Privacy-Preserving Reasoning with Knowledge-Distilled Parametric Retrieval Augmented Generation

Jinwen Chen[1,2] **Hainan Zhang**[1,2*] **Liang Pang**[3]
Yongxin Tong[4] Haibo Zhou[5] Wei Lin[5] Zhiming Zheng[1,2]
[1]Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing
[2]School of Artificial Intelligence, Beihang University, China
[3]Institute of Computing Technology, Chinese Academy of Sciences
[4]School of Computer Science and Engineering, Beihang University, China
[5]Meituan
{jwkami, zhanghainan}@buaa.edu.cn

Abstract

The current RAG system requires uploading plaintext documents to the cloud, risking private data leakage. Parametric RAG (PRAG) encodes documents as LoRA parameters within LLMs, offering a possible way to reduce exposure of raw content. However, it still faces two issues: (1) PRAG demands synthesizing QA pairs and fine-tuning LLM for each individual document to create its corresponding LoRA, leading to **unacceptable inference latency**. (2) The performance of PRAG relies solely on synthetic QA data while lacking internal alignment with standard RAG, resulting in **poor generalization** on out-of-distribution (OOD) inputs. Therefore, achieving high-efficiency parameterization while maintaining RAG-level performance remains a critical challenge for privacy-preserving reasoning. In this paper, we propose DistilledPRAG, a generalizable knowledge-distilled parametric RAG model aligned with standard RAG in document structure and parameter activation. We first synthesize QA pairs from single and multi-documents to enhance cross-document reasoning. Then, we mask the plaintext documents with a special token and translate them to LoRA via a parameter generator, maintaining the standard RAG document structure. Finally, guided by synthetic QA data, we train the parameter generator to match standard RAG's hidden states and output logits, enabling RAG-style reasoning without original documents. Experiments on four QA datasets show that DistilledPRAG outperforms baselines in accuracy and generalizes well on OOD data [1].

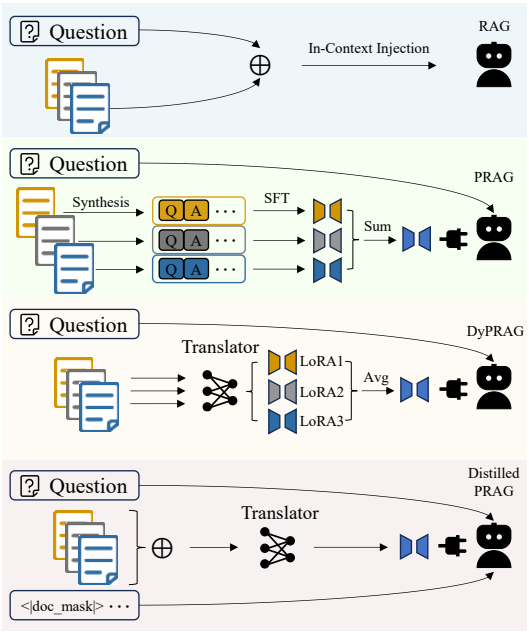


Figure 1: Inference Paradigms for standard RAG, PRAG, DyPRAG, and our DistilledPRAG. (1) Standard RAG inputs the plaintext documents and the question. (2) PRAG generates QA pairs per document to fine-tune LoRA adapters, and sums them to obtain document aggregated representations for LLM injection. (3) DyPRAG translates individual documents to its LoRA and averages them to achieve document aggregation for LLM injection. (4) DistilledPRAG generates cross-document LoRA from concatenated documents via a parameter generator, then takes masked documents and questions as input, similar to standard RAG.

1 Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Karpukhin et al., 2020; Zhang et al., 2024, 2026a,b; Wang et al., 2025a,b; Zheng et al., 2025; Zhang et al., 2025) enables real-time integration of external knowledge into LLMs and is widely used across fields like finance, law, and materials science. However, current RAG systems require uploading local documents to the cloud, raising privacy concerns when handling sensitive information such as corporate contracts, medical records, or personal notes (Zeng et al., 2024). Therefore, a new RAG paradigm without plaintext is necessary to support privacy-preserving reasoning.

* Corresponding author: zhanghainan@buaa.edu.cn
[1]https://github.com/JWQZ/DistilledPRAG

Findings of the Association for Computational Linguistics: ACL 2026, pages 18824–18838

Recently, parametric RAG (PRAG) (Su et al., 2025) is proposed to encode documents into LoRA parameters (Hu et al., 2022) and upload to a cloud server, thereby avoiding the transmission of plaintext documents. As shown in Figure 1, PRAG first synthesizes QA pairs for an individual document and then fine-tunes LLM with them to obtain its corresponding LoRA parameters. In inference, PRAG aggregates the LoRAs of retrieved documents to answer questions. However, the need for both data synthesis and fine-tuning per document leads to unacceptable inference latency in real-world scenarios.

To reduce this latency, DyPRAG (Tan et al., 2025) proposes training a dynamic parameter translator to replace data synthesis and fine-tuning at test-time. It still synthesizes QA pairs and fine-tunes LLM to obtain the corresponding LoRA like PRAG, but differs by training a linear parameter translator to map each document to its LoRA. As shown in Figure 1, during inference, each retrieved document is processed by the parameter translator to generate its LoRA, which is then averaged and injected into the LLM to generate the final answer.

However, both PRAG and DyPRAG rely solely on synthetic QA pairs to activate knowledge learning, lacking internal alignment with standard RAG, which may lead to **poor generalization** on out-of-distribution (OOD) inputs. This misalignment appears in two ways: (1) Document structure: training LoRA on individual documents and aggregating them disrupts cross-document reasoning. For instance, in Figure 1, summing or averaging LoRA1, LoRA2, and LoRA3 will miss cross-document reasoning cues, leading to incorrect answers(see in Appendix B.1). (2) Parameter activation: performing inference only with the query alters internal parameter activations compared to standard RAG, losing original reasoning capability. Therefore, efficient parameterization with RAG-level reasoning remains a major challenge for parametric RAG.

In this paper, we introduce DistilledPRAG, a knowledge-distilled parametric RAG model by aligning the document structure and parameter activations with a standard RAG for more robust generalization on OOD data. Specifically, we first use DeepSeek-V3 to synthesize QA pairs for individual documents and concatenated multi-documents to enhance cross-document reasoning. Then, we replace plaintext document tokens with a special token and translate them to LoRA by our LongT5-based parameter generator, forming the student model with the same document structure as standard RAG(teacher). Finally, guided by the synthetic QA data, we train the parameter generator by minimizing the differences between the student and teacher in both hidden states and output logits, enabling it to learn RAG-style reasoning without access to the original documents. In this setting, the student model inherits the teacher model's structure and activations, allowing rapid learning of standard RAG reasoning.

Experiments on four public QA datasets demonstrate that DistilledPRAG, trained only on 2WQA, outperforms baselines on three OOD datasets, validating its strong generalization capabilities. Further analysis on synthetic data and alignment functions confirms the effectiveness of our internal alignment mechanism. Our main contributions are:

- We identify that internal alignment between parametric RAG and standard RAG is crucial, facilitating efficient and highly generalizable document-specific parameter learning.
- We propose a knowledge-distilled parametric RAG by internal alignment with standard RAG in terms of both document structure and activation parameters, thereby enabling robust generalization to OOD data.
- Experiments show that DistilledPRAG delivers strong QA performance without uploading plaintext documents, and exhibits a superior ability to convert unseen documents into reasoning-capable LoRA modules.

