2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

UniversalRAG: Universal Retrieval-Augmented Generation Framework

Tianci Wu[1], Zikang Zhang[1], Dong Zhang[1] and Juntao Li[1]

[1] School of Computer Science and Technology, Soochow University, Suzhou 215006, China
20234227013@stu.suda.edu.cn

Abstract. Large Language Models (LLMs) excel in various tasks, yet hallucination limits their applicability in high-accuracy, domain-specific scenarios. Retrieval-Augmented Generation (RAG) mitigates this issue by integrating external knowledge retrieval, but existing systems struggle with multimodal, multi-format corpora common in industrial settings, and targeted evaluation datasets remain scarce. This paper introduces UniversalRAG, a modular, plug-and-play RAG framework supporting diverse document formats with adaptive indexing, retrieval, and generation agents, enhancing RAG adaptability and output quality. To validate its effectiveness, we develop the FACT dataset (Fact-based Augmented Corpus Testing) for RAG evaluation. Experimental results show that UniversalRAG, when paired with GPT-4o, achieves a 73.68 score, a 8.54-point improvement over the naive RAG baseline, significantly outperforming traditional methods. Ablation studies confirm the essential roles of indexing, retrieval, and generation agents in system performance. This work not only introduces a versatile RAG framework but also fills a critical gap in end-to-end evaluation, advancing RAG system development and assessment.

Keywords: Large Language Model, Hallucination Mitigation, Information Retrieval.

1 Introduction

With the rapid advancement of natural language processing (NLP), large language models (LLMs) have shown remarkable performance and robustness across domains. However, their training paradigms inherently limit their ability to fully eliminate hallucination [1], where generated content deviates from factual accuracy. This issue hinders their practical applicability, especially in high-accuracy domains like healthcare, law, and finance. Extensive research has thus focused on mitigating hallucination to enhance LLM reliability.

Retrieval-Augmented Generation (RAG) [7] effectively addresses hallucination by incorporating external knowledge bases, improving factual accuracy. However, a gap persists between academic and industrial applications. Industrial retrieval involves multimodal data sources—PowerPoint, PDF, and Word files—posing integration challenges. Conversely, academic RAG models primarily rely on clean, structured text, limiting their real-world applicability.

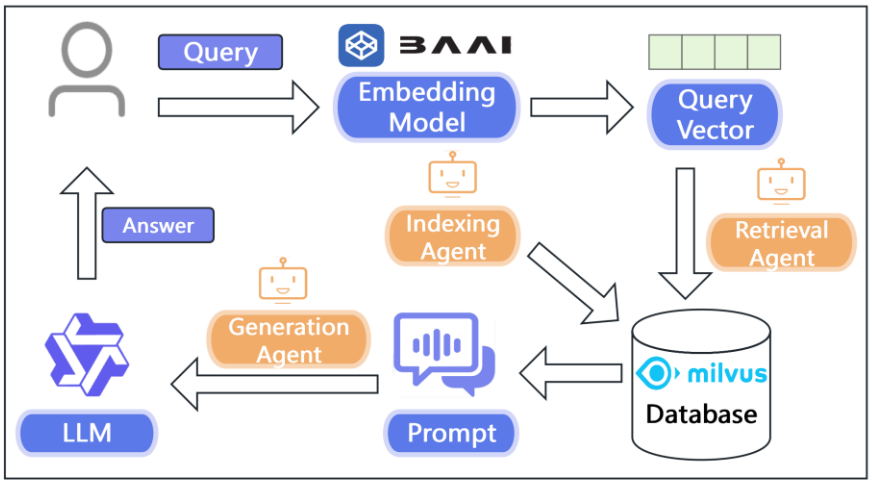

Fig. 1. The collaborative workflow of the UniversalRAG framework includes three core agents— **Indexing Agent**, **Retrieval Agent** and **Generation Agent.**

To bridge this gap, we propose UniversalRAG, a framework accommodating diverse input formats (**Fig. 1**). Through automation and modular design, it standardizes heterogeneous data for retrieval and generation. Additionally, it optimizes the three core RAG processes—indexing, retrieval, and generation—enhancing overall performance. The key contributions are:

FACT Dataset: A specialized dataset for RAG evaluation in industrial scenarios, encompassing diverse formats and fact-intensive, domain-specific queries. It enables rigorous assessment of multimodal information handling and end-to-end RAG performance.

