FlowRAG: Continual Learning for Dynamic Retriever in Retrieval-Augmented Generation

Senlei Zhang*
Huazhong University of Science and Technology
Wuhan, China
m202574111@hust.edu.cn

Tongjun Shi
Futu Holdings Limited
Shenzhen, China
tongjunshihhh@gmail.com

Dandan Song
Beijing Institute of Technology
Beijing, China
sdd@bit.edu.cn

Luan Zhang
Beijing Institute of Technology
Beijing, China
zl005888@163.com

Shuhao Zhang*†
Huazhong University of Science and Technology
Wuhan, China
shuhao_zhang@hust.edu.cn

Xiaofei Liao*
Huazhong University of Science and Technology
Wuhan, China
xfliao@hust.edu.cn

Hai Jin*
Huazhong University of Science and Technology
Wuhan, China
hjin@hust.edu.cn

Abstract

Retrieval-Augmented Generation (RAG) enhances *Large Language Models* (LLMs) by leveraging external knowledge, where retrieval accuracy directly affects generation quality. However, dense retrievers, commonly employed in RAG, suffer degraded performance in evolving corpora where new documents arrive continuously and distribution shifts accumulate over time. In such settings, continually updating retrievers is crucial, yet conventional retraining is computationally expensive and often impractical. To address this challenge, we propose FlowRAG, a lightweight and effective method for continual retriever adaptation in evolving corpora. FlowRAG augments the encoder with Layer-wise Prompt Embeddings and introduces a Cross-Layer Fusion mechanism to capture hierarchical semantic representations. In addition, a novel Generator-Guided Loss aligns retriever scores and intermediate representations with the LLM's generation likelihoods, encouraging retrieval decisions that are both semantically relevant and beneficial for generation. Experiments on datasets spanning four domains demonstrate that FlowRAG, which updates only about 0.64% of the total model parameters, consistently outperforms strong baselines in retrieval accuracy, generation quality, and robustness to forgetting in non-stationary settings. We release the code at https://github.com/CGCL-codes/FlowRAG.git.

CCS Concepts

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

Keywords

Retrieval-Augmented Generation, Continual Learning, Large Language Model

ACM Reference Format:
Senlei Zhang, Tongjun Shi, Dandan Song, Luan Zhang, Shuhao Zhang, Xiaofei Liao, and Hai Jin. 2026. FlowRAG: Continual Learning for Dynamic Retriever in Retrieval-Augmented Generation . In *Proceedings of the ACM Web Conference 2026 (WWW '26), April 13–17, 2026, Dubai, United Arab Emirates.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3774904.3792361

1 Introduction

Retrieval-Augmented Generation (RAG) has become a widely accepted approach to enhance the performance of *large language models* (LLMs) [37, 38], facilitating their ability to retrieve and integrate information from the web (such as Wikipedia [34]) or other external knowledge bases [5, 43]. By retrieving relevant documents, RAG bridges the gap between the static knowledge encoded in LLM parameters and the dynamic, continuously evolving real world. This capability allows LLMs to leverage up-to-date and domain-specific information without the need for extensive retraining [17, 25, 32]. As pointed out by previous studies [11, 16, 40], the effectiveness of retrieval augmentation strongly depends on the quality and relevance of the retrieved content, which is commonly achieved using dense retrievers [5].

Dense retrievers encode queries and documents into continuous semantic vectors, enabling similarity-based retrieval that captures conceptual relevance beyond exact lexical overlap [5, 9, 12]. While this approach is highly effective on static document collections, it

*Also with National Engineering Research Center for Big Data Technology and System, Service Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology.
†Shuhao Zhang is the corresponding author (shuhao_zhang@hust.edu.cn).

WWW '26, Dubai, United Arab Emirates.

ACM ISBN 979-8-4007-2307-0/2026/04
https://doi.org/10.1145/3774904.3792361

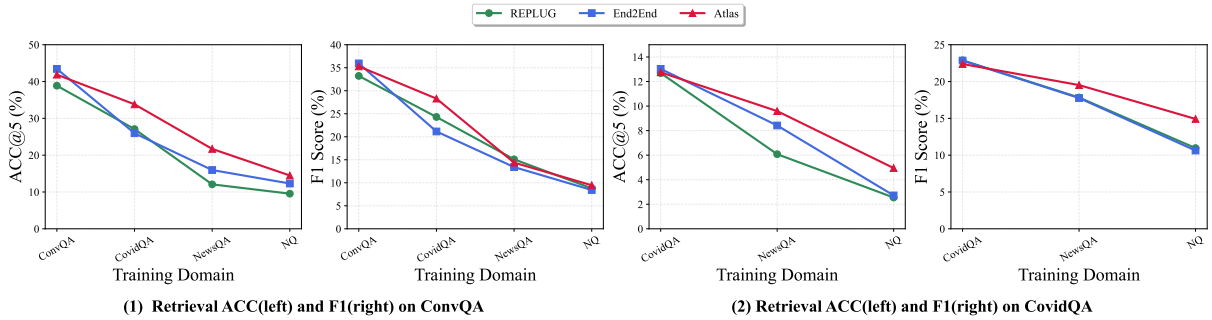

Figure 1: Catastrophic forgetting of representative LLM-guided retriever training methods (REPLUG [28], End2EndRAG [30], and Atlas [11]) under continual learning across multiple domains

faces significant challenges in real-world applications, where document corpora are continuously evolving due to technological advancements, societal changes, and the emergence of new topics. Distribution shifts across domains often cause significant performance degradation and catastrophic forgetting of previously learned knowledge [4], underscoring the need for continual learning for retrievers.

Existing approaches to continual learning for retriever, such as L^2R [2], mainly target dense retriever fine-tuning for first-stage retrieval. However, their optimization is tailored for standalone retrieval rather than retrieval for generation, which limits their applicability in RAG systems. To overcome this limitation, LLM-guided retriever training [11, 28, 30] has recently emerged as a promising paradigm, where generation likelihoods are leveraged to provide implicit supervision for retrieval. This paradigm offers two key advantages: (1) it leverages large language models to provide scalable supervision, thereby reducing the reliance on costly human annotations; and (2) it naturally aligns retrievers with downstream generation tasks, ensuring that retrieved information directly enhances generation quality. Despite these benefits, applying LLM-guided retriever training in a continual learning setting still encounters the challenge of catastrophic forgetting.

To better understand the constraints, we conducted a preliminary study on current representative LLM-guided retriever training methods, including REPLUG [28], End2EndRAG [30], and Atlas [11]. We sequentially trained the retriever across multiple domains. As shown in Figure 1, the performance of all models exhibited a clear degradation when shifting to new domains. For example, REPLUG achieved an initial F1 of 33.22% on ConvQA but dropped to 8.79% after training on subsequent datasets. Similar declines were observed for End2EndRAG and Atlas. These results demonstrate that existing methods are unable to retain previously acquired knowledge, leading to catastrophic forgetting in evolving corpora.

To address this challenge, we propose **FlowRAG**, a lightweight and effective method of continual learning for the retriever in evolving corpora. FlowRAG consists of three key components: (1) **Layer-wise Prompt Embeddings**: Building on insights from our preliminary results, we enhance the retriever's encoder by introducing layer-wise prompt embeddings. By incorporating these embeddings, the retriever can preserve previously learned knowledge and mitigate catastrophic forgetting. (2) **Cross-Layer Fusion**: To enhance the effectiveness of layer-wise prompt embeddings, we introduce a Cross-Layer Fusion mechanism. This fusion enables the model to blend multiple levels of representation, ensuring a more comprehensive understanding of the evolving context and improving retrieval accuracy. (3) **Generator-Guided Loss**: To align retrieval decisions with LLM's generation likelihoods, we propose a novel Generator-Guided Loss. This loss function optimizes the retriever's scores and intermediate representations to ensure they are not only semantically relevant but also conducive to high-quality generation. This process directly encourages retrieval choices that support more coherent and contextually appropriate generation outputs.