2 Related Work

Recently, Parametric Retrieval-Augmented Generation (PRAG) (Su et al., 2025) proposes encoding retrieved documents into model parameters (e.g., via LoRA) instead of appending them as context. This approach injects external knowledge directly into the LLMs, treating retrieval as parameter updates rather than input expansion. To obtain the document's parameters, PRAG requires synthesizing QA pairs and fine-tuning LLMs for each retrieved document, which has unacceptable inference latency in real-world testing scenarios. To address this, Dynamic PRAG (DyPRAG) (Tan et al., 2025) is proposed to introduce a parameter translator that maps documents to parameters at test-time, enabling document-specific parameterization without additional fine-tuning or storage overhead.

DyPRAG improves flexibility but often fails to generalize across OOD inputs, limiting its ability to replace traditional RAG under privacy constraints fully. To overcome these issues, we propose a knowledge-distilled parametric RAG method by cross-document data augmentation and aligning the model's behavior across the student and teacher models, significantly enhancing its generalization ability to OOD inputs.

3 Backgrounds

Now, we will introduce the process and notions of standard RAG and PRAG. Let q denote a user query and $D = \{d_1, \cdots, d_N\} \in \mathcal{C}$ denote the top-N documents retrieved from a large corpus $\mathcal{C}$ via a retriever $\mathcal{R}$. The standard RAG constructs the input of LLMs as a concatenation of the documents and the query:

$$x = [d_1, \cdots, d_N; q], \tag{1}$$

and generates y via LLM:

$$y = \arg\max_{y'} P(y' \mid x; \theta), \tag{2}$$

where θ is the parameter of LLM. However, this setup inevitably exposes private documents at inference time, raising serious privacy concerns in many practical scenarios.

In PRAG, given the query q and the retrieved documents D, it synthesizes QA pairs from individual document $d_i \in D$ and fine-tunes LLM to obtain its LoRA $\Delta\theta^i$. Then, the resulting LoRAs are summed to inject into LLM for reasoning y :

$$y = \arg\max_{y'} P(y' \mid q; \theta + \Delta\theta), \tag{3}$$

$$\Delta\theta = \sum_{i=1}^{N} \Delta\theta^i. \tag{4}$$

4 Model

This section introduces the DistilledPRAG model, covering synthetic data construction, knowledge distillation design, and online inference procedures, as shown in Figure 2.

4.1 Synthetic Data Construction

To supervise the parametric RAG model in answering questions, we construct a large-scale dataset with 289,079 synthetic QA pairs by DeepSeek-V3 (DeepSeek-AI, 2024) as the training dataset $\mathcal{D}$. An example is: "Document: [history of Ada Lovelace] Question: What was Ada Lovelace's main contribution to computer science? Answer: She is credited as the first computer programmer..." The proceeds are as follows:

1. We randomly sample 30,000 documents from the 2WikiMultihopQA (Ho et al., 2020) training set as $\mathcal{D}_{wiki}$, covering diverse topics and question styles.

2. For each document $d_i \in \mathcal{D}_{wiki}$, we use DeepSeek-V3 with carefully designed prompts(see in Appendix A.3) to automatically generate 2-5 high-quality question-answer pairs, donated as $\{(d_i, q_{ij}, a_{ij})\}$ where $2 \leq j \leq 5$, totally 139,723 single-document QA pairs into our training dataset $\mathcal{D}$.

3. To simulate realistic multi-document retrieval, for each $d_i \in \mathcal{D}_{wiki}$, we randomly sample a document d_i' from the corpus $\mathcal{D}_{wiki}$ and concatenate them. Then we use the same DeepSeek-V3 with carefully designed prompts (see in Appendix A.3) to automatically generate at least 5 additional cross-document QA pairs, focusing on reasoning that spans both documents, donated as $\{(d_i + d_i', q_{ij}', a_{ij}')\}$ where $j \geq 5$, totally 149,356 cross-document QA pairs into $\mathcal{D}$.

This construction yields a challenging multi-hop QA dataset with broad coverage, which encourages the model to encode document semantics into adapter parameters deeply. For the convenience of a formal definition, we uniformly define the training set as $\mathcal{D} = \{(D_i, q_i, a_i)\}_{i=1}^{N}$, which includes both single-document and cross-document QA pairs,

4.2 Knowledge Distillation

We define the teacher and the student models' inputs and outputs for each training triple $(D_i, q_i, a_i) \in \mathcal{D}$ as:

- The **original input** $x_i = [D_i; q_i]$ is passed to the standard LLMs f_θ.

- The **masked input** $\tilde{x}_i = [\texttt{<|doc\_mask|>}^{|D_i|}; q_i]$ is passed to the student's LLMs $f_{\theta+\Delta\theta_i}$. Here, $\texttt{<|doc\_mask|>}^{|D_i|}$ is formed by repeating the special token <|doc_mask|> to match the length of $|D_i|$, $\Delta\theta_i = \mathcal{T}_\phi(D_i)$, where $\mathcal{T}_\phi$ is a parameter generator network to map

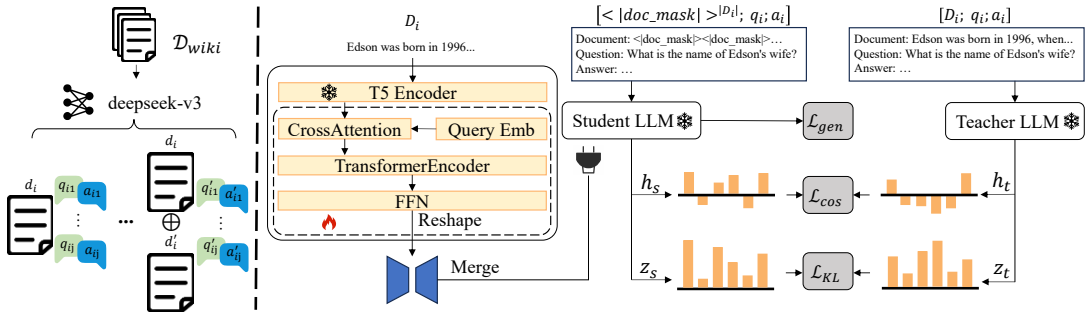


Figure 2: The Architecture of DistilledPRAG Model. ①Use DeepSeek-V3 to mine knowledge from a single document and augmented cross-documents by random concatenation. ②Train a parameter generator to map documents to a LoRA for student LLM, enabling it to mimic a teacher RAG's reasoning by minimizing differences in hidden states and logits on synthetic data.

documents D_i to its corresponding LoRA parameters $\Delta\theta_i$.

4.2.1 Special Token Initialization

To prevent raw document content from being leaked during inference, we introduce a special token <|doc_mask|> which replaces all tokens in the raw document. Naive initialization (e.g., random values) often leads to unstable training or degraded performance, as they are mismatched with the distribution of the pretrained embedding space(see in Experiments 5.3.3). Therefore, we propose a special token initialization that aligns with the first-order and second-order statistics of the model's pretrained vocabulary, thereby promoting stable training.