UniversalRAG Framework: A modular system that automates the processing of PPT, PDF, and Word files, converting them into a unified format for retrieval and generation. It features adaptive mechanisms and optimized indexing, retrieval, and generation agents, ensuring flexibility, efficiency, and seamless multimodal integration.

2 RELATED WORK

Retrieval-Augmented Generation (RAG) has gained significant traction with the rise of Transformer architectures, accelerating after the advent of ChatGPT. According to [3], RAG evolution can be categorized into three stages: Naive RAG, Advanced RAG, and Modular RAG. Naive RAG follows a traditional indexing-retrieval-generation pipeline but suffers from low retrieval accuracy, suboptimal generation quality, and limited information integration. Advanced RAG addresses these issues through enhanced indexing structures, query expressiveness, and candidate ranking [4]. Modular

RAG, the current mainstream paradigm, introduces iterative, recursive, and adaptive retrieval for greater flexibility and adaptability

With these advancements, RAG applications have expanded across question answering, conversational generation, and document summarization. Future research is expected to focus on longer contextual inputs, robustness to noisy data, hybrid sparse-dense retrieval, scalability, and multimodal adaptability.

To support RAG development, various benchmarks and tools assess system performance. RGB [7], RECALL [8], and CRUD [9] evaluate RAG models in generation tasks, while RAGAS [10] and ARES [11] leverage LLMs for quality assessment. These tools commonly employ Exact Match (EM), Accuracy, and BLEU as metrics. However, existing evaluations rely on well-structured, large-scale open-source text (e.g., Wikipedia) or LLM-generated synthetic data, neglecting the multimodal and multi-format nature of real-world domain-specific corpora. This evaluation gap limits their effectiveness in assessing indexing and retrieval capabilities for complex corpora, restricting insights into RAG's full potential.

3 Dataset

Existing RAG benchmarks mostly use well-structured text corpora, but real-world data often includes multimodal content (e.g., text, images, charts), which challenges file processing and reveals RAG frameworks' limitations. To address this, we introduce **FACT** (Fact-based Augmented Corpus Testing), a benchmark designed to evaluate RAG frameworks in industrial scenarios, with a specialized corpus and question set for assessing multimodal processing and complex reasoning.

3.1 Corpus Details

The FACT dataset consists of 115 files across multiple formats (PDF, Word, PPT), featuring diverse origins and complex content structures. It includes:

- **Introduction Documents:** Overview of a technology company's background, capabilities, and strategic direction, providing industry insights and updates on emerging technologies.
- **Programming Tutorials:** Syntax rules, code examples, and applications of a proprietary programming language, serving as key domain-specific technical resources.
- **Certification Records:** Technical certifications demonstrating the company's expertise in specialized fields.
- **Policy Documents:** Policy interpretations, implementation guidelines, and industry regulations, offering timely regulatory insights.

Although publicly accessible, this content is highly specialized, non-generic, and rarely encountered during LLM pretraining, making it an ideal corpus for evaluating RAG frameworks on domain-specific and unfamiliar information. To protect privacy, all entity information related to specific company names has been anonymized. Corpus analysis results are shown in **Fig. 2**.

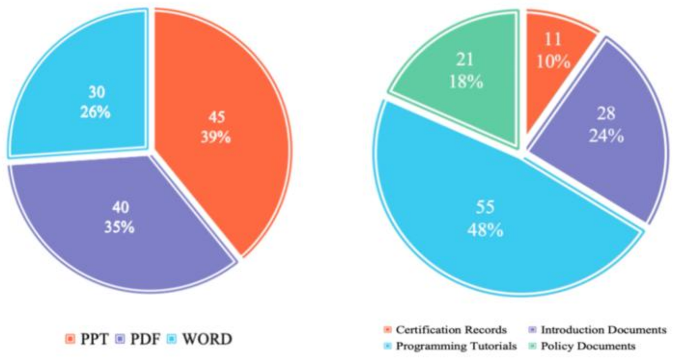

Fig. 2. Corpus Distribution. (left) illustrates the content distribution of the corpus. (right) shows the file type distribution, encompassing three formats with multimodal information such as text, images, and tables, posing significant challenges for the Indexing Agent.

