Boosting Retrieval-Augmented Generation with Generation-Augmented Retrieval: A Co-Training Approach

Yubao Tang*
CAS Key Lab of Network Data Science and Technology, ICT, CAS
University of Chinese Academy of Sciences
University of Amsterdam
Amsterdam, The Netherlands
y.tang3@uva.nl

Ruqing Zhang†
CAS Key Lab of Network Data Science and Technology, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
zhangruqing@ict.ac.cn

Jiafeng Guo†
CAS Key Lab of Network Data Science and Technology, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
guojiafeng@ict.ac.cn

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
m.derijke@uva.nl

Yixing Fan
CAS Key Lab of Network Data Science and Technology, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
fanyixing@ict.ac.cn

Xueqi Cheng
CAS Key Lab of Network Data Science and Technology, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
cxq@ict.ac.cn

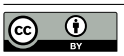
Abstract

Large language models (LLMs) have shown success in knowledge-intensive tasks, including closed-book question answering and entity linking. However, their susceptibility to hallucination undermines their reliability. Retrieval-augmented generation (RAG) partially addresses this issue by combining a retriever to locate relevant documents and a generator to produce responses grounded in the retrieved evidence. Despite its advantages, RAG faces challenges: (i) the structural gap between traditional dense retrievers and autoregressive generators, and (ii) limited generation performance due to insufficient contextual guidance returned by the retriever. To tackle these limitations, we propose MINT, a framework that enhances RAG by co-training Retrieval-augMented generatIon and geNeration-augmented reTrieval (GAR). MINT (i) bridges the gap between the retriever and generator using a unified encoder-decoder structure, (ii) incorporates an iterative co-training strategy between RAG and GAR, enabling mutual enhancement through pseudo-samples generation, and (iii) introduces three heuristic inference strategies to generate relevant document identifiers and answers. We conduct an empirical study on the KILT benchmark, and MINT is found to yield significant improvements in both retrieval and generation tasks compared with prevailing baselines.

CCS Concepts

• **Information systems** → **Retrieval models and ranking**.

*Yubao Tang is currently affiliated with the University of Amsterdam.
†Jiafeng Guo and Ruqing Zhang are the corresponding authors.



SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3729907

Keywords

Retrieval-augmented generation, Generative retrieval

ACM Reference Format:
Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2025. Boosting Retrieval-Augmented Generation with Generation-Augmented Retrieval: A Co-Training Approach. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3726302.3729907

1 Introduction

Large language models (LLMs) have demonstrated strong capabilities across a wide range of natural language processing (NLP) tasks [1, 21, 28, 42, 46, 59, 76, 77], including knowledge-intensive tasks (KILT) [41], such as closed-book question answering and fact verification [23, 60, 66]. Despite these successes, LLMs often suffer from hallucination [2, 67], resulting in erroneous or misleading outputs. This issue can significantly undermine the reliability of LLMs on KILT tasks, where accurate knowledge grounding is critical. A common approach to mitigating hallucination in LLMs is retrieval-augmented generation (RAG) [17, 19, 41]. As shown in Figure 1(a), RAG employs a two-step process: a retriever first identifies relevant documents from a corpus, and a generator then synthesizes the final response based on the retrieved evidence. By grounding generation in retrieved documents, RAG can improve factual consistency and reduce hallucinations, using explicit external evidence to guide the generative process [39, 68].

Research challenges. While RAG has achieved notable improvements, it still faces two key limitations: (i) RAG systems usually rely on dense retrievers, which build a corpus index and perform query-document matching. We argue that *the discriminative retriever introduces a gap between the retriever and the generator*. That is, the dense retriever operates on vector similarity matching, while the generator produces text in an autoregressive manner, leading

to suboptimal collaboration [9]. To alleviate this gap, prior work has explored unifying retrieval and generation with generative retrieval (GR) [36, 58]. GR encodes the corpus as model parameters based on an encoder-decoder architecture, allowing the model to directly generate document identifiers (docids) autoregressively. For example, as shown in Figure 1(c), UniGen [73] uses shared encoders for both retrieval and generation, with decoders designed to output relevant docids and answers. However, it lacks explicit interaction between the two decoders, limiting performance. Additionally, CorpusLM [29] decouples the two tasks into sequential decoding steps but suffer from inconsistencies between training and inference, leading to suboptimal results.

(ii) Another limitation in RAG is that *the quality of retrieval may impact the generation outcome*. If the retriever fails to retrieve relevant documents, the generator's performance suffers, as it relies on the evidence provided by the retriever [9, 72]. While the generator in RAG benefits from both the query and retrieved evidence, the retriever typically only has access to the query, making it relatively weaker. To alleviate this, some work has explored generation-augmented retrieval (GAR) [34], reversing the RAG process by first using a generator to produce pseudo-answers that guide the retriever, as shown in Figure 1(b). This approach helps the retriever better identify relevant documents by narrowing the query's scope. However, since the GAR generator only uses query information, it remains less robust, limiting its effectiveness.

Approach. To overcome these limitations, we propose MINT, a framework designed to enhance RAG through co-training of Retrieval-augMented generatIon and geNeration-augmented reTrieval. MINT consists of three core innovations: (i) *Unified structure*: We bridge the gap between the retriever and generator by unifying their structures. Unlike previous work [29, 73], MINT ensures consistent training and inference with parameter sharing across both components. (ii) *Iterative co-training*: Inspired by knowledge distillation [8, 14], we improve the performance of the weaker module (student) in RAG by using the stronger module (teacher) in GAR to generate high-quality pseudo-samples, and vice versa. These samples supervise the student's training, allowing it to benefit from the teacher's knowledge. We achieve this by iteratively co-training RAG and GAR. (iii) *Flexible inference strategies*: We explore three heuristic inference strategies to achieve the final results: using RAG only, using GAR only, and using RAG and GAR simultaneously.

Contributions. We conduct an empirical study on the widely-used KILT benchmark [41] with two variants of backbone models, i.e., T5 [45] and Llama2 [61]. Experimental results show that the proposed MINT framework is able to achieve significant improvements in both retrieval and generation tasks. For example, our method based on T5 achieves a 15.4% relative improvement in terms of EM on the NQ dataset over the state-of-the-art baseline CorpusLM (T5), under the closed-book setting.

2 Preliminaries

We briefly introduce generative retrieval, and describe and analyze the fundamental mechanisms underlying RAG and GAR.

Generative retrieval. Generative retrieval (GR) uses differentiable search indexes to directly generate relevant docids in response to a query based on auto-regressive generative models. GR targets two

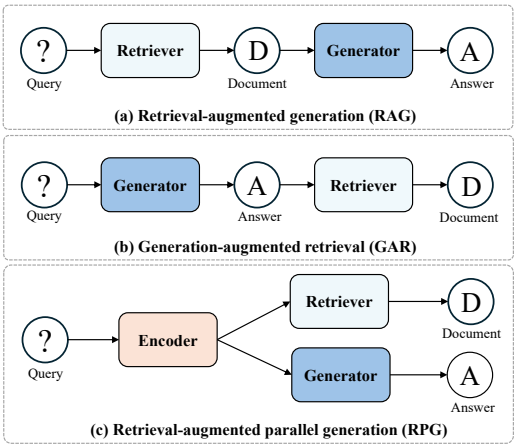


Figure 1: Three combinations of retrievers and generators: (a) Retrieval-augmented generation (RAG): Document retrieval is performed first, followed by answer generation. (b) Generation-augmented retrieval (GAR): Answer generation is performed first, followed by document retrieval. (c) Retrieval-augmented parallel generation (RPG): Document retrieval and answer generation are performed simultaneously with a shared query encoder.

