# RAG-Studio: Towards In-Domain Adaptation of Retrieval Augmented Generation Through Self-Alignment

**Kelong Mao**[1], **Zheng Liu**[2*], **Hongjin Qian**[2], **Fengran Mo**[3],
**Chenlong Deng**[1], **Zhicheng Dou**[1*]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Beijing Academy of Artificial Intelligence
[3]Université de Montréal, Québec, Canada
{mkl,dou}@ruc.edu.cn, zhengliu1026@gmail.com

## Abstract

Retrieval-Augmented Generation (RAG) has been widely received as an effective paradigm to enhance the quality of text generation by integrating large language models (LLMs) with external knowledge. However, the off-the-shelf RAG systems, which rely on LLMs and retrievers trained from general-purpose datasets, often fall short in handling specialized domains. To address the above challenge, we introduce RAG-Studio, a novel self-aligned training framework which autonomously adapts general RAG systems to specific domains. In a nutshell, RAG-Studio accepts a specialized domain corpus, where it identifies useful domain knowledge and synthesizes training data on top of it. Then, it leverages the synthetic data for the joint fine-tuning of the RAG system, such that the retriever can bring in more precise information, and the LLM can become more proficient at utilizing the retrieved information. We perform extensive experiments across diversified domain-specific QA datasets, spanning the Biomedical, Finance, Law, Computation, and Wiki, whose results validate the substantial improvements over the generally trained RAG.

## 1 Introduction

Retrieval-Augmented Generation (RAG) is widely received as a popular paradigm for the application of large language models (LLMs). By integrating a standing-by retrieval component, LLMs can leverage authoritative and up-to-date information, thus improving the truthfulness and credibility of their generated outputs (Asai et al., 2023; Siriwardhana et al., 2023). Additionally, frameworks like LangChain[1] and LlamaIndex[2] have significantly simplified the development and deployment of RAG systems by providing well-packaged components and automatically selecting the optimal combinations of generators and retrievers.

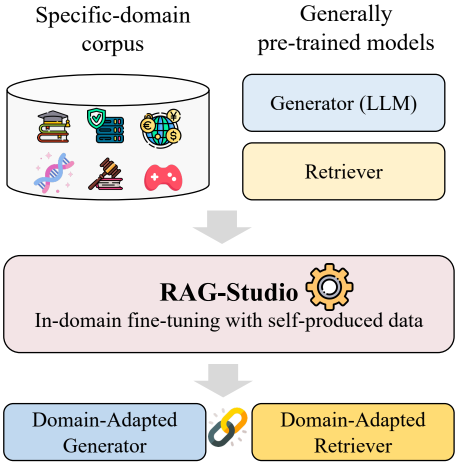

Figure 1: High-level illustration of RAG-Studio.

However, in specialized domains such as medical, finance, law, or private enterprise databases, existing off-the-shelf RAG systems, which utilize general LLMs and retrievers trained on generic data, often underperform. This underperformance arises because both the LLM and the retriever lack the updated, domain-specific knowledge and deep understanding required to handle in-domain data effectively. Furthermore, the retriever and LLM do not learn to interact optimally as a unified RAG system tailored for specialized contexts. As a result, achieving good performance often requires very intricate prompting, which can be cumbersome in practical applications. In contrast, fine-tuning has been shown to be an effective method for unlocking the potential of RAG systems in specific in-domain tasks (Zhang et al., 2024). However, in many specific domains or tasks, collecting and scaling the necessary training data for RAG (e.g., in-domain question-answer pairs and their relevant or irrelevant passages) can be challenging.

To address this issue, we propose RAG-Studio, a self-aligned training framework for the domain

[*]Corresponding author.
[1]LangChain: https://www.langchain.com
[2]LlamaIndex: https://www.llamaindex.ai

*Findings of the Association for Computational Linguistics: EACL 2024*, pages 725–735
November 12-16, 2024 ©2024 Association for Computational Linguistics

adaptation of RAG systems based entirely on synthetic data. All training ingredients in RAG-Studio are autonomously synthesized through self-alignment (Sun et al., 2023) without the need for external datasets or models, significantly enhancing the RAG system's ability to adapt to specialized domains with minimal manual intervention.

Our method starts with a target domain corpus, a general retriever, and a general LLM generator. Initially, the LLM is prompted to produce a collection of synthetic question-answer pairs based on sampled passages from the corpus. Subsequently, the retriever is employed to gather additional contextual passages relevant to these questions. The LLM then engages in a self-curation process, evaluating, refining, and filtering its own outputs to generate a set of high-quality, informative preference training samples. For the retriever, training signals are derived from the LLM's self-feedback, which is elicited through a Chain-of-Thought (Wei et al., 2020) prompting designed to assess the helpfulness of the retrieved contexts (Yu et al., 2023b; Zhang et al., 2024). Through our fine-tuning process, both the generator and the retriever are optimized to function as a cohesive, domain-specific RAG system. The LLM is adapted to integrate new domain knowledge effectively and to withstand noisy contexts provided by the retriever. Concurrently, the retriever is trained to align more closely with the LLM's preferences, ensuring the provision of helpful information and reducing the likelihood of misleading the LLM.

We conduct extensive experiments across five in-domain question answering datasets, encompassing the fields of biomedical, finance, law, computing, and general Wikipedia. The results demonstrate that RAG-Studio consistently outperforms baselines on these benchmarks. Notably, it outperforms the use of human-annotated data for fine-tuning by 5.1% and 3.2% in terms of automated metrics and GPT-4 evaluation scores, respectively.