Formally, let $\mathbf{E} \in \mathbb{R}^{V \times h}$ be the embedding matrix of LLMs, where V is the vocabulary size and h is the hidden state size. We compute the mean and variance of the embedding distribution as:

$$\boldsymbol{\mu} = \frac{1}{V}\sum_{i=1}^{V}\mathbf{E}_i, \quad \boldsymbol{\sigma} = \sqrt{\frac{1}{V}\sum_{i=1}^{V}(\mathbf{E}_i - \boldsymbol{\mu})^2}. \quad (5)$$

Then, we sample the special token's embedding from this distribution:

$$\mathbf{e}_{\text{mask}} = \boldsymbol{\mu} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \text{diag}(\boldsymbol{\sigma}^2)), \quad (6)$$

where diag(.) is the diagonal matrix.

After initialization, $\mathbf{e}_{\text{mask}}$ is frozen mask and serves as a stable, information-free placeholder used only for document masking, never appearing in queries or answers.

4.2.2 Parameter Generator

Here, $\mathcal{T}_\phi$ denotes a *parameter generator network* that transforms the full documents D_i into LoRA weights $\Delta\theta_i$. Specifically, it consists of two components: a document embedding model Enc_ψ and a parameter generator Gen_ω, such that:

$$\mathcal{T}_\phi(D_i) = \text{Gen}_\omega\left(\text{Enc}_\psi(D_i)\right), \quad (7)$$

where Enc_ψ is a pretrained LongT5 encoder (Guo et al., 2022) to encode documents D_i into embeddings $\mathbf{E}_{\mathbf{D_i}} \in \mathbb{R}^{L \times d}$, where L is the sequence length of D_i and d is the embedding dimension.

Parameter generator Gen_ω maps the document embeddings $\mathbf{E}_{\mathbf{D_i}}$ to the target LoRA. Specifically, given $\mathbf{E}_{\mathbf{D_i}}$ as key and value, a learnable query embeddings $\mathbf{Q} \in \mathbb{R}^{\tilde{L} \times d}$, where $\tilde{L}$ is the number of hidden layers in LLM, we utilize the multi-head cross-attention mechanism to selectively aggregate information from the documents:

$$\mathbf{H}_0 = \text{CrossAttention}(\mathbf{Q}, \mathbf{E}_{\mathbf{D_i}}, \mathbf{E}_{\mathbf{D_i}}). \quad (8)$$

Next, the output $\mathbf{H}_0$ undergoes a self-attention-based Transformer encoder, which models dependencies and integrates information within the query sequence itself:

$$\mathbf{H}_1 = \text{SelfAttentionEncoder}(\mathbf{H}_0). \quad (9)$$

Finally, a feed-forward network (FFN) maps the encoded representation to the desired LoRA parameter space:

$$\mathcal{T}_\phi(D_i) = \text{FFN}(\mathbf{H}_1). \quad (10)$$

This architecture enables efficient extraction, aggregation, and projection of document knowledge into model parameters. Importantly, both the base LLM parameters θ and the document encoder parameters ψ are frozen throughout training, and only the generator Gen_ω is updated.

4.2.3 Training Objectives

The training objectives of the parameter generator are: generative loss on synthetic QA data, alignment loss of internal representations, and KL divergence of output logits between the student model and the teacher model.

Generative Loss: Given the documents D_i and the query q_i, we minimize the negative log-likelihood for generating the answer a_i from the masked input $\tilde{x}_i$:

$$\mathcal{L}_{\text{gen}} = -\log P(a_i \mid \tilde{x}_i;\, \theta + \Delta\theta_i), \tag{11}$$

$$\Delta\theta_i = \mathcal{T}_\phi(D_i). \tag{12}$$

Internal Alignment Loss: To align the internal reasoning process, we match their hidden representations at each layer between the student model and the teacher model. Let $h_{\text{t}}^{(j)}$ and $h_{\text{s}}^{(j)}$ denote the hidden states from the j-th hidden layer of teacher model $f_\theta(x_i)$ and student model $f_{\theta+\Delta\theta_i}(\tilde{x}_i)$, respectively. The cosine similarity alignment loss for the j-th layer is defined as:

$$\mathcal{L}_{\text{cos}}^{(j)} = 1 - \cos(h_{\text{t}}^{(j)},\, h_{\text{s}}^{(j)}). \tag{13}$$

The overall alignment loss is then computed as the weighted average of the losses from all layers, where the weight of the j-th layer is proportional to j:

$$\mathcal{L}_{\text{cos}} = \frac{\sum_{j=1}^{M} j\, \mathcal{L}_{\text{cos}}^{(j)}}{\sum_{j=1}^{M} j} \tag{14}$$

where M is the total number of hidden layers in LLM. We believe that the layers closer to the output are more important.

KL Divergence of Output Logits: To further ensure output consistency, we align the predictive distributions at the answer tokens a_i by minimizing the Kullback-Leibler divergence (Kullback and Leibler, 1951) between the logits from the student model and the teacher model. Let $\mathbf{z}_{\text{t}}$ and $\mathbf{z}_{\text{s}}$ be the pre-softmax logits of the teacher model and the student model for answer positions:

$$\mathcal{L}_{\text{KL}} = \text{KL}\left(\text{softmax}(\mathbf{z}_{\text{t}}) \parallel \text{softmax}(\mathbf{z}_{\text{s}})\right). \tag{15}$$

Finally, we optimize only the generator parameters ω via gradient descent over the above objectives: $\mathcal{L}_{\text{gen}}$, $\mathcal{L}_{\text{cos}}$ and $\mathcal{L}_{\text{KL}}$. The base model and document encoder remain frozen during training. The overall objective can be written as:

$$\min_{\omega}\ \mathbb{E}_{(D_i,q_i,a_i)\sim\mathcal{D}}\left[\mathcal{L}_{\text{gen}} + \lambda_1\mathcal{L}_{\text{cos}} + \lambda_2\mathcal{L}_{\text{KL}}\right], \tag{16}$$

where λ_1, λ_2 are hyperparameters that balance hidden-states and output-logits alignment. Note that our distillation is not performed between a large teacher model and a smaller student model in the traditional sense, but rather between two different input paradigms.

4.3 Online Inference

At inference time, our approach diverges from previous methods such as PRAG and DyPRAG, which require explicit fusion of LoRA parameters from multiple documents. In PRAG, the LoRA A and B matrices generated on each document are concatenated along the LoRA rank dimension. In contrast, DyPRAG directly averages the LoRA A and B weights generated on each document. Instead, we adhere to the same input organization paradigm as in our training phase. During the fusion of LoRA weights, there may be loss or corruption of knowledge and information, as this fusion process is not explicitly modeled during training in their methods. In contrast, our inference paradigm remains consistent with training, helping enhance generalization performance.

