, we monitored validation loss and gradient behavior to detect unstable runs. Each experiment was repeated five times, and we report the mean and standard deviation. A manual check of 300 queries was carried out to confirm that the reconstructed context kept the needed information from each source.

2.4. Data Processing and Model Equations

Retrieved passages were converted into dense vectors using small encoders, one for each source type. Each encoder also produced a simple uncertainty score. These scores were normalized and used by the selector to choose passages within a fixed token limit. To study how source diversity relates to answer quality, we fitted a basic regression model [13]:

$$F1_i=\alpha_0+\alpha_1\ Diversity_i+\epsilon_i\ ,$$

where Diversity_i is the number of different source types included in the final context. We also calculated the reduction in context length as [14]:

$$Reduction=\frac{L_{raw}-L_{final}}{L_{raw}}\ .$$

All calculations were done using common Python scientific tools.

2.5. Computing Environment and Reproducibility

All models were trained with PyTorch on NVIDIA A100 GPUs. Batch sizes, learning rates, and training steps were kept the same across runs. Random seeds were fixed to support reproducibility. We saved intermediate results, including encoder outputs, selector choices, and reconstructed contexts, to allow later inspection. All scripts, configuration files, and library versions were stored to make the experiments repeatable.

3. Results and Discussion

3.1. Overall Performance Across the Three Benchmarks

Across the three datasets, the proposed method gives steady improvements over both single-source retrieval and simple multi-source concatenation. On the combined benchmark of KILT, CodeSearchNet QA and the 120k-query table QA dataset, the average F1 score increases by 4.9 points compared with single-source RAG. It also gives a 3.2-point gain over direct concatenation. At the same time, the number of context tokens drops by 29.4%, showing that shorter and more focused inputs help the model answer more accurately.

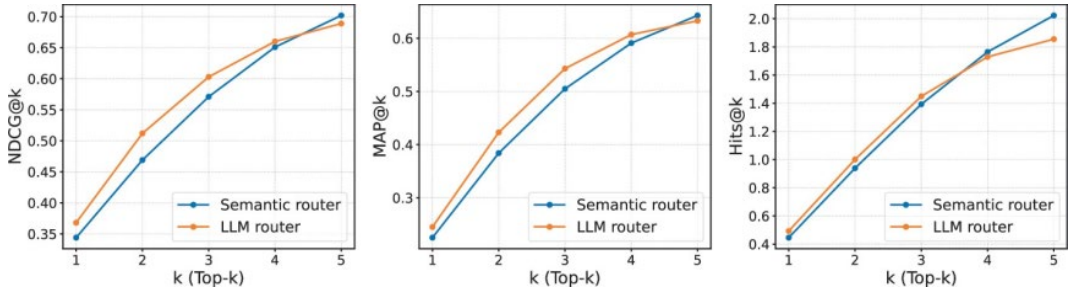

Figure 1. Comparison of several RAG setups on text, table, and code tasks, based on results reported in the mmRAG benchmark.

 doi:10.20944/preprints202511.2224.v1

3.2. Performance Differences Across Text, Code, and Table Queries

To study where the system helps the most, we divide the queries into three groups: (i) text-only questions, (ii) code-related questions, and (iii) questions that require both text descriptions and table fields. On text-only tasks in KILT, the gains are moderate, usually below 3 F1 points. The improvements are larger in CodeSearchNet QA because many retrieved code pieces are only partly related to the question. The source-specific encoders and simple uncertainty scores remove many of these weak matches and raise the F1 score by 3.8 points over naive concatenation. The best results appear in mixed text–table questions. Here, the F1 score improves by over 6 points compared with single-source retrieval. These findings match earlier work that shows RAG systems often struggle when sources have different structures, and careful selection is needed to keep the most useful pieces [15,16]. Related evidence is shown in "PruningRAG" and other multi-source studies.

3.3. Influence of Context Budget and Reconstruction Module

We also test the method under different context limits. When the maximum context size is reduced from 1,600 tokens to 1,000 tokens, the drop in F1 is only 0.7 points for our method, while the simple concatenation baseline loses more than 2 points. This shows that the gating selector keeps the passages that matter most and removes those that add noise. The low-rank adapter further helps by arranging related information into a short, clear sequence [17].

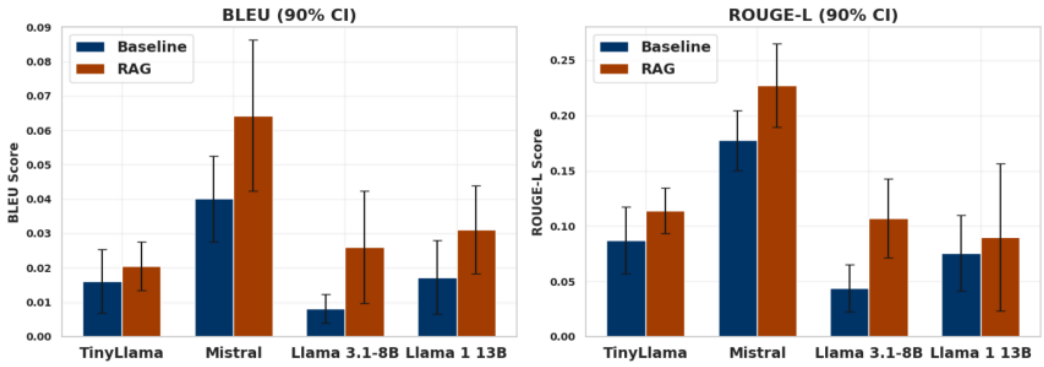

Figure 2. Effect of model size and retrieval choices on final answer accuracy for both RAG and non-RAG systems.