In summary, our contributions are as follows:

- We present FlowRAG, the first method to integrate layer-wise prompt embeddings into RAG retrievers for dynamic adaptivity and continual learning.
- We propose the innovative Cross-Layer Fusion mechanism and a Generator-Guided Loss to not only enhance the retriever's representational capacity but also align retrieval choices with the generation process, ensuring more contextually relevant and coherent outputs.
- The experimental results across four distinct domains demonstrate that FlowRAG consistently outperforms strong baselines in terms of retrieval accuracy, generation quality, and resilience to catastrophic forgetting. Further analysis reveals that FlowRAG maintains high efficiency, requiring only approximately 0.64% of the total model parameters to be updated in non-stationary settings. This efficiency underscores FlowRAG's potential as a robust solution for dynamic and continual learning.

2 Related Work

RAG Retriever Fine-tuning. Large language models have demonstrated limitations in effectively leveraging context [20, 39]. Several studies have explored improving the performance of *Retrieval-Augmented Generation* (RAG) systems through retriever fine-tuning. Recent efforts have aligned retriever training with generation objectives to enhance the overall RAG pipeline. Early methods such as Distill-RAG [10] and End2EndRAG [30] jointly optimize the retriever and generator by propagating the generation loss through cross-attention layers, thereby enabling retrieval decisions that directly benefit downstream generation. However, such joint training can be computationally expensive and sensitive to corpus shifts.

To mitigate computational overhead, recent methods decouple retriever optimization by leveraging the language model as a supervisory signal. REPLUG [28] proposes *LM-Supervised Retrieval* (LSR), which aligns retriever scores to the language model's preference distribution using a KL divergence loss. RA-DIT [19] adopts the same LSR objective to update the retriever without manual annotations. Similarly, Atlas [11] applies a Perplexity Distillation loss that trains the retriever to approximate the language model's preference distribution over documents by assessing their impact on perplexity, also using KL divergence.

Continual Learning. *Continual Learning* (CL) [3, 33] has been widely studied as a solution to catastrophic forgetting, the phenomenon where models overwrite previously acquired knowledge when exposed to new tasks or domains [18]. Existing CL approaches are commonly categorized into three families: replay-based methods, which preserve a subset of old samples and mix them with new data during training [24, 42]; regularization-based methods, which penalize updates to parameters deemed critical for past tasks [13, 41]; and parameter-isolation methods, which allocate task-specific subnetworks or modules to mitigate interference [1]. These approaches have achieved notable success in computer vision [22, 26, 27] and natural language processing [1, 7], establishing CL as a central paradigm for learning in non-stationary environments.

Despite advancements in CL, its application to RAG remains underexplored. Existing work focuses mainly on continuous dense retrievers [2], which are optimized for standalone retrieval rather than retrieval for generation. This narrow focus limits the flexibility needed for independent adaptation. In practical scenarios, document corpora evolve continuously, introducing distribution shifts that exacerbate forgetting in static retrieval models. While methods like SelfAug [8] address forgetting in the LLM component, we focus on the retriever component, which has been less explored. Our work complements existing methods by targeting a different aspect of the continuous learning challenges of RAG.

3 Task Formulation

We consider a RAG setting where the external document corpus evolves over time. Let $\{\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^T\}$ denote a sequence of corpora, where each $\mathcal{D}^t = \{d_1^t, d_2^t, \ldots, d_{n_t}^t\}$ represents the set of documents available at time step t. At each time step t, the input to the system consists of queries $\mathcal{Q}^t$ and corresponding answers $\mathcal{Y}^t$.

For a given query $q_i^t \in \mathcal{Q}^t$, the retriever selects the top-k relevant documents from $\mathcal{D}^t$ based on a similarity function computed between encoded representations:

$$\mathcal{R}_{q_i^t}^t = \mathrm{TopK}_{d_j^t \in \mathcal{D}^t}\left(\mathrm{sim}(f_q(q_i^t), f_d(d_j^t)), k\right), \tag{1}$$

where $f_q(\cdot)$ and $f_d(\cdot)$ denote the query and document encoders, respectively. The generator G then produces the answer a_i^t conditioned on the query and the retrieved context:

$$a_i^t = G(q_i^t, \mathcal{R}_{q_i^t}^t). \tag{2}$$

This continual learning scenario introduces considerable challenges: as the document distribution evolves, the RAG system must adapt to new domains while maintaining accuracy on previously encountered data.

4 Methodology

To address the challenges of adapting RAG systems to continually evolving document corpora, we propose FlowRAG, a framework that integrates prompt embedding, cross-layer fusion, and generator-guided supervision. As illustrated in Figure 2, our approach enhances traditional RAG pipelines in three main aspects: (1) Prompt Tuning (§4.1), where we insert task-specific prompt embeddings into multiple encoder layers to efficiently adapt to domain shifts; (2) Cross-Layer Fusion (§4.2), which employs a distribution-aware attention mechanism, *StateFusion*, to recursively aggregate prompt-enhanced representations across layers, enabling richer contextual modeling; and (3) Generator-Guided Loss (§4.3), which aligns retrieval and state-level distributions with generation-guided signals through KL-divergence based objectives. Together, these components provide a more efficient and robust adaptation strategy for RAG, mitigating catastrophic forgetting while maintaining strong performance across dynamic domains.

4.1 Prompt Tuning

To adapt to domain-specific document collections D_t at timestamp t, we follow [21] to insert trainable prompt embeddings into multiple layers of the encoder. We describe the model's operation in the order of forward propagation. Specifically, for each transformer layer j, we modify the self-attention computation by prepending the prompt embeddings exclusively to the key and value sequences, while the query input remains unchanged.

Formally, let $H^{(j-1)} \in \mathbb{R}^{B\times T\times D}$ denote the hidden states from the $(j-1)$-th layer, and let $P^{t,j} \in \mathbb{R}^{B\times p\times D}$ denote the prompt embeddings specific to task t at layer j, where B is the batch size, T is the sequence length, D is the hidden size, and p is the number of prompt tokens. Then, the input to the self-attention module at the j-th layer is defined as:

$$\begin{aligned} Q^{(j)} &= W_Q H^{(j-1)} \\ K^{(j)} &= W_K[P^{t,j}; H^{(j-1)}] \\ V^{(j)} &= W_V[P^{t,j}; H^{(j-1)}], \end{aligned} \tag{3}$$

where $[\cdot;\cdot]$ denotes concatenation along the sequence length dimension, and W_Q, W_K, W_V are standard linear projections.