3.2 Question Set

The question set consists of 180 questions categorized into three types to comprehensively assess the model's capabilities:

- **General Knowledge Questions:** Answerable using the model's internal knowledge without external retrieval. Example: "What type of hardware is specifically designed for image computation in computers?"
- **Domain-Specific Factual Questions:** Require the RAG framework to retrieve relevant information for accurate responses. Example: "What key features are supported by Company XX's proprietary programming language?"
- **Domain-Specific Reasoning Questions:** Involve logical reasoning or computation after retrieval. Example: "Based on the latest tax policy documents in Shanghai, what is the tax deduction amount for Company XX in 2024?"

This structured design enables a thorough evaluation, from simple knowledge retrieval to complex reasoning and decision-making. We applied the same anonymization process to the question set as we did to the corpus, ensuring consistent entity mapping across both components. For each question, **gold-label answer** and **key fact sets** were created to support metric calculation, which will be detailed in the experimental section.

4 Methods

4.1 Task Formulation

RAG is a task that integrates retrieval and generation to enhance the quality of answers by leveraging an external knowledge base $\mathcal{D}$ Given a query $q \in \mathcal{Q}$ (e.g., a natural language question), [19] the RAG task can be formalized into two steps: retrieval and generation.

In the retrieval step, the module identifies k most relevant documents $\{d_1, d_2, \dots, d_k\}$ from the knowledge base $\mathcal{D}$ based on the query q :

$$\{d_1, d_2, \dots, d_k\} = \text{Retrieval}(q, \mathcal{D}) \tag{1}$$

where the retrieval process is typically implemented using either sparse vector models [13] (e.g., BM25) or dense vector models (e.g., Transformer-based embedding models) [14].

In the generation step, the module conditions on q and the retrieved documents $\{d_i\}$ to generate the final answer a :

$$a = \text{Generate}(q, \{d_1, d_2, \dots, d_k\}) \tag{2}$$

where the generation process is commonly performed using conditional language models [20]

The objective of RAG is to jointly optimize the performance of the retrieval and generation components, maximizing the alignment between the generated answer a and the ground truth answer a^* . This can be formulated as:

$$\arg\max_{\mathcal{D}} \sum_{(q,a^*)\in\mathcal{Q}} \log P\,(a^*|q, \mathcal{D}; \theta) \tag{3}$$

where θ represents the set of model parameters and $\mathcal{D}$ represents the information retrieved by the system.

By integrating external retrieval with the generation module, RAG effectively addresses knowledge-intensive tasks, improving the accuracy and factual consistency of outputs [21].

4.2 Overviews

This section introduces UniversalRAG, a modular, plug-and-play RAG system designed for diverse corpus formats and easy offline deployment in specialized domains. Building upon the naive RAG pipeline, we comprehensively optimized the indexing, retrieval, and generation agents, enabling the system to adapt dynamically to varying question complexity and document volume. It addresses challenges in handling heterogeneous, multimodal real-world retrieval sources (e.g., PPT, Word, PDF), which existing research overlooks. UniversalRAG natively supports multi-format and multimodal inputs, as outlined in **Fig. 3** and detailed in the following sections.

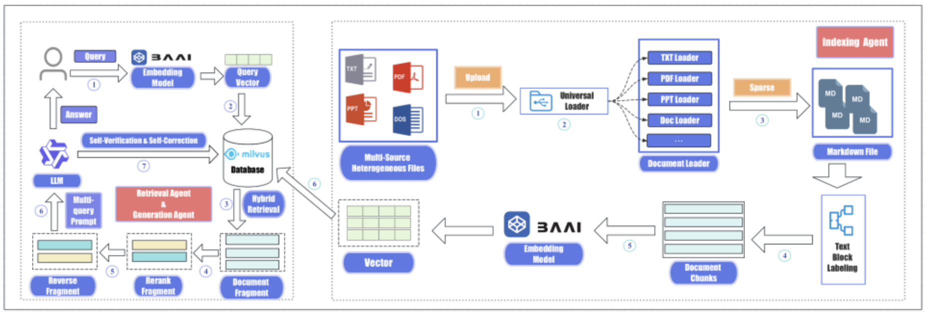

Fig. 3. The overview of UniversalRAG illustrates the detailed workflows of the Indexing Agent, Retrieval Agent, and Generation Agent. **(Left)** it highlights the collaboration between the Retrieval Agent and Generation Agent in efficiently retrieving and generating information, **(Right)** it demonstrates how the Indexing Agent processes various document formats and stores them in the vector database.

