Decoupling Knowledge and Context: An Efficient and Effective Retrieval Augmented Generation Framework via Cross Attention

Qian Dong
dq22@mails.tsinghua.edu.cn
DCST, Tsinghua University&
Quan Cheng Laboratory
Beijing, China

Qingyao Ai*
aiqy@tsinghua.edu.cn
Quan Cheng Laboratory&
DCST, Tsinghua University
Beijing, China

Hongning Wang
hw-ai@tsinghua.edu.cn
DCST, Tsinghua University
Beijing, China

Yiding Liu
liuyiding.tanh@gmail.com
TikTok
Singapore

Haitao Li
liht22@mails.tsinghua.edu.cn
DCST, Tsinghua University
Beijing, China

Weihang Su
swh22@mails.tsinghua.edu.cn
DCST, Tsinghua University
Beijing, China

Yiqun Liu
yiqunliu@tsinghua.edu.cn
DCST, Tsinghua University
Beijing, China

Tat-Seng Chua
dcscts@nus.edu.sg
National University of Singapore
Singapore

Shaoping Ma
msp@tsinghua.edu.cn
DCST, Tsinghua University
Beijing, China

Abstract

Retrieval-Augmented Generation (RAG) systems have become a crucial tool to augment large language models (LLMs) with external knowledge for better task performance. However, existing traditional RAG methods inject knowledge directly into the context, resulting in several limitations. First, these methods highly rely on the in-context learning capability of LLMs, which often leads to excessively long contexts. This is inefficient due to the quadratic complexity of self-attention, leading to significant increase in inference time. Second, the extended context and the nature of self-attention can cause the LLMs to lose important information in the context, thereby degrading the original capabilities of LLMs. Third, the effectiveness of knowledge injection is perturbed by the permutation of knowledge within the extended context, reducing the robustness of existing RAG methods. To tackle the above problems, we propose **DecoupledRAG**, a method that decouples external knowledge from the context within the RAG framework. Specifically, we introduce a cross-attention based method that injects retrieved knowledge directly into the inference process of LLM on the fly, without modifying its parameters or the input context, so that the external knowledge can be utilized robustly in a permutation-independent manner. To the best of our knowledge, this is the first work that explore how to utilize cross-attention to inject knowledge with low training cost in decoder-only LLM era. By leveraging cross-attention operation, DecoupledRAG enables seamless knowledge aggregation without creating extended context. Experimental results demonstrate that our method could achieve high efficiency while maintaining strong performance, which indicates that RAG frameworks have the potential to benefit further from more knowledge [1].

CCS Concepts

• **Information systems** → **Retrieval tasks and goals**.

Keywords

Retrieval Augmented Generation, Language Model, Knowledge Injection

ACM Reference Format:
Qian Dong, Qingyao Ai, Hongning Wang, Yiding Liu, Haitao Li, Weihang Su, Yiqun Liu, Tat-Seng Chua, and Shaoping Ma. 2025. Decoupling Knowledge and Context: An Efficient and Effective Retrieval Augmented Generation Framework via Cross Attention. In *Proceedings of the ACM Web Conference 2025 (WWW '25), April 28-May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3696410.3714608

1 Introduction

Retrieval-Augmented Generation (RAG) has emerged as a powerful approach for enhancing the capabilities of large language models (LLMs) by injecting external knowledge into their context. RAG enables LLMs to generate correct and timely responses based on knowledge retrieved from external corpus that may not be presented to the models during their training process, significantly boosting their performance across a range of knowledge-intensive natural language processing (NLP) tasks [23].

Despite the benefits, existing RAG methods face several issues. As shown in Figure 2(a), VanillaRAG (i.e., in-context manner) directly concatenates the retrieved external knowledge with the instruction and question. Although VanillaRAG can help mitigate the issue of incorrect or outdated responses, this approach inevitably leads to excessively long context, reducing both effectiveness and efficiency. The reasons lies in the following aspects. First, due to

*Corresponding author.

WWW '25, April 28-May 2, 2025, Sydney, NSW, Australia

ACM ISBN 979-8-4007-1274-6/2 5/04
https://doi.org/10.1145/3696410.3714608

[1] The codes are released at https://github.com/Deriq-Qian-Dong/DecoupledRAG

Figure 1: Performance of Llama3-8B-Chat with VanillaRAG and DecoupledRAG on T-REx. The maximum context length of this LLM is 8k.

the quadratic complexity of self-attention [37], the processing of extended context substantially increases inference time. Second, the extended context and the nature of self-attention can cause LLMs to lose important information in the context [17, 20, 27], thereby degrading the original capabilities of LLMs, as demonstrated in Appendix A. Third, due to the *lost in the middle* issue [17, 27], the effectiveness of knowledge injection could be perturbed by the permutation of knowledge within the extended context, reducing the robustness of VanillaRAG. The "lost in the middle" phenomenon describes the tendency of LLMs to focus on the beginning and end of long sequences, often struggling to effectively process information located in the middle of context. Therefore, as shown in our preliminary experiment (see Figure 1), increasing the number of knowledge documents injected into the context significantly reduces the performance of LLMs on the T-REx [12] dataset. The issues of long texts limit the VanillaRAG framework from utilizing extensive useful knowledge efficiently and effectively [17, 20, 27]. Therefore, a natural research question is: Can we construct a new RAG paradigm in which the injected knowledge and context are decoupled?