Given a user query q_{test}, we first employ a retriever $\mathcal{R}$ to obtain the top-k relevant documents $\{d_1, \ldots, d_k\}$. These documents are concatenated in their original order to form a composite document $d_{\text{inf}} = [d_1; \ldots; d_k]$. To preserve privacy, we mask the entire d_{inf} using our special token, resulting in the masked input $\tilde{x}_{\text{inf}} = [\texttt{<|doc\_mask|>}^{|d_{\text{inf}}|}; q_{\text{test}}]$. The composite document is then passed to the parameter generator network $\mathcal{T}_\phi$, which computes LoRA adapter weights $\Delta\theta_{\text{inf}} = \mathcal{T}_\phi(d_{\text{inf}})$ conditioned on the composite document. The LoRA-augmented model $f_{\theta+\Delta\theta_{\text{inf}}}$ processes the masked input and generates the final answer (prompts in the Appendix A.3):

$$g_{\text{test}} = f_{\theta+\Delta\theta_{\text{inf}}}(\tilde{x}_{\text{inf}}). \tag{17}$$

In summary, our online inference procedure is fully aligned with the data format during training and obviates the need for complex parameter fusion schemes commonly required in prior works. During inference, the document is never exposed in plaintext, and only the dynamically generated $\Delta\theta_i$ is used to condition the model.

5 Experiments

To demonstrate the performance of DistilledPRAG, we define the in-domain benchmark and OOD

Base LLM	Method	2WQA				HQA		PQA	CWQ	Avg
		Compare	Bridge	Inference	Compose	Bridge	Compare			
LLaMA-1B	Standard RAG	34.9	32.7	**28.2**	7.9	**18.9**	27.5	17.8	29.1	24.6
	PRAG	41.2	41.0	18.9	5.5	13.9	**42.2**	**21.3**	31.7	27.0
	DyPRAG	31.3	20.9	22.3	7.2	10.4	21.3	9.7	23.2	18.3
	DistilledPRAG	**42.0**	**43.1**	26.3	**14.5**	14.7	32.6	14.3	**38.9**	**28.3**
LLaMA-8B	Standard RAG	28.7	37.1	**31.9**	8.9	**33.6**	**55.2**	**33.0**	41.9	33.8
	PRAG	**45.1**	37.0	19.8	6.9	18.6	44.7	17.6	36.2	28.2
	DyPRAG	44.7	41.9	21.6	12.4	14.9	47.1	12.4	41.4	29.6
	DistilledPRAG	41.3	**45.6**	30.1	**16.2**	26.9	54.2	25.6	**49.0**	**36.1**
Mistral-7B	Standard RAG	26.7	28.1	26.7	6.8	16.2	23.6	**18.2**	18.4	20.6
	PISCO	28.9	27.9	**27.9**	6.6	**16.8**	24.0	14.6	**26.0**	21.6
	DistilledPRAG	**34.0**	**35.9**	25.8	**10.2**	15.4	**26.4**	12.6	24.7	**23.1**

Table 1: Overall F1(%) performance of DistilledPRAG and baselines on 2WQA, HQA, PQA and CWQ datasets. Bold indicates the best performance, and underlined indicates the second best. Our performance on Qwen-14B can be seen in Appendix B.3

datasets across four datasets and conduct comparisons with baselines.

5.1 Experimental Setup

5.1.1 Datasets and Metrics

We evaluate question-answering performance using the F1 score(%) for our DistilledPRAG method and baselines on four open-domain QA datasets: 2WikiMultihopQA(2WQA) (Ho et al., 2020), HotpotQA(HQA) (Yang et al., 2018), PopQA(PQA) (Mallen et al., 2023), and ComplexWebQuestions(CWQ) (Talmor and Berant, 2018). For a fair comparison between DistilledPRAG and baselines, following PRAG and DyPRAG, we use the same first 300 questions from the dev split of each sub-task dataset as the test set. **But their training sets are different.** PRAG requires no training and only loads offline-generated LoRA parameters at test time. DyPRAG proposes to train its parameter translator on questions 301-600 from each sub-task's dev split, which has a distribution similar to the test set. Notably, our DistilledPRAG only use the training split of 2WQA as a training dataset, and test on the 2WQA dev set as the in-domain benchmark, while HQA, PQA, and CWQ dev sets serve as OOD datasets for generalization evaluation. Therefore, **the generalization evaluation of our model is more rigorous**, demonstrating the effectiveness of our method. Details in Appendix A.1.

5.1.2 Implementation Details

We conduct experiments using two LLM backbones: **Llama-3.2-1B-Instruct** and **LLaMA3-8B-Instruct**, and compare with PISCO using **Mistral-7B-Instruct-v0.2**. All experiments run in PyTorch on NVIDIA A100 80GB PCIE and RTX PRO 6000 Blackwell GPUs. The parameter generator uses the **LongT5 encoder** as its embedding model. Training uses AdamW with a learning rate of 10^{-4}, 10% warm-up, a polynomial scheduler ending at 10^{-6}, batch size 4, one epoch, LoRA rank 2, and LoRA alpha 32. We set $\lambda_1 = 0.5$ and $\lambda_2 = 0.1$, keeping all other hyperparameters at defaults. In inference, we disable sampling `do_sample=False` and use greedy decoding for deterministic outputs.

5.1.3 Baselines

For DistilledPRAG and all baselines, we adopt a standard BM25 retriever (Robertson and Walker, 1994) to fetch the top-3 documents for each question. All baselines are **Standard RAG**, **PRAG** (Su et al., 2025), **DyPRAG** (Tan et al., 2025), **PISCO** (Louis et al., 2025). See Appendix A.2 for baseline details and Appendix B.2 for retriever and top-k analysis.

5.2 Main Results

Table 1 reports QA performance for DistilledPRAG and several baselines across datasets and backbone LLMs. DistilledPRAG attains the best or second-best results on most sub-tasks and consistently yields the highest average F1 score. With the LLaMA-8B backbone, it achieves the top average F1, surpassing standard RAG, PRAG, and DyPRAG by 2.3, 7.9, and 6.5, respectively.

Notably, DistilledPRAG is trained only on the 2WQA training set, whereas the baseline DyPRAG

Method	2WQA				HQA		PQA	CWQ	Avg
	Compare	Bridge	Inference	Compose	Bridge	Compare			
DistilledPRAG	**41.3**	45.6	30.1	16.2	**26.9**	**54.2**	**25.6**	**49.0**	**36.1**
w/o $\mathcal{L}_{\text{cos}}$	41.0	**46.3**	29.2	15.7	25.8	51.2	22.8	46.0	34.7
w/o $\mathcal{L}_{\text{KL}}$	33.1	33.9	29.5	16.5	24.5	42.5	24.5	44.6	31.1
w/o $\mathcal{L}_{\text{cos}}$, $\mathcal{L}_{\text{KL}}$	30.5	30.5	**30.2**	**17.2**	24.3	39.8	23.3	44.5	30.0

Table 2: Ablation study of alignment losses based on LLaMA3-8B-Instruct backbone model.

Method	2WQA				HQA		PQA	CWQ	Avg
	Compare	Bridge	Inference	Compose	Bridge	Compare			
DistilledPRAG	**42.0**	43.1	26.3	**14.5**	**14.7**	**32.6**	14.3	**38.9**	**28.3**
Llama-Synthesis	38.2	**44.6**	24.3	14.1	**14.7**	29.4	**18.6**	36.9	27.6
Single-Document	33.8	42.4	**27.0**	12.9	**14.7**	29.7	12.7	38.2	26.4

Table 3: The impact of QA synthesis on model performance using Llama-3.2-1B-Instruct. Llama-Synthesis uses Llama-3.2-1B-Instruct (vs. DeepSeek-V3) to generate QA data. Single-Document uses only single-document QA pairs of training dataset $\mathcal{D}$(see in Section 4.1).

is trained on a similar distribution of data with test data [2]. Nevertheless, DistilledPRAG shows strong OOD generalization on HQA, PQA, and CWQ, achieving a leading 49.0 on CWQ. Similar patterns hold for LLaMA-1B and LLaMA-8B, where DistilledPRAG consistently matches or surpasses the strongest baselines in both in-domain and OOD settings. Overall, DistilledPRAG demonstrates robust transferability and practical effectiveness for real-world open-domain QA.