4.2 Cross-Layer Fusion

Although prompt tuning (Section 4.1) adapts the model by independently inserting prompts $P_{t,j}$ at each layer j, this layer-wise isolation limits the formation of hierarchical semantic understanding. Drawing inspiration from methods like iVPT [44], which employ *Cross-Layer Dynamic Connections* (CDC) to enhance inter-layer information flow, we introduce Cross-Layer Fusion. Specifically, our approach leverages *StateFusion*, a distribution-aware attention mechanism, to recursively integrate prompt-enhanced representations from all layers into a single, context-aware embedding. This process unfolds through several key stages:

Inner-Layer Fusion. To implement inner-layer fusion, we design a mechanism called *StateFusion*, which refines standard self-attention by incorporating global statistical properties of the input, specifically the mean and standard deviation computed along the sequence dimension. For a given layer j, with hidden states $H^{(j-1)} \in \mathbb{R}^{B\times T\times D}$

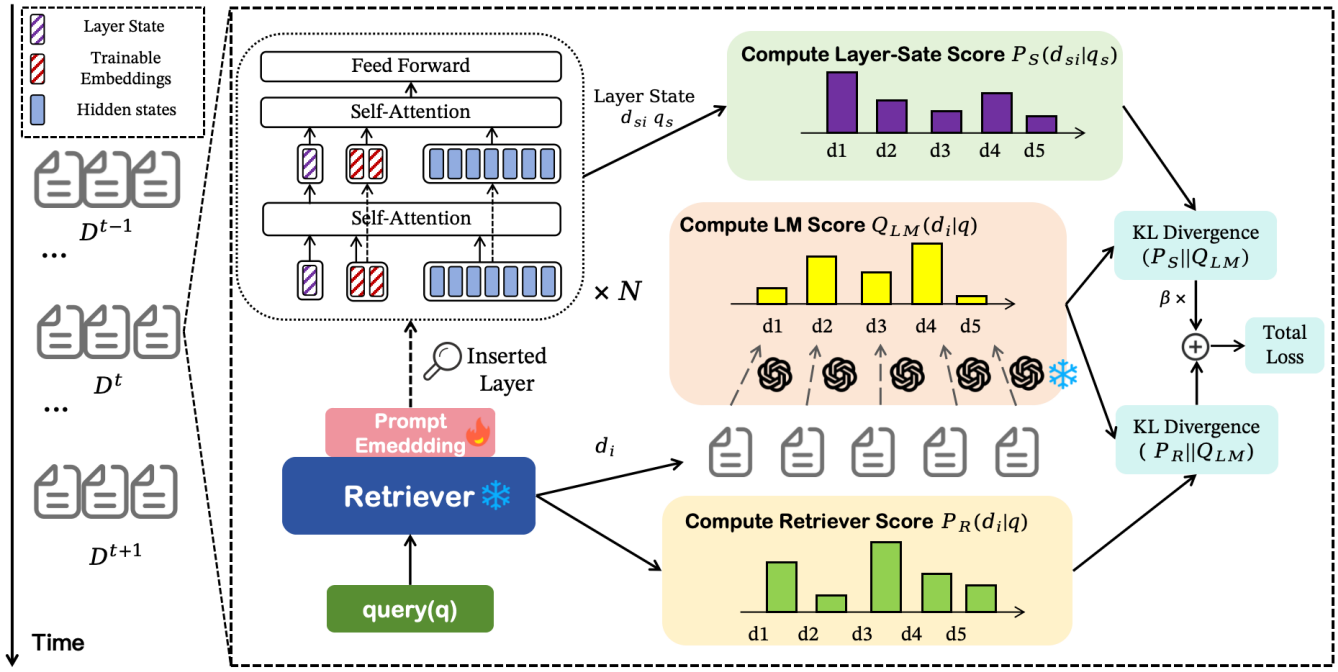

Figure 2: The overall framework of FlowRAG. For document collections D_t at timestamp t, FlowRAG inserts prompt embeddings into multiple layers of the retriever to efficiently adapt to domain shifts, applies cross-layer fusion for richer contextual modeling, and leverages generator-guided loss to align retrieval and state-level distributions with generation-guided signals.

Algorithm 1 StateFusion Mechanism

Input: Hidden states $H^{(j-1)} \in \mathbb{R}^{B\times T\times D}$, prompt slice $P^{t,j} \in \mathbb{R}^{B\times p\times D}$
Output: Fused output O
$\mu_h \leftarrow \text{MEAN}(H^{(j-1)})$, $\sigma_h \leftarrow \text{STD}(H^{(j-1)})$
$\mu_p \leftarrow \text{MEAN}(P^{t,j})$, $\sigma_p \leftarrow \text{STD}(P^{t,j})$
$Q \leftarrow W_Q H^{(j-1)}$, $K \leftarrow W_K P^{t,j}$, $V \leftarrow W_V P^{t,j}$
$Q \leftarrow \text{CONCAT}(Q, \mu_h, \sigma_h)$
$K \leftarrow \text{CONCAT}(K, \mu_p, \sigma_p)$
$V \leftarrow \text{CONCAT}(V, \mu_p, \sigma_p)$
$\text{C} \leftarrow \text{SOFTMAX}(QK^\top/\sqrt{D}) \cdot V$
$O \leftarrow \text{ATTNOUT}(\text{C})$
return O

and prompt embeddings $P^{t,j} \in \mathbb{R}^{B\times p\times D}$, their respective means (μ_h, σ_h) and standard deviations (μ_p, σ_p) are calculated. These statistics are then concatenated with the query (Q), key (K), and value (V) projections prior to the attention computation. The resultant attention output is subsequently processed through a self-output layer, yielding the Single Layer State. Algorithm 1 details this inner-layer fusion process.

Cross-Layer Fusion. To integrate enhanced prompt information across different layers, the Layer States of all preceding prompt insertion layers are recursively aggregated. Specifically, at layer j, the mean of all previously computed Layer States, stored in the queue `PreLayerStates`, is calculated to derive an intermediate aggregated state S. This state S is then fused with the current Layer State L^j using *StateFusion*, which produces an updated Aggregated State A^j. Such recursive fusion enriches contextual representations by facilitating inter-layer information exchange. For the initial layer where a prompt is inserted, A^j is formed from L^j directly, without prior aggregation. Algorithm 2 details this cross-layer fusion process.

Algorithm 2 Cross-Layer Fusion Mechanism

Input: Hidden states $H^{(j-1)} \in \mathbb{R}^{B\times T\times D}$, prompt slice $P^{t,j} \in \mathbb{R}^{B\times p\times D}$
Output: Aggregated state A^j, layer state L^j
$L^j \leftarrow \text{STATEFUSION}(H^{(j-1)}, P^{t,j})$ ▷ See Algorithm 1
Append L^j to `PreLayerStates`
if $j > 1$ **then**
  `P` $\leftarrow \text{STACK}(\texttt{PreLayerStates})$
  `S` $\leftarrow \text{MEAN}(\texttt{P}, \text{dim} = 1)$
  $A^j \leftarrow \text{STATEFUSION}(\texttt{S}, L^j)$
  if j is the last prompt-inserted index **then**
    Set A^j as aggregated query q_s or doc state d_s
  end if
else
  $A^j \leftarrow$ `None`
end if
return A^j, L^j

Aggregated State. To further enhance contextual modeling, the Aggregated State A^j is prepended to the key and value sequences within

the self-attention mechanism, alongside the prompt embeddings. This modifies the standard attention input projections as follows:

$$\begin{aligned} Q^{(j)} &= W_Q H^{(j-1)} \\ K^{(j)} &= W_K [A^j; P^{t,j}; H^{(j-1)}] \\ V^{(j)} &= W_V [A^j; P^{t,j}; H^{(j-1)}]. \end{aligned} \tag{4}$$

This update enables the model to attend not only to the prompt and hidden states within the current layer but also to the fused representations accumulated from previous layers.

At the final prompt insertion layer: if the encoder is the query encoder, A is assigned as the query state q_s; conversely, it is assigned as the document state d_s.

