Direct Retrieval-augmented Optimization: Synergizing Knowledge Selection and Language Models

ZHENGLIANG SHI, Leiden University, Leiden, The Netherlands
LINGYONG YAN, Baidu Inc., Beijing, China
WEIWEI SUN, LTI, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
YUE FENG, School of Computer Science, University of Birmingham, Birmingham, UK
PENGJIE REN, Shandong University, Qingdao, China
XINYU MA, SHUAIQIANG WANG, and DAWEI YIN, Baidu Inc., Beijing, China
MAARTEN DE RIJKE, University of Amsterdam, Amsterdam, The Netherlands
ZHAOCHUN REN, Leiden University, Leiden, The Netherlands

Retrieval-augmented Generation (RAG) integrates **Large Language Models (LLMs)** with retrievers to access external knowledge, improving the factuality of LLM generation in knowledge-grounded tasks. To optimize the RAG performance, most previous work independently fine-tunes the retriever to adapt to frozen LLMs or trains the LLMs to use documents retrieved by off-the-shelf retrievers, lacking end-to-end training supervision. Recent work addresses this limitation by jointly training these two components but relies on overly simplifying assumptions of document independence, which has been criticized for being far from real-world scenarios. Thus, effectively optimizing the overall RAG performance remains a critical challenge. We propose a **Direct Retrieval-augmented Optimization (DRO)** framework that enables end-to-end training of two key components: (i) a generative knowledge selection model and (ii) an LLM generator. DRO alternates between two phases: (i) document permutation estimation and (ii) re-weighted maximization, progressively improving RAG components through a variational approach. In the estimation step, we treat *document permutation* as a latent variable and directly estimate its distribution from the selection model by applying an importance sampling strategy. In the maximization step, we calibrate the optimization expectation using importance weights and jointly train the selection model and LLM generator. Our theoretical analysis reveals that DRO is analogous to policy-gradient methods in reinforcement learning. Extensive experiments conducted on five datasets illustrate that DRO outperforms the best baseline with 5–15% improvements in EM

Work done during Zhengliang Shi's visit to Leiden University.

This work was supported by the National Key Research and Development Program of China (NO. 2024YFE0203200), Dutch Research Council (NWO) under project numbers 024.004.022, NWA.1389.20.183, and KICH3.LTP.20.006, and the European Union's Horizon Europe program under grant agreement No. 101070212 (FINDHR) and No. 101201510 (UNITE). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

Authors' Contact Information: Zhengliang Shi, Leiden University, Leiden, The Netherlands; e-mail: zhengliang.shii@gmail.com; Lingyong Yan, Baidu Inc., Beijing, China; e-mail: lingyongy@gmail.com; Weiwei Sun, LTI, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA; e-mail: sunnweiwei@gmail.com; Yue Feng, School of Computer Science, University of Birmingham, Birmingham, UK; e-mail: y.feng.6@bham.ac.uk; Pengjie Ren, Shandong University, Qingdao, China; e-mail: jay.r@outlook.com; Xinyu Ma, Baidu Inc., Beijing, China; e-mail: xinyuma2016@gmail.com; Shuaiqiang Wang, Baidu Inc., Beijing, China; e-mail: shqiang.wang@gmail.com; Dawei Yin, Baidu Inc., Beijing, China; e-mail: yindawei@acm.org; Maarten de Rijke, University of Amsterdam, Amsterdam, The Netherlands; e-mail: m.derijke@uva.nl; Zhaochun Ren (corresponding author), Leiden University, Leiden, The Netherlands; e-mail: z.ren@liacs.leidenuniv.nl.

ACM 1558-2868/2026/5-ART94
https://doi.org/10.1145/3795527

and F1. We also qualitatively analyze the stability, convergence, and variance of DRO. (Code is available on DRO GitHub).

CCS Concepts: • **Information systems** → **Retrieval models and ranking**;

Additional Key Words and Phrases: Retrieval-augmented generation, List-wise knowledge selection, Importance sampling, Expectation-Maximization principle

ACM Reference format:
Zhengliang Shi, Lingyong Yan, Weiwei Sun, Yue Feng, Pengjie Ren, Xinyu Ma, Shuaiqiang Wang, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. 2026. Direct Retrieval-augmented Optimization: Synergizing Knowledge Selection and Language Models. *ACM Trans. Inf. Syst.* 44, 4, Article 94 (May 2026), 30 pages.
https://doi.org/10.1145/3795527

1 Introduction

Large Language Models (LLMs) have shown remarkable text generation abilities. However, they often provide factually incorrect content [4, 53, 73] due to hallucination [16] or out-of-date information [9]. To mitigate these limitations, **Retrieval-augmented Generation (RAG)** has been proposed to integrate external retrievers with LLMs, which enables the model to access extensive corpora and retrieve relevant documents for references, thereby enhancing factuality. By integrating the retriever with LLMs, RAG has shown superior performance in knowledge-intensive tasks such as question answering [49, 61] and conversational information seeking [5, 24, 68].

Following the most widely used architecture [9, 11, 23], RAG typically includes two components to answer an input query: (i) *knowledge selection*, where retrieval and re-ranking models select target documents, (ii) *answer generation*, where an LLM generator generates correct answers conditioned on the selected documents. To enhance coverage and improve answer quality, RAG models often provide multiple retrieved documents as input to the generator. The interrelationships among these documents are crucial for final performance [15, 28, 32, 72]. We refer to a specific selection of retrieved documents as a *document permutation*.

Improving RAG Performance. To optimize RAG performance, some studies improve knowledge accuracy by fine-tuning the retrieval or ranking model with relevance criteria [33, 38, 48]. Others enhance the robustness of LLMs against irrelevant content through **Supervised Fine-tuning (SFT)** [10, 71] or in-context learning [49], teaching them to summarize key points from retrieved documents. However, these approaches optimize either the *selection* or the *generation* component separately while neglecting a dual enhancement, which may lead to sub-optimal overall performance [27, 45].

To address the above limitation, some recent studies train both the retriever and the LLM generator [23, 25, 51]. Specifically, they first retrieve the top-K documents, feed each document into downstream LLMs *one by one* for answer generation, and fine-tune the *retrieval model* to assign higher similarity to the documents that lead to better downstream outcomes. Despite progress, they simplify the knowledge into independent point-wise documents, which is typically known as *independent top-k approximation* [23, 72]. *This independent document modeling is far from reality, as RAG models often gather useful content from multiple documents*, such as in multi-hop question answering [62, 72], leading to a pronounced limitation in complex scenarios. Besides, training a retrieval model (especially the dual-tower dense retrieval [20]) requires frequent updates to the document index [13], which is difficult to implement and may be incompatible with established retrieval applications. Thus, a **Natural Question (NQ)** is: *How to synergize (i) knowledge selection; and (ii) answer generation to optimize holistic RAG performance?*

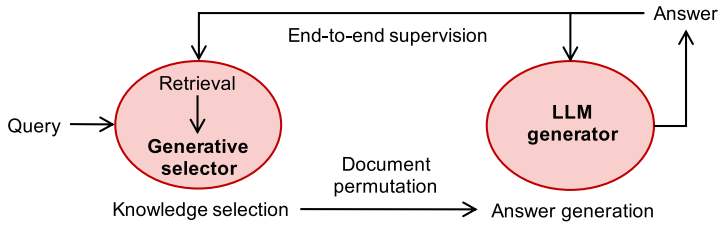

Fig. 1. Overview of DRO objective. The selection model directly estimates a document permutation for the generator to predict an answer, with both components trained jointly.

Direct RAG Optimization. In this article, we propose **Direct Retrieval-augmented Optimization (DRO)** method that synergizes (i) a list-wise selection model (a.k.a. selector) to generate target document permutations, and (ii) an LLM generator to predict the answer, enabling end-to-end improvement. As shown in Figure 1, the core idea is to directly treat the *document permutation* as a latent variable and estimate its distribution to maximize the log-likelihood of question answering. To achieve this, DRO iteratively alternates between two steps: (i) document permutation estimation and (ii) re-weighted maximization within the **Expectation-Maximization (EM)** principle [33].

In the *permutation estimation* step, we first define an ideal posterior distribution of document permutation inspired by the classic EM algorithm [33], introducing a tractable **Evidence Lower Bound (ELBO)** for the log-likelihood objective. Considering that exactly computing the posterior distribution is typically impractical, we employ an importance sampling strategy [8, 21] to directly estimate the document permutation distribution by sampling from the selection model. Specifically, for each input query, we first recall relevant documents using an off-the-shelf dense retriever, filtering documents with no semantic relevance to narrow down the candidate documents. The generative selector then selects a subset of documents by generating their document identifiers in an auto-regressive manner (e.g., `[1] > [2] > [3]`). In the *re-weighted maximization* step, we optimize the ELBO constructed in the estimation step, thereby improving the overall log-likelihood for question answering. We first re-weight the collected samples using importance weights to calibrate the bias introduced by sampling shifting from the importance sampling strategy. Then, we jointly optimize the selection model and the LLM generator by maximizing this re-weighted expectation, where both components are trained with end-to-end supervision. By alternating the estimation and maximization steps, DRO progressively improves the holistic RAG performance.

Theoretical Analysis. DRO differs from prior work such as [23, 25, 51] by: (i) *directly estimating* the distribution of useful knowledge by modeling document permutations as latent variables, which relaxes the *independent top-k approximation* assumption made in prior works; (ii) enabling *end-to-end optimization* for both knowledge selection and answer generation, rather than optimizing these components separately; and (iii) *iteratively aligning* the selection and generation models, achieving consistent performance improvements until convergence. To investigate the advantages of DRO, we provide a theoretical analysis of the learning objective and optimization process within our framework. We prove that DRO shares similarities with policy-based **Reinforcement Learning (RL)** approaches. In DRO, the selection module improves by reinforcing document permutations that enhance generation performance, while the generator, in turn, benefits from improved document permutations, creating a synergistic loop that optimizes the entire RAG process. Additionally, we provide a theoretical analysis about the training convergence and stability of DRO. We reveal that importance sampling with normalized weights can guarantee variance reduction and non-decreasing ELBO across iterations.

Experiments. We conduct extensive experiments across a wide range of datasets, including NQs [22], HotpotQA [69], **2WikiMultihopQA (2WikiQA)** [14], MusiQue [59], and **Wizard-of-Wikipedia (WoW)** [5]. The results show that the proposed DRO outperforms the best baselines with 5–15% improvement in EM and F1 metrics. Additionally, the selection model trained using our method achieves an average precision improvement of 17.78% in identifying target documents. We further conduct fine-grained analyses to examine the variance, convergence, and stability of the DRO during the training process. We observe substantial variance decay with increased sampling size, and consistent improvements (e.g., F1 score) over iterations, indicating stable and convergent optimization. These findings verify that DRO achieves not only strong performance but also robust training dynamics across datasets.

Contributions. The main contributions of this article are: (i) We propose DRO, a DRO method that treats the document permutation as a latent variable to enable an end-to-end improvement; (ii) We provide theoretical analysis for the learning objective of the proposed method and demonstrate its convergence and training stability; and (iii) extensive experiments conducted on 5 datasets show the improvement of our method, e.g., 5–15% improvement compared with state-of-the-art baselines.

2 Related Work

2.1 RAG

RAG aims to integrate external knowledge into LLMs, improving their factuality [11]. Given an input query, the first process of RAG is to select relevant knowledge, which is typically done by retrieval [20] or a ranking model [33]. Subsequently, an LLM generator incorporates these candidate documents to generate an answer. An active research question is how to improve the overall RAG performance. Some studies aim to improve the knowledge accuracy [6, 43], such as fine-tuning an answer-aware dense retriever [44, 48] or introducing additional modules for document filtering [63, 66]. Other work alternatively enhances the robustness of LLMs to irrelevant content, enabling LLMs to adaptively extract supporting facts from the retrieved documents [70, 76]. However, these methods either optimize the retrieval or the generation process without dual enhancement, potentially leading to sub-optimal performance [27]. Although existing work proposes the end-to-end training paradigm, they overly simplify a marginalization optimization through *independent top-k approximation* [43, 72], where they simply feed top-k documents into downstream LLMs one-by-one and re-score their relevance to optimize the retriever [23, 27]. This has been criticized far from the practical scenarios as the RAG system typically consumes multiple documents [72], while exhaustively enumerating all possible document permutations is cost-intensive and typically infeasible in practice. In this work, we propose DRO, which directly treats the document permutation as a latent variable and estimates its distribution for optimization.