5.3 Analysis

5.3.1 Alignment Loss

Table 2 shows that removing either alignment loss $\mathcal{L}_{\text{cos}}$ or $\mathcal{L}_{\text{KL}}$ reduces performance. Dropping $\mathcal{L}_{\text{cos}}$ decreases accuracy by 3.9%, while removing $\mathcal{L}_{\text{KL}}$ leads to a larger 13.9% drop. Eliminating both results in a 16.9% decline. These findings indicate that aligning hidden states and logits is crucial for capturing the teacher model's representations, improving semantic sensitivity, and enhancing generalization to unseen or out-of-distribution data.

5.3.2 QA Synthesis

Table 3 shows the impact of DeepSeek synthetic data and cross-document augmentation. Using QA pairs generated by Llama-3.2-1B leads to a 0.7 average performance drop, suggesting that synthetic QA quality is essential for training the parameter generator. High-quality QA pairs capture key facts and reasoning patterns, providing stronger supervision (see Appendix B.4). Superficial or incomplete QA generation produces LoRA parameters that miss key document knowledge, weakening downstream performance. Moreover, when training only on single-document QA pairs, performance drops by 1.9 because the training data lacks knowledge diversity. Cross-document QA synthesis adds questions that require reasoning across multiple sources, helping the model learn broader, more transferable representations. Without this augmentation, the model overfits to narrow, document-specific patterns and struggles with complex or open-domain queries. Additionally, we also compare DistilledPRAG with baselines on the same synthesizer and data size in Appendix B.5.

5.3.3 Special Token

Table 4 compares several mask-token configurations: no token, a randomly initialized token, and a trainable token. (1) Removing mask tokens causes a large F1 drop (17.7%), showing that preserving the model's structural paradigm is crucial for effective parameter-generator training. (2) Using a randomly initialized fixed token $\mathbf{e}_{\text{mask}} \sim \mathcal{N}\left(0, \sigma^2 I\right), \sigma = 0.02$ yields an 8.9% lower F1 than ours. Because such a token lacks semantic grounding, the model struggles to interpret it. In contrast, our token is sampled using the mean and variance of all vocabulary embeddings, aligning it with the pretrained embedding space

[2]DyPRAG proposes to train its parameter translator on questions 301-600 from each sub-task's dev split, which has a distribution similar to the test set.

Method	2WQA				HQA		PQA	CWQ	Avg
	Compare	Bridge	Inference	Compose	Bridge	Compare			
DistilledPRAG	**42.0**	**43.1**	**26.3**	**14.5**	**14.7**	32.6	14.3	**38.9**	**28.3**
No-token	35.5	29.8	25.3	12.5	13.6	23.3	13.3	33.5	23.3
Random-token	39.2	41.6	22.6	9.0	9.9	**39.3**	9.2	35.5	25.8
Trainable-token	31.9	39.2	22.8	8.4	13.7	25.3	**15.3**	31.7	23.5

Table 4: The impact of special tokens on masked documents using Llama-3.2-1B-Instruct: No-token deletes content without masking; Random-token uses default Transformer initialization for special token; Trainable-token has a learnable special token embedding.

Method	Latency		
	LLaMA-1B	LLaMA-8B	Mistral-7B
Standard RAG	0.14	0.52	0.76
PRAG	0.6(+18)	1.18(+100)	-
DyPRAG	0.52	0.8	-
PISCO	-	-	0.92
DistilledPRAG	0.3	0.81	1

Table 5: Average inference latency (s). For PRAG, extra time (in brackets) is added for offline LoRA synthesis and training when new documents are introduced.

	2WQA	HQA	PQA	CWQ
DyPRAG	19.9	17.1	17.5	25.4
Ours	19.8	17.5	17.8	16.6

Table 6: The overlap (%) between documents used in training and those retrieved from the test set.

and stabilizing training. (3) Making the mask token trainable results in a notable performance drop, likely because continual changes to the token destabilize document representations.[3]

5.3.4 Inference Latency

Table 5 shows the average inference latency of all methods under identical hardware (RTX PRO 6000 and Intel Xeon Gold 5318Y). Our method achieves the lowest latency for small models, except for RAG, because it requires only one round of parameter generation. In contrast, DyPRAG needs three rounds of parameter generation and aggregation, and PRAG repeatedly loads offline document parameters, which significantly increases overhead. Loading LoRA parameters accounts for 33.9%, 38.9%, and 6.9% of total inference time for PRAG, DyPRAG, and DistilledPRAG, respectively. As model size grows, our latency increases due to special tokens matching the length of the original documents. Still, our inference process is the same as standard RAG, and our goal is to match RAG-level performance without plaintext documents rather than to optimize speed.

[3]Document-specific mask tokens are impractical here, as each document has fewer than 15 QA pairs, insufficient for effective token training.

5.3.5 Overlap Between Train and Test

To further validate the generalization of DistilledPRAG and eliminate the risk of test set leakage, we estimated the maximum Jaccard similarity (Jaccard, 1901) between each test document and all training documents using an efficient MinHash and Locality-Sensitive Hashing(LSH) approach. For each test document, we recorded the highest similarity with any training document and averaged these scores. As shown in Table 6, the average maximum similarity across four datasets remains below 20%. This demonstrates limited textual overlap and confirms the robustness and authenticity of our method's generalization. Additionally, we add analysis of reconstruction attack in Appendix B.6.

6 Conclusion

In this work, we present DistilledPRAG, a novel parametric RAG model that addresses the generalization and efficiency limitations of existing approaches such as PRAG and DyPRAG. By aligning both input structure and parameter activations with a standard RAG teacher model, DistilledPRAG leverages knowledge distillation to achieve robust RAG-style reasoning without access to plaintext documents. Experimental results on four public QA datasets demonstrate that DistilledPRAG significantly outperforms strong baselines in OOD settings, even when trained on a single dataset. In future work, we plan to extend DistilledPRAG to support multi-modal inputs and explore its applicability in settings with more complex reasoning and open-domain generation tasks.

Limitations

Despite its promising performance, DistilledPRAG remains an approximation of standard RAG. While our internal alignment strategy improves reasoning consistency, the parametric nature of LoRA representations may still fall short in capturing nuanced cross-document interactions, especially in complex multi-hop scenarios. Furthermore, the privacy-preserving design by avoiding transmission of plaintext documents inevitably introduces a privacy-utility tradeoff. Encoding documents into LoRA may lose fine-grained contextual details, limiting answer accuracy on knowledge-intensive queries. Future work may explore richer alignment objectives and more expressive methods to further close the gap between parametric and standard RAG performance.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No.U25B2070 and No. 62406013, the Beijing Advanced Innovation Center Funds for Future Blockchain and Privacy Computing(GJJ-24-034), and the Fundamental Research Funds for the Central Universities.