4.3 Generator-Guided Loss

This section introduces the Generator-Guided Loss, which aligns retrieval and state-level distributions with generation signals to improve RAG performance. We propose two losses: Retrieval Likelihood Loss, which favors documents useful for generation, and Aggregated State Loss, which aligns internal representations through cross-layer fusion.

Retrieval Likelihood Loss. Following [29], we construct a generation-aware retrieval distribution by leveraging the likelihood of generating the answer given each retrieved document. Given a query q, its corresponding answer y, and the top n retrieved documents $D' = \{d_1, d_2, \ldots, d_n\}$, the LLM-based distribution is defined as:

$$Q_{\text{LM}}(d_i \mid q, y) = \frac{\exp\left(\text{PLM}(y \mid d_i, q)/\beta\right)}{\sum_{d_j \in D'} \exp\left(\text{PLM}(y \mid d_j, q)/\beta\right)}, \tag{5}$$

where $\text{PLM}(y \mid d_i, q)$ denotes the logarithmic probability of generating the answer y conditioned on the document d_i and the query q, with β serving as a temperature hyperparameter to adjust the sharpness of the distribution.

Meanwhile, the retriever produces its own distribution over the same set D' based on similarity scores $\text{sim}(d_i, q)$:

$$P_R(d_i \mid q) = \frac{\exp\left(\text{sim}(d_i, q)/\beta\right)}{\sum_{d_j \in D'} \exp\left(\text{sim}(d_i, q)/\beta\right)}, \tag{6}$$

where β is a temperature parameter. We minimize the Kullback-Leibler (KL) divergence between P_R and the LLM-derived distribution Q_{LM}, encouraging the retriever to favor documents that are more helpful for generation:

$$\mathcal{L}_{\text{KL}}^{\text{retrieval}} = \frac{1}{|B|} \sum_{x \in B} \text{KL}\left(P_R(d_i \mid q) \,\|\, Q_{\text{LM}}(d_i \mid q, y)\right), \tag{7}$$

where B denotes the training batch.

Aggregated State Loss. To further align internal representations, we introduce a state-level loss based on Cross-Layer Fusion outputs (see Algorithm 2). For each query-document pair (q, d_i), we extract the fused query state q_s and document state d_{si}. Based on similarity scores $\text{sim}(d_{si}, q)$, we generate aggregated state scores:

$$P_S(d_i \mid q) = \frac{\exp\left(\text{sim}(d_{si}, q_s)/\gamma\right)}{\sum_{d_j \in D'} \exp\left(\text{sim}(d_{si}, q_s)/\gamma\right)}. \tag{8}$$

Similar to the likelihood-based loss, we minimize the Kullback-Leibler (KL) divergence between the state-based distribution P_S and the generation-informed distribution Q_{LM}:

$$\mathcal{L}_{\text{KL}}^{\text{state}} = \frac{1}{|B|} \sum_{x \in B} \text{KL}\left(P_S(d_i \mid q) \,\|\, Q_{\text{LM}}(d_i \mid q, y)\right). \tag{9}$$

Total Loss. The final training objective jointly considers both the retrieval likelihood loss and the aggregated state loss. To balance their relative contributions, we introduce a weighting hyperparameter λ on the state-level term:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{KL}}^{\text{retrieval}} + \lambda \cdot \mathcal{L}_{\text{KL}}^{\text{state}}. \tag{10}$$

Here, $\mathcal{L}_{\text{KL}}^{\text{retrieval}}$ ensures that the retriever distribution aligns with generation-informed signals, while $\mathcal{L}_{\text{KL}}^{\text{state}}$ further regularizes the internal representations through state-level alignment. The hyperparameter λ provides flexibility in adjusting the strength of this regularization.

5 Experiment Setup

5.1 Datasets

We evaluate our model on the following datasets:

- **CovidQA** [30]: Derived from CORD-19 [35], it contains 250,000 100-word passages from 5,000 papers (excluding abstracts), with 225,000 synthetic QA pairs (90% training, 10% validation) and 2,000 human-annotated test pairs [23].
- **NewsQA** [30]: Based on the NewsQA corpus [31], it includes 85,000 100-word passages from 10,000 articles, with 100,000 human-annotated QA pairs (90,000 training, 5,000 validation, 5,000 test) from CNN/DailyMail [6].
- **ConvQA** [30]: Built from QAConv [36], it comprises 110,000 100-word passages from 10,000 conversations, each prefixed with ID and speaker, with 35,000 QA pairs (25,000 training, 5,000 validation, 5,000 test).
- **NQ** [14]: It contains 307,000 training QA pairs, 7,800 validation pairs, and 7,800 test pairs from Wikipedia and the web.

5.2 Baselines

We compare our proposed model with several representative baselines, all of which use the same backbone retriever, Contriever [9], as the standard dense retriever in our setting. For the RAG tasks, we employ Qwen2.5 [38] and ChatQA2 [37], a Llama3.0-based model. In this study, we focus on training the retriever while keeping the generator frozen. During evaluation, we use the vllm framework [15] to accelerate inference.

We compare our methods with the following baselines. (1) Contriever [9]: A strong unsupervised dense retriever trained on large-scale corpora. It serves as the standard RAG baseline and represents the performance of static retrievers without online adaptation. (2) Atlas [11]: It applies a Perplexity Distillation loss that trains the retriever to approximate the language model's preference distribution over documents by assessing their impact on perplexity, also using KL divergence. (3) End2EndRAG [30]: An end-to-end framework that jointly optimizes the retriever and generator, providing a more integrated retrieval-augmented generation system. (4) REPLUG [28]: Treats the language model as a black box, augmenting it with a tunable retriever that prepends retrieved documents, allowing the model to indirectly supervise retrieval. (5) L^2R [2]: A lifelong learning approach for first-stage retrieval that maintains a memory

Table 1: Comparison of methods across four QA datasets under three sequential training orders. We report both the final Retrieval ACC@5 (%) and F1 scores (%) on each dataset after sequential training, along with the averaged performance (AVE.) and the forgetting measure (Forget). We highlight the best results for each metric in bold.