2.2 Knowledge Selection for RAG

In RAG, the *knowledge selection* process aims to select target documents that can maximize LLM generation performance [12, 44]. To achieve this, prior work typically trains point-wise rankers (e.g., MonoT5 [33], BGE [65]) on conventional retrieval benchmarks (e.g., MS-MARCO [34]) and separately judges the relevance of each document to the input query. In contrast, our method applies a generative list-wise selection model [55], which selects target documents for the input query by generating corresponding document identifiers auto-regressively [37, 38]. Compared with point-wise ranking, our list-wise selection enables the comparison between multiple documents [30, 62], which can be inherently used to estimate the permutation distribution in our framework. Additionally, unlike previous ranking models trained with semantic relevance criteria [41], our selection model is jointly trained with the generator to maximize end-to-end RAG performance.

2.3 Variational Approach for Optimization

The variational approach has been widely applied in unsupervised scenarios and optimization involving latent variables [52, 56], such as GLEM in graph learning [75] and SSDM in speech modeling [26]. As a general principle, the variational approach, such as the EM algorithm [33], alternates between the *Expectation* step to compute the posterior distribution and the *Maximization* step to update the model parameter, optimizing the marginalization progressively. Inspired by this principle, in our work, we treat the document permutation as a latent variable. Besides, we directly estimate the permutation distribution using a list-wise selection model through the importance sampling [21]. This strategy diverges from the standard EM algorithm by avoiding the exact computation of the posterior distribution, a process that is impractical due to the large-scale document permutation space.

3 Preliminaries

3.1 Task Definition

Given an input query x, the task of RAG typically consists of two processes: (i) *knowledge selection* to acquire a set of relevant documents; and (ii) *answer generation* to generate a correct answer y by referring to the acquired documents. In the proposed DRO, our first process starts by using an off-the-shelf retrieval model (e.g., ColBERT [46]) to recall relevant documents $\boldsymbol{d} = \{d_1, d_2, \ldots, d_{|\boldsymbol{d}|}\}$ through semantic matching, thereby filtering the irrelevant contents to narrow down the candidates. Then, a generative re-ranking model θ_s reads these documents and selects a document permutation $\boldsymbol{z}$ by generating the corresponding document identifiers such as `[1] > [2] > ... > [k]`. Subsequently, an LLM generator θ_g predicts the answer y based on the input query x and selected documents $\boldsymbol{d_z}$, which can be formulated as $p(y \mid x, \boldsymbol{d_z}; \theta_g) = \prod_{t=1}^{|y|} p(y_t \mid y_{<t}, x, \boldsymbol{d_z}; \theta_g)$.

3.2 Generative Knowledge Selection for RAG

The generative selection model (also called the selector) identifies target documents in a list-wise manner. Given a query and a set of candidate documents, each associated with a unique identifier, the selector reads the query along with the entire document list and auto-regressively generates a sequence of document identifiers such as `[1] > [2] > [3]`. This sequence represents the K documents that the model considers most useful for answering the query.

Formally, the selector takes as input the query x and the candidate documents $\boldsymbol{d}$, and generates an ordered sequence of document *identifiers*:

$$p(\boldsymbol{z} \mid x; \theta_s) = \prod_{t=1}^{K} p(z_t \mid \boldsymbol{z}_{<t}, x, \boldsymbol{d}; \theta_s). \tag{1}$$

Here, $\boldsymbol{z} = \{z_t \mid t \in [K]\}$ denotes the generated document permutation of length K, where each token z_t is a document identifier (e.g., `[1]`, `[2]`).

By mapping the document permutations back to the original documents, we obtain K selected documents, denoted as $\boldsymbol{d_z} = \{\boldsymbol{d}_{z_t} \mid t \in [K]\}$. Compared with traditional point-wise ranking, which assigns each individual document a relevance score to the query, we use this generative selection model for two reasons: (i) it inherently enables the comparison of multiple candidate documents through the attention mechanism [54, 60]; and (ii) it allows the direct modeling of document permutations, ordered lists of documents that capture their interrelationships by autoregressively generating the docid list [31].

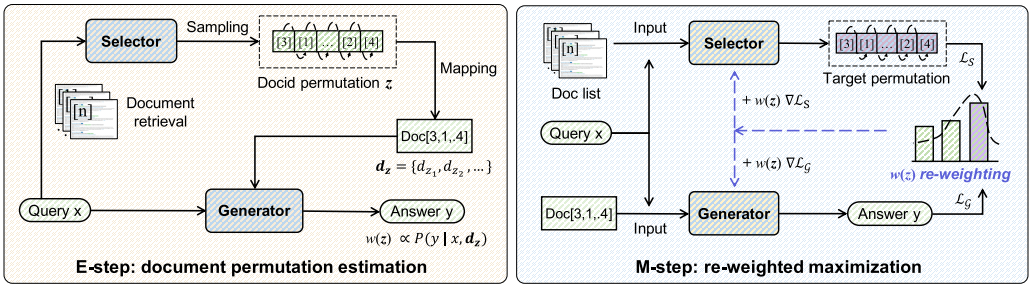

Fig. 2. The overall framework for DRO alternates between the (i) E-step: document permutation estimation (Section 4.2); and (ii) M-step: re-weighted maximization (Section 4.3) to progressively optimize the holistic RAG performance.

4 Direct RAG Optimization

In this work, the proposed DRO method, , as shown in Figure 2, improves the holistic performance of RAG by synergizing (i) knowledge selection and (ii) answer generation processes. To achieve this, DRO enables the end-to-end training of (i) a list-wise generative selector with parameters θ_s to estimate the target document permutation for the input query x; and (ii) an LLM generator with parameters θ_g to generate accurate answers. For simplicity, we use $\theta = (\theta_s, \theta_g)$ to represent the overall tunable parameters of DRO throughout the article. Below, we first derive the learning objective of DRO and then relate it to how to achieve the end-to-end optimization of the selection and generation model for such an objective.

4.1 Deriving the DRO Objective

As pointed out by prior work, generating a correct answer grounded on reference knowledge is identified as one of the most crucial goals of RAG tasks [11, 23, 45]. In DRO, we define the optimization objective as maximizing the marginal log-likelihood of generating the correct answer y to the input query x using external documents $\boldsymbol{d}_{\boldsymbol{z}}$. This can be formulated as:

$$\log p(y|x;\theta) = \log \sum_{\boldsymbol{z}} p(y, \boldsymbol{d}_{\boldsymbol{z}}|x;\theta). \tag{2}$$

Since summing over all possible $(y, \boldsymbol{z})$ is typically intractable, we employ a variational approach to construct a tractable lower bound for $\log p(y|x;\theta)$ that we then maximize. Specifically, we introduce a variational distribution $q(\boldsymbol{z}|x)$ and apply *Jensen's inequality* to $\log \sum_{\boldsymbol{z}} p(y, \boldsymbol{d}_{\boldsymbol{z}}|x;\theta)$:

$$\begin{aligned}\log \sum_{\boldsymbol{z}} q(\boldsymbol{z}|x) \frac{p(y, \boldsymbol{d}_{\boldsymbol{z}}|x;\theta)}{q(\boldsymbol{z}|x)} &\geq \sum_{\boldsymbol{z}} q(\boldsymbol{z}|x) \log \frac{p(y, \boldsymbol{d}_{\boldsymbol{z}}|x;\theta)}{q(\boldsymbol{z}|x)} \\ &= \underbrace{\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}|x)} \left[\log \frac{p(y, \boldsymbol{d}_{\boldsymbol{z}}|x;\theta)}{q(\boldsymbol{z}|x)}\right]}_{\text{Evidence Lower BOund (ELBO}(q,\theta))}.\end{aligned} \tag{3}$$

Here, we define the $\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}|x)} \left[\log \frac{p(y,\boldsymbol{d}_{\boldsymbol{z}}|x;\theta)}{q(\boldsymbol{z}|x)}\right]$ as an ELBO for $\log p(y|x;\theta)$. Based on Jensen's inequality, the $\text{ELBO}(q,\theta) = \log p(y|x;\theta)$ if and only if when the $\frac{p(y,\boldsymbol{d}_{\boldsymbol{z}}|x;\theta)}{q(\boldsymbol{z}|x)} \equiv c$, where c is a constant.

From Equation (3), we can derive the DRO objective as a progressive optimization process, which includes: (i) estimating the distribution of document permutation $\boldsymbol{z}$ to achieve $\log p(y|x;\theta) \approx \text{ELBO}$, and (ii) maximizing the ELBO to improve $\log p(y|x;\theta)$.

4.1.1 EM for DRO Training. To optimize the DRO objective, we adopt the EM algorithm with an importance sample strategy. In more detail, we start by demonstrating the condition for the alignment in Equation (3), which is formulated in the following lemma.

Lemma 1. *For ELBO$(q, \theta) = \log p(y|x;\theta)$, there exists a variational distribution $q(z|x)$ such that $q(z|x) = p(z|x, y;\theta)$.*

Proof. This lemma can be proved by considering the case where the importance weight $\frac{p(y,z|x;\theta)}{q(z|x)}$ is constant, denoted as c, across all z. This is formulated as below:

$$\frac{p(y, z \mid x;\theta)}{q(z \mid x)} \equiv c \quad \forall x, y.$$

Then, we can sum both sides over z and obtain: $\sum_z p(y, z \mid x;\theta) = c \sum_z q(z|x) \equiv c$. Here $q(z \mid x)$ is a probability distribution over z, i.e., it sums to 1. Therefore, we solve for $q(z \mid x)$ as:

$$q(z \mid x) = \frac{p(y, z \mid x;\theta)}{\sum_z p(y, z \mid x;\theta)} = p(z \mid x, y;\theta). \tag{4}$$

Therefore, the variational distribution $q(z \mid x)$ that matches the true posterior $p(z \mid x, y;\theta)$ achieves the exact ELBO $= \mathbb{E}_{z\sim q(z|x)}\left[\log \frac{p(y,\boldsymbol{d}_z|x;\theta)}{q(z|x)}\right]$. □

The Lemma 1 shows an intuitive solution to achieve $\log p(y|x;\theta) \approx \text{ELBO}(q, \theta)$ by exactly computing the posterior distribution $p(z \mid x, y;\theta)$ for latent document permutation z. However, it is often impractical due to the large-scale permutation space. To address this challenge, we use an importance sampling strategy, where the $q(z|x)$ is directly set to $p(z \mid x;\theta_s)$. Consequently, the training of DRO is achieved within an EM principle, including:

(i) E-step: *document permutation estimation* (Section 4.2) to estimate the distribution of document permutations by sampling from the selection model.
(ii) M-step: *re-weighted maximization* (Section 4.3) to jointly optimize the selection model and generator using the importance-weighted samples.

By iteratively alternating these two steps, DRO progressively improves the holistic RAG objective $\log p(y|x;\theta)$.