In summary, our main contributions are:

(1) We present RAG-Studio, a self-aligned training framework that enables efficient in-domain adaptation of RAG models through synthetic data, eliminating the need for external data or models.

(2) We propose a series of self-data curation approaches that autonomously create high-quality, contrastive training samples for the fine-tuning of both the generator and the retriever.

(3) Our extensive experiments across five domains demonstrate the viability and even superiority of synthetic data to build strong in-domain RAG systems.

## 2 Related Work

**RAG Fine-tuning** Large language models have exhibited limitations in effectively utilizing context (Liu et al., 2023a; Yoran et al., 2023). A few studies have investigated enhancing the performance of retrieval-augmented generation systems through fine-tuning. From the retriever's perspective, a key idea is to optimize the retriever using the generation signals from LLMs, thereby aligning the retriever with the LLM's preferences (Shi et al., 2023; Lin et al., 2023; Zhang et al., 2023). On the LLM side, research has focused on improving context utilization by training LLMs with noisy contexts (Yoran et al., 2023; Zhang et al., 2024), employing prefix tuning to compress or refine context (Cheng et al., 2024; Zhu et al., 2024), or training additional modules to enhance context refinement (Xu et al., 2023). Nevertheless, these works are mainly focusing on the open domain that has much training data.

The work most closely related to ours is RAFT (Zhang et al., 2024), an in-domain RAG fine-tuning method. However, RAFT primarily focuses on fine-tuning the generator using raw LLM-generated pseudo questions and answers within noisy contexts. In contrast, our approach involves unified RAG fine-tuning that includes both the generator and the retriever. We implement a holistic self-alignment process, utilizing self-generated raw data and self-curation to produce higher-quality contrastive training data, thereby offering a more comprehensive training framework.

**Self-Alignment** Self-alignment refers to using a model to improve its own performance and align its responses with desired behaviors (Rennie et al., 2020; Sun et al., 2023; Tao et al., 2024). This emerging approach shows promise in addressing data scarcity issues and achieving higher-level alignment in large model training. Current research explores enhancing various aspects of LLMs with self-alignment, such as coding (Jiang et al., 2023b), mathematics (Yu et al., 2023a), and general conversational helpfulness (Ding et al., 2023; Ulmer et al., 2024). Studies on self-evolved data curation encompass instruction tuning (Li et al., 2023), reward modeling (Yuan et al., 2024), self-generation of explanations (Stammer et al., 2023), critiques (Gou et al., 2023), and so on. Our work explores self-alignment to support fine-tuning a general retriever and LLM into a unified RAG system, addressing data scarcity issues in domain-specific adaptations of RAG.

# 3 Methodology

## 3.1 Preliminaries

In this work, we focus on the question answering (QA) scenario of RAG. Typically, given a question $q$, the RAG system first retrieves relevant information from an external knowledge database $D$ and then generates the answer $a$ by incorporating the retrieved information:

$$a = G(q, R(q, D)), \tag{1}$$

where $G$ is a generator and $R$ is a retriever. We utilize LLM as the generator $G$, and a dense retriever for $R$ due to their superior performance in handling such tasks.

However, in domain-specific applications, existing off-the-shelf RAG systems, which combine a general LLM and a general retriever trained on generic data, often underperform. In this work, we propose RAG-Studio to effectively and efficiently fine-tune the general RAG models ($G$ and $R$) into in-domain RAG models ($G^*$ and $R^*$) given any specific domain corpus ($D$).

## 3.2 Synthetic Data Curation with Self-Alignment

RAG-Studio eliminates the need for human-labeled data by autonomously generating high-quality synthetic data. In this section, we introduce our approach to synthetic data curation with self-alignment for fine-tuning RAG models.

### 3.2.1 Raw Data Generation

Specifically, the initial input to RAG-Studio consists of a domain corpus $D$ containing multiple passages ($D = \{p_1, ..., p_n\}$), a general LLM-based generator $G$, and a retriever $R$. First, we select a ground passage $p$ from $D$ and prompt the generator $G$ to generate a question and the corresponding answer based on $p$:

$$q, a = G(\text{Prompt}_{\text{gqa}}(p)), \tag{2}$$

where $q$ and $a$ are the generated question and answer, respectively. $\text{Prompt}_{\text{gqa}}(.)$ is a question-answer generation prompt function as shown in Figure **??**. Then, we retrieve the top-$K$ passages from the corpus ($p$ has been filtered out) for the generated question to provide additional context:

$$p_1^c, ..., p_K^c = R(q, D). \tag{3}$$

This process results in a collection of synthetic raw training samples, where each sample $s$ is represented as $s = (q, a, p, \{p_1^c, ..., p_K^c\})$. These samples are then refined and used to fine-tune the generator and the retriever.

### 3.2.2 Data Curation for Generator

The general LLMs lack domain-specific knowledge, and existing studies (Liu et al., 2023a; Yoran et al., 2023) have shown that irrelevant context can considerably confuse them and weaken their performance. Through fine-tuning, we aim to not only incorporate new domain knowledge but also improve the context utilization capability for the LLM generator.