Reader	Train Order	Methods	CovidQA		NewsQA		ConvQA		NQ		AVE.		Forget	
			ACC@5	F1	ACC@5	F1	ACC@5	F1	ACC@5	F1	ACC@5	F1	ACC@5	F1
Qwen2.5-7B [38]		Contriever	6.80	13.80	23.53	10.47	44.34	19.86	34.02	27.71	27.17	17.96	/	/
	Order 1	REPLUG	3.40	7.85	11.21	4.96	32.45	25.12	56.78	41.20	25.96	19.78	13.43	13.99
		End2EndRAG	3.57	7.98	13.89	6.99	34.82	25.25	56.12	40.26	27.10	20.12	13.12	14.36
		Atlas	2.36	5.17	12.52	8.80	28.22	22.72	52.21	38.41	23.83	18.78	12.83	12.53
		L^2R	4.59	10.29	21.39	13.92	40.03	34.00	54.79	41.40	30.20	24.90	6.57	6.57
	Order 2	REPLUG	4.96	9.59	22.42	13.25	46.75	39.91	24.22	21.90	24.59	21.16	15.95	12.56
		End2EndRAG	2.37	7.77	22.30	13.34	45.58	39.14	28.18	19.87	24.61	20.03	10.06	13.92
		Atlas	0.28	7.30	23.51	12.71	41.22	36.17	21.75	22.73	21.69	19.73	17.69	13.40
		L^2R	4.65	10.62	28.16	19.60	44.18	38.74	38.79	30.75	28.95	24.93	9.52	6.83
	Order 3	REPLUG	3.23	7.87	23.05	14.90	16.39	9.71	55.06	40.15	24.43	18.16	14.44	15.81
		End2EndRAG	3.46	7.60	19.50	15.01	12.27	6.14	**56.38**	40.61	22.90	17.34	16.33	16.44
		Atlas	4.42	5.66	24.99	12.87	12.27	5.24	43.85	38.03	21.38	15.45	14.67	18.75
		L^2R	5.15	13.20	27.98	17.57	24.87	19.13	55.07	39.21	28.27	22.28	9.11	8.03
		FlowRAG(ours)	**10.88**	**20.54**	**29.98**	**22.03**	**41.95**	**35.89**	53.24	**41.08**	**34.01** (+3.81)	**29.89** (+4.96)	**0** (+9.11)	**0** (+8.03)
Llama3-8B [37]		Contriever	6.80	17.29	23.53	19.40	44.34	29.38	34.02	40.11	27.17	26.55	/	/
	Order 1	REPLUG	2.98	12.16	13.35	11.02	25.23	28.60	56.46	53.86	24.51	26.41	13.22	10.66
		End2EndRAG	3.00	11.33	11.15	12.74	27.72	25.02	54.89	50.87	24.19	24.99	15.98	9.44
		Atlas	1.25	10.67	10.51	9.87	23.87	20.85	53.84	52.68	22.37	23.52	15.16	11.09
		L^2R	7.98	15.11	21.77	14.26	29.60	27.40	58.44	52.17	29.45	27.24	5.16	4.84
	Order 2	REPLUG	1.97	10.31	17.03	15.95	39.79	33.88	21.52	28.56	20.08	22.18	18.99	11.13
		End2EndRAG	1.51	10.42	15.15	16.63	**45.71**	38.00	12.61	30.75	18.75	23.95	22.71	10.32
		Atlas	2.53	12.97	16.99	16.70	44.66	36.67	17.86	26.70	20.51	23.26	21.09	11.42
		L^2R	6.15	16.98	27.85	22.03	43.70	37.80	39.84	35.84	29.39	28.16	8.57	3.85
	Order 3	REPLUG	3.55	12.96	10.45	10.78	9.54	9.79	50.39	42.63	18.48	19.04	15.84	13.29
		End2EndRAG	2.72	10.63	22.34	16.00	12.30	12.43	**55.20**	44.40	23.14	20.87	16.68	14.11
		Atlas	4.95	14.92	19.70	15.24	14.48	9.46	53.70	**44.72**	23.21	21.09	15.42	13.74
		L^2R	7.92	18.48	24.15	20.71	24.44	23.74	56.29	43.13	28.20	26.52	7.16	5.25
		FlowRAG(ours)	**12.34**	**22.92**	**32.36**	**24.21**	45.69	**37.38**	53.44	43.49	**35.96** (+6.51)	**32.00** (+3.84)	**0** (+7.16)	**0** (+5.25)

Table 2: Parameter efficiency comparison

Method	# Trainable Params	Percentage
Fine-tuning	110M	100%
FlowRAG (ours)	**0.63M**	**0.64%**

buffer of support negatives and incorporates a diverse negative selection strategy and ranking alignment objective, avoiding costly index rebuilding.

5.3 Evaluation Metrics

We evaluate our models using three complementary metrics: Top-k retrieval accuracy, average F1, and forgetting score.

Top-k Retrieval Accuracy. Following [30], we evaluate the effectiveness of the retriever using the Top-k retrieval accuracy. Specifically, this metric checks whether the ground-truth answer appears in any of the top k retrieved passages. In our experiments, we set $k = 5$. For brevity, we denote the Top-5 retrieval accuracy as **ACC@5** in all result tables.

F1 Score. The F1 score is calculated as the harmonic mean of precision and recall at the token level.

We present the mean F1 score across all test sets following training on the final task T.

Forgetting Score. To quantify catastrophic forgetting, we define the forgetting score for each intermediate task D_i ($i < T$) as:

$$\text{Forget}_{D_i} = \max_{t \le T} \text{F1}_t(D_i) - \text{F1}_T(D_i), \tag{11}$$

where $\text{F1}_t(D_i)$ denotes the F1 on D_i after training on task t. The total forgetting is then the average over all tasks except the last:

$$\text{Total Forgetting} = \frac{1}{T-1} \sum_{i=1}^{T-1} \text{Forget}_{D_i}. \tag{12}$$

A lower total forgetting score indicates better retention of earlier tasks.

6 Experimental Evaluation

6.1 Evaluation Framework

To evaluate the continual learning capability of our proposed method and baselines, we adopt a sequential training and evaluation framework across four datasets: CovidQA, NewsQA, ConvQA, and NQ. This setting simulates a realistic scenario where tasks arrive incrementally, requiring the retriever to adapt to new domains while preserving knowledge from previous tasks.

We consider three training orders by randomly permuting the datasets:

(1) CovidQA → NewsQA → ConvQA → NQ
(2) CovidQA → NQ → NewsQA → CovidQA

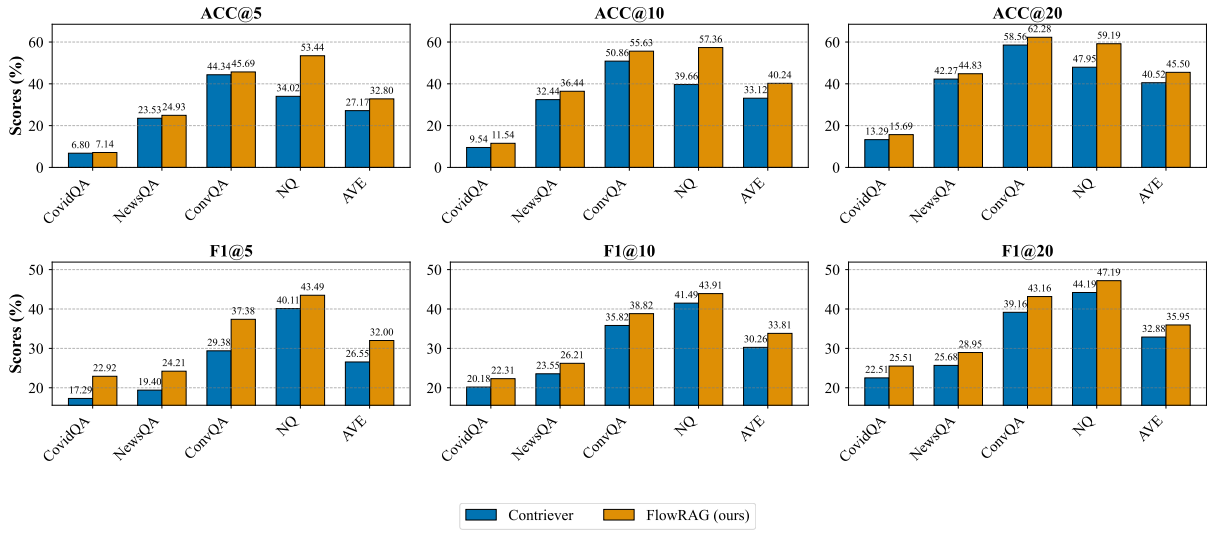

Figure 3: Comparison of Top-k Retrieval Accuracy (ACC) and F1 scores across multiple datasets for the Top-5 (left), Top-10 (middle), and Top-20 (right) retrieval settings

(3) ConvQA → CovidQA → NewsQA → NQ

Training Phase. In each phase, we sample 1,000 QA pairs from the training set of each dataset (approximately 1.6%–20% of the dataset), simulating a continual-learning scenario with T tasks. For all methods, we retrieve the top-5 documents per query to provide context to the generator; alternative choices are explored in Section 6.3.