4.2 E-step: Document Permutation Estimation

This step aims to estimate the distribution of document permutations. Specifically, at the tth iteration, we first assume $q(z \mid x) \approx p(z \mid y, x;\theta^t)$ and transform the ELBO(θ, θ^t) in Equation (3) into:

$$\begin{aligned} &\mathbb{E}_{z\sim p(z|x,y;\theta^t)}\left[\log p(y, z \mid x;\theta) - \log p(z \mid x, y;\theta^t)\right] \\ &= \mathbb{E}_{z\sim p(z|x,y;\theta^t)}\left[\log p(y, z \mid x;\theta)\right] + \mathcal{H}\left(p(z|x, y;\theta^t)\right), \end{aligned} \tag{5}$$

where the z is ideally sampled from the posterior distribution $p(z \mid x, y;\theta^t)$ to compute the expectation. The $\mathcal{H}(p(z|x,\theta^t))$ indicates the entropy of $p(z|x,\theta^t)$, which is independent to θ and can be viewed as a constant. Since the posterior distribution $p(z|y, x;\theta)$ is intractable, we alternatively adopt an importance sampling strategy to directly sample the document permutation z from our selection model θ_s via $z \sim p(z \mid x;\theta_s^t)$. To correct the bias introduced by the sampling shifting, we also employ an importance weight to calibrate the expectation. Formally, it is presented

Algorithm 1: The Algorithm for DRO, Which Alternates between `Estimation` and `Maximization` Steps to Progressively Improve the Overall RAG Performance

Input: selection model θ_s; LLM generator θ_g; training iteration number N; training data $\mathcal{T}$.

for $t = 1$ *to* N **do**

  `// Permutation Estimation (E-step)`

  for *each input query* x *in training set* **do**

    Sample document permutations: $z \sim p(z \mid x; \theta_s^t)$

    Compute importance weight: $w(z) = p(y \mid x, z; \theta_g^t)$

  `// Re-weighting (M-step)`

  $\mathcal{L}_{\mathcal{S}}(x; \theta_s) := -\mathbb{E}_{z \sim p(z|x;\theta_s^t)} \left[w(z) \log p(z \mid x; \theta_s) \right]$

  $\mathcal{L}_{\mathcal{G}}(x; \theta_g) := -\mathbb{E}_{z \sim p(z|x;\theta_s^t)} \left[w(z) \log p(y \mid x, \boldsymbol{d}_z; \theta_g) \right]$

  `// Re-weighted Maximization (M-step)`

  $\theta_s^{t+1} = \arg\max_{\theta_s} \mathbb{E}_{(x,y)\sim\mathcal{T}} \left[-\mathcal{L}_{\mathcal{S}}(x; \theta_s) \right]$

  $\theta_g^{t+1} = \arg\max_{\theta_g} \mathbb{E}_{(x,y)\sim\mathcal{T}} \left[-\mathcal{L}_{\mathcal{G}}(x; \theta_g) \right]$

  if *no improvement on validation set* **then**

    Stop training Maximization `// Early Stop`

Output: (θ_s, θ_g)

as follows:

$$\begin{aligned} \text{ELBO}(\theta, \theta^t) &= \mathbb{E}_{z \sim p(z|x;\theta_s^t)} \left[w(z) \log p(y, z|x; \theta) \right] + \text{constant} \\ w(z) &= \frac{p(z \mid x, y; \theta^t)}{p(z \mid x; \theta_s^t)}. \end{aligned} \tag{6}$$

Here, we denote the $w(z)$ as the *importance weight*. According to Bayes' theorem, where $p(z \mid y, x; \theta^t) \propto p(z \mid x; \theta_s^t) \times p(y \mid \boldsymbol{d}_z, x; \theta_g^t)$, we can then simplify the $w(z)$ as:

$$w(z) \propto \frac{p(z \mid x; \theta_s^t) \times p(y \mid \boldsymbol{d}_z, x; \theta_g^t)}{p(z \mid x; \theta_s^t)} = p(y \mid x, \boldsymbol{d}_z; \theta_g^t). \tag{7}$$

Intuitively, the weight $w(z)$ reflects the utility of a set of documents $\boldsymbol{d}_z$ for the LLM generator θ_g in generating ground-truth answers. By sampling from the generative selection model θ_s ($z \sim p(z \mid x; \theta_s)$) and calibrating the expectation with importance weights, we directly estimate the distribution of the latent variable z (i.e., the document permutation) for unbiased optimization, without explicitly computing the posterior distribution $p(z \mid x, y; \theta^t)$.

4.3 M-step: Re-weighted Maximization

After the permutation estimation step, the maximization step aims to update the tunable parameter θ in the RAG system. Formally, the optimization objective is defined as:

$$\theta^{t+1} = \arg\max_{\theta} \mathbb{E}_{z \sim p(z|x;\theta_s^t)} \left[w(z) \log p(y, z \mid x; \theta) \right]. \tag{8}$$

Here, the $\theta = (\theta_s, \theta_g)$ denotes the tunable parameters of knowledge selection model θ_s and LLM generator θ_g. Based on Bayes' theorem, we have $p(y, z \mid x; \theta) = p(z \mid x; \theta_s) \cdot p(y \mid x, \boldsymbol{d}_z; \theta_g)$. Then, the $\text{ELBO}(\theta, \theta^t)$ can be rewritten as a decomposition of two parts:

$$\mathbb{E}_{z \sim p(z|x;\theta_s^t)} \Bigg[\underbrace{w(z) \log p(z \mid x; \theta_s)}_{\text{Learning to select}} + \underbrace{w(z) \log p(y \mid x, \boldsymbol{d}_z; \theta_g)}_{\text{Learning to generate}} \Bigg]. \tag{9}$$

Thus, we derive two loss functions, namely (i) selection optimization $\mathcal{L}_{\mathcal{S}}$ and (ii) generation optimization $\mathcal{L}_{\mathcal{G}}$:

$$\mathcal{L}_{\mathcal{S}} := -\mathbb{E}_{z\sim p(z|x;\theta_s^t)}\left[w(z)\log p(z \mid x;\theta_s)\right], \tag{10}$$

$$\mathcal{L}_{\mathcal{G}} := -\mathbb{E}_{z\sim p(z|x;\theta_s^t)}\left[w(z)\log p(y \mid x, \boldsymbol{d}_z;\theta_g)\right]. \tag{11}$$

Learning to Select. The function $\mathcal{L}_{\mathcal{S}}$ optimizes the selection model θ_s for document permutation generation. Since $\mathcal{L}_{\mathcal{S}}$ is weighted by $w(z)$ and $w(z) \propto p(y|x, \boldsymbol{d}_z;\theta_s^t)$, the model θ_s learns to generate the document permutation that maximizes the end-to-end performance. Based on Equation (10), the gradient of $\mathcal{L}_{\mathcal{S}}$ with respect to θ_s is:

$$\nabla_{\theta_s}\mathcal{L}_{\mathcal{S}} = -\mathbb{E}_{z\sim p(z|x;\theta_s^t)}\left[w(z)\nabla_{\theta_s}\log p(z \mid x;\theta_s)\right]. \tag{12}$$

Learning to Generate. The loss function for θ_g is defined as:

$$\mathcal{L}_{\mathcal{G}} := -\mathbb{E}_{z\sim p(z|x;\theta_s^t)}\left[w(z)\log p(y \mid x, z;\theta_g)\right], \tag{13}$$

training the LLM generator to understand the selected documents and generate correct answers. The gradient of $\mathcal{L}_{\mathcal{G}}$ with respect to θ_g can be formulated as:

$$\nabla_{\theta_g}\mathcal{L}_{\mathcal{G}} = -\mathbb{E}_{z\sim p(z|x;\theta_s^t)}\left[w(z)\nabla_{\theta_g}\log p(y \mid x, \boldsymbol{d}_z;\theta_g)\right]. \tag{14}$$

Practical Implementation. In practice, the optimization of both the selection model and the generator can be viewed as a form of *re-weighted SFT*. The importance weight serves as a soft indicator of the utility of the sampled document permutation: a higher weight indicates that the selected documents are highly useful for generating the correct answer, while a lower weight suggests that the permutation is less beneficial. This weighting mechanism allows the model to focus on high-quality document selections during training, effectively aligning the knowledge selection process with the end-to-end generation performance.

4.4 Upshot: Explaining DRO with a Pseudo Algorithm

To provide a more intuitive explanation of the overall optimization process, we present a pseudo algorithm in Algorithm 1. In DRO, the document permutation distribution is first estimated using the selection model θ_s (E-step). Subsequently, the model θ_s learns to select documents that maximize the generation performance while the generator θ_s, in turn, further learns to leverage the selected documents (M-step). These two steps are iteratively alternated, enabling the progressive improvement of the holistic RAG system.

5 Theoretical Analysis

To further interpret DRO, we offer a theoretical analysis of its advantages and training stability, and prove its convergence.

5.1 What Do the Selector and Generator Learn?

The format of the optimization gradient in Equations (12) and (14) is similar to classic policy-gradient approaches [1, 57, 74] used in RL. Generally, given an input task x and the action τ generated by the policy model π, the policy-gradient objective $\mathcal{J}(\pi)$ to maximize an reward function $r(\tau)$ can be presented as:

$$\begin{aligned} \mathcal{J}(\pi) &= \mathbb{E}_{\tau\sim p(\tau|x;\pi)}\left[r(\tau)\right], \\ \nabla_\pi\mathcal{J}(\pi) &= \mathbb{E}_{\tau\sim p(\tau|x;\pi)}\left[r(\tau)\cdot\nabla_\pi\log p(\tau \mid x;\pi)\right]. \end{aligned} \tag{15}$$

Below, we make several comparisons to relate RL with our DRO.

- $w(z) \iff r(\tau)$: In our learning objectives, $w(z)$ plays a role similar to the reward $r(\tau)$ in RL. It represents the importance of the document permutation z and serves as a weighting factor to evaluate the utility of documents to downstream tasks, analogous to how rewards $r(\tau)$ shape the policy model in RL.
- $p(z \mid x; \theta_s^t) \iff p(\tau \mid x; \pi_\theta)$: The sampling distribution over permutations $p(z \mid x; \theta_s^t)$ reflects the state of the selection model at iteration t, similar to how $p(\tau \mid x; \pi_\theta)$ represents the current policy distribution over trajectories in RL.
- $\log p(y, z \mid x; \theta) \iff \log p(\tau \mid x; \pi)$: The $\log p(y, z \mid x; \theta)$ in our Equation (8) is analogous to the $\log p(\tau \mid x; \pi)$ in RL. In both cases, the gradient is updated to increase the likelihood of actions (or permutations z) that yield higher rewards (or weights $w(z)$).

Understanding DRO from an RL Perspective. The objective functions in Equation (9) can be interpreted as a two-agent system collaborating within the RAG task. In the estimation step, directly sampling a document permutation from the selection model θ_s is essentially a Monte Carlo process, while $w(z) = p(y \mid x, \boldsymbol{d}_z; \theta_g)$ in the maximization step serves as the reward. Similar to the RL, where policy models improve by reinforcing high-reward actions, the selection model θ_s improves by selecting documents that lead to better generation. The generator θ_g, in turn, learns to leverage the selected documents to generate correct answers, creating a synergistic loop that optimizes the entire RAG performance.

5.2 Impact of Importance Sampling

In the permutation estimation step, we apply an importance sampling strategy to directly sample document permutations from the knowledge selection model (i.e., $z \sim p(z \mid x; \theta_s)$) instead of the posterior distribution $p(z \mid x, y; \theta)$. We analyze its impact on the expectation and variance of the training objective. For simplicity, we denote $\log p(y, z \mid x; \theta^t)$ in the ELBO as a function $f(y, z)$.

5.2.1 Expectation Is Unchanged. In the permutation estimation step, we employ importance sampling to approximate the posterior distribution $p(z \mid x, y; \theta)$ with samples drawn from the proposal distribution $p(\mathbf{z} \mid \mathbf{x}; \theta_s)$, which is parameterized by the selection model. To compensate for the discrepancy between the target and proposal distributions, we apply an importance weight the weight $w(z) = p(y \mid \boldsymbol{d}_z, x; \theta)$ adjusts for the different between the target distribution $p(z \mid x; \theta_s^t)$ and the sampling distribution $p(z \mid x; \theta_s)$, which can be presented as:

$$\begin{aligned} \mathbb{E}_{p(z|x,y;\theta^t)}[f(z)] &= \mathbb{E}_{p(z|x;\theta_s^t)}\left[\frac{p(y \mid x, \boldsymbol{d}_z; \theta^t)}{p(z \mid x; \theta_s^t)} f(z)\right] \\ &= \mathbb{E}_{p(z|x;\theta_s^t)}[w(z) f(z)]. \end{aligned} \tag{16}$$

The key property of importance sampling is that it provides an unchanged estimator of the expectation under the true posterior. This fundamental property of importance sampling ensures that the expectation in the variational optimization objective remains unchanged.

5.2.2 Variance Decreases. While importance sampling preserves the expectation, it crucially affects the *variance* of the estimator, which directly impacts the stability and efficiency of gradient-based training. We begin by formulating the *vanilla variance*, i.e., the variance before applying importance sampling.