Decoupling knowledge from context offers several advantages in the RAG system. First, it enhances efficiency by enabling the offline caching of external knowledge, allowing the LLM to efficiently utilize knowledge representations during inference online. Second, decoupling mitigates the information loss issues that often arise in long contexts [17, 20, 27]. In this work, we propose utilizing cross-attention to inject the knowledge representations into the LLM, thereby decoupling the external knowledge from the context. The knowledge could be utilized in a permutation-independent manner via cross-attention, further alleviating the information loss issue [27]. To the best of our knowledge, this is the first work that explore how to utilize cross-attention to inject knowledge with low training cost in decoder-only LLM era.

However, equipping LLMs with the capability to inject external knowledge via cross-attention is non-trivial due to the following two challenges:

Challenge 1. The training cost should not be excessively high. LLMs already contain extensive parameters, resulting in inherently high computational overhead. Retraining the entire LLM or introducing excessive additional trainable parameters is impractical.

Challenge 2. Equipping the LLM with this capability should not degrade its original capabilities. This requires that training should build on the LLM's existing capabilities incrementally, ensuring they remain unaffected.

To tackle these challenges, our DecoupledRAG framework divides the workflow into two stages: knowledge encoding and knowledge aggregation, as shown in Figure 2(b). For knowledge encoding, we use the same LLM to pre-compute and cache representations of external knowledge, ensuring its alignment with the context representations. This alignment allows the LLM to effectively utilize the pre-computed knowledge representations with minimal additional training overhead, thereby addressing Challenge 1. For knowledge aggregation, the cached representations of the external knowledge are retrieved from the database and integrated with the next token's representation via cross-attention, producing the next token representation with respect to the external knowledge. Simultaneously, the next token undergoes a self-attention operation with the context to generate the representation with respect to the internal context. We then apply the coordinate-wise summation of these representations, with the weight for external knowledge initially set to zero. This setting ensures that the injected external knowledge does not disrupt the LLM's existing capabilities at the beginning of training, addressing Challenge 2. As training progresses, LLM learns to balance the internal and external knowledge appropriately. To further reduce the introduced parameters and maintain a low training cost, we decompose the zero matrix used in representation summation into two low-rank matrices. This parameter-efficient approach further addresses Challenge 1. We conduct extensive experiments across multiple tasks to validate the effectiveness of our method. The experimental results demonstrate that our approach achieves high efficiency while maintaining strong performance, paving the way for more efficient and effective RAG systems.

To sum up, our contributions lies in the following aspects:

- We introduce DecoupledRAG, a novel RAG paradigm in which the injected knowledge and context are decoupled.
- We propose a low-cost training method that enables LLMs to effectively inject external knowledge via cross-attention without compromising their original capabilities.
- We conduct comprehensive experiments across multiple tasks, demonstrating the high efficiency and strong performance of our method.

2 Related Works

2.1 Large Language Models

Large Language Models (LLMs) have become a foundational component in natural language processing (NLP), achieving remarkable results in various tasks through a training process that typically includes pre-training, supervised fine-tuning (SFT), and preference alignment [5, 9]. One of the earliest milestones is the introduction of Transformer [37], which revolutionize NLP by introducing attention mechanism. This architecture pave the way for models such as BERT [6], GPT [28], and T5 [29], which have demonstrated state-of-the-art performance across tasks like question answering, summarization, and translation.

Figure 2: Comparison between VanillaRAG and our proposed DecoupledRAG.

As model sizes grow, LLMs like GPT-3 [3] and PaLM [4] show the power of scaling, with billions of parameters enabling models to generalize better to a variety of prompts. Despite their strengths, LLMs rely on static knowledge acquired during pre-training, which limits their ability to adapt to real-time information. Additionally, hallucination remains a concern, where the model generates con dent but incorrect or nonsensical outputs, highlighting the need for external knowledge augmentation.