Baselines. For fine-tuning methods, we train the retriever with AdamW, using a learning rate of 1×10^{-5} and a warm-up ratio of 0.1. For L2R, we follow the experimental setup from the original paper [2].

Proposed method. We set the prompt length to 150 and insert the prompts into encoder layers 1–7. The retriever is trained with AdamW (learning rate 5×10^{-3}, warm-up ratio 0.1). The KL-divergence temperature for retrieval-likelihood alignment is 0.1, and for aggregated-state alignment it is 1. We set $\beta = 0.6$ in Eq. (10).

Evaluation Phase. After training on each dataset (phase t), we evaluate the model on the test sets of all previously seen datasets. This yields a $T \times T$ matrix of Top-5 Retrieval ACC and F1 scores, where T is the number of tasks.

6.2 Experimental Results and Analysis

Table 1 reports the final results on each dataset after sequential training. Detailed results on the changes throughout every sequential training step can be found in Appendix C. Overall, our FlowRAG demonstrates a significant advantage over all baseline methods across four datasets and different training orders, highlighting the effectiveness of our approach. We make the following key observations:

(1) **Superior Performance.** Our proposed FlowRAG consistently outperforms existing methods across a range of datasets and training orders. On the Llama model, FlowRAG achieves 35.96% ACC@5 and 32.00% F1, outperforming L^2R by 6.95% in ACC@5 and 4.7% in F1. In Orders 1, 2, and 3, FlowRAG surpasses L^2R by 6.51%, 6.57%, and 7.76% in ACC@5, and by 4.8%, 3.8%, and 5.5% in F1, respectively. These improvements are consistent across different training orders and generalize well to other models, such as Qwen2.5-7B. This demonstrates the broad applicability and robustness of our method.

(2) **Better Performance for Generation Tasks.** While L^2R performs well in sequential retrieval tasks, FlowRAG excels in generation tasks. This is reflected in the results, where FlowRAG consistently outperforms L^2R in both retrieval accuracy and F1 scores across multiple datasets and training orders. For example, on the Llama model, FlowRAG achieves a +6.51% improvement in ACC@5 and +4.96% improvement in F1 on the NQ dataset compared to L^2R.

(3) **Reduced Forgetting.** Sequential fine-tuning baselines exhibit substantial degradation on earlier tasks, and L^2R only partially mitigates this issue. In contrast, FlowRAG maintains stable performance across tasks without explicit replay. By assigning task-specific prompt embeddings, FlowRAG preserves previously learned knowledge and avoids interference during continual training. As a result, it achieves strong retrieval accuracy and F1 across different training orders while showing no noticeable forgetting, making it well-suited for continual retrieval on evolving corpora.

(4) **Parameter Efficiency.** Additionally, as shown in Table 2, fine-tuning the Contriever model[1] involves updating all 110M parameters. In contrast, FlowRAG introduces six layers of prompt embeddings, each with a length of 150 for a task, updating only 0.63M parameters. This represents just 0.64% of the total model parameters, showcasing the parameter efficiency of our approach.

6.3 Ablation Study

Table 3 presents an ablation study evaluating the contribution of each component in our framework on two representative readers. We compare three variants: **(1) w/ Contriever**, which uses a frozen

[1] https://huggingface.co/facebook/contriever

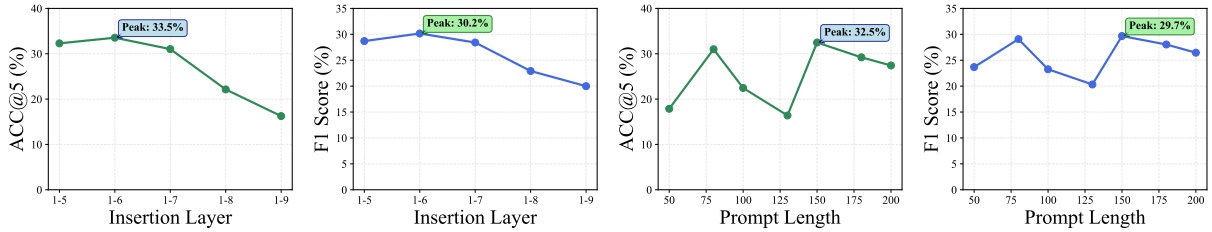

Figure 4: Impact of insertion layer (left) and prompt length (right) on the performance

Table 3: Ablation results

Method	Qwen2.5-7B		Llama3-8B	
	ACC@5	F1	ACC@5	F1
w/ Contriever	27.17	17.96	27.17	26.55
w/ Prompt + RLLoss	31.64	27.68	32.77	30.16
w/ Cross-Layer Fusion + ASLoss	**34.01**	**29.89**	**35.96**	**32.00**

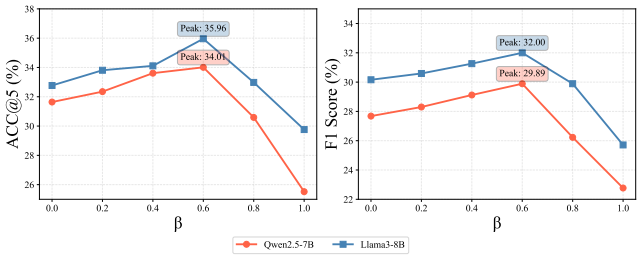

Figure 5: The impact of the hyperparameter λ. The left shows the Retrieval ACC@5, and the right shows the F1 score.

retriever without optimization; **(2) w/ Prompt + RLLoss**, which applies prompt tuning with the Retrieval Likelihood loss to better align retrieval with generation; and **(3) w/ Cross + ASLoss (Full)**, which further introduces cross-layer fusion and the Aggregated State loss.

Overall, adding prompt tuning and RLLoss consistently improves over the frozen retriever, with notable gains in F1, indicating stronger retrieval–generation alignment. Our full model achieves the best ACC@5 and F1 on both Qwen2.5-7B and Llama3-8B, validating the effectiveness of combining architectural fusion with state-aggregated supervision.

Figure 3 further compares FlowRAG with the Contriever baseline across datasets under different retrieval depths (top-5/10/20). FlowRAG consistently outperforms the baseline, and the gap generally widens as more documents are retrieved. For example, on NQ, ACC@5 increases from 34.02% to 53.44%, with further improvements at ACC@10 and ACC@20. Similar trends hold for F1, demonstrating the robustness of FlowRAG under deeper retrieval.

6.4 Sensitivity Analysis

We conduct a sensitivity analysis on three hyperparameters that affect FlowRAG's performance: (1) the number of insertion layers, (2) the input prompt length, and (3) the loss weighting factor. Unless otherwise stated, we vary the insertion layer with a fixed prompt length of 150 tokens, and vary the prompt length with a fixed insertion depth of 6. We also study the weighting parameter λ that balances the two generator-guided loss terms in Section 4.3.

Insertion Layer. The left two plots in Figure 4 show that both Retrieval ACC@5 and F1 peak when inserting at layers 1–7 (33.5% and 30.2%, respectively), while performance drops sharply at deeper layers (8–9). This suggests that too shallow insertion limits capacity, whereas overly deep insertion may introduce noise and dilute salient information, degrading both retrieval and generation.