References

DeepSeek-AI. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Mandy Guo, Joshua Ainslie, David C Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. Longt5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

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 *International Conference on Learning Representations*.

Paul Jaccard. 1901. Etude de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Societe Vaudoise des Sciences Naturelles*, 37:547–579.

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)*, pages 6769–6781, Online. Association for Computational Linguistics.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Yibin Lei, Liang Ding, Yu Cao, Changtong Zan, Andrew Yates, and Dacheng Tao. 2023. Unsupervised dense retrieval with relevance-aware contrastive pre-training. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10932–10940, Toronto, Canada. Association for Computational Linguistics.

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*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Jimmy J. Lin and Xueguang Ma. 2021. A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. *ArXiv*, abs/2106.14807.

Maxime Louis, Hervé Déjean, and Stéphane Clinchant. 2025. Pisco: Pretty simple compression for retrieval-augmented generation. *arXiv preprint arXiv:2501.16075*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822.

David Rau, Shuai Wang, Hervé Déjean, Stéphane Clinchant, and Jaap Kamps. 2025. Context embeddings for efficient answer generation in retrieval-augmented generation. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pages 493–502.

S. E. Robertson and S. Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, page 232–241, Berlin, Heidelberg. Springer-Verlag.

Weihang Su, Yichen Tang, Qingyao Ai, Junxi Yan, Changyue Wang, Hongning Wang, Ziyi Ye, Yujia Zhou, and Yiqun Liu. 2025. Parametric retrieval

augmented generation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1240–1250.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Yuqiao Tan, Shizhu He, Huanxuan Liao, Jun Zhao, and Kang Liu. 2025. Dynamic parametric retrieval augmented generation for test-time knowledge enhancement. *arXiv preprint arXiv:2503.23895*.

Yujing Wang, Hainan Zhang, Liang Pang, Binghui Guo, Hongwei Zheng, and Zhiming Zheng. 2025a. Maferw: query rewriting with multi-aspect feedbacks for retrieval-augmented large language models. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Yujing Wang, Hainan Zhang, Liang Pang, Yongxin Tong, Binghui Guo, Hongwei Zheng, and Zhiming Zheng. 2025b. Learning to erase private knowledge from multi-documents for retrieval-augmented large language models. *Preprint*, arXiv:2504.09910.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Shenglai Zeng, Jiankun Zhang, Pengfei He, Yiding Liu, Yue Xing, Han Xu, Jie Ren, Yi Chang, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. 2024. The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4505–4524, Bangkok, Thailand. Association for Computational Linguistics.

Qianchi Zhang, Hainan Zhang, Liang Pang, Yongxin Tong, Hongwei Zheng, and Zhiming Zheng. 2026a. Less is more: Compact clue selection for efficient retrieval-augmented generation reasoning. In *Proceedings of the ACM Web Conference 2026*, page 1971–1982.

Qianchi Zhang, Hainan Zhang, Liang Pang, Hongwei Zheng, and Zhiming Zheng. 2024. Adacomp: Extractive context compression with adaptive predictor for retrieval-augmented large language models. *arXiv preprint arXiv:2409.01579*.

Qianchi Zhang, Hainan Zhang, Liang Pang, Hongwei Zheng, and Zhiming Zheng. 2026b. Stable-rag: Mitigating retrieval-permutation-induced hallucinations in retrieval-augmented generation. *arXiv preprint arXiv:2601.02993*.

Shiwen Zhang, Lingxiang Wang, Hainan Zhang, Ziwei Wang, Sijia Wen, and Zhiming Zheng. 2025. Beyond the surface: A solution-aware retrieval model for competition-level code generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5237–5246, Suzhou, China. Association for Computational Linguistics.

JiaYing Zheng, HaiNan Zhang, Liang Pang, YongXin Tong, and ZhiMing Zheng. 2025. Can synthetic query rewrites capture user intent better than humans in retrieval-augmented generation? *Preprint*, arXiv:2509.22325.

A Experimental Settings

A.1 Datasets and Metrics

We evaluate question-answering performance using the F1 score(%) for our DistilledPRAG method and baselines on four open-domain QA datasets: 2WikiMultihopQA(2WQA) (Ho et al., 2020), HotpotQA(HQA) (Yang et al., 2018), PopQA(PQA) (Mallen et al., 2023), and ComplexWebQuestions(CWQ) (Talmor and Berant, 2018). 2WQA and HQA evaluate multi-hop reasoning by requiring integration of information from multiple sources. PQA tests factual recall and entity disambiguation. CWQ assesses multi-step reasoning over web-based content. The 2WQA and HQA datasets categorize questions by reasoning type, with four sub-tasks for 2WQA and two for HQA.

For a fair comparison between DistilledPRAG and baselines, following PRAG and DyPRAG, we use the same first 300 questions from the dev split of each sub-task dataset as the test set. **But their training sets are different.** PRAG requires no training and only loads offline-generated LoRA parameters at test time. DyPRAG proposes to train its parameter translator on questions 301-600 from each sub-task's dev split, which has a distribution similar to the test set. Notably, our Distilled PRAG only use the training split of 2WQA as a training dataset, and test on the 2WQA dev set as the in-domain benchmark, while HQA, PQA, and CWQ dev sets serve as OOD datasets for generalization evaluation. Therefore, **the generalization evaluation of our model is more rigorous**, demonstrating the effectiveness of our method.

A.2 Baselines

For DistilledPRAG and all baselines, we adopt a standard BM25 retriever (Robertson and Walker, 1994) to fetch the top-3 documents for each question. Our method and all baselines are as follows:

- **Standard RAG**: Using the retrieved documents as in-context input for backbone LLM.
- **PRAG** (Su et al., 2025): Parameterizing each document from the corpus into LoRA weights by offline synthesizing QA pairs and fine-tuning LLM. During inference, they retrieve the corresponding LoRAs for each query-relevant document and sum them according to the LoRA rank dimension.
- **DyPRAG** (Tan et al., 2025): During inference, dynamically generating document-specific LoRA using a translator network for each document, and averaging them as the aggregated document representation.
- **PISCO** (Louis et al., 2025): Optimized from COCOM (Rau et al., 2025), compressing each retrieved document into a few embeddings and concatenating them with the input's embedding for backbone LLM.
- **DistilledPRAG**: The parameter generator directly encodes the retrieved top-3 documents to the unified multi-document LoRA, without additional LoRA aggregation.

A.3 Prompts Used in Our Experiments

Figures 3, 4, and 5 show the prompts we use for single-document QA synthesis, cross-document QA synthesis, and inference.

Prompt for Single-Document QA Synthesis

I will provide a passage of text, and you need to generate two to five different questions based on the content of this passage. Each question should be answerable using the information provided in the passage. Additionally, please provide an appropriate answer for each question derived from the passage.
You need to generate the question and answer in the following format:

```
[
  {
    "question": "What is the capital of France?",
    "answer": "Paris"
    "full_answer": "The capital of France is Paris."
  },
]
```

This list should have at least two elements. As long as the passage information is sufficient, you should try to generate five elements as much as possible. You only need to output this list in the above format.
Passage:
{passage}

Figure 3: Prompt for single-document QA synthesis.