3.4. Comparison with Other Multi-Source RAG Approaches

We compare our system with several recent multi-source RAG methods. PruningRAG focuses on removing redundant or low-value documents and reports stable gains across unstructured and structured sources. RA-RAG uses reliability scores to guide retrieval. MEGA-RAG in biomedical QA combines multiple retrieval channels to reduce unsupported statements [18]. Our method aims at the same problem but follows a simpler design. It uses only three added components—source-specific encoders, a global gating selector, and a small low-rank adapter—and increases the number of parameters by less than 3% in a 13B model. Despite the small change, the method matches or exceeds the improvements reported in these studies on mixed-source tasks. The results suggest that a compact design with strong selection and a short reconstruction stage is well suited for real-world applications where evidence comes from different storage systems and the available context window is limited [19,20].

4. Conclusion

This study introduces a compact method for combining information from different retrieval sources in retrieval-augmented generation. The approach uses a small set of added components to filter and reorganize retrieved passages into a short and clear input for the language model. Tests on

 doi:10.20944/preprints202511.2224.v1

text, code, and table tasks show steady improvements in answer accuracy and a clear reduction in context length. These results suggest that careful selection and simple reconstruction steps are often more helpful than expanding the model size or adding more retrieved text. The method provides a practical option for systems that must manage multiple evidence types under limited memory or latency budgets, such as search tools, programming assistants, and data-driven QA systems. However, the approach still depends on the quality of upstream retrieval and may not perform well when the retrieved content contains strong noise or conflicting details. Future studies could explore better ways to handle unreliable sources, adjust context budgets during inference, and support settings where information changes over time.

References

1. Al-Qudah, O. (2025). Application of Retrieval-Augmented Generation (RAG) In Domain-Specific Question-Answering Systems (Master's thesis, Princess Sumaya University for Technology (Jordan)).
2. Palmer, N. (2017). Best Practices for Knowledge Workers: Innovation in Adaptive Case Management: Innovation in Adaptive Case Management. Future Strategies Inc..
3. Petroni, F., Piktus, A., Fan, A., Lewis, P., Yazdani, M., De Cao, N., ... & Riedel, S. (2021, June). KILT: a benchmark for knowledge intensive language tasks. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 2523-2544).
4. Rao, T. R., Mitra, P., Bhatt, R., & Goswami, A. (2019). The big data system, components, tools, and technologies: a survey. Knowledge and Information Systems, 60(3), 1165-1245.
5. Chen, S. A., Miculicich, L., Eisenschlos, J., Wang, Z., Wang, Z., Chen, Y., ... & Pfister, T. (2024). Tablerag: Million-token table understanding with language models. Advances in Neural Information Processing Systems, 37, 74899-74921.
6. Rai, S., Belwal, R. C., & Gupta, A. (2022). A review on source code documentation. ACM Transactions on Intelligent Systems and Technology (TIST), 13(5), 1-44.
7. Li, S., & Ramakrishnan, N. (2025, July). Oreo: A plug-in context reconstructor to enhance retrieval-augmented generation. In Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR) (pp. 238-253).
8. Pal, V., Lassance, C., Déjean, H., & Clinchant, S. (2023, March). Parameter-efficient sparse retrievers and rerankers using adapters. In European Conference on Information Retrieval (pp. 16-31). Cham: Springer Nature Switzerland.
9. Ding, Y., Wu, Y., & Ding, Z. (2025). An automatic patent literature retrieval system based on llm-rag. arXiv preprint arXiv:2508.14064.
10. Gao, Z., Qu, Y., & Han, Y. (2025). Cross-Lingual Sponsored Search via Dual-Encoder and Graph Neural Networks for Context-Aware Query Translation in Advertising Platforms. arXiv preprint arXiv:2510.22957.
11. Jin, J., Su, Y., & Zhu, X. (2025). SmartMLOps Studio: Design of an LLM-Integrated IDE with Automated MLOps Pipelines for Model Development and Monitoring. arXiv preprint arXiv:2511.01850.
12. Yin, Z., Chen, X., & Zhang, X. (2025). AI-Integrated Decision Support System for Real-Time Market Growth Forecasting and Multi-Source Content Diffusion Analytics. arXiv preprint arXiv:2511.09962.
13. Liang, R., Ye, Z., Liang, Y., & Li, S. (2025). Deep Learning-Based Player Behavior Modeling and Game Interaction System Optimization Research.
14. Wu, C., Zhang, F., Chen, H., & Zhu, J. (2025). Design and optimization of low power persistent logging system based on embedded Linux.
15. Zhu, W., Yao, Y., & Yang, J. (2025). Optimizing Financial Risk Control for Multinational Projects: A Joint Framework Based on CVaR-Robust Optimization and Panel Quantile Regression.
16. Wang, J., & Xiao, Y. (2025). Research on Transfer Learning and Algorithm Fairness Calibration in Cross-Market Credit Scoring.
17. Pal, V., Lassance, C., Déjean, H., & Clinchant, S. (2023, March). Parameter-efficient sparse retrievers and rerankers using adapters. In European Conference on Information Retrieval (pp. 16-31). Cham: Springer Nature Switzerland.

 doi:10.20944/preprints202511.2224.v1

18. Gu, X., Liu, M., & Yang, J. (2025). Application and Effectiveness Evaluation of Federated Learning Methods in Anti-Money Laundering Collaborative Modeling Across Inter-Institutional Transaction Networks.
19. Machidon, A. L., & Pejović, V. (2023). Deep learning for compressive sensing: a ubiquitous systems perspective. Artificial Intelligence Review, 56(4), 3619-3658.
20. Wu, Q., Shao, Y., Wang, J., & Sun, X. (2025). Learning Optimal Multimodal Information Bottleneck Representations. arXiv preprint arXiv:2505.19996.