Prompt Length. The right two plots in Figure 4 indicate a non-monotonic effect of prompt length. Retrieval ACC@5 is low with very short prompts, peaks at 150 tokens (32.5%), and then declines. F1 exhibits a similar but milder trend, also performing best at 150 tokens (29.7%). These results suggest that adequate context is beneficial, but overly long prompts may introduce irrelevant details.

Loss Weighting. We further examine the balance between the retrieval likelihood loss and the state prediction loss by setting

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{KL}}^{\text{retrieval}} + \lambda \cdot \mathcal{L}_{\text{KL}}^{\text{state}}.$$

Figure 5 reports results on both Qwen2.5-7B and Llama3-8B across different λ. When $\lambda = 0$, using only the retrieval loss yields suboptimal performance. Increasing λ improves both ACC@5 and F1, with the best results at $\lambda = 0.6$. Beyond this point, performance degrades, suggesting that over-weighting the state loss can overconstrain the retriever and hurt factual document selection. Overall, $\lambda = 0.6$ provides the best trade-off.

7 Conclusion

In this paper, we presented FlowRAG, a lightweight and effective approach for continual retriever adaptation in Retrieval-Augmented Generation (RAG) with evolving document collections. FlowRAG introduces Layer-wise Prompt Embeddings and a Cross Layer Fusion mechanism to capture hierarchical semantics, along with a Generator-Guided Loss that aligns retrieval with generation likelihoods. Experiments on dynamic retrieval benchmarks show that FlowRAG consistently outperforms strong baselines in both retrieval and generation, while mitigating catastrophic forgetting. Our method provides a robust and parameter efficient solution for adaptive RAG.

7.1 Acknowledgments

This work is supported by National Key Research and Development Program of China (No.2023YFB4502300) and NSFC-RGC under Grant No.62461160333.

References

[1] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. 2017. Expert Gate: Lifelong Learning with a Network of Experts. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 7120–7129. doi:10.1109/CVPR.2017.753

[2] Yinqiong Cai, Keping Bi, Yixing Fan, Jiafeng Guo, Wei Chen, and Xueqi Cheng. 2023. L2R: Lifelong Learning for First-stage Retrieval with Backward-Compatible Representations. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 183–192.

[3] Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. 2018. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. In *Computer Vision – ECCV 2018*. Springer International Publishing, Cham, 556–572.

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

[5] 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*. 6491–6501.

[6] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1* (Montreal, Canada) *(NIPS'15)*. MIT Press, Cambridge, MA, USA, 1693–1701.

[7] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 2790–2799. https://proceedings.mlr.press/v97/houlsby19a.html

[8] Yuqing Huang, Rongyang Zhang, Qimeng Wang, Chengqiang Lu, Yan Gao, Yi Wu, Yao Hu, Xuyang Zhi, Guiquan Liu, Xin Li, Hao Wang, and Enhong Chen. 2025. SelfAug: Mitigating Catastrophic Forgetting in Retrieval-Augmented Generation via Distribution Self-Alignment. arXiv:2509.03934 [cs.CL] https://arxiv.org/abs/2509.03934

[9] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. arXiv:2112.09118 [cs.IR] https://arxiv.org/abs/2112.09118

[10] Gautier Izacard and Edouard Grave. 2020. Distilling knowledge from reader to retriever for question answering. *arXiv preprint arXiv:2012.04584* (2020).

[11] 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. *J. Mach. Learn. Res.* 24, 1, Article 251 (Jan. 2023), 43 pages.

[12] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 6769–6781. doi:10.18653/v1/2020.emnlp-main.550

[13] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences* 114, 13 (2017), 3521–3526. arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.1611835114 doi:10.1073/pnas.1611835114

[14] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. doi:10.1162/tacl_a_00276

[15] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles* (Koblenz, Germany) *(SOSP '23)*. Association for Computing Machinery, New York, NY, USA, 611–626. doi:10.1145/3600006.3613165

[16] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 793, 16 pages.

[17] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs. arXiv:2304.08244 [cs.CL] https://arxiv.org/abs/2304.08244

[18] Yayong Li, Peyman Moghadam, Can Peng, Nan Ye, and Piotr Koniusz. 2025. Inductive Graph Few-shot Class Incremental Learning. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining* (Hannover, Germany) *(WSDM '25)*. Association for Computing Machinery, New York, NY, USA, 466–474. doi:10.1145/3701551.3703578

[19] 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. 2024. RA-DIT: Retrieval-Augmented Dual Instruction Tuning. arXiv:2310.01352 [cs.CL] https://arxiv.org/abs/2310.01352

[20] 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. doi:10.1162/tacl_a_00638

[21] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks. *CoRR* abs/2110.07602 (2021). arXiv:2110.07602 https://arxiv.org/abs/2110.07602

[22] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. 2022. Online continual learning in image classification: An empirical survey. *Neurocomputing* 469 (2022), 28–51. doi:10.1016/j.neucom.2021.10.021

[23] Timo Möller, Anthony Reina, Raghavan Jayakumar, and Malte Pietsch. 2020. COVID-QA: A Question Answering Dataset for COVID-19. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Karin Verspoor, Kevin Bretonnel Cohen, Mark Dredze, Emilio Ferrara, Jonathan May, Robert Munro, Cecile Paris, and Byron Wallace (Eds.). Association for Computational Linguistics, Online. https://aclanthology.org/2020.nlpcovid19-acl.18/

[24] Alexander Pluska, Pascal Welke, Thomas Gärtner, and Sagar Malhotra. 2024. Logical distillation of graph neural networks. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning* (Hanoi, Vietnam) *(KR '24)*. Article 86, 11 pages. doi:10.24963/kr.2024/86

[25] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. arXiv:2307.16789 [cs.AI] https://arxiv.org/abs/2307.16789

[26] Kandan Ramakrishnan, Rameswar Panda, Quanfu Fan, John Henning, Aude Oliva, and Rogerio Feris. 2020. Relationship Matters: Relation Guided Knowledge Transfer for Incremental Learning of Object Detectors. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 1009–1018. doi:10.1109/CVPRW50498.2020.00133

[27] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. iCaRL: Incremental Classifier and Representation Learning. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5533–5542. doi:10.1109/CVPR.2017.587

[28] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-Augmented Black-Box Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Kevin Duh, Helena Gomez, and Steven Bethard (Eds.). Association for Computational Linguistics, Mexico City, Mexico, 8371–8384. doi:10.18653/v1/2024.naacl-long.463

[29] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-Augmented Black-Box Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Kevin Duh, Helena Gomez, and Steven Bethard (Eds.). Association for Computational Linguistics, Mexico City, Mexico, 8371–8384. doi:10.18653/v1/2024.naacl-long.463

[30] Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. 2023. Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering. *Transactions of the Association for Computational Linguistics* 11 (2023), 1–17. doi:10.1162/tacl_a_00530

[31] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A Machine Comprehension Dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, Phil Blunsom, Antoine Bordes, Kyunghyun Cho, Shay Cohen, Chris Dyer, Edward Grefenstette, Karl Moritz Hermann, Laura Rimell, Jason Weston, and Scott Yih (Eds.). Association for Computational Linguistics, Vancouver, Canada, 191–200. doi:10.18653/v1/W17-2623

[32] Shangqing Tu, Yuanchun Wang, Jifan Yu, Yuyang Xie, Yaran Shi, Xiaozhi Wang, Jing Zhang, Lei Hou, and Juanzi Li. 2024. R-Eval: A Unified Toolkit for Evaluating Domain Knowledge of Retrieval Augmented Large Language Models. In

Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 5813–5824. doi:10.1145/3637528.3671564