2.2 Knowledge Injection in Language Models

Several recent methods have been proposed to enable e ective knowledge injection into LLMs. One common strategy involves the use of Supervised Fine-Tuning (SFT), where out-of-domain knowledge is injected by ne-tuning models like Llama-3 [1] with new datasets. This approach has demonstrated signi cant improvements in question answering accuracy, particularly in domains where the model's pre-existing knowledge is insu cient. Other approaches, such as those explored by KnowGPT [40], inject external knowledge from structured sources like knowledge graphs into LLMs, helping to mitigate hallucination and improve factual consistency. Moreover, graph neural network [10, 11, 26] is also utilized to enhance reasoning and knowledge injection in language models, further pushing the limits of context comprehension and factuality. These methods enable more precise and structured knowledge integration, providing models with the ability to reference external data e ciently. Retrieval-augmented generation (RAG) [24] has garnered signi cant attention in recent years. It extends knowledge injection by dynamically retrieving relevant documents during the generation process, making it a powerful tool for enhancing the factual accuracy and relevance of LLM outputs.

2.3 Retrieval-Augmented Generation

RAG is initially introduced by this study [24], combining the capabilities of information retrieval and generative language models. The original RAG architecture uses a Dense Passage Retrieval (DPR) [18] model to index documents and a sequence-to-sequence model (BART) to generate the responses [22].

Dense retrieval [8, 13, 25, 30, 33] is a foundational retrieval method that enables precise retrieval from large knowledge corpus like Wikipedia, and serves as the retrieval backbone for extensive RAG models [14, 24, 31, 34]. RAG leverages dense retrieval to retrieve relevant documents, conditioning generation on these documents, and improving generation quality.

Recent agent-based RAG systems, such as PaperQA [29], Graph-Reader [26], and PersonaRAG [39], further introduce specialized frameworks that combine retrieval and generation with agent-based capabilities. These systems aim to improve performance in long-context handling, reduce hallucination, and adapt to real-time user data.

3 Methodology

3.1 Preliminaries

Before delving into the details of our proposed method, we provide a formal de nition of self-attention operation, as it plays a crucial role in both the knowledge encoding and aggregation stages. Self-attention is used in LLM to compute the hidden state for generating the next token. Given a context = ›G1 • G2 • """ • G=4 the self-attention operation computes the attention scores to determine the importance of each token relative to its preceding tokens, producing a contextualized hidden state. For an LLM with layers, at each layer, 2 f1 • 2 • """ • g, the output of the previous layer 1; 1º serves as the input to the current layer, where 10º represents the output of the embedding layer. To ensure that the model captures the sequential information of the context, positional encoding operation (Pos 1 º) is incorporated into the attention mechanism. The attention mechanism itself is invariant to the ordering of tokens, which means that without Pos 1 º, the model would have no information about the position of each token. Commonly used Pos 1 º functions include sinusoidal positional encoding [37], learnable positional encoding [6], and rotary position embedding (RoPE) [32]. Formally, the self-attention output at layer is computed as

$$- \; ^{1;\,o} = \text{softmax}\left(\frac{\text{Pos}\,\&^{1;\,o}\,\text{Pos}\,^{1;\,o})^{o}}{p\overline{3.}}\right)^{!} + \; ^{1;\,o} \bullet \qquad (1)$$

where the query, key, and value matrices & ^{1; o}, ^{1; o}, and + ^{1; o} at layer ; are computed from the hidden states of the previous layer

$-^{1;\ 1^o}$. Speci cally, these matrices could be computed as

$$\&^{1;o} = -^{1;\ 1^o}, ^{1;o}\bullet \quad ^{1;o} = -^{1;\ 1^o}, ^{1;o}\bullet \quad +^{1;o} = -^{1;\ 1^o}, ^{1;o}\bullet, \tag{2}$$

Here, $\&^{1;o}$, , $^{1;o}$, and, $+^{1;o}$ are learned projection matrices at layer ; that map the hidden states of the input $^{1;\ 1^o}$ into the query, key, and value spaces, respectively. After processing through all layers of the LLM, the hidden state from the nal layer of last token $G_{=}^{1!\ o}$ is used to generate the next token. Formally, the next token $G_{=,1}$ is obtained by

$$G_{=,1} = \text{argmax}^{\ast}\text{softmax}^{\ast}G_{=}^{1!\ o}, _{\$}, 1_{\$}{}^{oo}\bullet \tag{3}$$

where, $_{\$}$ is the output projection matrix, and $1_{\$}$ is the bias term. The softmax function is applied to produce a probability distribution over the vocabulary, allowing the model to predict the next token based on the nal layer's hidden state. After generating $G_{,1}$, it is added back into the context, i.e., $= »G_1\bullet G_2\bullet"""\bullet G_{=,1}¼$.

VanillaRAG methods directly inject knowledge documents into the input context to improve the generation quality. Given a question Q and a set of corresponding retrieved documents $D^{\&} = f\ ^{\&}_1\bullet\ ^{\&}_2\bullet"""\bullet\ ^{\&}_{\#}g$, the input is formed by concatenating the question and the documents through a template, denoted as

$$-\ = T^{1}T\bullet Q\bullet D^{Qo}\bullet \tag{4}$$

where T represents task-speci c instructions. The structure of the template T and the task-speci c instructions T are de ned in the Appendix B. Then, the LLM auto-regressively predicts each next token one by one using Eq. 3. This process is repeated for each subsequent token until the complete answer is generated.