key tasks [58]: (i) indexing: memorizing corpus information by associating the content of each document with its docid; and (ii) retrieval: assessing relevance by learning a mapping from queries to their relevant documents. Formally, given a corpus $\mathcal{D} = \{d_1, \ldots, d_{|\mathcal{D}|}\}$, each document d_i has a corresponding docid id_i. Let $\mathcal{I}_\mathcal{D}$ denote all the docids of $\mathcal{D}$. Given a query set $\mathcal{Q} = \{q_1, \ldots, q_{|\mathcal{Q}|}\}$, its relevant docid set is $\mathcal{I}_\mathcal{Q}$. The indexing and retrieval operations are typically learned in a multi-task setup and optimized via maximum likelihood estimation (MLE) [40]:

$$\mathcal{L}_{GR}(\mathcal{Q}, \mathcal{I}_\mathcal{D}, \mathcal{D}; \theta) = \mathcal{L}_{indexing}(\mathcal{D}, \mathcal{I}_\mathcal{D}; \theta) + \mathcal{L}_{retrieval}(\mathcal{Q}, \mathcal{I}_\mathcal{Q}; \theta), \quad (1)$$

where θ represents the GR model parameters, and the two terms $\mathcal{L}_{indexing}(\mathcal{D}, \mathcal{I}_\mathcal{D}; \theta)$ and $\mathcal{L}_{retrieval}(\mathcal{Q}, \mathcal{I}_\mathcal{Q}; \theta)$ are defined as:

$$\mathcal{L}_{indexing}(\mathcal{D}, \mathcal{I}_\mathcal{D}; \theta) = -\sum_{d_i \in \mathcal{D}} \sum_{id_i \in \mathcal{I}_\mathcal{D}} \log P_\theta(id_i \mid d_i), \quad (2)$$

$$\mathcal{L}_{retrieval}(\mathcal{Q}, \mathcal{I}_\mathcal{Q}; \theta) = -\sum_{q_i \in \mathcal{Q}} \sum_{id_i \in \mathcal{I}_\mathcal{Q}} \log P_\theta(id_i \mid q_i). \quad (3)$$

Through GR, relevant documents can be retrieved all within a generative model, facilitating joint learning with answer generation tasks in RAG and GAR.

Retrieval-augmented generation. As shown in Figure 2 (a), given a query, the retriever first retrieves relevant documents by generating a ranked docid list. Then these docids are mapped to corresponding documents. These documents subsequently serve as the evidence to aid the generator in producing high-quality answers. The warm-up of RAG involves two basic tasks using labeled data and several auxiliary tasks using LLM-generated data. To equip the model with both retrieval and generation capabilities simultaneously, the parameters are shared between them.

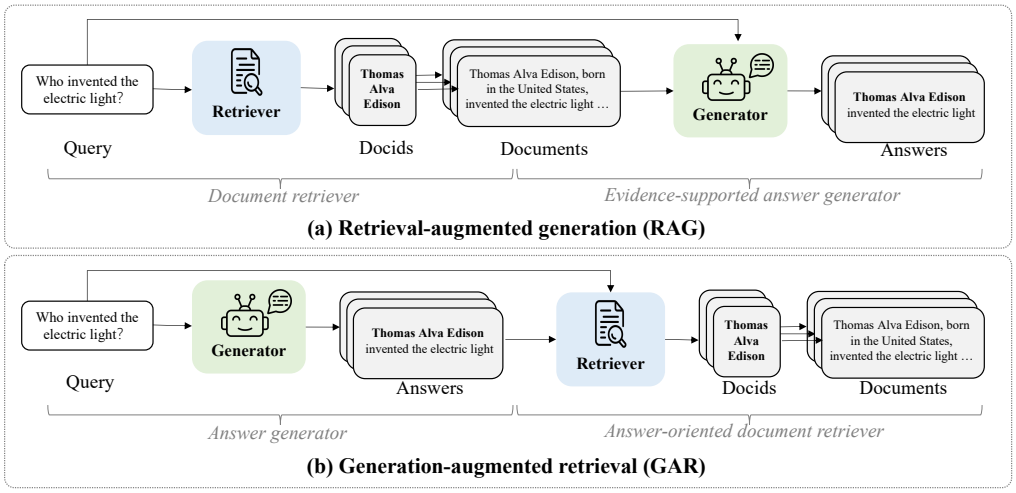


Figure 2: Comparison between RAG (a) and GAR (b), where the evidence-supported answer generator in RAG uses information retrieved by the document retriever to produce answers, and the answer-oriented document retriever uses the answers generated by the answer generator to retrieve documents.

Basic tasks. The retriever is trained based on labeled query-docid pairs and document-docid pairs via Eq. (1). For the generator, given $\mathcal{Q}$ with $\mathcal{I}_Q$ and the corresponding answer set $\mathcal{A} = \{a_1, \ldots, a_{|\mathcal{A}|}\}$, where a_i is the answer of $q_i \in \mathcal{Q}$, the training objective $\mathcal{L}^G_{RAG}(\mathcal{Q}, \mathcal{A}, \mathcal{I}_\mathcal{D}, \mathcal{D}; \beta)$ with MLE is formalized as:

$$-\sum_{q_i \in \mathcal{Q}, a_i \in \mathcal{A}, id_i \in \mathcal{I}_\mathcal{D}, d_i \in \mathcal{D}} \log P_\beta(a_i \mid q_i, d_i, id_i; \beta), \tag{4}$$

where β denotes the parameters of the evidence-supported answer generator. Here, we develop a unified language model for retrieval and generation in RAG, so β and θ of the GR model are shared.

Auxiliary tasks. For the retriever, to augment the docid understanding by pre-trained language models, we introduce two unsupervised docid understanding tasks: (i) generating a ranked docid list based on a pseudo-query to improve ranking performance; and (ii) predicting a ranked docid list related to a given docid to capture document-level relevance [29]. For the generator, to enhance pre-trained language models' capability to integrate supporting knowledge, we use pseudo-training triplets (pseudo-query, query context, answer) to generate answers from pseudo-queries and query contexts. Pseudo-queries and pseudo-training triplets are generated using the Llama2-7B model [61] with specific prompts. For pseudo-queries, the prompt is:

"`Generate five pseudo queries for the following document. The queries should be relevant and diverse, reflecting the key points and topics discussed: {d}.`"

For the evidence-supported question generator, the prompt is:

"`Given the document and its title, generate five alternative titles for the document. Ensure the new titles are relevant to the document's content and different from the provided title. The document is: {d}. The title is:{title}.`"

Pseudo-docids are formed by concatenating the titles of documents and their respective multiple sections. All auxiliary tasks are formulated as seq2seq generative tasks.

Warm-up. The warm-up of RAG aims to optimize objectives via a combined loss function with the above basic and auxiliary tasks:

$$\mathcal{L}_{RAG}(\mathcal{Q}, \mathcal{A}, \mathcal{I}_\mathcal{D}, \mathcal{D}; \theta) = \lambda_1 \mathcal{L}_{GR} + \lambda_2 \mathcal{L}^G_{RAG} + \lambda_3 \mathcal{L}^{AUX}_{\text{RAG}}, \tag{5}$$

where $\mathcal{L}^{AUX}_{\text{RAG}}$ is the loss function for the auxiliary tasks. λ_1, λ_2 and λ_3 are weighting coefficients used to balance these tasks.

Generation-augmented retrieval. As depicted in Figure 2(b), given a query, the generator first produces answers and then the retriever retrieves related documents using the query and the generate answers. This process is akin to navigating a maze game, where the initial guess guides subsequent exploration [37]. The basic and auxiliary tasks for initializing GAR are as follows.

Basic tasks. For the generator, given a query $q_i \in \mathcal{Q}$, the objective is to maximize the likelihood of generating the target answer, defined as:

$$\mathcal{L}^G_{GAR}(\mathcal{Q}, \mathcal{A}; \psi) = -\sum_{q_i \in \mathcal{Q}, a_i \in \mathcal{A}} \log P_\psi(a_i \mid q_i; \psi), \tag{6}$$

where ψ represents the model parameters of GAR. For the retriever, given a query $q_i \in \mathcal{Q}$ and corresponding answer a_i the inputs, it generates relevant docids. The learning objective $\mathcal{L}^R_{GAR}(\mathcal{Q}, \mathcal{I}_\mathcal{D}, \mathcal{A}; \psi)$ is formalized as:

$$-\sum_{q_i \in \mathcal{Q}, id_i \in \mathcal{I}_\mathcal{D}, a_i \in \mathcal{A}} \log P_\psi(id_i \mid q_i, a_i; \psi), \tag{7}$$

where the generator and the retriever in GAR share parameters ψ.