Chain-of-Thought (CoT) prompting, where the model generates its final answer along with the reasoning process (i.e., a rationale), has proven effective in RAG (Yu et al., 2023b; Zhang et al., 2024). We adopt this CoT-based generation and design a specific structure for the rationale: the model is instructed to evaluate and classify each context passage into one of three categories—*helpful*, *irrelevant*, or *misleading*—before generating the final answer based on this context evaluation.

Specifically, given a raw training sample $s$, we first employ the generator $G$ to generate the rationale and the answer under the context $C$:

$$e', a' = G(\text{Prompt}_{\text{rag}}(q, C)), \tag{4}$$

where $e'$ and $a'$ are the generated rationale and the answer, respectively. $\text{Prompt}_{\text{rag}}(.)$ is the CoT-based RAG generation prompt function shown in Figure **??**. We consider two types of context: $C_1 = \{p\} \cup \{p_1^c, ..., p_K^c\}$ and $C_2 = \{p_1^c, ..., p_K^c\}$, where $C_1$ includes the ground-truth passage $p$ while $C_2$ does not. Then, we prompt $G$ to evaluate the correctness of $a'$ by additionally providing it with the question $q$, the ground truth passage $p$, and the gold answer $a$. Correctly answered samples are filtered out at this stage, as we believe they provide limited benefit for further fine-tuning.

For the samples that are not correctly answered, we collect their ground-truth rationales $e$ by prompting $G$ with the ground-truth answer $a$:

$$e = G(\text{Prompt}_{\text{rationale}}(q, C, a)). \tag{5}$$

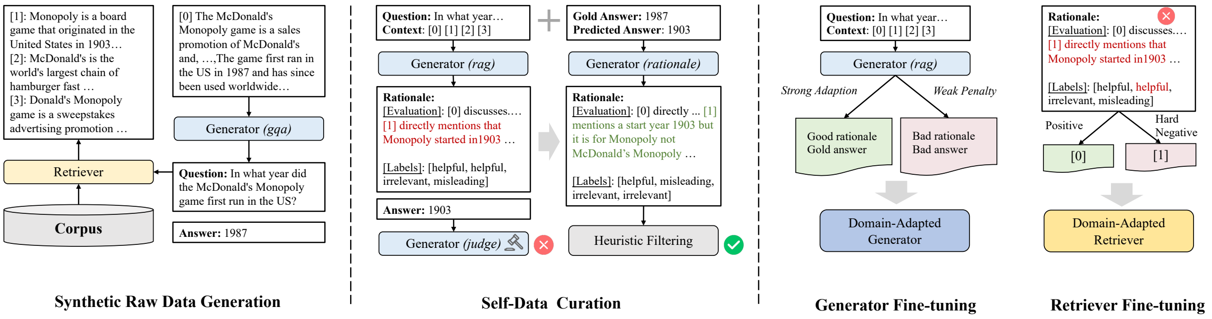
[1]: Monopoly is a board game that originated in the United States in 1903…
[2]: McDonald's is the world's largest chain of hamburger fast …
[3]: Donald's Monopoly game is a sweepstakes advertising promotion …
Retriever
Corpus
[0] The McDonald's Monopoly game is a sales promotion of McDonald's and, …,The game first ran in the US in 1987 and has since been used worldwide…
Generator (gqa)
Question: In what year did the McDonald's Monopoly game first run in the US?
Answer: 1987
Synthetic Raw Data Generation
Question: In what year…
Context: [0] [1] [2] [3]
Generator (rag)
Rationale:
[Evaluation]: [0] discusses.… [1] directly mentions that Monopoly started in1903 …
[Labels]: [helpful, helpful, irrelevant, misleading]
Answer: 1903
Generator (judge)
Gold Answer: 1987
Predicted Answer: 1903
Generator (rationale)
Rationale:
[Evaluation]: [0] directly ... [1] mentions a start year 1903 but it is for Monopoly not McDonald's Monopoly …
[Labels]: [helpful, misleading, irrelevant, irrelevant]
Heuristic Filtering
Self-Data Curation
Question: In what year…
Context: [0] [1] [2] [3]
Generator (rag)
Strong Adaption
Weak Penalty
Good rationale
Gold answer
Bad rationale
Bad answer
Domain-Adapted Generator
Generator Fine-tuning
Rationale:
[Evaluation]: [0] discusses.… [1] directly mentions that Monopoly started in1903 …
[Labels]: [helpful, helpful, irrelevant, misleading]
Positive
Hard Negative
[0]
[1]
Domain-Adapted Retriever
Retriever Fine-tuning

Figure 2: Overview of RAG-Studio. Synthetic training data are generated and curated autonomously, without external data or models. The generator and retriever are fine-tuned contrastively to achieve RAG domain adaptation.

$\text{Prompt}_{\text{rationale}}(.)$ is shown in Figure **??**. Despite providing the ground-truth answer in the prompt, we observe that the generated rationales are still not guaranteed to be correct. To further improve data quality, we filter out a few "contradictory" samples using the following heuristic rules:

- For $C = C_1$, if the ground-truth passage $p$ is not labeled as *helpful* in $e$, the sample is discarded.
- For $C = C_2$, if more than two passages are labeled as *helpful* in $e$, the sample is discarded.

After these steps, we gather the final preference data for fine-tuning the generator. Each sample can be represented as a triple $(q + C, e + a, e' + a')$, where $q + C$ is the input, $e + a$ is the preferred response, and $e' + a'$ is the non-preferred response.

