Technical Report on CARROT: A Learned Cost-Constrained Retrieval Optimization System for RAG

Ziting Wang[†], Haitao Yuan[†], Wei Dong[†], Gao Cong[†], Feifei Li[‡]
[†]*Nanyang Technological University, Singapore*
[‡]*Alibaba Group, China*
[†]ziting001@e.ntu.edu.sg, [†]{haitao.yuan, wei_dong, gaocong}@ntu.edu.sg, [‡]lifeifei@alibaba-inc.com

I. Non-Additivity Analysis of Chunk Benefits

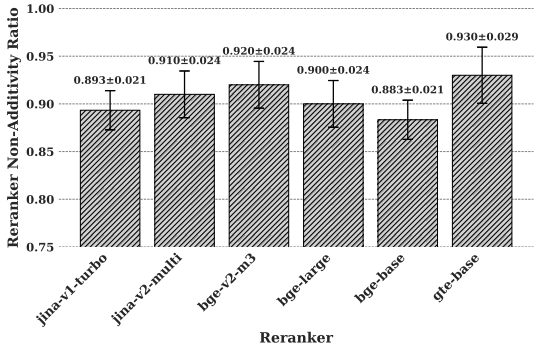

Fig. 1: Non-additivity ratios across different rerankers.

We evaluate the benefit of each chunk using a reranker model. To examine whether rerankers exhibit additive behavior, we analyze six state-of-the-art rerankers on the WikiPassageQA dataset. As shown in Figure 1, we define the node benefit $W(v)$ as the intermediate score assigned by the reranker to a specific node, representing the query relevance of the corresponding chunk combination. We then measure the *non-additivity ratio*, which quantifies the proportion of cases where the relation $W(a) + W(b) \neq W(a + b)$ holds for text chunks a and b. A high non-additivity ratio indicates that the reranker's scoring function cannot be decomposed into independent additive components, confirming that these models exhibit strong non-additive behavior in evaluating chunk benefits.

II. NP-Hardness of Optimal Chunk Combination Order Selection

Definition 1 (Optimal Chunk Combination Order Selection)**.** *Let $\{\chi_1, \chi_2, \ldots, \chi_k\}$ be the set of candidate chunks, $\mathcal{B}$ the cost budget, and $\Phi = \langle \chi_{\phi_1}, \ldots, \chi_{\phi_m} \rangle$ the ordered chunk combination, where ϕ_i denotes the position of each chunk. The reranker assigns a benefit $W(\Phi)$ to each combination. The objective is to find the order $\hat{\Phi}$ that maximizes the total benefit under a cost constraint:*

$$\hat{\Phi} = \arg\max_{\Phi} W(\Phi) \quad s.t. \quad \sum_{\chi_i \in \Phi} cost(\chi_i) \leq \mathcal{B}. \tag{1}$$

Theorem 1. *The Optimal Chunk Combination Order Selection problem is NP-hard.*

Proof. We prove NP-hardness by reduction from the *Maximum Weighted Hyperclique Problem (MWHP) [1]*, which is known to be NP-hard.

1) Problem definition of MWHP: Given a hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E}, w_1, w_2)$, where $\mathcal{V}$ is the set of vertices, $\mathcal{E}$ is the set of hyperedges, where each contains a subset of $\mathcal{V}$. $w_1 : v \to \mathbb{R}$ and $w_2 : e \to \mathbb{R}$ are functions assigning a benefit to each vertex and hyperedge, respectively. Given a subset of vertices $\mathcal{V}' \subseteq \mathcal{V}$, we say a hyperedge e belongs to $\mathcal{V}'$, i.e., $e \in \mathcal{V}'$, if $\mathcal{V}'$ covers all vertices of e. The objective is to find k vertices maximizing the benefit sum of these vertices and their covered hyperedges:

$$\arg \max_{\mathcal{V}' \subseteq \mathcal{V}, |\mathcal{V}'|=k} \Big(\sum_{v \in \mathcal{V}'} w_1(v) + \sum_{e \in \mathcal{V}'} w_2(e) \Big). \tag{2}$$