Auxiliary tasks. For the generator, we use pseudo-training pairs (pseudo-query, pseudo-answer) to predict answers for queries. For the retriever, the unsupervised docid understanding task is to generate a ranked docid list based on the pseudo-query and pseudo-answer. Similar to RAG, the pseudo-data is generated using the Llama2-7B model [61] with the specific prompt:

"`Given the document, generate a relevant query and provide an answer based on the document's content. The document is: {d}.`"

Table 1: Comparison of warmed-up RAG and GAR under the retrieval and RAG settings.

Method	Retrieval	RAG setting
RAG	71.32	90.15
GAR	72.57	87.47

The loss function of the auxiliary tasks is represented by $\mathcal{L}^{AUX}_{\text{GAR}}$.

Warm-up. Similar to RAG, the training objective of GAR is defined as:

$$\mathcal{L}_{GAR}(\mathcal{Q}, \mathcal{A}, \mathcal{I}_{\mathcal{D}}, \mathcal{D}; \psi) = \lambda_4 \mathcal{L}^{G}_{GAR} + \lambda_5 \mathcal{L}^{R}_{GAR} + \lambda_6 \mathcal{L}^{AUX}_{GAR}, \quad (8)$$

where λ_4, λ_5 and λ_6 are hyperparameters, which are used to weight each module.

Discussion. We warm up both RAG and GAR as described above and then evaluate their retrieval and answer generation performance. We conduct a comparison on the FEV dataset under the retrieval and RAG setting, with R-precision and accuracy as the metrics, respectively. Detailed experimental settings are provided in Section 4. As shown in Table 1, each approach exhibits distinct strengths. (i) For RAG, the performance of the answer generation is better than that of GAR under the RAG setting, validating that RAG has a stronger ability of evidence-supporting answer generation. (ii) For GAR, the retrieval performance is better than RAG, while the answer generation performance is weaker than RAG. Therefore, GAR tends to have a stronger ability for retrieval. *Can we develop a novel framework that harnesses the diverse capabilities of RAG and GAR and uses their respective strengths to enhance the overall performance of retrieval and generation?*

3 Methodology

In this section, we introduce the co-training process of RAG and GAR as shown in Figure 3. The key idea is that stronger modules can generate high-quality pseudo-samples to teach and supervise weaker modules [14], improving their performance iteratively. The process involves three key steps: (i) pseudo-sample generation, (ii) confidence-based sample filtering, and (iii) iterative co-training. Then, we introduce the inference process.

Pseudo-samples generation. As described in Section 2, we warm up RAG and GAR using their respective basic and auxiliary tasks. The model parameters after warm-up for RAG and GAR are denoted as θ^0 and ψ^0, respectively. Based on the initialized θ^0 and ψ^0, we randomly select M documents, denoted as D^-, from unlabeled data to construct pseudo-data for subsequent co-training. Inspired by [7], we use a leading passage and a leading sentence from these documents as pseudo-queries, allowing RAG and GAR to generate relevant docids and answers. The generated pseudo-samples, i.e., triplets (pseudo-query, docids, answers), are denoted as D^0_{RAG}. Similarly, GAR-generated samples are denoted as D^0_{GAR}.

Confidence-based sample filtering. The generated pseudo-samples might suffers from noise introduced by erroneous predictions, which can degrade model performance. To address this, we prioritize high-confidence samples, inspired by the principle that reliable data leads to better training outcomes [48]. We apply a confidence

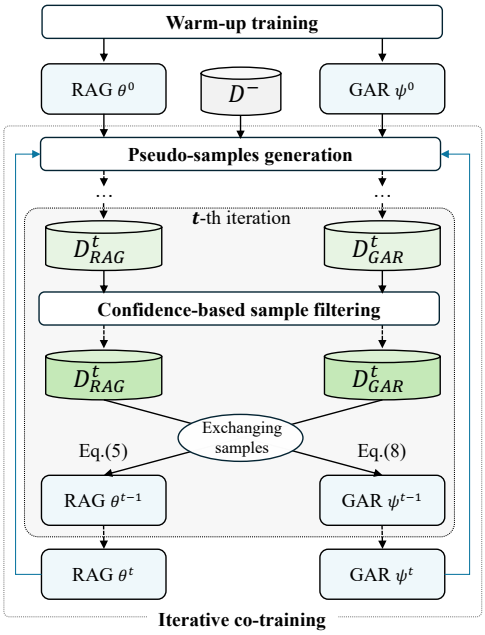


Figure 3: The co-training process of RAG and GAR in the t-th iteration.

interval to filter the data based on their confidence scores. By normalizing these scores and applying a confidence interval based on the model's behavior on ground-truth data, the method filters out low-confidence, noisy samples, retaining only reliable ones for training. This process is iteratively refined to enhance the reliability of the pseudo-samples. (i) For the aforementioned (pseudo-query, docids, answers) triplet, we can obtain the conditional likelihood probability scores of the model's output during generation. For example, in RAG, the retriever, given a query, outputs relevant docids along with their corresponding probability scores. Similarly, the generator, given a query and a docid with the corresponding documents as input, outputs an answer and its corresponding probability score. We further normalize these probability scores by length and take their absolute values as their respective confidence scores [38]. (ii) Regarding the confidence interval, considering the reliability of existing labeled data, we set the interval based on the information from the labeled data. We first calculate the probability scores of the model generating ground-truth outputs given the corresponding inputs. Next, we normalize these probability scores by length and take their absolute values. Finally, we use the maximum and minimum of these values as the boundaries of the confidence interval. (iii) We retain only the data pairs with confidence scores within this interval, denoting the filtered pseudo-relevant data as D^0_{RAG} and D^0_{GAR}. Similarly, based on the parameters θ^t and ψ^t after the t-th round, the filtered data pairs generated by RAG and GAR are denoted as D^t_{RAG} and D^t_{GAR}, respectively.

Iterative co-training. In each iteration, the co-training algorithm uses filtered generated pseudo-samples with high confidence scores from RAG or GAR, respectively, to form an auto-labeled dataset. The other component is then updated with the exchanged auto-labeled dataset. (i) For the 1-st round of co-training, based on the

initialized θ^0 and ψ^0, we exchange the filtered pseudo data for training both components: training θ^0 (RAG) with D^0_{GAR} via Eq. (5) and ψ^0 (GAR) with D^0_{RAG} via Eq. (8). Subsequently, the model parameters are updated from θ^0 to θ^1 and from ψ^0 to ψ^1, followed by new pseudo-samples generation for subsequent iterations. (ii) For the t-th ($t \geq 1$) round, based on θ^{t-1}, ψ^{t-1}, the filtered pseudo-data D^{t-1}_{RAG} and D^{t-1}_{GAR}, we train θ^{t-1} (RAG) with D^{t-1}_{GAR} using Eq. (5), and ψ^{t-1} (GAR) with D^{t-1}_{RAG} using Eq. (8). And the parameters are updated from θ^{t-1} to θ^t and from ψ^{t-1} to ψ^t, respectively, followed by new pseudo-samples generation obtaining D^t_{RAG} and D^t_{GAR} for the next iteration. (iii) After enough iterations with the properly selected unlabeled data, the performance of RAG and GAR would be stable generally.

Inference. After iterative co-training optimization, we explore three inference mechanisms:

- Using RAG only: RAG's evidence-supported answer generator exhibits stronger knowledge integration and generation capabilities than GAR. Hence, we use RAG for question answering tasks.
- Using GAR only: GAR's answer-oriented document retriever has stronger retrieval capabilities than RAG. Hence, we use GAR for retrieval tasks.
- Using RAG and GAR simultaneously: We first use the document retriever and the evidence-supported answer generator in RAG to generate the final answers. This is followed by the answer-oriented document retriever in GAR to generate the relevant docids, as the final supporting documents. However, this approach incurs higher inference costs than the first two, thus requiring some trade-offs and future exploration.