4.3 Indexing Agent

Universal Document Loader. The Universal document loader first separates documents (e.g., PDF, PPT, Word) into textual and image components [22]. For text, it analyzes layouts (e.g., single/double-column, horizontal/vertical) to ensure accurate extraction while minimizing formatting inconsistencies. For images, masking techniques prevent interference before Optical Character Recognition (OCR) [15] extracts text. A large language model then evaluates the extracted text's relevance to remove redundancy. Finally, textual data is stored in Markdown for maintainability, while charts and graphics are preserved in HTML for structural integrity.

Information Source Identification and Text Block Labeling. During corpus indexing, filenames and section titles provide distinctive semantic value, serving as both file identifiers and text block labels. Traditional text chunking often omits such metadata, weakening contextual coherence. To address this, we embed filenames and section titles into the chunking process, strengthening semantic relationships. This enhances clarity and retrieval precision.

4.4 Retrieval Agent

Hybrid Retrieval. Traditional RAG systems rely on dense vector retrieval, but challenges arise with low-resource corpora (e.g., short texts, domain-specific terms). We propose a hybrid model combining BM25 for sparse retrieval and a pre-trained BGE model for dense retrieval, improving performance by leveraging both long-tail corpora and deep semantics.

To optimize retrieval across tasks, we integrate both methods using a weighted score:

$$S_{final} = \alpha \cdot S_{sparse} + \beta \cdot S_{dense} \tag{4}$$

where $\alpha = 0.2$ and $\beta = 0.8$. These fusion balances precise keyword matching with deep semantic understanding, enhancing retrieval accuracy across diverse documents and tasks.

Reranker Component and Optimization. Our framework integrates a BGE-based reranker into the retrieval agent to refine candidate documents [17]. Unlike traditional rerankers that merely reorder results, ours also filters redundant content for better output quality. To prevent irrelevant documents [24], we employ a threshold-based strategy that re-scores candidates and retains only the top 50%. Inspired by [32], we arrange the final documents in reverse order, placing the most relevant ones nearest to the query to enhance response quality.

4.5 Generation Agent

Multi-query Strategy. To improve generation quality, our framework improves generation quality by using a multi-query approach, generating three queries for retrieval to enhance information coverage and reduce bias compared to single-query retrieval [25]. This approach boosts accuracy and completeness, especially for multimodal or long-text generation. To merge results efficiently, we use the Rank Fusion Framework (RFF) [18], optimizing relevance, coverage, and maintaining high accuracy.

Self-Verification and Self-Correction Mechanism. We implement a self-verification and self-correction mechanism inspired by [6] to enhance answer accuracy and reliability. After generating an initial answer, an LLM evaluates errors and reference confidence using a verification prompt. If confidence is low, retrieval parameters (e.g., scope, document count) are adjusted, and the refined results are reprocessed to improve quality.

5 Experiments

5.1 Metric

To comprehensively assess the quality of generated results, we designed two primary evaluation metrics that collectively evaluate performance from the dimensions of semantic relevance and key fact relevance.

Key Fact Relevance Score. This metric assesses whether the generated answer contains predefined keyword groups specific to each question, derived through corpus analysis. A higher hit rate corresponds to a higher score, indicating stronger factual accuracy. More relevant keyword groups in the answer signify better alignment with key facts.

To calculate the **Key Fact Relevance Score** ($\boldsymbol{S}_{\textbf{key}}$), Let A denote the generated answer, and let $F = \{f_1, f_2, \dots, f_n\}$ represent the predefined set of key fact phrases, where each f_i corresponds to a specific key fact phrase. The formula for S_{key} is expressed as:

$$S_{\text{key}}(A, F) = \frac{\sum_{i=1}^{n} 1(f_i \in A)}{n} \tag{5}$$

where:

- $1(f_i \in A)$ is an indicator function that equals 1 if f_i is present in the answer A and 0 otherwise.
- n is the total number of key fact phrases in F.