### 3.2.3 Data Curation for Retriever

Existing general retrievers lack domain-specific knowledge for matching. Additionally, since the retriever and the generator are trained separately, the information retrieved may not effectively support the generator and could even mislead it. To address this, we fine-tune the retriever to better align itself with the preference of the generator for better in-domain retrieval-augmented generation.

Specifically, we mine training signals for the retriever from the generator's own prediction $e'$ and $a'$. Specifically, when $a'$ is correct, passages labeled as *helpful* are treated as positive samples, while those labeled as *misleading* serve as hard negatives for the question $x$. When $a'$ is incorrect, we inversely treat passages labeled as *helpful* as hard negatives. The original grounding passage $p$ always remains a positive sample.

## 3.3 Fine-tuning RAG Models

For the generator, we use the ORPO training approach, which combines supervised fine-tuning (SFT) and preference alignment. This approach efficiently discourages the model from learning the non-preferred response $e' + a'$ during the SFT to learn the preferred response $e + a$ and its specific output style. The training loss function is:

$$\mathcal{L}_{\mathrm{G}} = -\lambda \log \sigma(\log \frac{P_\theta(y^w|x)(1 - P_\theta(y^l|x))}{P_\theta(y^l|x)(1 - P_\theta(y^w|x))}) - \frac{1}{m}\sum_{t=1}^{m} \log P_\theta(y_t^w|x, y_{<t}^w), \quad (6)$$

where $x$, $y^w$, and $y^l$ are the input $(q + C)$, the preferred response $(e + a)$, and the non-preferred response $(e' + a')$, respectively. $m$ is the number of tokens of $y^w$. $\lambda$ is a hyper-parameter set to 0.1.

For the retriever, after collecting the positive passages $P^+$ and the hard negative passages $P^-$ for the question $x$ from the rationale of the generator, we employ the contrastive ranking loss function to fine-tune the dense retriever to align with the generator's preference:

$$\mathcal{L}_{\mathrm{C}} = -\log \frac{\phi(x, p^+)}{\phi(x, p^+) + \sum_{p^- \in P^-} \phi(x, p^-)}, \quad (7)$$

where $p^+ \in P^+$, $\phi(x, p) = \exp((E(p) \cdot E(p))/\tau)$, $E(\cdot)$ is the text encoder of the retriever, and $\tau$ is a hyper-parameter temperature set to 0.01.

# 4 Experiments

## 4.1 Setup

**Datasets and evaluation metrics** We evaluate our method primarily on four domain-specific QA datasets (Xu et al., 2020): **Biomedical**, **Finance**, **Law**, and **Computing**. These datasets are derived from the MS MARCO dataset (Nguyen et al., 2016) using topic modeling and filtering techniques. The MS MARCO dataset provides the top 10 retrieved passages from Bing and a human-generated answer

| Statistic | Biomedical | Finance | Law | Computing | TriviaQA (wiki) |
|---|---|---|---|---|---|
| #Questions (train) | 26,877 | 8,245 | 3,770 | 3,668 | 61,888 |
| #Questions (test) | 4,743 | 1,455 | 666 | 648 | 7,993 |
| #Passages | 287K | 89K | 42K | 41K | 2M |

Table 1: Statistics of the five datasets.

for each question. To construct the domain-specific retrieval corpora, we aggregate all the context passages corresponding to the questions in each domain. Additionally, we test our method on an open-domain QA dataset, **TriviaQA** (Joshi et al., 2017) (Wiki subset), to assess the generalizability of our approach. We form the retrieval corpora of TriviaQA by chunking all the documents into at most 200-word passages. Table 1 provides the statistics of these five datasets.

For evaluation, we use the official metrics: ROUGE-L for the four domain-specific QA datasets, and EM for TriviaQA. Considering the limitations of these heuristic metrics in capturing the nuances of answer quality, we also employ GPT-4 to evaluate answer accuracy (Liu et al., 2023b). GPT-4 is provided with the question, the reference answer, and the predicted answer to assess the quality of the predictions more robustly.

**Baselines** We compare RAG-Studio against the following baselines: (1) **GPT-3.5**: Directly prompt GPT-3.5 to perform question answering. (2) **Prompt**: Directly prompt the generator for question answering. (3) **DSF**: Fine-tune the generator on the domain-specific training data provided by each dataset. (4) **RAFT** (Zhang et al., 2024): Fine-tune the generator on synthetic training data within noisy contexts, which is our main competitor. For **Prompt**, **DSF**, and **GPT-3.5**, we consider both their settings with and without retrieval (i.e., RAG).

**Implementations** For the basic RAG system, we utilize Llama-3-8B-Instruct[3] as the generator and BGE (Xiao et al., 2023) as the retriever due to their superiority. We generate 3,000 synthetic training samples for each dataset using our self-alignment strategy. The impact of data size is examined in Section 4.6. The number of additional context passages, $K$, is set to 3. The generator is fine-tuned with LoRA (Hu et al., 2022) for 3 epochs, using a learning rate of 1e-5 and a total batch size of 64. The retriever is fine-tuned for 1 epoch with a learning rate of 1e-6 and a batch size of 64. Code is released at https://github.com/kyriemao/rag_studio.

For our main competitor, RAFT, we adhere to their original pipeline. Specifically, we use the synthetic raw samples and follow their empirical results to set 60% of the training samples to include the gold passage, while the remaining 40% contain only distractor passages. To ensure a fair comparison, we apply the same CoT prompting.