We refer to these three types of mechanism as MINT_R, MINT_G and MINT_{RG}, respectively.

4 Experimental Settings

Datasets. We conduct experiments on the KILT benchmark [41] with 11 datasets spanning 5 knowledge-intensive natural language tasks, to evaluate the retrieval and generation capabilities of MINT. For detailed statistics, please refer to Table 2. We evaluate the performance in retrieval, closed-book generation, and retrieval-augmented generation tasks.

Evaluation metrics. To measure the retrieval performance, we use R-precision following the official KILT evaluation metrics and previous works [3, 7, 29]. To measure the generation performance, we employ specific metrics tailored for each task akin to [41]: for FEV, AY2, WnWi, WnCw, T-REx, and zsRE, we use Accuracy; For NQ, TQA, and HoPo, we use Exact Match (EM); for ELI5 and WoW, we use ROUGEL and F1, respectively. Specifically, R-precision is calculated as the fraction of relevant contexts within the top-R retrieved contexts, where R is the number of contexts in the provenance set. Accuracy measures the percentage of correctly answered samples out of the total number of samples. EM is a stricter version of accuracy where all labels have to match exactly for the sample to be correctly classified. And ROUGEL measures the longest matching sequence of words between a generated text and a reference text, emphasizing both precision and recall in terms of sequence length. Additionally, for a fair comparison with non-finetuned generators, we include the percentage of outputs containing gold answers ("has_answer").

Table 2: Datasets statistics of the KILT benchmark. #Train and #Dev denote the number of queries in the training set and development set, respectively. And "-" denotes that the task does not provide a ground-truth label in the training set. (FC: Fact checking; EL: Entity linking; SF: Slot filling; ODQA: Open domain question answering; D: Dialogue.)

Label	Dataset	Task	#Train	#Dev
FEV [60]	FEVER	FC	104,966	10,444
AY2 [16]	AIDA CoNLL-YAGO	EL	18,395	4,784
WnWi [15]	WNED-WIKI	EL	-	3,396
WnCw [15]	WNED-CWEB	EL	-	5,599
T-REx [15]	T-REx	SF	2,284,168	5,000
zsRE [25]	Zero Shot RE	SF	147,909	3,724
NQ [23]	Natural Questions	ODQA	87,372	2,837
HoPo [66]	HotpotQA	ODQA	88,869	5,600
TQA [20]	TriviaQA	ODQA	61,844	5,359
ELI5 [13]	ELI5	ODQA	272,634	1,507
WoW [12]	Wizard of Wikipedia	D	63,734	3,054

Baselines for retrieval tasks. Following [7, 29], the baselines for retrieval tasks include two main types: (i) Sparse & dense retrieval: (a) BM25 [47] is a typical strong sparse retrieval method which represents the classical probabilistic retrieval model. (b) DPR [22] is a BERT-based dual-encoder architecture to obtain the dense vectors of the query and document. (c) MT-DPR [33] is a multi–task variant of DPR. (d) RAG [27] combines dense retrieval with Seq2Seq models tailored for knowledge-intensives tasks. (e) E5 [62] is a strong text-embedding model, which uses weakly-supervised contrastive pre-training. (f) SimLM [63] designs pre-training for the dense passage retrieval. (ii) Generative retrieval: (a) T5 [44] is a pre-trained text-to-text model for multitask learning based on an encoder-decoder structure. (b) BART [26] designs several text denoising pre-training tasks for text generation tasks. (c) SEAL [3] takes all the n-grams in a document as its docids base on FM-index. (d) CorpusBrain [7] designs pre-training tasks tailored for the KILT benchmark. (e) GenRet [49] learns to generate a string of discrete numbers as the docids, specifically for the retrieval task. (f) Llama2 [61] is an available open-sourced LLM. (g) UniGen [30] optimizes the retrieval and QA with a shared encoder and two independent decoders. (h) GCoQA [31] has a generative retriever firstly, followed by a generator for QA. (i) CorpusLM (T5) [29] a unified language model that uses an external corpus to tackle various based on T5-Base. (j) CorpusLM (Llama2) [29] is based on Llama2-7b-Chat. All retrieval models are fine-tuned with labeled data from KILT datasets. While the DPR model undergoes fine-tuning on each specific dataset, other dense and GR models are multi-task fine-tuned using labeled data from all datasets.

Baselines for downstream tasks. Following [29], two primary streams are considered, including closed-book generation and retrieval-augmented generation. For closed-book generation, the baselines are: T5 [44], BART [26], Llama270B [61], UniGen [30], GCoQA [31], CorpusLM(T) [29], CorpusLM(L) [29].

For RAG, baselines are: (i) DPR+BART [41] uses DPR as the retriever and BART as the generator. (ii) RAG [27] is an end-to-end retriever-generator model. (iii) MT-DPR+FID uses the multi-task variant of DPR [33] as the dense retriever, and a fusion-in-decoder model [18] as the generator. (iv) BM25+Llama2-70B uses BM25 as the initial retriever and Llama2 as the generator. (v) E5+Llama2-13B&70B [17] uses E5 as the retriever and Llama2 as the generator.

Implementation details. We use titles as docids, since the articles in Wikipedia have unique high-quality titles. We use the T5-base [45] (T5) and Llama2-7B-Chat [61] (Llama2) as our backbone models. For T5-Base, the hidden size is 768, the feed-forward layer size is 12, the number of self-attention heads is 12, and the number of transformer layers is 12. For LLama2-7B-chat, it consists of 32 transformer layers with 4096 hidden embedding sizes.

In the warm-up phase, we first independently train each module within each framework using the corresponding annotated data and constructed data pairs. For the indexing task of GR, we use Eq. 2 to learn document-docid pairs. It is learned together with the retrieval task (Eq. 3) in a multi-task manner. During training, to balance the importance between tasks, all λ series parameters, i.e., λ_i ($i = \{1, 2, 3, 4, 5, 6\}$), are set to 1. For hyperparameters, we specify the learning rate as 5e-5, label smoothing as 0.1, weight decay as 0.01, sequence length of documents as 512, maximum training steps as 50K, and batch size as 80. And we specify M as 100K, considering the training efficiency. For the retrieval tasks, the total iteration number t is set to 6, while for the generation tasks, t is 7.

To improve the training efficiency of MINT (Llama2), we use QLoRA [11] and DeepSpeed [25] technologies. We train MINT on eight NVIDIA Tesla A100 80GB GPUs. During training, we use the Adam optimizer with a linear warm-up over the first 10% steps. During inference, we set the beam size to 50. We will open-source the code and trained models upon publication.

5 Experimental Results

5.1 Retrieval performance

We conduct retrieval inference using the proposed three inference mechanisms; see Table 3.

Comparison with sparse & dense baselines. (i) Although dense retrievers like MT-DPR, E5, RAG, and SimLM show competitive performance through multi-task fine-tuning, MINT outperforms them by a significant margin on most datasets. For instance, on the WnWi dataset, MINT$_{RG}$ (T5) outperforms SimLM by 41.4% in terms of R-precision. (ii) Joint training of the DPR model on multiple datasets (MT-DPR) consistently yields better results than individual dataset training, highlighting the effectiveness of joint training for enhanced retrieval performance.

Comparison with GR baselines. (i) Generative retrievers generally perform better than dense retrievers, indicating the effectiveness of generative models for knowledge-intensive tasks. (ii) CorpusLM (T5) sometimes performs better than CorpusLM (LLama2), possibly because T5's pre-training corpus is based on Wikipedia, allowing its knowledge to be effectively applied to these datasets. (iii) Compared to most GR baselines, MINT$_{RG}$ consistently achieves superior performance. For example, on the HoPo dataset, MINT$_{RG}$ (Llama2) performs better than the SOTA CorpusLM (Llama2) by 5% in terms of R-precision. (iv) MINT$_G$ and MINT$_{RG}$ demonstrate similar performance. This is likely because iterative co-training has allowed the modules in GAR and RAG to complement each other, resulting in similar generative effectiveness.