B Additional Analysis

B.1 Cross-document Testing

We extract 100 challenging cross-document QA pairs synthesized by DeepSeek-V3 as test data to evaluate the robustness of different methods under strong cross-document reasoning requirements. As shown in Table 7, DistilledPRAG achieves an F1 score of 30.9, which is remarkably close to the performance of standard RAG (34.6). This indicates that DistilledPRAG preserves most of RAG's cross-document reasoning capability even without accessing the original documents at inference time.

In contrast, PRAG and DyPRAG show severe degradation in this setting, with F1 scores dropping

Prompt for Cross-Document QA Synthesis

I will provide multiple passages of text. Your task is to generate five different questions that can only be answered by synthesizing information from multiple passages. Each question must require combining information across all passages. Do not create questions that can be answered using only a single passage.

To help you think more broadly, here are some types of questions you can consider:

- **Comparison questions** (e.g., comparing entities or events mentioned in different passages)
- **Cause-and-effect questions** (e.g., identifying how one event described in one passage influences or results in another event in a different passage)
- **Temporal reasoning questions** (e.g., constructing a timeline or understanding chronological dependencies)
- **Entity synthesis questions** (e.g., asking about a person/organization/concept that appears in multiple passages with different attributes)
- **Theme or topic integration** (e.g., identifying common themes, contrasting perspectives, or overarching insights across documents)

Each question should be answerable using the combined information provided in the set of passages. For each question, provide both a short answer and a full sentence answer.

You must follow this format strictly:

```
[
  {
    "question": "What is the capital of France?",
    "answer": "Paris",
    "full_answer": "The capital of France is Paris."
  },
  ... (at least five in total)
]
```

Only output the list in this exact format.

Passages:

{passage}

Figure 4: Prompt for single-document QA synthesis.

Prompt for Inference

user_template:
You should answer the question by referring to the knowledge provided below and integrating your own knowledge. You must answer in a concise manner without any additional explanation.
Passages:
{passage}
Question: {question}
assistant_template:
The answer is

Figure 5: Prompt for inference.

	RAG	PRAG	DyPRAG	DistilledPRAG
F1	34.6	11.8	5.0	30.9

Table 7: Performance on LLaMA3-8B-Instruct in strong cross-document scenarios.

to 11.8 and 5.0, respectively. Such near-collapse suggests that their LoRA aggregation strategies struggle to capture the multi-hop relationships required for cross-document questions. The sharp performance gap further demonstrates that the distillation procedure in DistilledPRAG, particularly the cross-document QA design, plays a critical role in maintaining reasoning ability across dispersed evidence.

B.2 Retrieval Quality

From Figure 6, across BM25, uniCOIL (Lin and Ma, 2021), and Contriever (Lei et al., 2023), we observe a consistent pattern: performance decreases as top-k increases from 1 to 5, and drops further when passages are replaced with irrelevant documents. This confirms that retrieval relevance plays a central role in DistilledPRAG.

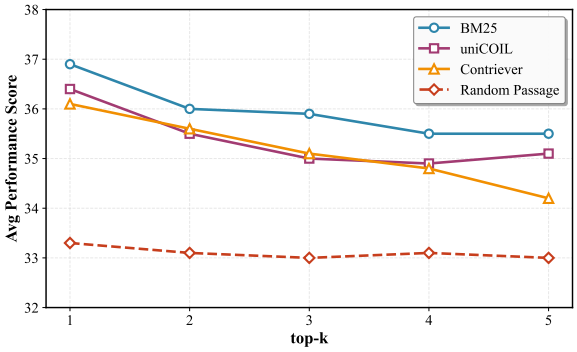


Figure 6: Performance under different search engines and different top-k values.

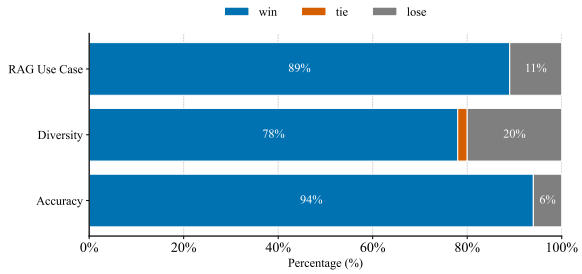
Method	2WQA				Avg
	Compare	Bridge	Inference	Compose	
PRAG	38.0	38.7	32.1	9.7	29.6
DistilledPRAG	45.7	45.8	25.1	12.5	32.3
Method	**HQA**		**PQA**	**CWQ**	**Avg**
	Bridge	Compare			
PRAG	35.0	52.1	38.2	41.3	41.7
DistilledPRAG	22.1	60.2	25.4	48.6	39.1

Table 8: Performance on Qwen-14B.

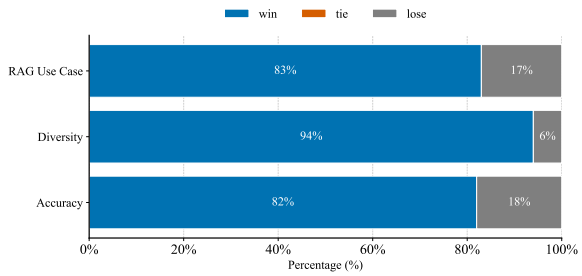
The decline with larger k likely stems from DistilledPRAG's design: all retrieved passages are merged into a single input before producing one LoRA update. Increasing k, therefore, introduces longer and noisier merged contexts, causing weakly relevant or irrelevant content to be directly encoded into parameters. Compared with standard RAG, where distractors mainly dilute attention, this parameterization is inherently more sensitive to heterogeneous document sets. Handling very long or noisy document clusters is a promising direction for future exploration. BM25 performs best because its lexical retrieval introduces fewer misleading distractors, whereas dense retrievers more often retrieve semantically similar but non-answer-bearing passages that can "pollute" the update. The irrelevant-document results further support this explanation: performance decreases but remains above collapse, indicating that DistilledPRAG retains some general reasoning ability while still reflecting the quality of retrieved evidence.

B.3 Larger Model

To explore the potential of DistilledPRAG on newer, larger, and architecturally different models, we further conduct experiments on Qwen3-14B. This evaluation allows us to examine whether the benefits observed on LLaMA3-8B generalize



(a)



(b)

Figure 7: Comparison between DeepSeek-V3 and Llama3-8B on synthetic QA quality across three dimensions: accuracy, diversity, and fitness for RAG use cases. (a) Results in the single-document setting. (b) Results in the cross-document setting.

to a stronger base model. As shown in Table 8, DistilledPRAG achieves an average performance that is nearly on par with standard RAG across all benchmarks. These results suggest that DistilledPRAG retains much of its advantage even when scaled to larger models, and that its design is broadly compatible with architectures beyond LLaMA-family models.

B.4 Quality Evaluation of Synthetic QA

In this section, we compare the quality of synthetic QA pairs generated by DeepSeek-V3 and Llama3-8B. We employ GPT-5 as the evaluator and assess the QA pairs along three dimensions: accuracy, diversity, and fitness for RAG use cases. As shown in Figure 7, DeepSeek-V3 consistently produces substantially higher-quality QA pairs than Llama3-8B in both single-document and cross-document scenarios.