[33] Gido M. van de Ven and Andreas S. Tolias. 2019. Three scenarios for continual learning. arXiv:1904.07734 [cs.LG] https://arxiv.org/abs/1904.07734

[34] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (Sept. 2014), 78–85. doi:10.1145/2629489

[35] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex Wade, Kuansan Wang, Nancy Xin Ru Wang, Chris Wilhelm, Boya Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. CORD-19: The COVID-19 Open Research Dataset. arXiv:2004.10706 [cs.DL] https://arxiv.org/abs/2004.10706

[36] Chien-Sheng Wu, Andrea Madotto, Wenhao Liu, Pascale Fung, and Caiming Xiong. 2022. QAConv: Question Answering on Informative Conversations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 5389–5411. doi:10.18653/v1/2022.acl-long.370

[37] Peng Xu, Wei Ping, Xianchao Wu, Chejian Xu, Zihan Liu, Mohammad Shoeybi, and Bryan Catanzaro. 2025. ChatQA 2: Bridging the Gap to Proprietary LLMs in Long Context and RAG Capabilities. arXiv:2407.14482 [cs.CL] https://arxiv.org/abs/2407.14482

[38] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] https://arxiv.org/abs/2505.09388

[39] Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. arXiv:2310.01558 [cs.CL] https://arxiv.org/abs/2310.01558

[40] Zichun Yu, Chenyan Xiong, Shi Yu, and Zhiyuan Liu. 2023. Augmentation-Adapted Retriever Improves Generalization of Language Models as Generic Plug-In. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 2421–2436. doi:10.18653/v1/2023.acl-long.136

[41] Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) *(ICML'17)*. JMLR.org, 3987–3995.

[42] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shutao Xia. 2019. Maintaining Discrimination and Fairness in Class Incremental Learning. arXiv:1911.07053 [cs.CV] https://arxiv.org/abs/1911.07053

[43] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. 2024. Retrieval-Augmented Generation for AI-Generated Content: A Survey. arXiv:2402.19473 [cs.CV] https://arxiv.org/abs/2402.19473

[44] Nan Zhou, Jiaxin Chen, and Di Huang. 2024. iVPT: Improving Task-relevant Information Sharing in Visual Prompt Tuning by Cross-layer Dynamic Connection. arXiv:2404.05207 [cs.CV] https://arxiv.org/abs/2404.05207

A Experiemnt Details

A.1 Prompt Template

The prompt template to generate the response in the experiment is show in Figure 6.

```
System: This is a chat between a user and an artificial
intelligence assistant.
The assistant gives helpful, detailed, and polite answers
to the user's questions based on the context.
The assistant should also indicate when the answer
cannot be found in the context.

#Retrieved Passages:
{document}

#User query:
{query}
```

Figure 6: The prompt to generate the response in the experiment.

B Device

This information about the device currently used in the experiment is list in 4.

Table 4: device information.

	Detail
CPU	Intel Xeon Platinum 8368Q
GPU	2× NVIDIA A100 80GB
CUDA Version	12.8

C Result Details

Table 5: Detailed Performance on Training Order 1

Train on →	CovidQA		NewsQA		ConvQA		NQ	
	ACC@5	F1	ACC@5	F1	ACC@5	F1	ACC@5	F1
REPULG								
CovidQA	11.22	21.77						
NewsQA	8.70	18.42	31.84	24.10				
ConvQA	4.79	15.44	22.85	19.90	38.16	37.89		
NQ	2.98	12.16	13.35	11.02	25.23	28.60	56.46	53.86
EndEndRAG								
CovidQA	10.76	21.16						
NewsQA	7.47	16.51	32.62	23.65				
ConvQA	4.45	15.59	22.13	18.90	46.44	32.61		
NQ	3.00	11.33	11.15	12.74	27.72	25.02	54.89	50.87
Atlas								
CovidQA	11.90	20.57						
NewsQA	7.14	17.56	29.35	20.78				
ConvQA	5.11	14.12	19.31	15.30	39.87	33.30		
NQ	1.25	10.67	10.51	9.87	23.87	20.85	53.84	52.68
L^{2}R								
CovidQA	11.27	21.07						
NewsQA	10.68	19.94	29.17	21.47				
ConvQA	9.14	18.29	26.08	18.47	39.56	33.57		
NQ	7.98	15.11	21.77	14.26	29.60	27.40	58.44	52.17

Table 6: Detailed Performance on Training Order 2

Train on →	CovidQA		NQ		NewsQA		ConvQA	
	ACC@5	F1	ACC@5	F1	ACC@5	F1	ACC@5	F1
REPULG								
CovidQA	11.22	21.16						
NQ	6.12	18.94	57.21	44.93				
NewsQA	2.70	14.95	40.71	35.84	29.05	22.13		
ConvQA	1.97	10.31	21.52	28.56	17.03	15.95	39.79	33.88
EndEndRAG								
CovidQA	10.76	20.57						
NQ	5.34	14.96	54.43	43.96				
NewsQA	2.64	12.96	37.83	37.98	32.21	24.22		
ConvQA	1.51	10.42	12.61	30.75	15.15	16.63	45.71	38.00
Atlas								
CovidQA	11.90	21.77						
NQ	7.34	18.77	55.84	44.68				
NewsQA	3.42	15.99	35.81	33.25	32.91	24.18		
ConvQA	2.53	12.97	17.86	26.70	16.99	16.70	44.66	36.67
L^{2}R								
CovidQA	11.75	21.88						
NQ	9.50	19.46	55.56	44.13				
NewsQA	7.97	18.59	47.34	39.08	32.24	24.24		
ConvQA	6.15	16.98	39.84	35.84	27.85	22.03	43.70	37.80

Table 7: Detailed Performance on Training Order 3

Train on →	ConvQA		CovidQA		NewsQA		NQ	
	ACC@5	F1	ACC@5	F1	ACC@5	F1	ACC@5	F1
REPULG								
ConvQA	38.87	33.22						
CovidQA	29.09	25.30	12.69	22.88				
NewsQA	17.07	18.10	6.08	17.83	19.49	17.30		
NQ	9.54	9.79	3.55	12.96	10.45	10.78	50.39	42.63
EndEndRAG								
ConvQA	43.44	35.94						
CovidQA	32.96	29.15	13.03	22.85				
NewsQA	19.96	20.41	7.42	15.76	30.92	22.61		
NQ	12.30	12.43	2.72	10.63	22.34	16.00	55.20	44.40
Atlas								
ConvQA	41.87	35.30						
CovidQA	33.81	28.29	12.75	22.38				
NewsQA	21.70	14.38	9.59	19.52	30.76	23.16		
NQ	14.48	9.46	4.95	14.92	19.70	15.24	53.70	44.72
L^{2}R								
ConvQA	41.60	35.98						
CovidQA	37.39	32.86	12.95	24.59				
NewsQA	28.10	27.13	10.31	21.99	30.60	23.35		
NQ	24.44	23.74	7.92	18.48	24.15	20.71	56.29	43.13

Table 5 – 7 report detailed results for all methods under three sequence orders, using the Llama model as the generator:

(1) CovidQA → NewsQA → ConvQA → NQ

(2) CovidQA → NQ → NewsQA → CovidQA

(3) ConvQA → CovidQA → NewsQA → NQ

We report two evaluation metrics: (1) Top-5 retrieval accuracy. (2) F1 score.

Baselines. For fine-tuning methods, we train the retriever with AdamW, using a learning rate of 1×10^{-5} and a warm-up ratio of 0.1. For L2R, we follow the experimental setup from the original paper [2].