Before using importance sampling, the *vanilla variance* $\mathrm{Var}_{p(z|x,y;\theta^t)}[f(y, z)]$ is formulated as:

$$\mathbb{E}_{p(z|x,y;\theta^t)}\left[f(y, z)^2\right] - \left(\mathbb{E}_{p(z|x;\theta_s^t)}[f(y, z)]\right)^2. \tag{17}$$

After using importance sampling, we denote the *new variance* as $\mathrm{Var}_{p(z|x;\theta_s^t)}\left[w(z)f(y,z)\right]$, which becomes:

$$\mathbb{E}_{p(z|x;\theta_s^t)}\left[w(z)^2 f(y,z)^2\right] - \left(\mathbb{E}_{p(z|x;\theta_s^t)}\left[w(z)f(y,z)\right]\right)^2. \tag{18}$$

To evaluate the effect of importance sampling on variance, we derive the difference $\Delta\mathrm{Var}$ between the new and old variances:

$$\begin{aligned}\Delta\mathrm{Var} &= \mathrm{Var}_{p(z|x;\theta_s^t)}\left[w(z)f(y,z)\right] - \mathrm{Var}_{p(z|x,y;\theta^t)}\left[f(y,z)\right] \\ &= \mathbb{E}_{p(z|x,y;\theta^t)}\left[f(y,z)^2\,(w(z)-1)\right].\end{aligned} \tag{19}$$

Since $w(z) \propto p(y \mid x, \boldsymbol{d}_z; \theta_g^t)$ and the $0 \le p(y \mid x, \boldsymbol{d}_z; \theta_g^t) \le 1$, it follows that $\Delta\mathrm{Var} \le 0$ if $w(z)$ is normalized. This expression reveals a key insight: if $w(z) \le 1$ for all z, then $\Delta\mathrm{Var} \le 0$, indicating that the variance of the importance-weighted estimator is strictly *lower* than the original variance. To further guarantee numerical stability and prevent rare outlier samples with disproportionately large weights, we apply normalization to $w(z)$ in practice:

$$w(z) = \frac{w(z)}{\sum_{z'} w(z')}. \tag{20}$$

This normalization step ensures bounded gradients and facilitates smoother convergence throughout the DRO training process.

5.2.3 Training Process Gradually Stabilizes. From Equations (17) and (18), we notice that the $w(z)$ plays a key role in shaping the variance, which affects the training stability. Since $w(z) \propto p(y \mid x, \boldsymbol{d}_z; \theta_g^t)$, we have the following analysis to demonstrate the training stability of our DRO: (i) initially, the generator θ_g^t is less optimized, which potentially results in high variance in $p(y \mid x, z; \theta_g^t)$; (ii) as generator θ_g^t is optimized through the loss function in Equation (11), $p(y \mid x, z; \theta_g^t)$ stabilizes, which leads to a progressively reduced variance.

5.3 Convergence of the Optimization Process

To prove the convergence of DRO, we show the non-decreasing and upper-bounded property of $\log p(y|x, \theta^t)$ during the training.

We first prove that the log-likelihood $\log p(y \mid x, \theta^{t+1})$ is non-decreasing after each training iteration, which is formulated as: $\log p(y \mid x, \theta^{t+1}) \ge \log p(y|x, \theta^t)$. Here $\theta = (\theta_s, \theta_g)$ consists of the parameters of the selection model θ_s and the LLM generator θ_g.

Proof. At each iteration t, we start by estimating the distribution of the latent variable z, i.e., document permutation. Since we apply an importance sampling strategy to directly sample z from the selection model θ_s^t, the ELBO is transformed as in Equation (6):

$$\mathrm{ELBO}(\theta, \theta^t) := \mathbb{E}_{z\sim p(z|x;\theta_s^t)}\left[w(z)\log p(y, z \mid x; \theta)\right].$$

In the maximization step, we update θ by maximizing the ELBO: $\theta^{t+1} = \arg\max_\theta \mathrm{ELBO}(\theta, \theta^t)$. This ensures that $\mathrm{ELBO}(\theta^{t+1}, \theta^t) \ge \mathrm{ELBO}(\theta^t, \theta^t)$. We further observe that the marginal log-likelihood in Equation (3) satisfies:

$$\begin{aligned}\log p(y \mid x; \theta^{t+1}) &\ge \mathrm{ELBO}(\theta^{t+1}, \theta^t) \\ &\ge \mathrm{ELBO}(\theta^t, \theta^t) = \log p(y \mid x; \theta^t).\end{aligned} \tag{21}$$

Thus, we establish that $\log p(y \mid x; \theta^{t+1}) \ge \log p(y \mid x; \theta^t)$, demonstrating the non-decreasing nature of the optimization process. Next, we examine the boundedness of the log-likelihood. Given that $0 \le p(y \mid x; \theta) \le 1$, it follows that $-\infty \le \log p(y \mid x; \theta) \le 0$. Then, we introduce an existing theorem from [3] as follows.

Table 1. Statistics of Our Experimental Datasets

Experimental benchmarks	Training data size	Query length (Train)	Evaluation data size	Query length (Evaluation)	Retrieval corpus
Nature Question [22]	58,622	9.21	6,489	9.16	Wiki2018
HotpotQA [69]	90,185	17.85	7,384	15.63	Wiki2018
MusiQueQA [59]	19,938	15.96	2,417	18.11	Wiki2018
2WikiQA [14]	167,454	12.74	12,576	11.97	Wiki2018
WoW [5]	63,734	70.41	3,054	70.25	Wiki2018

We provide the amount of training and evaluation dataset, the average length of input query (word) as well as the retrieval corpus.

Theorem 5.1 (Monotone Convergence Theorem). *If a sequence $\{a_n\}$ is monotonic (either non-decreasing or non-increasing) and bounded, then it converges to a finite limit.*

Applying Theorem 5.1, the non-decreasing and upper-bounded nature of $\{\log p(y \mid x; \theta^t)\}_{t=1}^{\infty}$ ensures that the sequence converges to a finite limit, proving the convergence of the DRO training. In practice, when the performance on the validation set shows no further improvement, the model is considered to have converged. □

5.4 Upshot

Below, we summarize three key insights from the above theoretical analysis:

(1) The objective of DRO can be interpreted as optimizing a collaborative two-agent system within the RAG framework. The selection model θ_s improves by identifying document permutations that enhance the generator's performance. In turn, the generator θ_g learns to better utilize the selected documents to produce correct answers. This mutual improvement forms a synergistic loop that jointly optimizes the overall RAG performance.
(2) The use of importance sampling in DRO preserves the expectation of the learning objective while reducing variance, particularly when the importance weights are normalized. This contributes to more stable and efficient training.
(3) The convergence of DRO is theoretically guaranteed by the monotonic increase of the log-likelihood and its upper boundedness. In practice, this implies that the model either improves or maintains its performance over iterations, and training can be safely terminated when validation performance saturates, ensuring both convergence and training efficiency.

6 Experimental Setup

6.1 Datasets

Following previous work [23, 24, 71, 72], we conduct experiments on a *full development set* of five commonly used question-answering benchmarks: NQ [22], HotpotQA [69], MuSiQue [59], 2WikiQA [14], and WoW [5]. Table 1 presents the statistics of these datasets.

6.2 Evaluation Metrics

In line with previous studies [25, 49, 64], we use *F1* and EM metrics from KILT [36] for evaluation. The *F1* score is used to measure the token-level overlap between the generated answer and the ground-truth answer, which represents the harmonic mean of precision and recall, where the recall is determined by considering the number of overlaps with the correct answer tokens, while

precision is determined by considering the number of overlaps with all generated tokens. The *EM* metric checks if the predicted strings exactly match the ground truth. Besides the end-to-end evaluation, we also use the *Recall@K* ($K = 1, 3, 5$) to evaluate the document re-ranking performance of our selection model following previous work [24, 36, 45], in which the Recall@K is set to 1 if and only if the top-K documents contain the ground-truth answer.

6.3 Baselines

We compare the proposed DRO with four categories of baselines based on how they integrate the knowledge selection process with the answer generation process, namely *Prompting-based methods*, *Retrieval Tuning*, *LLM Fine-tuning*, and *End-to-end training* methods.

6.3.1 Prompting-based Methods. These methods guide LLMs to leverage external knowledge through prompt learning. We evaluate:

(1) *RetGen* [47], which interleaves query formulation, retrieval, and answer generation iteratively.
(2) *GenGround* [49], which instructs an LLM to generate answers and then revise them using retrieved documents.
(3) *In-context RAG* [40], which truncates the top-k retrieved documents as context for the LLM to generate answers.

For the first two baselines, we use GPT-3.5-turbo as the backbone, following their official implementations. For *in-context RAG*, we evaluate various LLMs with the same three in-context examples.

6.3.2 Retrieval Tuning. These methods improve the entire RAG performance by enhancing the retrieval process. We evaluate:

(1) *REPLUG* [48] and *DPA-RAG* [6], which adapt a dense retriever or point-wise ranker to a frozen LLM by re-scoring the relevance.
(2) *FILCO* [63] and *RECOMP*, which filter irrelevant content from retrieved documents.
(3) *Re-ranking Models*, which re-rank retrieved documents, passing the top-K ranked results for LLMs.

We benchmark point-wise (MonoT5 [35] and BGE [65]) and list-wise rankers (RankVicuna [37] and RankZephyr [38]).

6.3.3 LLM Fine-tuning. These methods fine-tune LLMs through instruction tuning, improving LLMs' ability to utilize retrieved documents. We evaluate:

(1) Vanilla SFT, which trains LLMs by maximizing the answer likelihood based on query and documents;
(2) *ChatQA* [29] and *RankRAG* [71], which post-train LLMs on diverse knowledge-grounded tasks;
(3) *RetRobust* [70], *InstructRAG* [64], *Self-RAG* [2], and *RAAT-7B* [10], which train LLMs to identify relevant content from retrieved documents for QA.

6.3.4 End-to-end Training. These methods train RAG components with an end-to-end objective. We benchmark:

(1) *Atlas* [17], a pretrained retrieval-augmented LLM.
(2) *RA-DIT* [27], which initially trains a generator and subsequently fine-tunes a dual-encoder retriever.

Table 2. Efficiency Statistics across Datasets

Metrics	NQ	HotpotQA	2WikiQA	MuSiQue	WoW
Training time (*hour*)	15	18	17	18	20
Retrieval time (*seconds*)	0.08	0.87	0.09	0.09	0.09
Cost (*tokens*)					
—Document selection	600.46/19.44	635.43/18.50	648.07/19.02	629.87/19.89	728.52/18.44
—Answer generation	533.45/3.59	567.58/3.98	580.15/4.20	561.88/4.40	661.150/23.48

For token usage, we report the input/output token counts.

(3) *DDR-RAG* [25], which employs Direct Preference Optimization [39] to jointly train a pointwise ranker and an answer generator.

To ensure fairness, we set the size of the set of retrieval documents to 20 for all the baselines and the $K = 5$, aligning with the implementation of DRO. Since the code or model checkpoints are unavailable for some baselines, we mark their incomplete results as "–". In such cases, we report results from the original papers but only for reference.

6.4 Implementation Details

We use Llama-3-8B [7] to initialize the parameters for both our ranker and LM generator. Although they share the same pretrained backbone, they are instantiated as two independent parameter sets and updated separately according to their respective loss functions (i.e., Equations (11) and (10)). We also alternate it with another model, i.e., Mistral-7B [18], to evaluate the generalizability of DRO across different backbones.

Given a query x, we initially use the off-the-shelf ColBERTv2.0 [46] to retrieve the top 20 documents as input for the selection model θ_s, which selects a permutation containing $K = 5$ documents to the generator θ_g. Following previous work [20, 25, 67], the document corpus for the retrieval is based on the Wikipedia passage dump from 20 December, 2018. We set the maximum training iteration $N = 5$ (Section 4.3) and report the performance for each iteration checkpoint for a comprehensive analysis. The sampling number for document permutation is set to 8. We use the DeepSpeed ZeRO strategy [42] during training, with learning rates of $1e^{-5}$ and a weight decay coefficient of 0.01. We also report the training and inference cost of DRO in Table 2.