5.2 Generation performance

As shown in Table 4, we evaluate the generation performance in both closed-book and open-retrieval (i.e., RAG) settings on the dev set. For the closed-book setting, we conduct inference using the evidence-supported answer generator of RAG without documents as input, as well as using the answer generator of GAR. For the RAG setting, we conduct inference using RAG to generate answers with retrieved information.

Closed-book setting. (i) With the same backbone, MINT outperforms the current SOTA method CorpusLM. For example, on the NQ dataset, MINT$_R$ (T5) improves by 15.4% over CorpusLM (T5), and MINT$_R$ (Llama2) improves by 13.8% over CorpusLM (Llama2) in terms of the EM metric. These results demonstrate the effectiveness of our co-training approach. (ii) MINT shows a performance improvement when the backbone is switched from T5 to Llama2, highlighting the importance of the backbone in our approach. (iii) MINT$_R$ consistently outperforms MINT$_G$, even though both use a generator to produce answers without input documents. This again highlights the stronger answer generation ability of RAG compared to GAR. (iv) Further, MINT$_R$ (T5) outperforms GCoQA by 21.4% on the NQ dataset in terms of the EM metric, validating the effectiveness of the co-training between two components.

RAG setting. (i) Similarly, with the same backbone, MINT$_R$ consistently outperforms the current SOTA method CorpusLM, demonstrating the effectiveness of our approach. Specifically, on the WnWi dataset, MINT$_R$ (T5) achieves a 5.7% improvement over CorpusLM (T5) in terms of the accuracy metric. (ii) Additionally, MINT$_R$ (T5) even performs better than the larger baseline E5+Llama-70B across most datasets. MINT$_R$ (T5) outperforms GCoQA by 45.2% on the zsRE dataset in terms of accuracy. These results confirm the effectiveness of our collaborative co-training approach. (iii) Among the two variants, the one using Llama2 as the backbone demonstrates stronger performance, highlighting that larger model capacity enables the parameterization of more knowledge.

5.3 Ablation studies

We conduct an ablation analysis using GAR inference for retrieval tasks. We investigate the impact of co-training, auxiliary tasks and warm-up training based on T5-base. The results are shown in Table 5. (i) When co-training is not used (i.e., 2nd row), that is, we only perform the warm-up training. We can observe that there is a significant performance drop compared to MINT$_G$. This confirms that the collaboration between RAG and GAR enhances both modules. (ii) When no auxiliary tasks are used for any modules (i.e., 3rd row), and only annotated data is used for warm-up, there is a noticeable decline in performance across the datasets. This demonstrates the importance of the quality of the warm-up and the effectiveness of the auxiliary tasks we designed. (iii) When skipping the warm-up phase (i.e., 4th row), that is, RAG without using Eq. 5 and GAR without using Eq. 5 for learning, we observe a significant drop in retrieval performance. This indicates that the

Table 3: R-Precision (%) for the retrieval task on the KILT dev set. † indicates results from [29]. And "-" indicates that the baseline does not report data the specified metric, due to the lack of ground-truth labels in the training set. ∗ indicates statistically significant improvements over all the baselines ($p \leq 0.05$).

Methods	FC	Entity linking			Slot filling		Open domain QA				Dial
	FEV	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
Sparse & dense retrieval											
BM25†	30.29	2.82	1.38	3.84	32.04	43.37	12.34	31.31	14.40	1.20	17.20
DPR†	59.10	79.51	-	-	60.61	70.91	31.13	39.47	35.48	-	37.66
MT-DPR†	64.05	81.69	49.20	46.95	57.64	73.81	32.80	38.42	36.29	10.86	38.00
RAG†	66.04	76.40	48.28	46.01	53.57	67.97	38.25	34.61	41.38	10.70	38.04
E5†	68.52	79.72	50.47	48.10	54.48	70.01	39.40	37.35	42.62	11.02	39.16
SimLM†	68.06	80.11	51.98	49.54	55.42	72.11	38.58	36.11	41.80	10.36	38.31
Generative retrieval											
T5†	71.63	86.71	67.34	62.20	64.87	78.51	38.69	38.09	45.73	10.35	42.51
BART†	69.90	87.43	67.22	60.71	61.57	76.13	39.84	38.44	47.26	10.09	40.19
SEAL†	70.55	82.05	57.09	58.70	55.91	74.89	39.67	40.54	44.16	9.32	41.59
CorpusBrain†	72.23	88.79	69.40	63.23	63.42	79.05	40.09	39.45	47.97	10.68	42.19
GenRet	72.45	88.92	69.57	63.56	63.77	79.52	40.25	39.78	48.21	10.67	42.26
Llama2†	74.39	85.53	66.55	61.45	66.12	77.90	40.59	40.37	48.43	10.66	42.69
UniGen	72.57	89.12	69.77	63.74	63.86	79.74	40.43	39.84	48.41	10.73	42.53
GCoQA	70.83	87.43	67.12	61.03	61.35	77.23	38.77	37.25	46.93	10.20	40.78
CorpusLM (T5)†	75.64	90.96	70.35	65.43	68.89	81.08	41.46	39.31	48.80	10.90	44.96
CorpusLM (Llama2)†	76.21	88.59	69.39	64.18	69.17	80.79	44.10	42.06	50.62	10.88	43.92
Co-training RAG and GAR (Ours)											
MINT_R (T5)	75.69	90.99	70.42	65.52	68.91	81.00	41.52	39.37	48.84	10.91	44.97
MINT_G (T5)	78.84	92.24	73.47	67.83	70.03	83.24	43.89	42.01	50.13	10.93	45.73
MINT_{RG} (T5)	78.87	**92.26**∗	**73.51**∗	**67.85**∗	70.08	**83.27**∗	43.95	42.16	50.22	**10.95**	**45.81**∗
MINT_R (Llama2)	76.51	88.84	69.85	64.82	69.25	81.32	44.78	42.76	51.21	10.90	43.98
MINT_G (Llama2)	77.56	89.38	70.13	65.47	70.56	81.84	45.83	43.81	51.78	10.91	44.24
MINT_{RG} (Llama2)	**79.13**∗	90.67	71.46	66.32	**71.48**∗	82.47	**46.52**∗	**44.15**∗	**52.81**∗	10.92	44.87

warm-up training is essential, as it enables the model to acquire fundamental retrieval and generation capabilities, which are crucial for supporting subsequent co-training. We also conduct an ablation analysis using RAG inference for generation tasks. As shown in Table 6, we observe the same trend as with GAR, which further confirms the effectiveness of the proposed mechanisms.

5.4 Impact of the number of iterations

The number of co-training iterations is a crucial factor affecting performance. Therefore, we evaluate the model's R-precision of the retrieval and accuracy of closed-book generation and RAG, on the FEVER dataset across different iteration counts. For the retrieval, we use MINT_G (T5); for the closed-book generation and RAG, we use MINT_R (T5). From Figure 4, we can observe the following: (i) Retrieval performance gradually improves from 1 to 6 iterations, reaching its peak at the 6-th iteration (the green triangle in Figure 4). Beyond this point, performance gradually declines with additional iterations. Similarly, we notice that in closed-book generation and RAG settings, performance increases from 1 to 7 iterations and then gradually decreases. This indicates the necessity and effectiveness of iterative training. (ii) Additionally, we observe

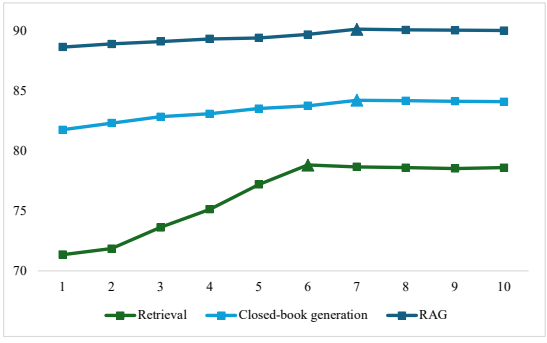


Figure 4: Results of different numbers of the iterations under the retrieval, closed-book generation and RAG settings, on the FEVER dataset.

that when retrieval performance reaches its peak, closed-book generation and RAG performance have not yet reached their highest values. This might be because the capabilities required for retrieval and closed-book generation tasks are somewhat conflicting: retrieval demands a broader range of relevance, whereas closed-book generation requires more fine-grained accuracy.