## 4.2 Main Results

The experimental results are shown in Table 2. We have the following findings:

(1) RAG-Studio outperforms all baselines across the five datasets, except for EM on TriviaQA, where it ranks second to DSF. The average relative improvement in ROUGE-L and accuracy scores are 4.94% and 3.73%, respectively, over the second-best results. By relying solely on self-curated synthetic data, RAG-Studio matches or exceeds the performance of DSF, which uses labeled data. This highlights the promising potential of synthetic data in fine-tuning RAG models and demonstrates the effectiveness of our proposed method.

(2) The performance improvements of RAG-Studio on the four domain-specific datasets are more significant than on the open-domain TriviaQA. On TriviaQA, the performance of different methods is quite close, with less than a 1% absolute difference in EM and accuracy. We hypothesize that this is because models like Llama3 and GPT-3.5 have already learned a large amount of open Wikipedia knowledge, which TriviaQA is based on, resulting in marginal improvements. This also indicates that general LLMs perform well on open-domain tasks but benefit more from fine-tuning in specific domains.

(3) Fine-tuning without retrieval may not improve performance. Comparing Prompt and DSF on four domain-specific datasets, we see a 2.6 average improvement in ROUGE-L but a 5.0 average decrease in GPT-4 evaluated accuracy score. On TriviaQA, both EM and GPT-4 scores drop. Since

[3] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

| Method | FT | Synthetic Data | Biomedical | | Finance | | Law | | Computing | | TriviaQA (wiki) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | R-L | Acc | R-L | Acc | R-L | Acc | R-L | Acc | EM | Acc |
| | | | | | | *without Retrieval* | | | | | | |
| GPT-3.5 | ✗ | ✗ | 19.5 | 43.1 | 16.9 | 34.2 | 20.5 | 51.4 | 19.6 | 42.9 | 62.9 | 60.0 |
| Prompt | ✗ | ✗ | 22.2 | 45.4 | 18.4 | 35.7 | 19.7 | 50.1 | 21.6 | 46.3 | 64.8 | 60.5 |
| DSF | ✔ | ✗ | 23.3 | 33.7 | 23.2 | 31.8 | 23.1 | 35.9 | 22.1 | 34.4 | 59.7 | 56.7 |
| | | | | | | *with Retrieval* | | | | | | |
| GPT-3.5 | ✗ | ✗ | 31.2 | 55.0 | 36.5 | 52.0 | 35.0 | 57.5 | 31.8 | 54.6 | 69.5 | 69.5 |
| Prompt | ✗ | ✗ | 31.9 | 54.7 | 37.7 | 52.2 | 34.1 | 56.2 | 34.9 | 56.9 | 69.3 | 69.2 |
| DSF | ✔ | ✗ | 34.8 | 57.5 | 40.0 | 55.4 | 37.4 | 60.4 | 44.2 | 57.9 | **70.2** | **69.9** |
| RAFT | ✔ | ✔ | 34.3 | 55.9 | 36.5 | 51.7 | 36.7 | 59.8 | 45.4 | 59.2 | 69.4 | 69.6 |
| RAG-Studio | ✔ | ✔ | **37.5** | **59.2** | **41.8** | **57.6** | **40.0** | **63.5** | **47.1** | **60.3** | 70.0 | **69.9** |

Table 2: Overall results on five datasets. FT indicates fine-tuning. R-L and Acc represent ROUGE-L and the accuracy evaluated by GPT-4, respectively. The best results are in bold, and the second-best results are underlined.

| Method | $\overline{\Delta}$ | Bio. | Fin. | Law | Comp. | TQA |
|---|---|---|---|---|---|---|
| Prompt | -4.3 | 54.7 | 52.2 | 56.2 | 56.9 | 69.2 |
| DSF | -1.9 | 57.5 | 55.4 | 60.4 | 57.9 | 69.9 |
| SFT | -1.2 | 58.5 | 56.7 | 62.2 | 57.9 | 69.3 |
| w/o G-FT | -3.9 | 54.9 | 53.4 | 57.4 | 56.5 | 68.8 |
| w/o R-FT | -0.8 | 58.7 | 57.2 | 62.4 | 58.8 | 69.6 |
| RAG-Studio | 0 | 59.2 | 57.6 | 63.5 | 60.3 | 69.9 |

Table 3: Results of ablation study. The metric is Acc. $\overline{\Delta}$ indicates the average difference with RAG-Studio. The dataset names are abbreviated.

GPT-4 evaluation is more reliable, this suggests that fine-tuning without retrieval reduces model performance. This likely occurs because fine-tuning on specific domains without access to relevant external information limits the model's ability to generalize and accurately respond to queries, highlighting the importance of retrieval in enhancing model performance for in-domain applications.

### 4.3 Ablation Study

In RAG-Studio, we curate contrastive training data with model self-alignment for both generator and retriever fine-tuning. We investigate different fine-tuning strategies in RAG-Studio to evaluate their impact on the final RAG performance. Specifically, we build the following three ablations: (1) *w/o G-FT*: We fine-tune only the retriever, leaving the generator unchanged. (2) *w/o R-FT*: We fine-tune only the generator, leaving the retriever unchanged. (3) *SFT*: For the original training sample $(x+C, e+y, e'+y')$, we consider only the preferred output $e+y$ and perform supervised fine-tuning, replacing ORPO. In addition, we also experimented with using DPO (Rafailov et al., 2023) for fine-tuning. However, we found that the model could not learn a stable output style through DPO, making it difficult to parse the outputs. Therefore, we omit the DPO results from this study.