7 Experimental Results

7.1 Overall Performance

7.1.1 Single-hop QA Benchmarks. We first analyze the performance of the proposed DRO on open-domain QA and dialogue datasets, including NQ and WoW. The inputs in these datasets are complex, posing challenges for neural models to comprehend [22]. Table 3 presents the results, where we observe that our DRO achieves the best performance (green lines), such as pushing the EM from 40.74 to 45.76 in the NQ dataset (12.32% relative improvements). To investigate the reasons for our superior performance, we conduct a coarse-to-fine analysis. We begin by identifying some evaluation cases that are incorrectly answered by strong baselines (e.g., InstructRAG) while being correctly answered by our method. Then we compare the outputs from both approaches for these cases. The selection model, trained collaboratively with the LLM generator in DRO, selects more useful documents for answer generation. This demonstrates that our selection model learns

Table 3. Experimental Results on Five Benchmarks

Tasks		NQ		HotpotQA		MuSiQue		2WikiQA		WoW	Avg.	
Metrics	**Scale**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**F1**	**EM**	**F1**
Prompting-based method												
RetGen[a] [47]	175B	37.75	39.78	39.04	42.10	17.69	21.04	33.00	39.17	16.97	31.87	31.81
GenGround[a] [49]	175B	40.60	42.31	41.27	44.71	20.77	24.36	39.61	42.58	17.76	35.56	34.34
In-context RAG [40]												
—*w/* Llama3-inst.-70B [7]	70B	39.38	47.26	37.38	39.62	16.43	21.16	37.26	41.46	17.06	32.61	33.31
—*w/* Llama2-70B-chat [58]	70B	38.07	44.69	37.14	40.27	16.78	20.11	38.51	41.02	15.75	32.62	32.37
—*w/* Mixtral-inst.-8x7B [19]	56B	39.34	46.34	39.63	42.53	16.65	20.73	37.03	38.50	16.66	33.16	32.95
—*w/* Llama2-13B-chat [58]	13B	35.27	41.54	35.43	40.37	16.87	18.37	35.47	38.63	13.12	30.76	30.41
Retrieval tuning												
REPLUG [66]	65B	28.80	–	32.00	–	–	–	–	–	–	–	–
RECOMP [66]	20B	37.47	42.67	38.72	42.72	17.34	24.96	32.17	38.26	15.11	31.43	32.74
DPA-RAG [6]	8B	37.29	44.31	37.15	40.53	18.45	20.36	39.02	39.66	14.73	32.98	31.92
FILCO [63]	7B	35.32	37.24	38.74	39.21	14.53	16.81	29.74	33.05	14.23	29.58	28.11
MonoT5 [35] *w/* Mistral[†]	7B	31.78	37.68	32.38	38.76	14.31	19.16	35.96	37.11	14.27	28.61	29.40
RankVicuna [37] *w/* Mistral[†]	7B	33.78	39.58	34.74	41.25	15.38	21.87	36.72	38.93	14.45	30.16	31.22
RankZephyr [38] *w/* Mistral[†]	7B	34.77	45.22	36.08	42.77	16.03	21.75	35.07	38.31	14.94	30.49	32.60
BGE-ranker [65] *w/* Mistral[†]	7B	35.21	40.50	37.61	38.10	16.61	21.37	37.16	38.49	14.77	31.65	30.65
LLM fine-tuning												
RetRobust [70]	13B	37.03	43.82	35.59	40.54	18.11	18.16	38.65	39.11	17.04	32.34	31.73
ChatQA [66]	8B	23.64	34.54	33.40	44.60	16.64	17.05	26.80	31.90	16.22	25.12	28.86
RankRAG [66]	8B	–	–	35.30	46.70	–	–	31.40	36.90	–	–	–
InstructRAG [64]	8B	35.90	38.21	30.69	34.71	14.94	25.88	35.92	20.01	14.57	29.36	26.68
Vanilla SFT *w/* Llama3 [58]	8B	35.25	38.46	25.07	32.57	14.35	17.82	30.65	30.43	13.91	26.33	26.64
Vanilla SFT *w/* Mistral [18]	7B	34.65	37.52	25.75	30.65	13.35	17.97	30.43	30.65	13.83	26.05	26.12
Self-RAG [2]	7B	29.74	31.63	16.30	27.30	9.43	21.50	23.52	27.33	13.24	19.75	24.2
RAAT [10]	7B	33.53	37.85	33.01	31.67	17.08	21.79	29.69	32.68	15.37	28.33	27.87
End-to-end optimization												
RA-DIT [27]	65B	35.20	–	39.70	–	–	–	–	–	–	–	–
Atlas [17]	11B	26.70	–	34.70	–	–	–	–	–	–	–	–
DDR-RAG [25]	8B	40.74	28.76	31.71	40.04	13.54	10.57	35.44	38.40	16.21	30.36	26.80
DRO-Mistral-7B (**Ours**)	7B	42.41	51.01	40.37	47.87	**21.36**	25.32	**42.12**	43.65	18.31	36.56	37.23
DRO-Llama-3-8B (**Ours**)	8B	**45.76**[‡]	**55.42**[‡]	**42.23**[†]	**49.27**[‡]	20.64[†]	**25.97**[†]	40.12[†]	**44.12**[‡]	**18.76**[‡]	**37.19**[‡]	**38.71**[‡]

We highlight the best results in bold and underline the second-best results. We also compute the average EM or F1 across all datasets. *Scale* indicates the parameter size of the LLM generator.

[a]The baselines based on closed source *gpt-3.5*.

[†] and [‡]Significant improvements over best open source baselines with p-value < 0.05 and 0.01, respectively, measured by two-tailed paired t-test across three independent runs (n = 3) with fixed random seeds 42, 43, 44. The green shaded cells indicate the results of our proposed method.

to understand utility-oriented document selection criteria in RAG tasks, contributing to holistic improvement.

7.1.2 Multi-hop QA Benchmarks. We further validate the superiority of the proposed method on multi-hop QA tasks, which are more challenging since they require the model to aggregate evidence from multiple documents to derive the answer. For example, based on our statistics, queries in HotpotQA involve at least 2.21 hops, while those in MuSiQue require an average of 2.65 hops. As shown in Table 3, our DRO substantially outperforms existing baselines, such as achieving a F1 score of 49.27 and an EM score of 42.23 in the HotpotQA dataset. The reasons for this improvement are twofold: (1) *Reward in the selection model*: The weight $w(z)$ in the selection model's learning objective serves as a reward function. This weight reinforces the model's ability to comprehensively select documents that maximize the overall performance of the end-to-end generation process, which is crucial for multi-hop reasoning. and (2) *Synchronous optimization of the LLM generator*: The LLM generator is optimized synchronously to leverage the selected document, thereby further enhancing overall effectiveness.

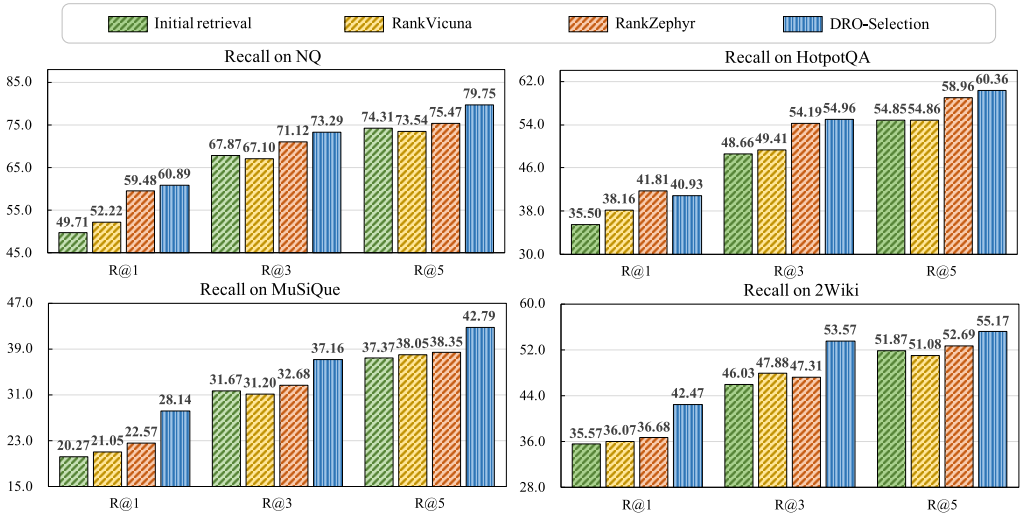

Fig. 3. Recall@K (k = 1, 3, 5) score of the initial retrieval (Colbertv2.0), two re-ranking baselines (i.e., RankVicuna and RankZephyr) and our selection model θ_s, respectively.

7.1.3 Document Selection Evaluation. In addition to the end-to-end evaluation presented in Table 3, we further evaluate the effectiveness of our selection model in the context of document re-ranking using the Recall@K metric. This metric evaluates the ability of the model to recall relevant documents, with a particular emphasis on how well the top-K retrieved documents align with the ground-truth answer in the context of RAG tasks. As defined in previous works [20, 22], we set Recall@K to 1 if the ground-truth answer is present within the top-K documents returned by the retrieval model.

The results of this evaluation are shown in Figure 3. The selection model, integrated into our DRO framework, demonstrates significant improvements in recall over the initial retrieval method, ColBERTv2.0. For example, DRO increases Recall@K by 17.78% on average in terms of the NQ dataset, which highlights the effectiveness of our selection model in enhancing the usefulness of retrieved documents. Figure 3 also shows the comparison with other similar list-wise re-ranking baselines to further illustrate the performance of our selection model. We find that the DRO-selection model outperforms these baselines, achieving the highest recall across multiple datasets. This finding underscores the importance of end-to-end supervision in the knowledge selection process, demonstrating that joint training with the LLM generator yields superior document selection, which directly contributes to the overall quality of the answer generation process.

Considering the potential data contamination issues highlighted by prior work [55], we additionally evaluate our selector, along with widely used document re-rankers, on a newly collected dataset, i.e., NovelEval [55]. As shown in Table 4, DRO-selection achieves substantially better performance than previous baselines. These results further validate the robustness and effectiveness of our proposed method under less contaminated evaluation settings.

7.2 Training Convergence

As illustrated in Algorithm 1, DRO iteratively optimizes the model parameters $\theta = (\theta_s, \theta_g)$ to improve performance. To analyze convergence, we evaluate the model checkpoint at each training iteration. Table 5 presents the results of four experimental datasets. For both DRO-Mistral-7B and DRO-LLaMA-3-8B, we observe a consistent and substantial increase in F1 scores during the first

Table 4. Experimental Results on the NovelEval Dataset, a Newly Proposed Dataset with Less Data Contamination

Model	Recall@1	Recall@3	Recall@5	nDCG@1	nDCG@3	nDCG@5
Initial Retrieval	12.42	32.90	46.55	64.29	59.88	58.25
Qwen-32B	16.19	48.18	70.24	80.95	80.96	83.10
LLaMA-3.1-70B	15.79	50.95	75.40	80.95	82.53	87.11
Qwen-72B	17.18	49.60	69.48	90.48	86.40	86.51
Rankzepyhr	21.71	61.64	77.20	73.81	74.55	75.42
DRO-selection	27.22	74.13	1.00	88.10	83.90	92.64

Table 5. F1 Score for Checkpoints in Each Training Iteration

Iteration	**NQ**	**HotpotQA**	**MuSiQue**	**2WikiQA**	**Average Δ**
DRO-*Mistral-7B*					
1	41.56	39.65	20.79	36.50	–
2	$47.19_{\uparrow 11.9\%}$	$43.71_{\uparrow 9.3\%}$	$_1$ $23.26_{\uparrow 10.6\%}$	$39.53_{\uparrow 7.7\%}$	9.8%
3	$49.98_{\uparrow 5.6\%}$	$46.03_{\uparrow 5.0\%}$	$24.87_{\uparrow 6.9\%}$	$41.54_{\uparrow 4.8\%}$	5.5%
4	$50.82_{\uparrow 1.7\%}$	$47.24_{\uparrow 2.6\%}$	$25.07_{\uparrow 0.8\%}$	$41.75_{\uparrow 0.5\%}$	1.4%
5	$50.97_{\uparrow 0.3\%}$	$47.87_{\uparrow 1.3\%}$	$25.32_{\uparrow 1.0\%}$	$42.12_{\uparrow 0.9\%}$	0.9%
6	$50.52_{\downarrow 0.9\%}$	$48.03_{\uparrow 0.3\%}$	$25.63_{\uparrow 1.2\%}$	$41.87_{\uparrow 0.5\%}$	0.2%
7	$50.96_{\uparrow 0.3\%}$	$47.90_{\downarrow 0.3\%}$	$25.55_{\downarrow 0.3\%}$	$42.15_{\uparrow 0.5\%}$	0.1%
DRO-*Llama-3-8B*					
1	44.84	41.07	21.43	38.43	–
2	$49.11_{\uparrow 8.7\%}$	$44.73_{\uparrow 8.2\%}$	$_1$ $24.06_{\uparrow 10.9\%}$	$41.27_{\uparrow 6.9\%}$	8.7%
3	$_1$ $54.98_{\uparrow 10.7\%}$	$48.24_{\uparrow 7.3\%}$	$25.24_{\uparrow 4.7\%}$	$43.02_{\uparrow 4.1\%}$	6.7%
4	$55.01_{\uparrow 0.1\%}$	$49.06_{\uparrow 1.7\%}$	$25.94_{\uparrow 2.7\%}$	$43.65_{\uparrow 1.4\%}$	1.5%
5	$55.42_{\uparrow 0.7\%}$	$49.27_{\uparrow 0.4\%}$	$25.97_{\uparrow 0.1\%}$	$44.12_{\uparrow 1.1\%}$	0.6%
6	$55.24_{\downarrow 0.3\%}$	$49.42_{\uparrow 0.3\%}$	$25.41_{\downarrow 2.1\%}$	$44.28_{\uparrow 0.4\%}$	−0.4%
7	$55.46_{\uparrow 0.4\%}$	$49.57_{\uparrow 0.3\%}$	$25.37_{\downarrow 0.2\%}$	$44.32_{\uparrow 0.1\%}$	0.2%

three iterations. DRO outperforms competitive baselines such as BGE and RankZephyr after just two iterations, demonstrating the promising performance of our method. Besides, performance gains taper off between iterations 3 and 5, suggesting that the model has entered a convergence phase. To further verify this observation, we conduct significance testing between iterations 4 and 5. Across three independent runs, a two-tailed paired t-test at the 0.05 significance level detects no statistically significant difference.