Notably, the main challenges in equipping LLM with the ability to utilize cross-attention for knowledge injection are: (1) training the LLM e ciently and (2) preserving its original capabilities. We tackle these challenges by two stages in DecoupledRAG. The framework of DecoupledRAG is depicted in Figure 3, which includes tow stages, knowledge encoding and knowledge aggregation. Next, we present the details of each stage of DecoupledRAG.

3.2 Knowledge Encoding

The knowledge encoding stage pre-computes the representations of external knowledge for use in subsequent knowledge aggregation in an on-the- y manner. To reduce training di culty, we ensure the compatibility between the external knowledge representations and the internal context representations. Therefore, we use the same LLM to encode external knowledge.

To formally de ne the knowledge encoding stage, we denote $= »3_1\bullet 3_2\bullet"""\bullet 3_<¼$ as a external knowledge sequence, where consist of $<$ tokens. At layer, of an LLM, the hidden states $^{1;o}$ is computed from the hidden states of the previous layer $^{1;\ 1^o}$ by Eq. 1. Then, we store the key-value representations of the external knowledge. Speci cally, at each layer, the key and value representations $^{1;o}_D$ and $+^{1;o}_D$ are computed as

$$^{1;o}_D = ^{1;\ 1^o}, ^{1;o}\bullet \quad +^{1;o}_D = ^{1;\ 1^o}, ^{1;o}\bullet \tag{5}$$

where, $^{1;o}$ and, $^{1;o}$ are the learned projection matrices from the same LLM. The cached key-value representations for all layers are stored as

$$K_D = f^{1}\ ^{1;o}_D\bullet +^{1;o}_D g_{,=1}{}^{"} \tag{6}$$

Then, the cached key-value representations K_D, can be directly used for knowledge aggregation. It is worth noting that when a LLM uses Grouped-Query Attention [2] for acceleration, we only need to store the key-value representations for each group, signi cantly reducing memory overhead.

3.3 Knowledge Aggregation

Once the external knowledge is encoded and cached, the next step is to inject it into the LLM through cross-attention. Since we may inject multiple external knowledge, we concatenate all the key-value representations of external knowledge before performing the cross-attention.

Speci cally, the context used in DecoupledRAG can be formed as

$$-\ = T^{1}T\bullet Q^{o}\bullet \tag{7}$$

which decouples the external knowledge $D^{\&}$ from the instruction T and the question Q. Since the subsequent formulas all focus on a speci c question Q, we omit the subscript Q in $D^{\&}$ for clarity. For multiple external knowledge $D = f\ _1\bullet\ _2\bullet"""\bullet\ _{\#}g$, we concatenate the key-value representations at layer of all $\#$ external knowledge sequences as

$$^{1;o}_{\text{ext}} = »^{1;o}_{D_1}\bullet ^{1;o}_{D_2}\bullet"""\bullet ^{1;o}_{D_{\#}}¼, \quad +^{1;o}_{\text{ext}} = »+^{1;o}_{D_1}\bullet +^{1;o}_{D_2}\bullet"""\bullet +^{1;o}_{D_{\#}}¼, \tag{8}$$

For the aggregation process, the last token's hidden state is rst integrated through the self-attention operation with the internal context representations, followed by integration with the concatenated external knowledge representations through the cross-attention operation. Formally, the self-attention operation for the last token $G_=$ at layer; can be de ned as

$$G^{1;o}_{=,\text{int}} = \text{softmax}\left(\frac{\text{Pos}^{1}\&^{1;o}_{G_=}\text{Pos}^{1}{}^{^{1;o})\ o}}{\sqrt{3_:}}\right)^{!} +^{1;o}\bullet \tag{9}$$

$G^{1;o}_{=,\text{int}}$ is the token representation with respect to internal context. Next, the representation of the last token $G_=$ undergoes a cross-attention operation with the concatenated external knowledge. This step aggregates information from the external knowledge to produce the token representation $G^{1;o}_{=,\text{ext}}$, which could be de ned as

$$G^{1;o}_{=,\text{ext}} = \text{softmax}\left(\frac{\&^{1;o}_{G_=}{}^{^{1;o})}_{\text{ext}}}{\sqrt{3_:}}\right)^{!} +^{1;o}_{\text{ext}}{}^{"} \tag{10}$$

Notably, the knowledge aggregation is permutation-independent due to the absence of $\text{Pos}^{1\ o}$ function, enabling DecoupledRAG to inject knowledge without considering the order of knowledge documents.