The results are shown in Table 3. We observe that the complete RAG-Studio outperforms all the ablations, demonstrating that the RAG performance benefits from our designed contrastive generator and retriever fine-tunings. Specifically, our generator fine-tuning yields greater improvements compared to retriever tuning (3.9 vs 0.8). By comparing Prompt and w/o G-FT, we find that only fine-tuning the retriever may not robustly gain improvements, as seen in the Law and Computing datasets (56.9 vs 56.5 and 69.2 vs 68.8). These results underscore the importance of both generator and retriever fine-tuning for optimal RAG performance.

### 4.4 Effects of Chain-of-Thought

The generator utilizes our tailored CoT to identify the helpfulness of context passages. This CoT also serves as a bridge for the retriever to align with the generator, facilitating the retrieval of more relevant passages while avoiding irrelevant or misleading ones. In this section, we investigate the effects of the CoT by removing it. Specifically, during generator training, we prompt the generation of both preferred and non-preferred answers without incorporating the CoT. For retriever training, we rely solely on the gold (positive) passages with in-batch negatives, omitting any hard negatives. The results, depicted in Figure 3, show that CoT fine-tuning in RAG-Studio significantly enhances model performance. Specifically, we observe an average improvement of 13.2% in ROUGE-L scores and 7.7% in accuracy. By jointly analyzing with Table 2, we find that the model struggles to sur-

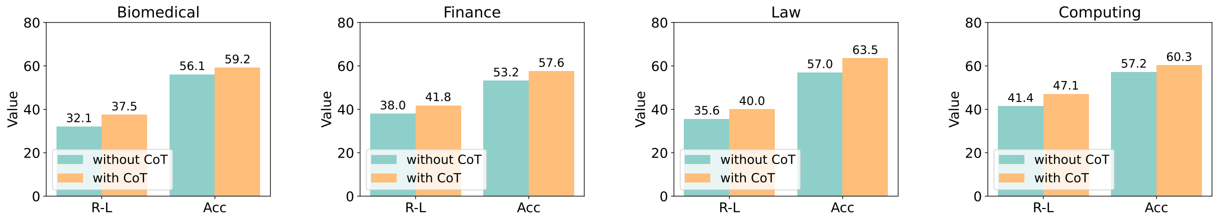

Figure 3: Comparison of RAG-Studio performance with and without our tailored chain-of-thought for fine-tuning.

| Retriever | Biomedical | | Finance | | Law | | Computing | | TriviaQA (wiki) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | R@5 | NDCG@5 | R@5 | NDCG@5 | R@5 | NDCG@5 | R@5 | NDCG@5 | R@5 | NDCG@5 |
| Before FT | 50.8 | 36.1 | 66.6 | 49.4 | 77.3 | 58.6 | 77.5 | 56.5 | 23.1 | 34.1 |
| After FT | 51.4 | 37.9 | 66.2 | 51.9 | 77.1 | 59.5 | 78.3 | 59.0 | 22.5 | 34.4 |

Table 4: Comparison of retrieval performance before and after in-domain fine-tuning.

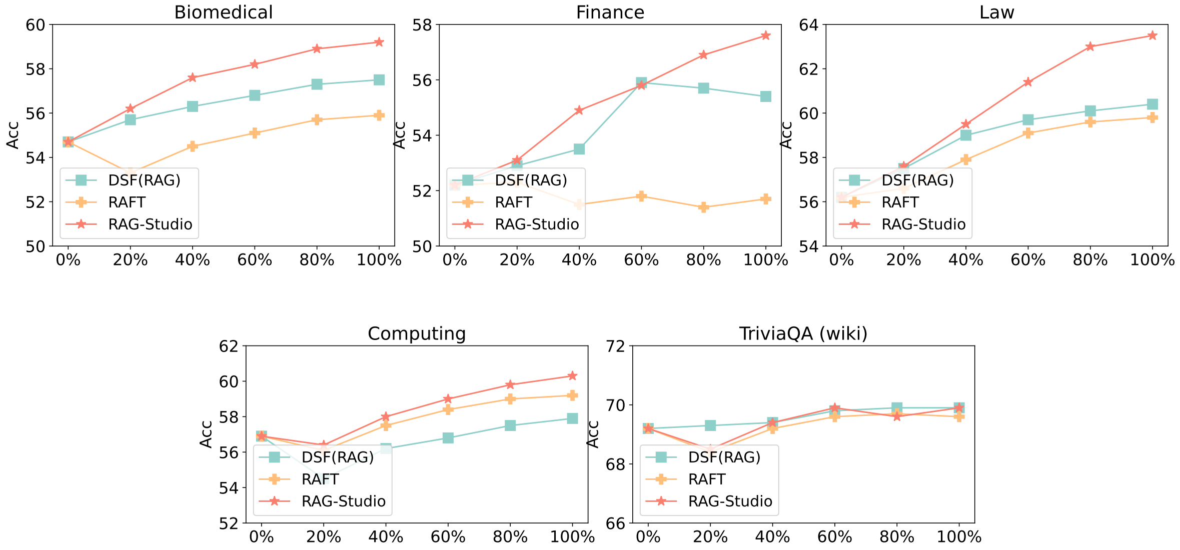

Figure 4: Performance of different methods using different percentages of training data.

pass the performance achieved with human-labeled data for fine-tuning (i.e., DSF) without CoT, yet it still outperforms models that are not fine-tuned (i.e., Prompt). This demonstrates the crucial role of CoT in enhancing the quality and informativeness of synthetic data for in-domain RAG fine-tuning.