Semantic Relevance Score. This metric evaluates the semantic similarity between the target answer and the generated answer by embedding both into a semantic space and calculating their cosine similarity.

To compute the **Semantic Relevance Score**, let the target answer T and the generated answer G be embedded into the semantic space as vectors $\boldsymbol{v_T} = \text{Embed}(T)$ and $\boldsymbol{v_G} = \text{Embed}(G)$, respectively, using an embedding model $\text{Embed}(\cdot)$**.** The cosine similarity is then calculated as follows:

$$S_{cos}(\mathbf{v_T}, \mathbf{v_G}) = \frac{\mathbf{v_T} \cdot \mathbf{v_G}}{|\mathbf{v_T}||\mathbf{v_G}|} \tag{6}$$

where:

- $\boldsymbol{v_T} \cdot \boldsymbol{v_G}$ is the dot product of vectors $\boldsymbol{v_T}$ and $\boldsymbol{v_G}$.
- $|\boldsymbol{v_T}|$ and $|\boldsymbol{v_G}|$ are the norms (i.e., magnitudes) of $\boldsymbol{v_T}$ and $\boldsymbol{v_G}$, respectively.

The $S_{key}(\mathbf{v_T}, \mathbf{v_G})$, ranges from 0 to 1. A score closer to 1 indicates a higher level of semantic similarity between the generated answer and the target answer.

Final Score. Each question's score is calculated as a weighted combination of the Semantic Relevance Score and the Key Fact Relevance Score. Specifically, the weight of the Semantic Relevance Score is set to 0.4, while the weight of the Key Fact Relevance Score is 0.6.

For a given question i , the score S_i is defined as:

$$S_i = 0.4 \times S_{cos} + 0.6 \times S_{key} \tag{7}$$

$$S_{\text{final}} = \frac{1}{N} \sum_{i=1}^{N} S_i \tag{8}$$

where N is the total number of questions in the dataset. This formula represents the mean score across all questions, serving as the ultimate evaluation metric for the system’s overall performance.

5.2 Experiment Setup

We evaluated our approach on the FACT dataset using RTX 3090 GPUs. Given its linguistic diversity, we employed multilingual models for embedding (i.e., bge-multilingual-gemma2 [16] for dense text embeddings.) and ranking (i.e., bge-reranker-v2-m3 [16] for document re-ranking.) which were trained and open-sourced by Beijing Academy of Artificial Intelligence **BAAI**). And we employ Qwen2.5-7B-Instruct [28] and GPT-4o [27] as generation models.

Furthermore, the length of inputs and outputs is capped at 2048 tokens, with greedy decoding ensuring deterministic evaluation.

5.3 Baselines

For evaluation on FACT, we designed four baseline settings to examine LLM performance under different retrieval-augmented conditions:

Zero-shot (LLM without Retrieval Augmentation): The model answers questions using only its internal knowledge, assessing knowledge retention without external retrieval.

Few-shot (LLM without Retrieval Augmentation): Following In-context Learning (ICL), a few high-quality demonstrations assist the model in answering questions, evaluating Few-shot learning's effectiveness.

LLM + Sparse Retriever (BM25): The BM25 algorithm retrieves relevant documents based on keyword matching, providing context for LLM-generated answers. This setting assesses traditional keyword-based retrieval in retrieval-augmented generation.

LLM + Dense Retriever (bge-multilingual-gemma2): The bge-multilingual-gemma2 model generates dense semantic vectors to retrieve the most relevant documents, evaluating the impact of semantic retrieval on retrieval-augmented generation.

LLM + naive RAG: Following the setup in , we implemented a naive RAG framework consisting of standard indexing, retrieval, and generation components. Specifically, the indexing component directly reads the document text and generates embedding representations using the bge-multilingual-gemma2 model. The retrieval component selects the top three documents with the highest cosine similarity to the question. During the generation stage, the question and the retrieved documents are concatenated into a prompt and fed into a large language model to produce the final answer.