Finally, the hidden state of the last token at layer, is obtained by combining the self-attention and cross-attention outputs, which can be computed as

$$G^{1;\ 1^o}_{=} = G^{1;o}_{=,\text{int}}, , ^{1;o}_{V}G^{1;o}_{=,\text{ext}}\bullet \tag{11}$$

where, $^{1;o}_{V}$ is introduced as a learnable weight matrix to control the in uence of external knowledge during the aggregation of internal and external hidden states. We initialize $^{1;o}_{V}$ as a zero matrix,

Figure 3: The illustration of DecoupledRAG framework.

ensuring that at the start of training, LLM relies entirely on its internal knowledge, with the contribution from external knowledge gradually learned during fine-tuning. This initialization prevents the collapse of the LLM's original capabilities and facilitates the smooth aggregation of external knowledge. Zero initialization of $W_V^{(l,o)}$ is crucial because if external knowledge is weighted too heavily at the start of training, it could significantly disrupt the LLM's behavior. This would necessitate a complete retraining of the model, introducing excessive training costs.

Inspired by low rank adaptation [16], we further reduce the number of trainable parameters in W_V by decomposing it into two low-rank matrices and a scaling factor

$$W_V^{(l,o)} = \alpha B_V^{(l,o)} A_V^{(l,o)}, \tag{12}$$

where $B_V^{(l,o)} \in \mathbb{R}^{d \times r}$ and $A_V^{(l,o)} \in \mathbb{R}^{r \times d}$ are low-rank matrices, with $r \ll d$. $A_V^{(l,o)}$ is initialized with Gaussian noise, while $B_V^{(l,o)}$ is initialized as a zero matrix. This decomposition enables the model to learn how to integrate external knowledge efficiently, with minimal additional training costs. Therefore, the Eq. 11 could be rewritten as

$$H^{(l+1)} = H_{int}^{(l,o)} + \alpha B_V^{(l,o)} A_V^{(l,o)} H_{ext}^{(l,o)}. \tag{13}$$

The next token is obtained in the same manner as in Eq. 3.

3.4 Model Training

Both VanillaRAG and our DecoupledRAG are optimized using the next token prediction objective, commonly employed for auto-regressive language models training.

Following recent work [17], VanillaRAG is trained using the standard next token prediction objective with teacher forcing, where the model is provided with the ground truth answer during training. This method is denoted as RAG FT [17]. The input to the model is represented as $\mathcal{S} = \mathcal{P} \oplus \mathcal{A}$, where $\mathcal{P} = T(T, Q, D)$ and $\mathcal{A}$ is the ground truth answer. Formally, the training objective is to minimize the cross-entropy loss as follows

$$\mathcal{L} = -\sum_{i=1}^{|A|} \log P(\hat{a}_{i+1} = a_{i+1} \mid \mathcal{P}, a_1, a_2, \ldots, a_i), \tag{14}$$

where $\hat{a}_{i+1}$ represents the predicted token.

In contrast, our DecoupledRAG follows the same next token prediction objective but decouples the external knowledge from the input context, i.e., $\mathcal{S} = \mathcal{P} \oplus \mathcal{A}$, where $\mathcal{P} = T(T, Q)$. While the training objective remains a next-token prediction task, DecoupledRAG generates the next token based on preceding tokens and pre-cached knowledge representations $K_D = \{K_1, K_2, \ldots, K_N\}$, which can be denoted as

$$\mathcal{L} = -\sum_{i=1}^{|A|} \log P(\hat{a}_{i+1} = a_{i+1} \mid K_D, \mathcal{P}, a_1, a_2, \ldots, a_i). \tag{15}$$

3.5 Computational Complexity

In this subsection, we analyze the computational complexity of aggregating external knowledge using self-attention and cross-attention operations, respectively. Let $|D|$, $|Q|$, and $|A|$ denote the number of tokens in the external knowledge, question, and answer, respectively.

Self-Attention for External Knowledge Aggregation. When using self-attention to aggregate external knowledge, the external knowledge is treated as part of the context, together the question and answer. This results in the following computational complexity

$$\$ \ \ ^1\# \ \ jDj \ , \ j \ Qj^{o2} \ , \ j \ Aj \ \ ^1\# \ \ jDj \ , \ j \ Qj^{o} \ \bullet \qquad (16)$$

where# represents the number of external knowledge. Since jQj jDj, the Equation (16) could be simpli ed as

$$\$ \ \ ^1\# \ \ jDj^{o2} \ \ , \ \ jAj \ \ ^1\# \ \ jDj^{o} \ \ ^{\prime\prime} \qquad (17)$$

(Knowledge encoding; Answer generation; Online inference cost)

As# increases, the online inference cost of VanillaRAG grows exponentially.