## 4.5 Impact of In-Domain Fine-Tuning on Retriever Performance

In this section, we examine the effect of fine-tuning on the retriever's performance, using Recall@5 and NDCG@5 as evaluation metrics. For TriviaQA, which provides only document-level relevance labels, we generate passage-level relevance labels using GPT-4. GPT-4 was given the question, the gold answer, and the candidate passage, and prompted to judge whether the passage supports the answer. The four domain-specific datasets already include gold passage labels. Table 4 presents the retriever's performance before and after fine-tuning. From the results, we find that our retriever fine-tuning effectively improves top-ranking performance, as evidenced by the NDCG@5 scores, while its impact on recall performance is less pronounced. This suggests that retriever fine-tuning may primarily enhance the final RAG performance by refining the ranking order of the top retrieved passages.

## 4.6 Data Analysis

**Data quantity** The performance of different methods using various percentages of fine-tuning data is shown in Figure 4. A common trend observed across different methods is that performance improves with increasing training data but gradually saturates. Notably, in the Biomedical, Finance, and Law domains, RAG-Studio's accuracy signif-

| Method | Biomedical | | Finance | | Law | | Computing | | TriviaQA (wiki) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | R-L | Acc | R-L | Acc | R-L | Acc | R-L | Acc | EM | Acc |
| RAG-Studio | 37.5 | 59.2 | 41.8 | 57.6 | 40.0 | 63.5 | 47.1 | 60.3 | 70.0 | 69.9 |
| Using Raw Data | 35.0 | 56.5 | 37.6 | 53.1 | 38.0 | 60.8 | 44.9 | 58.8 | 69.2 | 69.5 |

Table 5: Performance comparisons of fine-tuning on data with and without filtering.

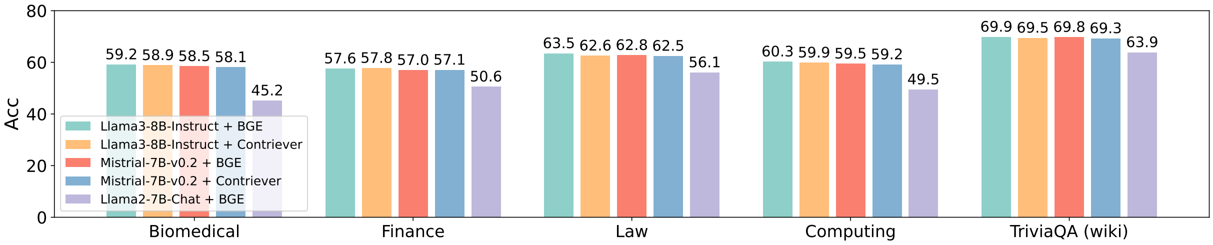

Figure 5: Performance of RAG-Studio variations based on different initial generator and retriever combinations.

icantly surpasses that of DSF (RAG) and RAFT, particularly with larger amounts of training data. In the Computing and open Wikipedia domains, the differences in performance and the rates of increase are less pronounced, though RAG-Studio generally maintains a lead. Interestingly, RAFT, which also uses synthetic fine-tuning data, often shows a performance decrease compared to no fine-tuning when fine-tuned with a small percentage of data. In contrast, RAG-Studio only exhibits this phenomenon in two domains: Computing and open Wikipedia, indicating better data efficiency for RAG-Studio. These results highlight the critical role of the quantity of training data in enhancing model performance, with RAG-Studio demonstrating the most substantial improvements.

**Data quality** In RAG-Studio, we use a series of data filtering strategies in our self-curation process to enhance the quality of the training data. To evaluate their impact, we remove these filters and use only the raw data for fine-tuning the RAG models, maintaining the same number of training samples as before. The results are shown in Table 5. We observe a significant performance decline, particularly on the four domain-specific datasets, when using raw training data without filtering. Specifically, the average relative drops in R-L and accuracy across these datasets are 6.6% and 4.8%, respectively. These findings underscore the importance of data quality in RAG fine-tuning and demonstrate that the data filtering in RAG-Studio can substantially improve model performance.

### 4.7 Experiments with Different Generator and Retriever Combinations

As discussed in Section 4.1, our primary experiments utilize the Llama-3-8B-Instruct and BGE models. To evaluate the generalizability and robustness of our framework, we conduct additional tests with other model combinations. Specifically, we use another strong LLM, Mistral-7B-Instruct-v0.2 (Jiang et al., 2023a), and a weaker one, Llama-2-7B-Chat, as the generators. For the retriever, we employ Contriever (Izacard et al., 2022), which is relatively weaker compared to BGE.

The results, presented in Figure 5, show no significant differences in performance when using strong LLMs, indicating that RAG-Studio maintains robustness across different combinations of high-quality generators and retrievers. However, there is a notable decline in performance when Llama-2-7B-Chat is used as the generator. This drop in performance can be attributed to two main factors: firstly, the synthetic data generated by the weaker LLM is of lower quality, adversely affecting the fine-tuning process. Secondly, the intrinsic performance of the weaker LLM in RAG tasks is inferior compared to stronger LLMs.