Table 4: Generation performance on KILT dev set under closed-book and RAG setting. † indicates results from [29]. And "-" indicates that the original work [29] does not report numbers for baselines under the specific metric. ∗ indicates statistically significant improvements over all the baselines ($p \leq 0.05$).

Methods	FC	Entity linking			Slot filling		Open domain QA				Dial
	FEV	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
Closed-book generation											
T5†	-	81.84	47.35	46.58	47.24	1.58	25.20	12.66	25.79	21.02	13.15
BART†	80.67	86.62	47.91	48.01	43.84	3.03	26.15	16.86	32.54	22.69	13.77
Llama2-70B†	33.60	39.80	42.80	39.20	28.50	11.30	19.60	13.90	**67.40**	23.00	13.30
UniGen	81.01	86.75	48.23	48.34	44.51	3.11	26.56	17.01	33.43	22.71	13.81
GCoQA	81.55	86.85	48.77	48.51	44.82	3.24	26.74	17.23	33.51	22.74	13.85
CorpusLM (T5)†	81.93	87.16	49.68	51.65	49.65	3.61	28.14	17.32	25.02	21.33	13.90
CorpusLM (Llama2)†	85.34	86.91	56.66	50.37	51.05	18.22	29.57	26.13	41.34	22.94	14.85
Co-training RAG and GAR (Ours)											
MINT_R (T5)	84.24	**91.32**∗	53.57	**54.41**∗	53.36	3.81	32.46	19.75	28.25	23.51	14.37
MINT_G (T5)	82.95	88.19	50.97	52.68	50.68	3.74	29.17	18.35	26.05	22.35	14.01
MINT_R (Llama2)	**89.42**∗	90.14	**58.93**∗	52.26	**55.74**∗	**20.31**∗	**33.65**∗	**28.59**∗	43.68	**24.78**∗	**15.84**∗
MINT_G (Llama2)	87.53	88.32	57.25	51.19	53.84	19.28	31.21	27.13	42.51	23.64	15.06
Retrieval-augmented generation											
DPR+BART†	88.11	-	44.96	45.70	56.70	34.96	45.05	25.75	59.28	18.53	15.51
RAG†	87.70	77.40	49.00	46.70	61.48	47.42	48.78	27.68	61.73	16.11	13.28
MT-DPR+FID†	88.49	79.77	49.52	47.15	79.43	69.09	50.07	36.50	69.92	15.77	15.60
BM25+Llama2†	46.20	18.00	19.10	14.20	25.90	31.40	25.30	25.90	65.80	21.30	12.20
E5+Llama2-13B†	66.30	51.20	48.60	45.60	17.20	31.70	36.10	14.30	56.30	20.90	12.30
E5+Llama2-70B†	49.90	51.20	48.60	45.60	28.90	35.00	36.40	28.10	71.10	21.50	13.20
UniGen	88.23	78.24	49.32	46.81	61.52	47.51	48.84	28.42	62.32	18.42	13.31
GCoQA	88.94	86.79	49.67	48.94	63.67	49.69	50.67	30.56	64.84	20.46	15.89
CorpusLM (T5)†	89.81	87.09	50.02	49.77	80.68	70.34	53.39	40.96	70.94	22.13	16.65
CorpusLM (Llama2)†	90.22	85.03	56.54	50.32	81.57	72.79	55.38	42.23	72.43	23.46	16.96
Co-training RAG and GAR (Ours)											
MINT_R (T5)	90.15	**89.53**∗	52.85	51.94	82.45	72.13	55.16	42.02	71.89	23.14	16.87
MINT_R (Llama2)	**92.84**∗	87.73	**58.91**∗	**53.82**∗	**83.81**∗	**74.88**∗	**57.35**∗	**44.87**∗	**74.82**∗	**25.19**∗	**17.25**∗

Table 5: Ablation studies on retrieval performance. The best results are in bold.

Methods	FC	Entity linking			Slot filling		Open domain QA				Dial
	FEV	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
MINT_G (T5)	**78.84**	**92.24**	**73.47**	**67.83**	**70.03**	**83.24**	**43.89**	**42.01**	**50.13**	**10.93**	**45.73**
w/o co-training	72.57	87.84	68.42	61.46	61.63	78.93	39.14	37.84	47.14	10.25	41.57
w/o auxiliary	76.82	90.73	69.46	64.28	68.24	81.01	40.67	39.86	48.23	10.7	44.83
w/o warm-up	68.54	84.73	65.27	58.75	58.52	75.63	36.21	34.79	44.72	8.73	37.64

5.5 Inference efficiency

As shown in Table 7, we analyze the inference efficiency, considering model parameters, memory footprint and query latency. Among them, query latency refers to the amount of time it takes for the model to inference to a query. Here, for our method, we only consider the RAG to conduct inference. MINT_R (T5) significantly reduces the number of parameters and memory usage compared to methods like RAG and MT-DPR+FID. Specifically, (i) MINT has around 2.8 times fewer parameters than the RAG model, which needs a substantially lower computational cost. (ii) In terms of the memory footprint, MINT achieves 201.2-fold decrease compared with MT-DPR+FID, mainly because the model only needs to store document titles as docids instead of a large-scale dense document index. However, CorpusLM requires storing document titles and section titles as docids. (iii) In terms of query latency, CorpusLM (T5) performs the best due to its continuous decoding strategy. The query latency of our method is comparable to RAG and is also acceptable. Overall, our method demonstrates advantages in terms of reduced parameter size and memory footprint, ensuring efficient

Table 6: Ablation studies on generation performance in RAG setting. The best results are in bold.

Methods	FC	Entity linking			Slot filling		Open domain QA				Dial
	FEV	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
MINT$_R$ (T5)	**90.15**	**89.53**	**52.85**	**51.94**	**82.45**	**72.13**	**55.16**	**42.02**	**71.89**	**23.14**	**16.87**
w/o co-training	88.97	86.96	49.73	49.13	79.67	69.87	52.84	49.67	69.73	22.01	16.23
w/o auxiliary	89.94	88.51	50.67	50.36	81.57	71.75	54.26	41.36	71.56	22.78	16.64
w/o warm-up	84.26	83.46	46.05	46.58	75.73	64.18	47.36	45.67	65.42	17.18	14.42

Table 7: Results about inference efficiency.

Method	Parameters	Storage	Latency
RAG	626M	59.3G	106.7ms
MT-DPR+FID	440M	51.2G	160.9ms
CorpusLM (T5)	220M	426.1M	78.4ms
MINT$_R$ (T5)	220M	260.5M	103ms

resource utilization. Additionally, it maintains a moderate query latency, striking a balance between performance and responsiveness.

6 Related Work

Generative retrieval. Key aspects [51, 55–57] of recent GR research include: (i) Docid designs: GR approaches typically use fixed docids during training, including number-based docids [4, 35, 58, 74, 75] and word-based docids like document titles [6, 7, 10, 24, 29], n-grams [3, 5, 32], important word sets [71], pseudo-queries [50]. Some research designs learnable docids for the retrieval [49, 65, 69, 70], but these require more complex design. For the KILT benchmark, we use Wikipedia's document titles as docids, since the documents have well-written unique titles. (ii) Training methods: The widely used training strategy is jointly optimizing indexing and retrieval tasks with maximum likelihood estimation [58]. Various improvements [43, 50, 52–54, 64, 78] have since enhanced performance. (iii) Inference strategies: Ensuring valid generated docids involves three main approaches, including constrained beam search [6, 7, 10, 49, 50] and FM-index based constrained decoding [3, 5, 65].