In summary, this convergence analysis not only demonstrates the stability and effectiveness of DRO but also showcases its promising performance, as it surpasses competitive baselines early in the training process.

7.3 Training Stability

In our maximization step, we employ an importance weight $w(\boldsymbol{z})$ to calibrate the optimization expectation. Our theoretical analysis in Section 5.2.2 shows that the training variance decreases as training progresses, since $w(\boldsymbol{z}) \propto p(y \mid x, \boldsymbol{d_z}; \theta_g)$. To validate this finding, we compute the variance of sampling from $p(y \mid x; \boldsymbol{d_z})$ at each training iteration. We vary the number of samples from 1

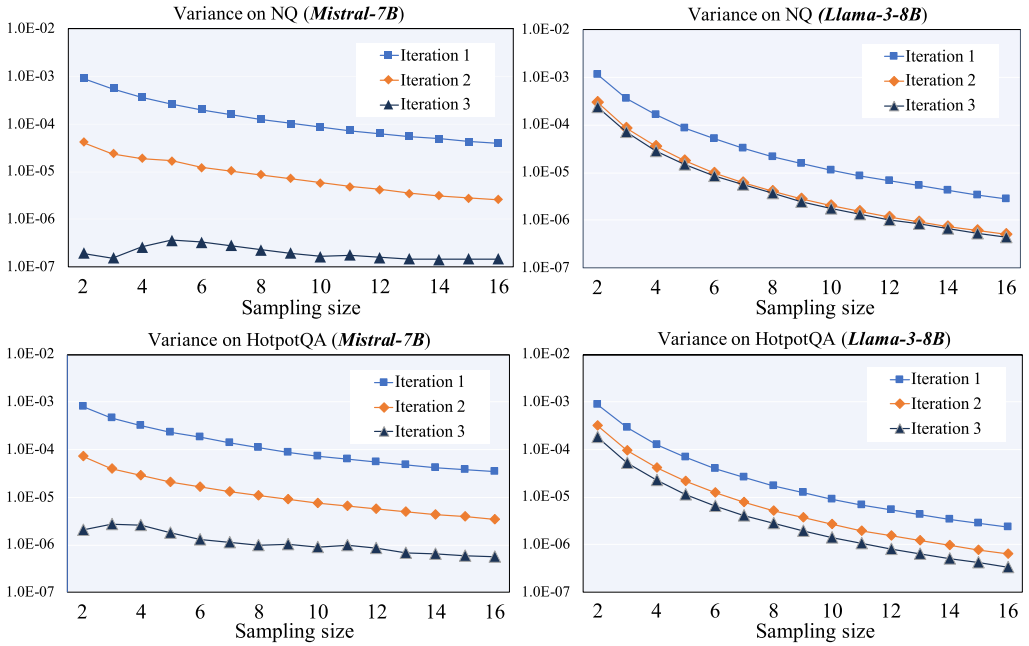

Fig. 4. Variance during the training process of our method (logarithmic scale).

to 16 to compute the variance, respectively (see Figure 4). The variance substantially decreases with the optimization of model θ_g during training progress, validating the correctness of our theoretical analysis from Section 5.2.3. We also find that increasing the number of samples per iteration can reduce variance at each training step, which is a straightforward solution to improve the training stability. Besides, in our experiments, we observe that increasing the number of samples per iteration reduces variance at each training step, offering a straightforward strategy to improve training robustness, especially during early-stage training (see Section 8.2 for more details).

Remark 7.1. The reduction of variance during training is a direct consequence of the improved confidence of the generator θ_g in predicting correct answers. As $p(y \mid x, \boldsymbol{d}_z; \theta_g)$ stabilizes, the importance weight $w(z)$ becomes less volatile, leading to lower variance and enhanced training stability. This validates the use of importance weighting in DRO as not only theoretically sound but also practically stabilizing.

7.4 Ablation Studies

To investigate the individual contributions of the two tightly coupled components in our DRO method, namely the selection model θ_s and the LLM generator θ_g, we conduct a comprehensive ablation study. In this study, we isolate the training of each module while freezing the other to examine their individual impact on overall performance. Specifically, we compare the full DRO method against two ablated variants:

(i) *DRO-w/o Selector*: In this variant, the selection model θ_s is fixed, and only the LLM generator θ_g is updated during training.
(ii) *DRO-w/o Generator*: In this variant, the LLM generator θ_g is fixed, and only the selection model θ_s is updated.

Fig. 5. Ablation study on five datasets to demonstrate the effectiveness of training the selection model θ_s and generator θ_g.

Figure 5 illustrates the F1 scores across five datasets, i.e., NQ, HotpotQA, MuSiQue, 2WikiQA, and WoW, over five training iterations. The results demonstrate the importance of jointly optimizing both components of the model. For example, on the **NQ** dataset, the vanilla DRO method achieves an F1 score of *51.01* at the 5th training iteration, while the two ablated variants perform notably worse. Specifically, DRO-*w/o Selector* achieves an F1 score of 46.77, suffering a drop of 4.24 points, and DRO-*w/o Generator* obtains 48.71, which is a decrease of 2.30 points. Similar trends can also be observed across other datasets:

(i) *HotpotQA*: The full DRO method achieves an F1 score of 47.87, outperforming the ablated variants, with DRO-*w/o Selector* scoring 44.60 and DRO-*w/o Generator* scoring 43.11.
(ii) *MuSiQue*: The complete model achieves 25.32 F1, surpassing DRO-*w/o Selector* (22.59) and DRO-*w/o Generator* (21.77).
(iii) *2WikiQA*: DRO achieves 43.65 F1, while DRO-*w/o Selector* and DRO-*w/o Generator* score 40.12 and 39.01, respectively.
(iv) *WoW*: DRO reaches 18.31 F1, outperforming DRO-*w/o Selector* (16.47) and DRO-*w/o Generator* (15.73).

These results demonstrate two crucial findings. *First*, training the selection model θ_s is critical for learning high-quality document permutations that substantially benefit the answer generation stage. *Second*, optimizing the generator θ_g to better utilize the selected documents further improves performance. Besides, as the full DRO method consistently outperforms the ablated variants, the synergy between both components is evident. The combined training of both modules yields cumulative gains that neither component can achieve independently.

Remark 7.2. The ablation study strongly highlights the necessity of joint optimization in the DRO method. It validates that the co-training of the selector and generator can boost the holistic performance of RAG.

8 Discussion

8.1 On Document Order and Duplicates in DRO

Handling of Duplicate Document Identifiers. In DRO, the generative selector autoregressively produces a sequence of document identifiers. We explicitly instruct the model to generate a permutation of distinct documents in the system prompt. We observe that duplicate IDs are rare in practice, as existing LLMs have strong instruction-following capabilities. If duplicates do occur, we follow the common practice in prior list-wise ranking and generative selection methods [55] by performing deduplication, retaining only the first occurrence of each document. The final input to the generator is thus a set of unique documents, ensuring that no redundant information is passed to the generation stage.

Table 6. Performance of DRO When Shuffling the Documents Selected by DRO-Selector before Feeding Them to the DRO-Generator

Method	NQ		HotpotQA	
	EM	F1	EM	F1
Mistral-7B-inst. as backbone				
Vanilla DRO	42.41	51.01	40.37	47.87
Shuffle DRO	42.33	50.41	40.54	47.43
Δ DRO	$-0.08\downarrow$	$-0.60\downarrow$	$+0.17\uparrow$	$-0.44\downarrow$
Δ% DRO	$-0.19\downarrow$	$-1.18\downarrow$	$+0.42\uparrow$	$-0.92\downarrow$

We initialize both the selector and generator using Mistra-7B-Inst.
Δ denotes the changes with respect to the *vanilla* setting.

Document Order in Generator Conditioning. The selector sequentially decodes an ordered list of document identifiers, reflecting its estimation of the relative utility of the selected documents. We directly concatenate the documents according to this order to form the generator's context. To investigate the generator's sensitivity to document order, we further shuffle the selected documents before feeding them into the DRO-generator and re-evaluate its performance. As illustrated in Table 6, we observe almost no performance difference between the *vanilla* and *shuffled* versions of DRO. We attribute this robustness to two main factors. First, the DRO-selector consistently selects the most informative top-K documents (typically $K \leq 5$), ensuring that the generator receives highly relevant contextual information regardless of order. Second, the DRO-generator is built on a 7B-parameter backbone with strong language understanding capabilities, which provides inherent resilience to variations in document ordering within a small document set.

8.2 Impact of the Number of Samples

In our training procedure, for each input query, we sample $m = 8$ document permutations z from the selection model to construct the ELBO objective via importance sampling. To further examine the effect of this sampling number on training dynamics and model performance, we vary m across $\{1, 2, 4, 6, 8, 10, 12, 14\}$ and evaluate the resulting models under the same experimental setup described in Table 3. Figure 6 summarizes the results. We derive several observations from the analysis: (i) *Higher sampling improves performance.* Overall, increasing the sampling number m consistently improves the final F1 score. This trend suggests that using more samples provides a better approximation of the expected objective, thereby guiding the optimization process more accurately. (ii) *Sampling number affects convergence speed.* Larger values of m not only lead to better performance but also accelerate convergence. For example, when $m = 14$, the model reaches peak performance within just two training iterations. In contrast, with only $m = 1$, the model requires up to four iterations to achieve similar results. This aligns with our variance analysis in Section 5.2.2, as increased sampling reduces training variance and facilitates faster optimization. (iii) *Tradeoff between cost and effectiveness.* While higher values of m yield better and faster results, they also incur greater computational overhead. In practice, we find that using $m = 3$ or $m = 4$ strikes a good balance, offering competitive performance with reasonable training efficiency. These findings highlight the importance of sampling strategies in importance-weighted optimization and suggest practical guidelines for choosing sampling numbers based on resource availability and convergence requirements.

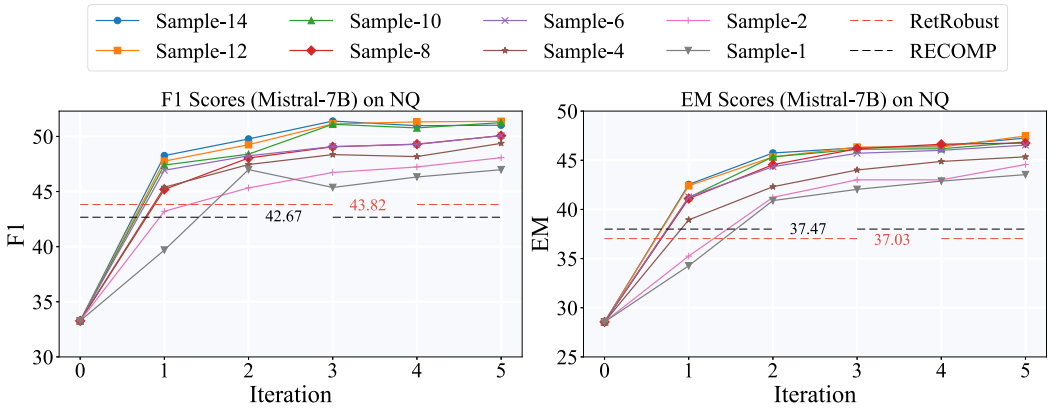

Fig. 6. Performance under different sampling numbers (Section 8.2).