Specifically, DeepSeek-V3 wins 89% / 78% / 94% of comparisons in the single-document setting and 83% / 94% / 82% in the cross-document setting. These consistently high margins indicate that DeepSeek-V3 not only generates more accurate and varied questions but also produces QA pairs that better reflect real RAG retrieval. The strong performance in cross-document assessments is particularly notable, as it suggests that DeepSeek-V3 excels at producing compositional and multi-hop reasoning questions that are crucial for robust RAG training.

B.5 Same Synthesizer and Data size

Although the previous experiments already demonstrate the effectiveness of our method, we further conduct a QA pair-controlled comparison, ensuring that all methods use the same number of QA pairs per document. We adjust the training documents to exactly match those used in DyPRAG, and we ensure that PRAG, DyPRAG, and DistilledPRAG use the same number of synthetic QA pairs per document (10 each). In our case, half of the QA pairs are used for single-document training and the other half for cross-document training. As shown in Table 9, DistilledPRAG still achieves the best performance, further indicating that the cross-document QA design and the introduction of mask tokens play a crucial role.

As shown in Table 9, DistilledPRAG continues to outperform all baselines across nearly all categories, even when the training supervision is strictly matched. PRAG and DyPRAG exhibit modest improvements when using DeepSeek-V3-generated QA pairs ("ds"), but their gains remain limited, suggesting that their LoRA aggregation and parameterization strategies do not fully exploit cross-document reasoning signals. In contrast, DistilledPRAG shows substantial and consistent improvements in both 2WQA and HQA subsets, indicating that its cross-document QA design and the use of mask tokens enable the model to better capture relational and compositional reasoning patterns. This controlled comparison further confirms that the benefits of DistilledPRAG stem from its learning mechanism rather than from differences in QA quality and quantity.

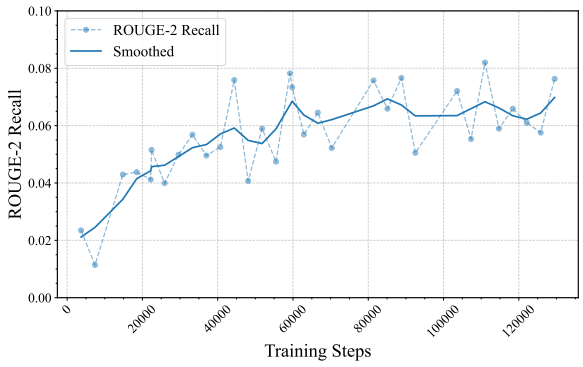
B.6 Reconstruction Attack

Although our approach prevents plaintext exposure of user documents during cloud-based inference, there remains a potential risk that an attacker could attempt to reconstruct documents from generated LoRA weights. We consider a worst-case scenario in which an attacker obtains a set of document-LoRA weight pairs. To simulate such an attack, we use 30,000 documents, selecting 100 as a test set and the rest for training a T5 decoder[4] (Guo

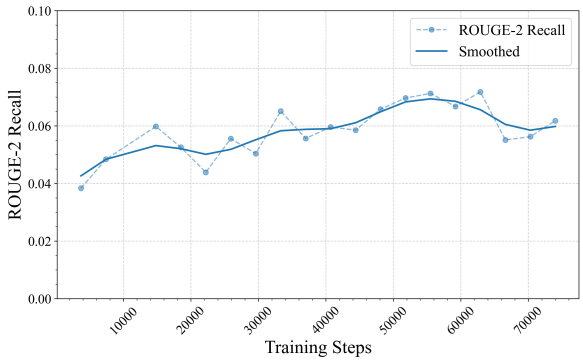
[4]https://huggingface.co/google/long-t5-tglobal-base

Method	2WQA				HQA		PQA	CWQ	Avg
	Compare	Bridge	Inference	Compose	Bridge	Compare			
PRAG	30.9	31.4	19.3	7.3	16.2	33.8	15.7	33.8	23.6
PRAG(ds)	**46.8**	**46.0**	20.5	10.1	18.3	30.8	19.3	36.4	28.5
DyPRAG	24.9	18.5	20.6	7.6	15.5	34.5	19.5	39.2	22.5
DyPRAG(ds)	33.0	19.5	20.1	9.1	16.6	42.2	10.4	35.3	23.3
DistilledPRAG	32.2	41.1	**29.6**	**17.5**	**26.3**	**47.8**	**24.3**	**46.1**	**33.1**

Table 9: Fairer Baseline Comparison on LLaMA3-8B-Instruct. The "ds" setting refers to using DeepSeek-V3 for QA synthesis.



(a)



(b)

Figure 8: Reconstruction attack results of Rouge-2 recall values for documents reconstructed at multiple checkpoints. (a) Results on T5. (b) Results on Llama.

et al., 2022) to reconstruct documents from LoRA parameters. We use ROUGE-2 recall to measure reconstruction quality. As shown in Figure 8a, even after extensive training (over 10 epochs), the reconstruction recall does not exceed 9%. We further design a stronger reconstruction attack by using Llama-3.2-1B-Instruct as the reconstruction model and feeding it the LoRA parameters, document mask token, and start token for autoregressive decoding. As shown in Fig. 8b, the stronger model does not yield a more threatening attack.

Figure 9 presents an example of document reconstruction. This represents one of the best reconstruction cases observed. In most instances, the T5-based reconstruction consists primarily of repetitive or meaningless phrases such as "Count of the Count of the...". Even in this relatively successful example, key information such as dates, names, and titles is completely incorrect. The Llama-based reconstruction consistently produces fluent text in our observations, but it still fails to recover key information. This demonstrates that our parameter generator is trained to encode abstract knowledge, rather than to directly reconstruct the original text. As a result, recovering the original document content, especially precise private information, from LoRA weights is extremely difficult. This further confirms the reliability of our method.

Figure 9: An example of a document reconstruction. Red indicates successful reconstruction, and blue indicates incorrect reconstruction.

B.7 Impact of LoRA Rank

To investigate the impact of the LoRA rank (r) on model performance, we conduct additional experiments under the LLaMA-1B setting by varying $r \in \{2, 4, 6\}$. The average F1 scores across all datasets are summarized in Table 10.

LoRA Rank (r)	Avg F1
2	28.3
4	30.3
6	30.3

Table 10: Performance across different LoRA ranks under the LLaMA-1B setting.

While individual sub-datasets exhibit slight fluctuations, the overall trend is clear: increasing the rank from $r = 2$ to $r = 4$ leads to a noticeable performance improvement. However, increasing r further to 6 does not yield additional gains, indicating a point of diminishing returns.

Importantly, we observe no noticeable increase in end-to-end inference time as the rank increases. The additional A/B matrix multiplications introduced by a larger LoRA rank contribute to less than 1% of the total LLM computation, making their impact on latency negligible relative to the full forward pass.

From an optimization perspective, increasing the rank linearly enlarges the hypernetwork output dimension (i.e., $out\_dim = r \times (h + f)$). This expansion significantly increases the parameter generation space, which makes the optimization process more challenging and can potentially affect training stability. The performance saturation observed at $r = 6$ suggests that further scaling of the rank may require improved optimization strategies to handle large-scale parameter generation. Exploring more efficient and stable generation methods for higher-dimensional adapter parameters remains an interesting direction for future work.