Cross-Attention for External Knowledge Aggregation. In contrast, DecoupledRAG encodes knowledge documents independently from each other and from the question. This reduces the computational complexity, which can be expressed as

$$\$ \ \ \# \ \ jDj^2 \ , \ j \ Qj^2 \ , \ j \ Aj \ \ ^1\# \ \ jDj \ , \ j \ Qj^{o} \ \ ^{\prime\prime} \qquad (18)$$

Similarly, this equation could be simpli ed as

$$\$ \ \ \# \ \ jDj^2 \ \ , \ \ jAj \ \ ^1\# \ \ jDj^{o} \ \ ^{\prime\prime} \qquad (19)$$

(Knowledge encoding; Answer generation; Offline inference cost; Online inference cost)

Here, the quadratic term# jDj^2 re ects the o ine cost of encoding each external knowledge separately. This results in more e cient handling of larger external knowledge sets, as the question and external knowledge are all decoupled. Notably, the encoded knowledge is question-independent and can therefore be used for all questions, rather than being tied to a speci c one.

As# increases, DecoupledRAG demonstrates increasingly superior e ciency compared to VanillaRAG, since the online inference cost of DecoupledRAG grows linearly.

4 Experiments

In this section, we present the experiments conducted to evaluate the performance of our proposed DecoupledRAG. We rst describe the datasets and evaluation metrics used, followed by details of the experimental setup. Finally, we present the results and provide a comprehensive analysis.

4.1 Datasets and Metrics

We evaluate our proposed method across three distinct tasks, using ve datasets to comprehensively assess performance.

Multi-hop Question Answering. 2WikiMultihopQA [15] is designed to test the multi-hop reasoning capabilities across multiple Wikipedia articles, requiring models to gather and synthesize information from multiple sources to answer complex questions. The ComplexWebQuestions [35] dataset involves answering multi-step, web-based questions, further challenging the LLM's ability to retrieve and reason over large-scale web content. We use accuracy and F1 score to evaluate these tasks. Accuracy is measured as the percentage of answers that exactly match the ground truth (EM) divided by the total number of questions in the test set, while the F1 score captures the balance between precision and recall by accounting for partially correct answers.

Slot Filling. For slot- lling tasks, we evaluate our method on the Zero-Shot RE [21] and T-REx [12] datasets. Zero-Shot RE [21] is used for zero-shot relation prediction, where LLMs are tested on relations they haven't explicitly seen in training. T-REx is a large-scale factual knowledge dataset used to evaluate the LLM's ability to ll in factual slots using external knowledge from Wikipedia. For these datasets, we use both accuracy and F1 score.

Dialogue. We use the Wizard of Wikipedia (WoW) [7] dataset for dialogue tasks. In this dataset, the LLM is expected to engage in knowledgeable conversations by leveraging external knowledge retrieved from Wikipedia articles. The task tests both the knowledge integration capabilities of the LLM and its ability to maintain coherent dialogue. For this task, we use F1 score to evaluate the LLM's ability to generate relevant and correct responses during conversations.

4.2 Experimental Setup

We implement our approach on top of pre-trained LLMs, speci cally Llama3-8B-Instruct [1] and Llama2-7B-Chat [36], using the Hugging Face Transformers library. Both training and evaluation take place on up to 8 NVIDIA A100 GPUs with 40GB of memory. The training process runs for 5 epochs, with a learning rate of 1e-3 and a batch size of 16.

We use Wikipedia as the knowledge corpus. Documents are divided into non-overlapping segments, each consisting of exactly 256 tokens. Any nal segment with fewer than 128 tokens is discarded. After segmentation, the corpus comprises approximately 21 million knowledge documents. For the experiments, 1, 3, 5, 10, and 20 knowledge documents are injected to assess performance, respectively. To accelerate training and evaluation, we apply stride sampling to sample 64 tokens from each document. We use RetroMAE [38] as the retrieval module to recall relevant external knowledge. In each experiment, all baselines and our method utilize the same set of retrieved documents to ensure a fair comparison and consistent evaluation across di erent approaches.

During LLM ne-tuning, we apply LoRA with rank$A = 16$ and $U = 32$, introducing a total of 6.82M additional parameters. Similarly, our proposed DecoupledRAG also employs$A = 16$ and$U = 32$ for Eq. 13, resulting in an 4.19M additional parameters.

4.3 Experimental Results

Table 1 compares the performance of our proposed DecoupledRAG method with VanillaRAG after ne-tuning (denoted as RAG FT) across three tasks: Slot Filling, Multi-hop Question Answering, and Dialogue. We evaluate the e ectiveness of both methods by injecting 1, 3, 5, 10, and 20 external knowledge documents. From this table, we can draw the following ndings:

DecoupledRAG demonstrates superior performance with more knowledge documents injected. This performance gain can be attributed to DecoupledRAG's ability to avoid increasing the context length while e ectively aggregating external knowledge. Consequently, the LLM bene ts from external

Table 1: Performance comparison between RAG FT and DecoupledRAG. The best performances are highlighted in bold.