Table 7. Human Evaluation on 100 Randomly Sampled Cases

	DPA-RAG	RetRobust	DDR-RAG	DRO-Mistral
Correctness	37/100	32/100	33/100	41/100

8.3 Performance with Scaling-down Parameter

In our experiment, we follow prior work (e.g., RankVicuna [37]) and employ a general-purpose LLM (e.g., Llama) as the backbone for list-wise selection. Recent studies typically scale up parameter sizes to explore performance upper bounds. However, since the document selection task involves long context sequences as input with a concern of increased latency, we investigate a more practical low bound by scaling down the size of the selection model θ_s in DRO. Specifically, we implement our method using Llama-3-8B as the generator and pairing it with smaller LLMs as selection models. We evaluate the performance under the same conditions as Table 3 and present the results in Figure 7. We observe that as the parameter size of the model θ_s increases from 1B to 8B, performance improves substantially. The Llama-3-8B generator, when paired with the smallest selector (Llama-3-1B), outperforms strong baselines such as RetRobust, pushing the F1 score from 43.82 to 46.83 on the NQ dataset. Additionally, as an empirical suggestion, training a 3B model (e.g., Llama-3-3B) as the selection model offers a balanced tradeoff between performance and computational cost in our method.

8.4 Human Evaluation

Considering the potential bias of automatic metrics [50], we conduct a human evaluation with three educated individuals assessing the *correctness* of 100 randomly sampled cases from five benchmarks, using a three-point scale. Each query is paired with the corresponding golden documents and ground-truth answers from the original datasets, which serve as references for the human evaluators. We ask at least two annotators to evaluate the same case repeatedly. If there is a discrepancy between two annotators, we ask a third annotator to recheck it. The results are presented in Table 7. The DRO achieves a correctness score of 0.41, while strong open source baselines only score between 0.32 and 0.37, demonstrating the advantage of our proposed method. The average Kappa statistic for our human evaluation is 0.751, indicating strong agreement.

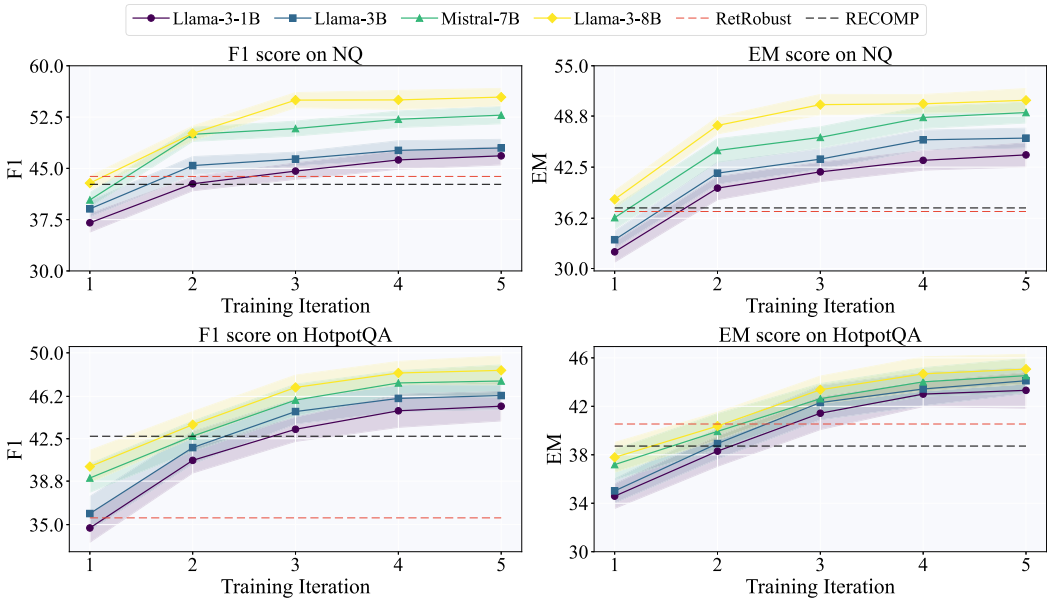

Fig. 7. Performance with different parameter sizes of selection model θ_s.

8.5 Training Cost of DRO

While DRO involves iterative joint training of both the selector and generator, we identify several empirically effective strategies to reduce its computational cost without sacrificing performance.

Empirical Early Stopping. We can employ early stopping when no significant performance improvement is observed on the validation set for consecutive iterations. In practice, we observe rapid convergence across all experiments. For instance, models like Llama-3-8B typically stabilize by the third iteration, and Qwen-2.5-7B converges by the fourth, as measured by validation set performance. This consistent pattern indicates that only a small number of DRO iterations (e.g., 3–5) is sufficient in practice, and training can be safely halted once performance plateaus.

Reduced Sampling. As analyzed in Section 8.2, the number of document permutations sampled per query directly impacts cost. We find that sampling only 3 or 4 permutations per query already yields strong performance, striking a favorable balance between efficiency and effectiveness.

By combining these strategies, the overall training cost of DRO can be significantly reduced, making it feasible for practical deployment.

8.6 Case Study

We conduct case studies to intuitively analyze the advantages and disadvantages of DRO. Below, we first show the system prompt to the document selection model θ_s and answer generation model θ_g. Then, we present the correctly completed cases and the bad cases, respectively, for comparison.

8.6.1 System Prompt for Document Selection Models. In the prompt for the selection model θ_s, we instruct the model to generate a ranked list of useful documents in a generative, auto-regressive manner, where document identifiers are produced in descending order of utility. Documents that contain the correct answer are considered more useful and are expected to be ranked higher.

```
You are RankLLM, an intelligent assistant that can rank passages based on
 their relevance and usefulness to the user's query.

I will provide you with {n_docs} passages. Please rank these passages
based on their usefulness in answering the user's search query: "{
question}".

A passage's usefulness is defined as:
1. Relevance to the question.
2. Contains necessary information to help the user.

The passages are listed below, each with a numerical identifier [].
{docs}

Rank the {n_docs} passages based on their usefulness in descending order.
 Use the format [] > [], e.g., [2] > [1]. Only respond with the ranking
results; do not provide explanations.

Search Query: {question}
Your output:
```

8.6.2 System Prompt for Answer Generation Model. In the prompt for the answer generation model θ_g, we provide the documents selected by the selection model as references, and instruct the model to generate the final answer grounded in the information contained within these documents.

```
You are an artificial intelligence assistant. You should gives helpful
and precise answers to the user's questions based on the context. The
context here refer to external passages related to user's questions.

Please answer the question: "{question}" using provided passages. Each
passage is indicated by a numerical identifier [].
Here are the related passages for your reference.
{docs}
Question: {question}
Your answer:
```

8.6.3 Correctly Completed Cases. In DRO, the DRO-selector is trained to identify and provide the most informative documents that maximize the answer generation performance of the DRO-generator. Our case study reveals that the selector is effective at selecting relevant evidence even when the required information is distributed across multiple documents.

Below, we provide a case in which the input query asks: *Which games, Strange Synergy or Qwirkle, is a card game published by Steve Jackson Games?* with the ground truth being *Strange Synergy*. To answer this question, it is essential to retrieve not only detailed descriptions of Strange Synergy but also background information about Qwirkle for comparative analysis.

Among the initially retrieved documents, documents [1], [2], and [17] are all individually necessary to resolve the query. Specifically, passages [1] and [2] provide overlapping but essential descriptions of Strange Synergy, consistently identifying it as a card game published by Steve Jackson Games. Meanwhile, passage [17], though it offers necessary contrastive information about Qwirkle, ranked much lower initially. In DRO, the selection model successfully identifies [1], [2], and [17] as the top-ranked passages, demonstrating its ability to select the most relevant target information. Based on these selected passages, the generator compares the publisher information from [1] and [17], accurately associating Strange Synergy with Steve Jackson Games as the

correct answer. This example highlights the strength of our DRO in enabling the cooperation between the selection and generation models.

```
INPUT QUERY: Which games, Strange Synergy or Qwirkle,  is a card game
published by Steve Jackson Games?

INITIALLY RETRIEVED DOCUMENTS:
[0] Title: Strange Synergy. Content: Strange Synergy Strange Synergy is a
 card game published by Steve Jackson Games in which players build a team
 of super heroes to battle an opponent's team. The game supports up to
four players... [1] Title: Strange Synergy. Content: enemies for three
turns in a row, or capture an opposing team's flag. Strange Synergy
Strange Synergy is a card game published by Steve Jackson Games in which
players build a team of super heroes to battle an opponent's team. The
game supports up to four players. However, there are bases and flags for
two more players.
[2] ...
[3] ...
[4] ...
[5] ... [6] ... [7] ... [8] ... [9] ... [10] ... [11] ... [12] ... [13]
... [14] ... [15] ... [16]...
[17] Title: Qwirkle. Content: Qwirkle Qwirkle is a tile-based game for 2
to 4 players, designed by Susan McKinley Ross and published by MindWare.
Qwirkle shares some characteristics with the games Rummikub and Scrabble.
 It is distributed in Canada by game and puzzle company, Outset Media.
Qwirkle is considered by MindWare to be its most awarded game of all time
.
[18] ...
[19] ...
(... indicates omitted document contents for a clearer presentation.)

OUTPUT OF SELECTION MODEL: [1] > [2] > [17]
OUTPUT OF GENERATOR MODEL: Strange Synergy

--------------------------------------------------------

GROUND TRUTH: Strange Synergy
EXACTLY MATCH: 1.0
```

8.6.4 Bad Case of DRO. We also investigate the bad case of DRO to analyze the potential disadvantages of our method. Table 8 summarizes two key types of errors: incorrect selection and generation mismatch. First, the *incorrect selection* indicates that the selector fails to comprehensively include all documents that are necessary to answer the question. Without such contrastive evidence, the generator tends to produce incorrect answers, even when relevant documents are present in the retrieved pool. This suggests the need for diversity-aware selection strategies that explicitly promote comparative reasoning. Second, the *generation mismatch* indicates that the generator model produces an incorrect answer, though the ground-truth documents have been incorporated into the context. This often occurs when the input contains partially relevant or distracting content, i.e., the noise. These cases highlight the challenge of robust generation.

8.6.5 Integrating DRO with Post-verification. To address the *generation mismatch* issue identified in our case studies (Table 6), we introduce an optional post-verification module. After the standard

Table 8. Representative Failure Cases of DRO

Case type	Description	Observed behavior	Potential remedies
Selector misses contrastive document	The selector fails to comprehensively include documents (e.g., different publishers), which are necessary for disambiguation.	The generator outputs an incorrect answer due to the lack of comparative evidence.	Incorporate multi-view selection or contrastive supervision to enforce diversity in document selection.
Generator is misled by noisy content	Although the selector provides useful documents, the generator fails to generate the correct answer.	The LLM misunderstands the input documents and predicts an incorrect answer.	Introduce an answer verification module chain-of-thought reasoning process for LLM, before generating the final answer.

We summarize two typical types of failure, i.e., selection and generation mismatch, and suggest potential remedies.

Table 9. Performance of DRO-Mistral-7B with Post-Verification Using *Qwen-2.5-7B* as the Verifier Model

Method	NQ		HotpotQA	
	EM	F1	EM	F1
Vanilla DRO-Mistral-7B	42.41	51.01	40.37	47.87
—*w/* Verifier	$43.82_{\uparrow 1.41}$	$52.34_{\uparrow 1.33}$	$42.91_{\uparrow 2.54}$	$48.43_{\uparrow 0.57}$

DRO pipeline generates an answer, this answer, along with the selected documents, is presented to a separate verifier model. We implement this verifier using non-trained LLMs, specifically evaluating models of varying scales such as Qwen-7B. The verifier is prompted to cite supporting sentences from the documents that justify the generated answer. If the verifier fails to find sufficient evidence or identifies a contradiction, the generator is triggered to produce a revised answer

As illustrated in Table 9, our preliminary experiments show that this post-verification mechanism leads to a consistent, though modest, improvement in answer faithfulness, reducing instances of factual hallucination. This demonstrates the potential of incorporating lightweight verification steps to enhance the robustness of RAG systems.