Retrieval-augmented generation. RAG is an advanced approach that combines the strengths of IR and text generation. In RAG, a retriever first searches for relevant documents or passages from a large corpus based on a given query. The retrieved information is then used to augment the context for a generator, which generates more accurate and contextually relevant responses or answers. For the mainstream methods, the retriever and reader modules are individually trained, using dense retrieval models [41]. Recently, some work has jointly trained these two modules, such as CorpusLM [29] and UniGen [30].

Differences. (i) CorpusLM aims for end-to-end retrieval and answer generation with a single model. Its training optimizes query-docid and query-answer pairs jointly, while inference follows a two-stage process: first generating the docid, linking it to the document, and then generating the answer. This gap between training and inference makes it difficult to align the intended behavior of generating both docid and answer simultaneously. (ii) UniGen employs a shared encoder and two separate decoders for docid and answer generation. The absence of explicit interaction between the two decoders reduces the effectiveness of joint generation.

We agree that the two-stage process is intuitive. Building on this idea, our approach ensures consistency between training and inference. Inspired by knowledge distillation [14], we use the stronger module as a teacher to generate pseudo-samples, which provide guidance and supervision to improve the weaker module.

7 Conclusion

This study proposes a novel cooperative framework to enhance the specific answer generation and document retrieval abilities of RAG and GAR, thereby improving the performance of retrieval, closed-book generation, and RAG. The proposed method effectively activates and utilizes intrinsic knowledge within RAG and GAR through iterative co-training, exchanging high-confidence pseudo-labeled samples. MINT is able to improve the retrieval and generation performance of RAG, while GAR can aid in understanding RAG from another perspective.

Limitations include: (i) The selection of the confidence interval relies on limited annotated data. The features contained in the annotated data are not comprehensive. Therefore, the confidence interval may not be fully suitable for all documents in the corpus. (ii) Additionally, since our inference phase still requires two-stage inference, there is room for improvement in inference speed.

For future work, we are interested in combining modality-agnostic data augmentation to make RAG and GAR co-training applicable to other tasks. We will also investigate an adversarial retriever-generator framework to boost RAG performance.

Acknowledgments

We thank the reviewers for their valuable feedback. This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 62472408, 62372431 and 62441229, the Strategic Priority Research Program of the CAS under Grants No. XDB0680102 and XDB0680301, the National Key Research and Development Program of China under Grants No. 2023YFA1011602, the Youth Innovation Promotion Association CAS under Grants No. 2021100, the Lenovo-CAS Joint Lab Youth Scientist Project, and the project under Grants No. JCKY2022130C039. This work was also (partially) funded by Ahold Delhaize, through AIRLab Amsterdam, by the Dutch Research Council (NWO), under project numbers 024.004.022, NWA.1389.20.183, and KICH3.LTP.20.006, and by the European Union's Horizon Europe program under grant agreement No. 101070212. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

References

[1] Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, et al. 2023. Information Retrieval Meets Large Language Models: A Strategic Report from Chinese IR Community. *AI Open* 4 (2023), 80–90.

[2] Razvan Azamfirei, Sapna R Kudchadkar, and James Fackler. 2023. Large Language Models and the Perils of their Hallucinations. *Critical Care* 27, 1 (2023), 120.

[3] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Wen-tau Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive Search Engines: Generating Substrings as Document Identifiers. *Advances in Neural Information Processing Systems* 35 (2022), 31668–31683.

[4] Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, Yixing Fan, and Xueqi Cheng. 2023. Continual Learning for Generative Retrieval over Dynamic Corpora. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 306–315.

[5] Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yiqun Liu, Yixing Fan, and Xueqi Cheng. 2023. A Unified Generative Retriever for Knowledge-Intensive Language Tasks via Prompt Learning. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*. 1448–1457.

[6] Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. 2022. GERE: Generative Evidence Retrieval for Fact Verification. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2184–2189.

[7] Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yiqun Liu, Yixing Fan, and Xueqi Cheng. 2022. CorpusBrain: Pre-train a Generative Retrieval Model for Knowledge-Intensive Language Tasks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 191–200.

[8] Jang Hyun Cho and Bharath Hariharan. 2019. On the Efficacy of Knowledge Distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4794–4802.

[9] Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The Power of Noise: Redefining Retrieval for RAG Systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 719–729.

[10] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *International Conference on Learning Representations*.

[11] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. QLoRA: Efficient Finetuning of Quantized LLMs. *Advances in Neural Information Processing Systems* 36 (2024).

[12] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of Wikipedia: Knowledge-powered Conversational Agents. *arXiv preprint arXiv:1811.01241* (2018).

[13] Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long Form Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 3558–3567.

[14] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge Distillation: A Survey. *International Journal of Computer Vision* 129, 6 (2021), 1789–1819.

[15] Zhaochen Guo and Denilson Barbosa. 2018. Robust Named Entity Disambiguation with Random Walks. *Semantic Web* 9, 4 (2018), 459–479.

[16] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 conference on empirical methods in natural language processing*. 782–792.

[17] Yasuto Hoshi, Daisuke Miyashita, Youyang Ng, Kento Tatsuno, Yasuhiro Morioka, Osamu Torii, and Jun Deguchi. 2023. RaLLe: A Framework for Developing and Evaluating Retrieval-Augmented Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 52–69.

[18] Gautier Izacard and Edouard Grave. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In *EACL 2021-16th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 874–880.

[19] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot Learning with Retrieval Augmented Language Models. *Journal of Machine Learning Research* 24, 251 (2023), 1–43.

[20] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 1601–1611.

[21] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large Language Models Struggle to Learn Long-tail Knowledge. In *International Conference on Machine Learning*. 15696–15707.

[22] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. 6769–6781.

[23] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, and Ankur Parikh. 2019. Natural Questions: a Benchmark for Question Answering Research. *Transactions oof the Association for Computational Linguistics* 7 (2019), 452–466.

[24] Hyunji Lee, Jaeyoung Kim, Hoyeon Chang, Hanseok Oh, Sohee Yang, Vladimir Karpukhin, Yi Lu, and Minjoon Seo. 2023. Nonparametric Decoding for Generative Retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023*. 12642–12661.

[25] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-Shot Relation Extraction via Reading Comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. 333–342.

[26] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[27] Patrick 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. 2020. Retrieval-augmented Generation for Knowledge-intensive NLP Tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[28] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: Bootstrapping Language-image Pre-training with Frozen Image Encoders and Large Language Models. In *International Conference on Machine Learning*. PMLR, 19730–19742.

[29] Xiaoxi Li, Zhicheng Dou, Yujia Zhou, and Fangchao Liu. 2024. CorpusLM: Towards a Unified Language Model on Corpus for Knowledge-Intensive Tasks. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*.

[30] Xiaoxi Li, Yujia Zhou, and Zhicheng Dou. 2023. UniGen: A Unified Generative Framework for Retrieval and Question Answering with Large Language Models. In *AAAI Conference on Artificial Intelligence*. 8688–8696.

[31] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Generative Retrieval for Conversational Question Answering. *Information Processing & Management* 60, 5 (2023), 103475.

[32] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Learning to Rank in Generative Retrieval. In *The 38th Annual AAAI Conference on Artificial Intelligence*. 8716–8723.

[33] Jean Maillard, Vladimir Karpukhin, Fabio Petroni, Wen-tau Yih, Barlas Oguz, Veselin Stoyanov, and Gargi Ghosh. 2021. Multi-Task Retrieval for Knowledge-Intensive Tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 1098–1111.

[34] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2020. Generation-augmented Retrieval for Open-domain Question Answering. *arXiv preprint arXiv:2009.08553* (2020).

[35] Sanket Vaibhav Mehta, Jai Gupta, Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Jinfeng Rao, Marc Najork, Emma Strubell, and Donald Metzler. 2023. DSI++: Updating Transformer Memory with New Documents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 8198–8213.