# Docs	Method	Slot Filling				Multi-hop Question Answering				Dialogue
		Zero-Shot RE		T-REx		2WikiMultihopQA		ComplexWebQuestions		WoW
		Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	F1
		Llama-3-8B-Instruct								
+1 doc	RAG FT	38.1	49.9	61.2	63.2	17.2	12.1	23.4	31.9	22.3
	DecoupledRAG	28.6	37.1	61.1	63.9	15.2	11.1	9.8	16.8	24.4
+3 doc	RAG FT	39.9	51.3	69.3	72.8	29.5	34.3	15.3	21.3	22.0
	DecoupledRAG	45.4	53.1	73.7	76.2	28.1	32.5	29.2	37.0	25.5
+5 doc	RAG FT	25.0	29.9	35.7	37.4	26.3	31.4	10.1	17.5	18.9
	DecoupledRAG	49.0	56.3	76.3	78.3	29.9	34.3	33.8	41.8	25.9
+10 doc	RAG FT	22.4	28.0	30.2	32.0	13.1	23.4	12.5	19.6	19.3
	DecoupledRAG	49.2	56.9	78.4	80.3	32.7	37.6	37.6	45.1	25.3
+20 doc	RAG FT	18.1	24.8	25.3	27.4	3.8	9.6	15.7	24.0	19.3
	DecoupledRAG	50.3	57.5	80.2	81.9	33.4	38.1	39.6	47.1	26.1
		Llama-2-7B-Chat								
+1 doc	RAG FT	3.7	5.1	18.1	19.2	13.1	18.9	11.0	14.7	18.5
	DecoupledRAG	2.2	4.1	11.8	13.0	13.2	19.3	10.5	14.0	17.3
+3 doc	RAG FT	3.1	4.6	18.3	19.5	15.5	20.7	12.5	15.9	19.2
	DecoupledRAG	4.2	5.6	17.4	18.7	16.6	21.8	12.2	15.8	17.3
+5 doc	RAG FT	3.4	4.8	17.7	18.9	12.2	16.5	9.8	12.4	15.6
	DecoupledRAG	4.2	5.5	19.0	20.1	17.4	22.6	13.4	16.7	17.8
+10 doc	RAG FT	2.5	4.2	6.2	7.4	6.5	8.2	5.1	6.3	7.4
	DecoupledRAG	4.6	5.9	18.5	19.7	17.7	23.1	13.1	16.5	18.2
+20 doc	RAG FT	1.7	3.8	6.9	7.3	3.2	4.0	2.3	3.0	3.6
	DecoupledRAG	4.8	6.1	20.6	21.8	18.2	23.5	13.8	17.0	20.0

(a) Zero-Shot RE (b) T-REx (c) 2WikiMultihopQA (d) ComplexWebQuestions (e) WoW

Figure 4: Performance of Llama-3-8B-Instruct across datasets

(a) Zero-Shot RE (b) T-REx (c) 2WikiMultihopQA (d) ComplexWebQuestions (e) WoW

Figure 5: Performance of Llama-2-7B-Chat across datasets

knowledge to generate accurate responses while preserving its instruction following capability.

When a limited number of external knowledge documents are injected, RAG FT performs slightly better than DecoupledRAG. The reason lies in that self-attention provides more comprehensive interaction across the entire context. The knowledge representations could aggregate information from the instruction and the question, making LLM can e ectively follow the instruction and focus on the question. RAG FT requires a trade-o between the number of external knowledge and the length of internal context. The optimal

number of injected knowledge documents varies across different datasets and models. For example, with Llama-3-8B-Instruct, the best performance is observed with two injected documents in Zero-Shot RE, T-REx, and 2WikiMultihopQA, while three documents yield the highest results in ComplexWebQuestions. In WoW, RAG FT achieves best performance with just one injected document.
- Overall, Llama-3-8B-Instruct outperforms Llama2-7B-Chat consistently across all datasets, owing to its stronger foundational capabilities and enhanced ability to handle longer contexts effectively.

To facilitate the analysis of trends, we present Figure 4 and 5, which compare the performance of DecoupledRAG and RAG FT across multiple datasets using Llama-3-8B-Instruct and Llama-2-7B-Chat, respectively. Since the trends for Accuracy and F1 are consistent in the Slot Filling and Multi-hop Question Answering tasks, we only present the Accuracy results. From these figures, we can draw several key observations:

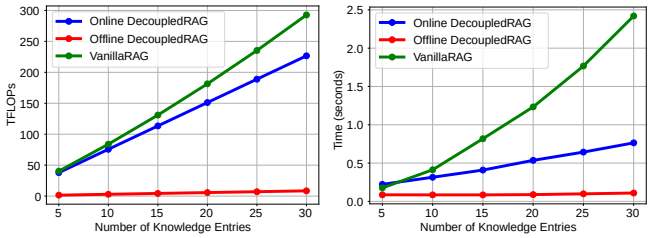
- Compared to RAG FT, DecoupledRAG shows a steady improvement in performance as more external knowledge documents are injected. This upward trend stems from DecoupledRAG's ability to decouple external knowledge from the context, ensuring that injecting additional knowledge does not overwhelm the important information in the context. However, with few external documents, the performance of DecoupledRAG is limited due to cross-attention interactions being less comprehensive than self-attention.
- In contrast, RAG FT experiences a significant drop in performance as the number of injected documents increases, particularly in tasks like Zero-Shot RE and ComplexWebQuestions. This decline further demonstrates the self-attention mechanism in RAG FT is inefficient with longer contexts, hindering the LLM' capability to utilize important information in instruction and knowledge.
- DecoupledRAG consistently outperforms the highest performance of RAG FT across all datasets, demonstrating its superior ability to effectively leverage knowledge without compromising the essential information in the context.