5.4 Results

The experimental results (**Table 1**) validate the effectiveness of UniversalRAG. Using Qwen2.5-7B-Instruct, UniversalRAG outperformed the naive RAG by 7.54 points. In contrast, the weakest performances came from LLM (Zero-shot) and LLM (Few-shot), as expected. Since FACT contains numerous domain-specific factual and reasoning questions, these tasks rely heavily on high-relevance corpora. Without external knowledge, the LLM struggles to generate correct answers, leading to lower scores in these settings.

Table 1. Performance comparison of different methods and models. The best performance is highlighted **in bold**.

Method & Model	S_{cos}	S_{key}	S_{final}
LLM (zero-shot)	40.0	47.3	44.38
LLM (few-shot)	41.5	49.8	46.48
LLM + Sparse Retrievers	51.7	56.0	54.28
LLM + Dense Retrievers	56.1	54.3	55.02
LLM + naive RAG	60.4	62.1	61.42
LLM + UniversalRAG (ours)	**65.3**	**71.4**	**68.96**
GPT-4o	45.2	52.7	50.10
GPT-4o + naive RAG	62.5	66.9	65.14
GPT-4o + UniversalRAG (ours)	**69.6**	**76.4**	**73.68**

Further analysis shows that even GPT-4o, a state-of-the-art closed-source model, scores only 50.10 without external knowledge augmentation. However, with UniversalRAG, the score increases by 23.58 points to 73.68, outperforming the naive RAG by a significant margin of 8.54 points and achieving the best overall performance. This underscores UniversalRAG's role in enhancing LLM capabilities for complex professional tasks, demonstrating the critical importance of external knowledge integration in improving generation quality.

5.5 Ablation Study

To assess the contributions of different agents in UniversalRAG, we conducted an ablation study (**Table 2**), analyzing performance changes as key agents were progressively removed. It is worth noting that in the ablation experiments, **"w/o [component]"** refers to replacing our optimized module with the corresponding module from the naive RAG.

Table 2. Ablation study results.

Methods	S_{final}
UniversalRAG	68.96
UniversalRAG *w/o indexing agent*	59.32
UniversalRAG *w/o retrieval agent*	62.95
UniversalRAG *w/o generation agent*	65.60

Impact of Removing the Indexing Agent: Removing the Indexing Agent caused a 10+ point drop, highlighting its importance. The loss of adaptive reading strategies and source identification weakened the corpus' semantic structure, impairing retrieval and generation quality.

Impact of Removing the Retrieval or Generation Agent: Removing either the Retrieval Agent or Generation Agent caused significant performance declines. Without the Retrieval Agent, the framework failed to extract relevant information, limiting contextual richness for downstream tasks. Without the Generation Agent, it could not synthesize retrieved content into coherent, high-quality answers, impacting overall performance.

6 Conclusion and future work

In this paper, we proposed **UniversalRAG**, a modular and plug-and-play RAG framework for multimodal and multi-format corpora. It enhances generation performance through adaptive indexing, efficient retrieval, and a multi-query generation chain. To evaluate its effectiveness, the authors introduce the **FACT** dataset. Experiments show that UniversalRAG outperforms traditional methods, especially with state-of-the-art LLMs. Future work includes refining indexing, optimizing retrieval, improving generation with reinforcement learning, and expanding the FACT dataset for broader industry applications.