[36] Donald Metzler, Yi Tay, Dara Bahri, and Marc Najork. 2021. Rethinking Search: Making Domain Experts out of Dilettantes. *ACM SIGIR Forum* 55 (2021), 1–27.

[37] Swati Mishra and Pankaj Bande. 2008. Maze solving algorithms for micro mouse. In *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*. IEEE, 86–93.

[38] Jay I Myung, Daniel J Navarro, and Mark A Pitt. 2006. Model Selection by Normalized Maximum Likelihood. *Journal of Mathematical Psychology* 50, 2 (2006), 167–179.

[39] Chaima Njeh, Haïfa Nakouri, and Fehmi Jaafar. 2024. Enhancing RAG-Retrieval to Improve LLMs Robustness and Resilience to Hallucinations. In *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 201–213.

[40] Jian-Xin Pan, Kai-Tai Fang, Jian-Xin Pan, and Kai-Tai Fang. 2002. Maximum Likelihood Estimation. *Growth curve models and statistical diagnostics* (2002), 77–158.

[41] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. 2021. KILT: a Benchmark for Knowledge Intensive Language Tasks. In *Proceedings of the 2021 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2523–2544.

[42] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language Models as Knowledge Bases? *arXiv preprint arXiv:1909.01066* (2019).

[43] Ronak Pradeep, Kai Hui, Jai Gupta, Adam Lelkes, Honglei Zhuang, Jimmy Lin, Donald Metzler, and Vinh Tran. 2023. How Does Generative Retrieval Scale to Millions of Passages?. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 1305–1321.

[44] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-text Transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[45] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2021. T5 Base. https://huggingface.co/t5-base.

[46] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context Retrieval-augmented Language Models. *Transactions of the Association for Computational Linguistics* 11 (2023), 1316–1331.

[47] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline M. Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference (NIST Special Publication, Vol. 500-225)*. 109–126.

[48] Severin Stalder, Helmut Grabner, and Luc Van Gool. 2010. Cascaded Confidence Filtering for Improved Tracking-by-Detection. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part I 11*. Springer, 369–382.

[49] Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. 2023. Learning to Tokenize for Generative Retrieval. In *Thirty-seventh Conference on Neural Information Processing Systems*. 46345–46361.

[50] Yubao Tang, Ruqing Zhang, Jiafeng Guo, Jiangui Chen, Zuowei Zhu, Shuaiqiang Wang, Dawei Yin, and Xueqi Cheng. 2023. Semantic-Enhanced Differentiable Search Index Inspired by Learning Strategies. In *29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4904–4913.

[51] Yubao Tang, Ruqing Zhang, Jiafeng Guo, and Maarten de Rijke. 2023. Recent Advances in Generative Information Retrieval. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*. 294–297.

[52] Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, and Xueqi Cheng. 2024. Generative Retrieval Meets Multi-Graded Relevance. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 72790–72817.

[53] Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, and Xueqi Cheng. 2024. Listwise Generative Retrieval Models via a Sequential Learning Process. *ACM Transactions on Information Systems* (2024).

[54] Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2024. Bootstrapped Pre-training with Dynamic Identifier Prediction for Generative Retrieval. In *Findings of the Association for Computational Linguistics ACL 2024*. 10303–10317.

[55] Yubao Tang, Ruqing Zhang, Zhaochun Ren, Jiafeng Guo, and Maarten de Rijke. 2024. Recent Advances in Generative Information Retrieval. In *Advances in Information Retrieval*, Nazli Goharian, Nicola Tonellotto, Yulan He, Aldo Lipani, Graham McDonald, Craig Macdonald, and Iadh Ounis (Eds.). Springer Nature Switzerland, Cham, 363–368.

[56] Yubao Tang, Ruqing Zhang, Zhaochun Ren, Jiafeng Guo, and Maarten de Rijke. 2024. Recent Advances in Generative Information Retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval)*. Association for Computing Machinery, New York, NY, USA, 3005–3008. https://doi.org/10.1145/3626772.3661379

[57] Yubao Tang, Ruqing Zhang, Weiwei Sun, Jiafeng Guo, and Maarten de Rijke. 2024. Recent Advances in Generative Information Retrieval. In *Companion Proceedings of the ACM Web Conference 2024* (Singapore, Singapore) *(The Web Conference 2024)*. Association for Computing Machinery, New York, NY, USA, 1238–1241. https://doi.org/10.1145/3589335.3641239

[58] Yi Tay, Vinh Q Tran, Mostafa Dehghani, Jianmo Ni, and Dara Bahri. 2022. Transformer Memory as a Differentiable Search Index. In *Advances in neural information processing systems*. 21831–21843.

[59] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large Language Models in Medicine. *Nature medicine* 29, 8 (2023), 1930–1940.

[60] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: A Large-scale Dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 809–819.

[61] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open Foundation and Fine-tuned Chat Models. *arXiv preprint arXiv:2307.09288* (2023).

[62] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text Embeddings by Weakly-supervised Contrastive Pre-training. *arXiv preprint arXiv:2212.03533* (2022).

[63] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2023. SimLM: Pre-training with Representation Bottleneck for Dense Passage Retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2244–2258.

[64] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Hao Sun, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. 2022. A Neural Corpus Indexer for Document Retrieval. In *Advances in Neural Information Processing Systems*. 25600–25614.

[65] Zihan Wang, Yujia Zhou, Yiteng Tu, and Zhicheng Dou. 2023. NOVO: Learnable and Interpretable Document Identifiers for Model-Based IR. In *32nd ACM International Conference on Information and Knowledge Management*. 2656–2665.

[66] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2369–2380.

[67] Hongbin Ye, Tong Liu, Aijia Zhang, Wei Hua, and Weiqiang Jia. 2023. Cognitive mirage: A review of hallucinations in large language models. *arXiv preprint arXiv:2309.06794* (2023).

[68] Jun Yu, Yunxiang Zhang, Zerui Zhang, Zhao Yang, Gongpeng Zhao, Fengzhao Sun, Fanrui Zhang, Qingsong Liu, Jianqing Sun, Jiaen Liang, et al. 2024. RAG-guided Large Language Models for Visual Spatial Description with Adaptive Hallucination Corrector. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 11407–11413.

[69] Hansi Zeng, Chen Luo, Bowen Jin, Sheikh Muhammad Sarwar, Tianxin Wei, and Hamed Zamani. 2024. Scalable and Effective Generative Information Retrieval. In *The Web Conference*.

[70] Hansi Zeng, Chen Luo, and Hamed Zamani. 2024. Planning Ahead in Generative Retrieval: Guiding Autoregressive Generation through Simultaneous Decoding. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*.

[71] Peitian Zhang, Zheng Liu, Yujia Zhou, Zhicheng Dou, and Zhao Cao. 2023. Term-Sets Can Be Strong Document Identifiers For Auto-Regressive Search Engines. *arXiv preprint arXiv:2305.13859* (2023).

[72] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, and Bin Cui. 2024. Retrieval-augmented Generation for AI-generated Content: A Survey. *arXiv preprint arXiv:2402.19473* (2024).

[73] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2023. Enhancing Generative Retrieval with Reinforcement Learning from Relevance Feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 12481–12490.

[74] Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, Peitian Zhang, and Ji-Rong Wen. 2022. Ultron: An Ultimate Retriever on Corpus with a Model-based Indexer. *arXiv preprint arXiv:2208.09257* (2022).

[75] Yu-Jia Zhou, Jing Yao, Zhi-Cheng Dou, Ledell Wu, and Ji-Rong Wen. 2023. DynamicRetriever: A Pre-trained Model-based IR System Without an Explicit Index. *Machine Intelligence Research* 20, 2 (2023), 276–288.

[76] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models. *arXiv preprint arXiv:2304.10592* (2023).

[77] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107* (2023).

[78] Shengyao Zhuang, Houxing Ren, and Linjun Shou. 2022. Bridging the Gap between Indexing and Retrieval for Differentiable Search Index with Query Generation. In *Gen-IR@SIGIR 2023: The First Workshop on Generative Information Retrieval*.