2) Reduction process: We now construct a corresponding Chunk Combination Optimization Problem instance from the given MWHP instance. For each node $v \in \mathcal{V}$, we create a corresponding chunk $\mathcal{X}_v$. We define its token cost $\text{cost}(X_v) \equiv 1$. Then, a chunk combination order Φ corresponds to a subset of vertices of $\mathcal{V}$, which is denoted as $\mathcal{V}(\Phi) \subseteq \mathcal{V}$. Since real utility computation for $\mathcal{V}(\Phi)$ relies on rerankers—black-box processes involving complex nonlinear relationships among chunks—we employ the mathematical reduction in (2) to approximate this process. Thus, we define its benefit as

$$W(\Phi) = \sum_{v \in \mathcal{V}(\Phi)} w_1(v) + \sum_{e \in \mathcal{V}(\Phi)} w_2(e). \tag{3}$$

Finally, we set $\mathcal{B} = k$ and our objective is

$$\arg\max_{\Phi} B(\Phi) \quad \text{s.t.} \quad \sum_{\chi_i \in \Phi} \text{cost}(\chi_i) = |\Phi| \leq k. \tag{4}$$

Denote Φ^* as the solution of (4), then, it is obvious $\mathcal{V}(\Phi^*)$ is the solution of (2), and the reduction can be done in time of $O\big(|\mathcal{V}| \cdot |\mathcal{E}|\big)$.

□

Remark. The NP-hardness reduction holds for arbitrary rerankers, and the property is unaffected by specific instances where a simple, additive reranker is used. As demonstrated in Section 1, practical rerankers exhibit significant non-additive behavior (with $W(a) + W(b) \neq W(a \cup b)$ in majority of

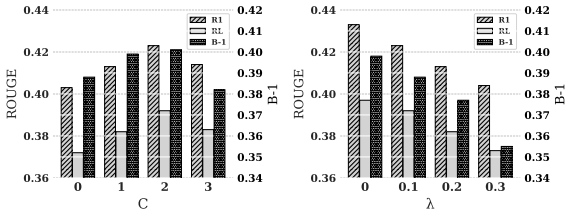

Fig. 2: Effectiveness Comparison vs. different coefficients

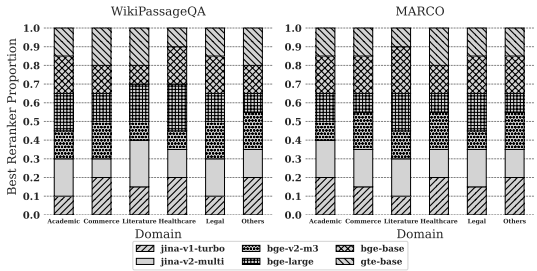

Fig. 3: Best Reranker Distribution

cases), supporting our reduction's assumption that rerankers can model the complex vertex-hyperedge interactions required by MWHP.

III. Experiment

A. MCTS Configuration Evaluation

1) Evaluation on Utility Coefficients: As illustrated in Figure 2, we evaluated the impact of exploration coefficient (C) and cost coefficient (λ) on system performance using WikiPassageQA. Results show that about $C = 2$ provides optimal performance by balancing exploration and exploitation during the search process, while both lower and higher values led to suboptimal outcomes. For the cost coefficient, introducing constraints ($\lambda > 0$) causes a slight decrease in ROUGE and BLEU scores (under 5%) as a higher cost coefficient produces shorter outputs. These findings emphasize the importance of our configuration agent in tuning these coefficients for optimal balance between quality and efficiency.

2) Evaluation on Different Rerankers' Domain Distribution: To evaluate the impact of different rerankers on retrieval performance, we conduct an ablation experiment using six Hugging face [2] widely recognized reranker models: *jina-reranker-v1-turbo-en*, *jina-reranker-v2-multilingual*, *bge-reranker-v2-m3*, *bge-reranker-large*, *bge-reranker-base*, and *gte-multilingual-reranker-base*. As shown in Figure 3, we initially classify the query domains following the general method [3], analyze the proportion of the best reranker. Our ablation study shows that the distribution of the best reranker for queries is varied among different settings (i.e., different datasets and different query domains).