Reference

1. Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen et al., "Siren's song in the ai ocean: A survey on hallucination in large language models," arXiv preprint arXiv:2309.01219, 2023.
2. X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, "Query rewriting for retrieval-augmented large language models," arXiv preprint arXiv:2305.14283, 2023.
3. Y. Gao, Y. Xiong, X. Gao, et al. Retrieval-augmented generation for large language models: A survey[J]. arXiv preprint arXiv:2312.10997, 2023.
4. W. Peng, G. Li, Y. Jiang, Z. Wang, D. Ou, X. Zeng, E. Chen et al., "Large language model based long-tail query rewriting in taobao search," arXiv preprint arXiv:2311.03758, 2023.
5. X. Wang, Q. Yang, Y. Qiu, J. Liang, Q. He, Z. Gu, Y. Xiao, and W. Wang, "Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases," arXiv preprint arXiv:2308.11761, 2023.
6. A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-rag: Learning to retrieve, generate, and critique through self-reflection," arXiv preprint arXiv:2310.11511, 2023.
7. P. Lewis, E. Perez, A. Piktus, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks." Advances in Neural Information Processing Systems, 2020, 33: 9459-9474.
8. J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language image pre-training with frozen image encoders and large language models," arXiv preprint arXiv:2301.12597, 2023.
9. W. Zhu, A. Yan, Y. Lu, W. Xu, X.E. Wang, M.Eckstein, and W. Y.Wang, "Visualize before you write: Imagination-guided open-ended text generation," arXiv preprint arXiv:2210.03765, 2022
10. J. Zhao, G. Haffar, and E. Shareghi, "Generating synthetic speech from spoken vocab for speech translation," arXiv preprint arXiv:2210.08174, 2022.
11. J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," arXiv preprint arXiv:2001.08361, 2020.
12. U. Alon, F. Xu, J. He, S. Sengupta, D. Roth, and G. Neubig, "Neurosymbolic language modeling with automaton-augmented retrieval," in International Conference on Machine Learning. PMLR, 2022, pp. 468–485.

13. S. Robertson, and H.Zaragoza, “The probabilistic relevance framework: BM25 and beyond.” Foundations and Trends® in Information Retrieval, 2009, 3(4): 333-389.
14. V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, “Dense passage retrieval for open-domain question answering,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 6769–6781.
15. C. Li, W. Liu, R. Guo, X. Yin, K. jiang, Y. Du et al, “PP-OCRv3: More attempts for the improvement of ultra lightweight OCR system,” arXiv preprint arXiv:2206.03001, 2022.
16. J. Chen, S. Xiao, P. Zhang, K. Luo, D. lian, and Z. liu, “Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation.” arXiv preprint arXiv:2402.03216, 2024.
17. Y. Ma, Y. Cao, Y. Hong, and A. Sun, “Large language model is not a good few-shot information extractor, but a good reranker for hard samples!” ArXiv, vol. abs/2303.08559, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257532405
18. G. Cormack, C. Clarke, and S. Buettcher, “Reciprocal rank fusion outperforms condorcet and individual rank learning methods” Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval. 2009: 758-759.
19. D. Arora, A. Kini, S. R. Chowdhury, N. Natarajan, G. Sinha, and A. Sharma, “Gar-meets-rag paradigm for zero-shot information retrieval,” arXiv preprint arXiv:2310.20158, 2023.
20. S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J.-B. Lespiau, B. Damoc, A. Clark et al., “Improving language models by retrieving from trillions of tokens,” in international conference on machine learning. PMLR, 2022, pp. 2206–2240.
21. Z. Luo, C. Xu, P. Zhao, X. Geng, C. Tao, J. Ma, Q. Lin, and D. Jiang, “Augmented large language models with parametric knowledge guiding,” arXiv preprint arXiv:2305.04757, 2023.
22. L. Zha, J. Zhou, L. Li, R. Wang, Q. Huang, S. Yang, J. Yuan, C. Su, X. Li, A. Su et al., “Tablegpt: Towards unifying tables, nature language and commands into one gpt,” arXiv preprint arXiv:2307.08674, 2023.
23. S. Yang, “Advanced rag 01: Small-to-big retrieval,” https://towardsdatascience.com/ advanced-rag-01-small-to-big-retrieval-172181b396d4, 2023.
24. W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W.-t. Yih, “Replug: Retrieval-augmented black-box language models,” arXiv preprint arXiv:2301.12652, 2023.
25. G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave, “Few-shot learning with retrieval augmented language models,” arXiv preprint arXiv:2208.03299, 2022.
26. Z. Ke, W. Kong, C. Li, M. Zhang, Q. Mei, and M. Bendersky, “Bridging the preference gap between retrievers and llms,” arXiv preprint arXiv:2401.06954, 2024.
27. A. Hurst, A. Lerer, P.Goucher, P. Adam, et al. “Gpt-4o system card,” arXiv preprint arXiv:2410.21276, 2024.
28. A. Yang, B. Yang, B. Zhang,B. Hui, et al. “Qwen2. 5 Technical Report,” arXiv preprint arXiv:2412.15115, 2024.