9 Conclusions and Future Work

We have presented DRO, a DRO method for the RAG task that (i) treats document permutations of the retriever as latent variables, and (ii) enables the co-training of a list-wise document selection model and an LLM generator through an EM approach. Specifically, DRO alternates between two steps: (1) directly estimating the distribution of document permutations from the selection model using an importance sampling strategy; and (2) maximizing the importance-weighted ELBO to jointly train the selection model and the LLM generator. Through theoretical analysis, we have proven that the learning objectives in DRO are analogous to policy-gradient RL, reinforcing the selection and generation models with an end-to-end training reward. Extensive experiments

conducted on five datasets have validated the superiority of our DRO. In DRO, we formalize retrieval and generation as a unified latent-variable optimization problem, offering a principled alternative to traditional RAG pipelines that decouple retriever and generator training. This latent optimization technique may also inspire advances in other retrieval-augmented applications, such as tool-calling agents and long-document reasoning.

Despite its strengths, DRO has several limitations. First, our current framework focuses on the retrieval and generation components within the RAG pipeline, without incorporating other potentially useful modules such as query decomposition or reformulation. Second, we use the LLM as the backbone for both selection and generation. This design limits DRO to tasks involving only textual inputs without rich multi-modal contexts such as images, tables, or audio. In future work, we plan to incorporate additional retrieval-enhancing components, such as query rewriting, decomposition, and document re-ranking, into the training loop. These enhancements could allow the system to better adapt to complex queries and dynamic information landscapes, leading to more flexible and generalizable RAG. Furthermore, we aim to extend DRO to multi-modal RAG settings, enabling the integration of image, audio, and other non-textual evidence into the retrieval and generation process.

References

[1] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 12248–12267.

[2] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of the 12th International Conference on Learning Representations (ICLR '24)*. OpenReview.net.

[3] John Bibby. 1974. Axiomatisations of the average and a further generalisation of monotonic sequences. *Glasgow Mathematical Journal* 15, 1 (1974), 63–65. DOI: https://doi.org/10.1017/S0017089500002135

[4] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. arXiv:2309.11495. Retrieved from https://arxiv.org/abs/2309.11495

[5] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of Wikipedia: Knowledge-powered conversational agents. arXiv:1811.01241. Retrieved from https://arxiv.org/abs/1811.01241

[6] Guanting Dong, Yutao Zhu, Chenghao Zhang, Zechen Wang, Zhicheng Dou, and Ji-Rong Wen. 2024. Understand what LLM needs: Dual preference alignment for retrieval-augmented generation. arXiv:2406.18676. Retrieved from https://arxiv.org/abs/2406.18676

[7] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 Herd of models. arXiv:2407.21783. Retrieved from https://arxiv.org/abs/2407.21783

[8] Víctor Elvira and Luca Martino. 2021. Advances in importance sampling. arXiv:2102.05407. Retrieved from https://arxiv.org/abs/2102.05407

[9] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[10] Feiteng Fang, Yuelin Bai, Shiwen Ni, Min Yang, Xiaojun Chen, and Ruifeng Xu. 2024. Enhancing noise robustness of retrieval-augmented language models with adaptive adversarial training. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

[11] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. arXiv:2312.10997. Retrieved from https://arxiv.org/abs/2312.10997

[12] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W. Bruce Croft, and Xueqi Cheng. 2020. A deep look into neural ranking models for information retrieval. *Information Processing & Management* 6 (2020), 102067.

[13] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the International Conference on Machine Learning*. PMLR.

[14] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics.

[15] Sebastian Hofstätter, Jiecao Chen, Karthik Raman, and Hamed Zamani. 2023. FiD-Light: Efficient and effective retrieval-augmented text generation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[16] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems* 43, 2 (2023), 1–55.

[17] 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* 251 (2023), 11912–11954.

[18] Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. arXiv:2310.06825. Retrieved from https://arxiv.org/abs/2310.06825

[19] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. arXiv:2401.04088. Retrieved from https://arxiv.org/abs/2401.04088

[20] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. 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 (EMNLP '20)*. Association for Computational Linguistics.

[21] Teun Kloek and Herman K. van Dijk. 1978. Bayesian estimates of equation system parameters: An application of integration by Monte Carlo. *Econometrica* 46, 1 (1978), 1–19.

[22] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[23] Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen Tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*.

[24] 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*.

[25] Xinze Li, Sen Mei, Zhenghao Liu, Yukun Yan, Shuo Wang, Shi Yu, Zheni Zeng, Hao Chen, Ge Yu, Zhiyuan Liu, et al. 2024. RAG-DDR: Optimizing retrieval-augmented generation using differentiable data rewards. arXiv:2410.13509. Retrieved from https://arxiv.org/abs/2410.13509

[26] Jiachen Lian, Xuanru Zhou, Zoe Ezzes, Jet Vonk, Brittany Morin, David Baquirin, Zachary Mille, Maria Luisa Gorno Tempini, and Gopala Krishna Anumanchipalli. 2024. SSDM: Scalable speech dysfluency modeling. arXiv:2408.16221. Retrieved from https://arxiv.org/abs/2408.16221

[27] Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, et al. 2023. RA-DIT: Retrieval-augmented dual instruction tuning. arXiv:2310.01352. Retrieved from https://arxiv.org/abs/2310.01352

[28] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics* 12 (2024), 157–173.

[29] Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. 2024. ChatQA: Surpassing GPT-4 on conversational QA and RAG. arXiv:2401.10225. Retrieved from https://arxiv.org/abs/2401.10225

[30] Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. arXiv:2305.02156. Retrieved from https://arxiv.org/abs/2305.02156

[31] Sewon Min, Kenton Lee, Ming-Wei Chang, Kristina Toutanova, and Hannaneh Hajishirzi. 2021. Joint passage ranking for diverse multi-answer retrieval. arXiv:2104.08445. Retrieved from https://arxiv.org/abs/2104.08445

[32] Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. arXiv:2004.10645. Retrieved from https://arxiv.org/abs/2004.10645

[33] Todd K. Moon. 1996. The expectation-maximization algorithm. *IEEE Signal Processing Magazine* 13, 6 (1996), 47–60.

[34] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches 2016 co-Located with the 30th Annual Conference on Neural Information Processing Systems (NIPS '16)*. Retrieved from https://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf

[35] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Proceedings of the Findings of the Association for Computational Linguistics (EMNLP '20)*. Association for Computational Linguistics, 708–718.

[36] 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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2523–2544.

[37] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. RankVicuna: Zero-shot listwise document reranking with open-source large language models. arXiv:2309.15088. Retrieved from https://arxiv.org/abs/2309.15088

[38] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. RankZephyr: Effective and robust zero-shot listwise reranking is a breeze! arXiv:2312.02724. Retrieved from https://arxiv.org/abs/2312.02724

[39] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*, 53728–53741.

[40] 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.

[41] Jinfeng Rao, Linqing Liu, Yi Tay, Wei Yang, Peng Shi, and Jimmy Lin. 2019. Bridging the gap between relevance matching and semantic matching for short text similarity modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

[42] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. DeepSpeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 3505–3506.

[43] Devendra Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. 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)*. Association for Computational Linguistics, 6648–6662.

[44] Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-Tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. arXiv:2204.07496. Retrieved from https://arxiv.org/abs/2204.07496

[45] Alireza Salemi and Hamed Zamani. 2024. Evaluating retrieval quality in retrieval-augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 2395–2400.

[46] Keshav Santhanam, O. Khattab, Jon Saad-Falcon, Christopher Potts, and Matei A. Zaharia. 2021. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 3715–3734.

[47] Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Proceedings of the Findings of the Association for Computational Linguistics (EMNLP '23)*.

[48] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-Tau Yih. 2023. Replug: Retrieval-augmented black-box language models. arXiv:2301.12652. Retrieved from https://arxiv.org/abs/2301.12652

[49] Zhengliang Shi, Weiwei Sun, Shen Gao, Pengjie Ren, Zhumin Chen, and Zhaochun Ren. 2024. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. arXiv:2406.14891. Retrieved from https://arxiv.org/abs/2406.14891

[50] Zhengliang Shi, Weiwei Sun, Shuo Zhang, Zhen Zhang, Pengjie Ren, and Zhaochun Ren. 2023. RADE: Reference-assisted dialogue evaluation for open-domain dialogue. arXiv:2309.08156. Retrieved from https://arxiv.org/abs/2309.08156

[51] Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. In *Proceedings of the 35th International Conference on Neural Information Processing Systems (NIPS '21)*, 25968–25981.

[52] Alessandro Sordoni, Eric Yuan, Marc-Alexandre Côté, Matheus Pereira, Adam Trischler, Ziang Xiao, Arian Hosseini, Friederike Niedtner, and Nicolas Le Roux. 2024. Joint prompt optimization of stacked LLMs using variational inference. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*, 58128–58151.

[53] Weihang Su, Yichen Tang, Qingyao Ai, Changyue Wang, Zhijing Wu, and Yiqun Liu. 2024. Mitigating entity-level hallucination in large language models. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*.

[54] Weiwei Sun, Pengjie Ren, and Zhaochun Ren. 2023. Generative knowledge selection for knowledge-grounded dialogues. In *Proceedings of the Findings of the Association for Computational Linguistics (EACL '23)*.

[55] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? Investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

[56] Zhiqing Sun and Yiming Yang. 2020. An EM approach to non-autoregressive conditional sequence generation. In *Proceedings of the International Conference on Machine Learning*. PMLR.

[57] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the Proceedings of the 13th International Conference on Neural Information Processing Systems (NIPS '99)*, 1057–1063.

[58] 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:2307.09288. Retrieved from https://arxiv.org/abs/2307.09288

[59] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics* 10 (2022), 539–554.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS '17)*, 6000–6010.

[61] Shuting Wang, Jiongnan Liu, Shiren Song, Jiehan Cheng, Yuqi Fu, Peidong Guo, Kun Fang, Yutao Zhu, and Zhicheng Dou. 2024. DomainRAG: A Chinese benchmark for evaluating domain-specific retrieval-augmented generation. arXiv:2406.05654. Retrieved from https://arxiv.org/abs/2406.05654

[62] Shuting Wang, Xin Yu, Mang Wang, Weipeng Chen, Yutao Zhu, and Zhicheng Dou. 2024. RichRAG: Crafting rich responses for multi-faceted queries in retrieval-augmented generation. arXiv:2406.12566. Retrieved from https://arxiv.org/abs/2406.12566

[63] Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, and Graham Neubig. 2023. Learning to filter context for retrieval-augmented generation. arXiv:2311.08377. Retrieved from https://arxiv.org/abs/2311.08377

[64] Zhepei Wei, Wei-Lin Chen, and Yu Meng. 2024. InstructRAG: Instructing retrieval-augmented generation via self-synthesized rationales. arXiv:2406.13629. Retrieved from https://arxiv.org/abs/2406.13629

[65] Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-Pack: Packaged resources to advance general Chinese embedding. arXiv:2309.07597. Retrieved from https://arxiv.org/abs/2309.07597

[66] Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. RECOMP: Improving retrieval-augmented LMs with compression and selective augmentation. arXiv:2310.04408. Retrieved from https://arxiv.org/abs/2310.04408

[67] Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. 2024. Search-in-the-chain: Towards accurate, credible and traceable large language models for knowledge-intensive tasks. In *Proceedings of the ACM Web Conference 2024 (WWW '24)*, 1362–1373.

[68] Diji Yang, Jinmeng Rao, Kezhen Chen, Xiaoyuan Guo, Yawen Zhang, Jie Yang, and Yi Zhang. 2024. IM-RAG: Multi-round retrieval-augmented generation through learning inner monologues. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[69] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[70] Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. arXiv:2310.01558. Retrieved from https://arxiv.org/abs/2310.01558

[71] Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. 2024. RankRAG: Unifying context ranking with retrieval-augmented generation in LLMs. arXiv:2407.02485. Retrieved from https://arxiv.org/abs/2407.02485

[72] Hamed Zamani and Michael Bendersky. 2024. Stochastic RAG: End-to-end retrieval-augmented generation through expected utility maximization. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[73] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren's song in the AI ocean: A survey on hallucination in large language models. arXiv:2309.01219. Retrieved from https://arxiv.org/abs/2309.01219

[74] Hanyang Zhao, Wenpin Tang, and David Yao. 2024. Policy optimization for continuous reinforcement learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*, 13637–13663.

[75] Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2022. Learning on large-scale text-attributed graphs via variational inference. arXiv:2210.14709. Retrieved from https://arxiv.org/abs/2210.14709

[76] Junda Zhu, Lingyong Yan, Haibo Shi, Dawei Yin, and Lei Sha. 2024. ATM: Adversarial tuning multi-agent system makes a robust retrieval-augmented generator. arXiv:2405.18111. Retrieved from https://arxiv.org/abs/2405.18111

Received 31 May 2025; revised 2 December 2025; accepted 10 January 2026