B. Baseline Method Detailes

We compare *CARROT* with several typical RAG baselines:

- **RAPTOR [4]:** Constructs a hierarchical summary tree by recursively embedding, clustering, and summarizing chunks for multi-layer abstraction, aligning with graph-based methods. The tree is built within the cost constraint.
- **NaiveRAG [5]:** A basic retrieval approach that follows a standard process: conducting vector similarity search for candidate chunks, followed by a reranker model for reranking. To meet the cost constraint, we employ the greedy budget allocation strategy, retrieving chunks until the budget is fully exhausted.
- **HYDE [6]:** Often used as a RAG retrieval optimization module, it employs LLMs to generate a hypothetical document from the query, then retrieves chunks based on it. Also, a greedy strategy is employed to exhaust the budget efficiently.
- **ColBERT [7]:** A tuning-based method using a late interaction ranking model that adapts BERT for efficient retrieval by independently encoding queries and documents. Due to substantial computational costs during offline training and indexing, we only set budget constraints for online inference.
- **GraphRAG [8]:** A graph-based method that uses LLMs to extract entities/relationships as nodes/edges, aggregates them into communities, and produces a community summary for global context. Due to the high cost of index building of GraphRAG, our comparative analysis for GraphRAG excludes budget constraints.
- **GraphCOT [9]:** A framework that augments LLMs with graph reasoning capabilities through an iterative process of LLM reasoning, LLM-graph interaction, and graph execution. It enables LLMs to leverage both textual content and structural relationships in knowledge graphs.
- **G-Retriever [10]:** A retrieval-augmented generation approach for question answering on textual graphs that formulates retrieval as a Prize-Collecting Steiner Tree optimization problem. G-Retriever enables conversational interaction with graphs while mitigating hallucination and handling graphs that exceed LLM context windows.
- **FLARE [11]:** An active retrieval augmentation method that iteratively anticipates upcoming content to determine retrieval timing. It regenerates low-confidence text segments after retrieving context-relevant documents, demonstrating superior performance on long-form knowledge-intensive generation tasks.
- **HippoRAG2 [12], [13]:** A neurobiologically inspired framework which can also be categorized into graph-based methods that enhances retrieval performance with integrated knowledge graphs, query contextualization, and recognition memory.

References

[1] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer Science & Business Media, 2012.

[2] "Hugging face," https://huggingface.co/models, 2025.

[3] B. Jin, C. Xie, J. Zhang, K. K. Roy, Y. Zhang, Z. Li, R. Li, X. Tang, S. Wang, Y. Meng, and J. Han, "Graph chain-of-thought: Augmenting large language models by reasoning on graphs," in *ACL*, 2024, pp. 163–184.

[4] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, "RAPTOR: recursive abstractive processing for tree-organized retrieval," in *ICLR*, 2024.

[5] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, "Query rewriting in retrieval-augmented large language models," in *EMNLP*, 2023.
[6] L. Gao, X. Ma, J. Lin, and J. Callan, "Precise zero-shot dense retrieval without relevance labels," in *ACL*, 2023, pp. 1762–1777.
[7] O. Khattab and M. Zaharia, "Colbert: Efficient and effective passage search via contextualized late interaction over BERT," in *SIGIR 2020*. ACM, 2020, pp. 39–48.
[8] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, "From local to global: A graph RAG approach to query-focused summarization," *CoRR*, vol. abs/2404.16130, 2024.
[9] B. Jin, C. Xie, J. Zhang, K. K. Roy, Y. Zhang, Z. Li, R. Li, X. Tang, S. Wang, Y. Meng, and J. Han, "Graph chain-of-thought: Augmenting large language models by reasoning on graphs," in *ACL*, 2024.
[10] X. He, Y. Tian, Y. Sun, N. V. Chawla, T. Laurent, Y. LeCun, X. Bresson, and B. Hooi, "G-retriever: Retrieval-augmented generation for textual graph understanding and question answering," in *NeurIPS*, 2024.
[11] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," in *EMNLP*, 2023.
[12] B. J. Gutierrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, "Hipporag: Neurobiologically inspired long-term memory for large language models," in *NeurIPS*, 2024.
[13] B. J. Gutiérrez, Y. Shu, W. Qi, S. Zhou, and Y. Su, "From RAG to memory: Non-parametric continual learning for large language models," *CoRR*, vol. abs/2502.14802, 2025.