## 5 Conclusion

In this paper, we introduced RAG-Studio, a self-aligned training framework for domain adaptation of RAG systems entirely based on synthetic data. Our method eliminates the need for external data or models, enabling efficient fine-tuning of both retriever and generator components. Through extensive experiments across various specialized domains, RAG-Studio consistently demonstrated superior performance, highlighting its effectiveness and practicality. We believe our work will advance the development and deployment of domain-specific RAG systems, making them more accessible and robust for specialized applications.

## Limitations

While RAG-Studio shows promising results as a strong autonomous framework for in-domain RAG adaption, we acknowledge the following limitations of our work:

(1) The current work focuses solely on single-hop question answering scenarios. More complex multi-hop or multi-turn question answering scenarios, which require iterative retrieval and reasoning steps, are not considered. Extending RAG-Studio to handle such intricate scenarios remains an open challenge.

(2) For simplicity, the current method performs only one round of optimization using the self-generated and curated data. However, multi-round self-alignment approaches (Yuan et al., 2024; Li et al., 2023) have shown promise in further enhancing performance. We will continue to explore multi-round extensions of RAG-Studio to better handle updated knowledge and increasingly complex scenarios.

(3) The RAG-Studio framework currently optimizes only the core retriever and generator components. However, practical RAG systems can incorporate additional modules like rewriters, rerankers, etc. to further boost performance. Extending RAG-Studio to jointly optimize these supplementary modules alongside the retriever and generator remains an open challenge.

## Acknowledgement

This work was supported by Beijing Natural Science Foundation No. L233008, National Natural Science Foundation of China No. 62272467, the fund for building world-class universities (disciplines) of Renmin University of China, Public Computing Cloud, Renmin University of Chin, and the Outstanding Innovative Talents Cultivation Funded Programs 2024 of Renmin University of China. The work was partially done at the Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE.

## References

Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. 2023. Retrieval-based language models and applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pages 41–46.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. xrag: Extreme context compression for retrieval-augmented generation with one token.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 3029–3051. Association for Computational Linguistics.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. CRITIC: large language models can self-correct with tool-interactive critiquing. *CoRR*, abs/2305.11738.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *CoRR*, abs/2310.06825.

Shuyang Jiang, Yuhao Wang, and Yu Wang. 2023b. Selfevolve: A code evolution framework via large language models. *CoRR*, abs/2306.02907.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023. Self-alignment with instruction backtranslation. *CoRR*, abs/2308.06259.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Scott Yih. 2023. RA-DIT: retrieval-augmented dual instruction tuning. *CoRR*, abs/2310.01352.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023a. Lost in the middle: How language models use long contexts. *CoRR*, abs/2307.03172.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 2511–2522. Association for Computational Linguistics.

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 *CoCo@ NIPS*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Steven J. Rennie, Etienne Marcheret, Neil Mallinar, David Nahamoo, and Vaibhava Goel. 2020. Unsupervised adaptation of question answering systems via generative self-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1148–1157. Association for Computational Linguistics.

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. *CoRR*, abs/2301.12652.

Shamane Siriwardhana, Rivindu Weerasekera, Tharindu Kaluarachchi, Elliott Wen, Rajib Rana, and Suranga Nanayakkara. 2023. Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering. *Trans. Assoc. Comput. Linguistics*, 11:1–17.

Wolfgang Stammer, Felix Friedrich, David Steinmann, Hikaru Shindo, and Kristian Kersting. 2023. Learning by self-explaining. *CoRR*, abs/2309.08395.

Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David D. Cox, Yiming Yang, and Chuang Gan. 2023. Principle-driven self-alignment of language models from scratch with minimal human supervision. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. 2024. A survey on self-evolution of large language models. *CoRR*, abs/2404.14387.

Dennis Ulmer, Elman Mansimov, Kaixiang Lin, Justin Sun, Xibin Gao, and Yi Zhang. 2024. Bootstrapping llm-based task-oriented dialogue agents via self-talk. *CoRR*, abs/2401.05033.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2020. Chain of thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. 2023. C-pack: Packaged resources to advance general chinese embedding. *CoRR*, abs/2309.07597.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. RECOMP: improving retrieval-augmented lms with compression and selective augmentation. *CoRR*, abs/2310.04408.

Ying Xu, Xu Zhong, Antonio José Jimeno-Yepes, and Jey Han Lau. 2020. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–8. IEEE.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. *CoRR*, abs/2310.01558.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023a. Metamath: Bootstrap your own mathematical questions for large language models. *CoRR*, abs/2309.12284.

Wenhao Yu, Hongming Zhang, Xiaoman Pan, Kaixin Ma, Hongwei Wang, and Dong Yu. 2023b. Chain-of-note: Enhancing robustness in retrieval-augmented language models. *CoRR*, abs/2311.09210.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *CoRR*, abs/2401.10020.

Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. 2023. Retrieve anything to augment large language models. *CoRR*, abs/2310.07554.

Tianjun Zhang, Shishir G. Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E. Gonzalez. 2024. RAFT: adapting language model to domain specific RAG. *CoRR*, abs/2403.10131.

Yutao Zhu, Zhaoheng Huang, Zhicheng Dou, and Ji-Rong Wen. 2024. One token can help! learning scalable and pluggable virtual tokens for retrieval-augmented large language models. *CoRR*, abs/2405.19670.