4.4 Efficiency Analysis

In this section, we compare the efficiency of DecoupledRAG and VanillaRAG. Based on Equations (17) and (19), the efficiency of answer generation is identical for both methods; therefore, we focus on comparing their efficiency in knowledge encoding. DecoupledRAG offers two configurations: the first encodes knowledge online without pre-caching, referred to as Online DecoupledRAG, while the second pre-encodes knowledge, referred to as Offline DecoupledRAG. In contrast, VanillaRAG encodes all knowledge into the context online. We compared the relationship between these methods' FLOPs and running time with the number of injected knowledge entries. The experimental results are illustrated in Figure 6, from which the following conclusions can be drawn.

Online DecoupledRAG demonstrates a significant advantage over VanillaRAG during the knowledge encoding phase, due to the decoupling of knowledge entries. While the inference time for knowledge encoding in VanillaRAG grows exponentially with the number of knowledge entries, it increases linearly in Online DecoupledRAG. This disadvantage of VanillaRAG mainly stems from putting all knowledge entries into the context window, resulting in very long contexts. In contrast, DecoupledRAG encodes knowledge entries separately, avoiding lengthy contexts and thus alleviating the efficiency bottleneck of self-attention. Similarly, lengthy contexts in VanillaRAG also results in higher TFLOPs requirements when encoding the same number of knowledge entries.

Furthermore, the ability to pre-cache knowledge allows Offline DecoupledRAG to exhibit minimal growth in time overhead as the number of knowledge entries increases. However, this approach requires substantial storage space to pre-store the key-value representations of knowledge, which is a notable limitation. In practical applications, it is necessary to balance storage and inference overhead, such as by caching only frequently accessed knowledge. Despite this limitation, Figure 6 clearly demonstrates that Online DecoupledRAG retains a significant efficiency advantage over VanillaRAG without incurring additional storage overhead.



(a) FLOPs vs Number of Knowledge Entries **(b) Running Time vs Number of Knowledge Entries**

Figure 6: Performance comparison of different RAG approaches.

5 Conclusion

In this work, we present DecoupledRAG, a novel framework designed to address the inherent limitations of traditional context-based RAG systems. By decoupling external knowledge from the context and utilizing cross-attention for knowledge injection, DecoupledRAG mitigates the issues related to long context, such as increased inference latency and degradation of fundamental capabilities. Besides, DecoupledRAG is more robust to the permutation of knowledge, as the knowledge is injected in a permutation-independent manner. Our approach enables efficient external knowledge aggregation while preserving the original capabilities of LLMs, ensuring robust performance across various tasks. Extensive experiments across multiple datasets demonstrate that DecoupledRAG not only maintains high efficiency but also achieves superior performance.

Acknowledgments

This research is part of NExT++ project, supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@Singapore Funding Initiative. This work is also supported by Quan Cheng Laboratory (Grant No.QCLZD202301).

References

[1] AI@Meta. 2024. Llama 3 Model Card. (2024). https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[2] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245* (2023).

[3] Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[4] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.

[5] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30 (2017).

[6] Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241* (2018).

[8] Qian Dong, Yiding Liu, Qingyao Ai, Haitao Li, Shuaiqiang Wang, Yiqun Liu, Dawei Yin, and Shaoping Ma. 2023. Iˆ 3 Retriever: Incorporating Implicit Interaction in Pre-trained Language Models for Passage Retrieval. *arXiv preprint arXiv:2306.02371* (2023).

[9] Qian Dong, Yiding Liu, Qingyao Ai, Zhijing Wu, Haitao Li, Yiqun Liu, Shuaiqiang Wang, Dawei Yin, and Shaoping Ma. 2024. Unsupervised large language model alignment for information retrieval via contrastive feedback. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 48–58.

[10] Qian Dong, Yiding Liu, Suqi Cheng, Shuaiqiang Wang, Zhicong Cheng, Shuzi Niu, and Dawei Yin. 2022. Incorporating explicit knowledge in pre-trained language models for passage re-ranking. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1490–1501.

[11] Qian Dong and Shuzi Niu. 2021. Latent Graph Recurrent Network for Document Ranking. In *Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11–14, 2021, Proceedings, Part II 26*. Springer, 88–103.

[12] Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[13] Yan Fang, Jingtao Zhan, Qingyao Ai, Jiaxin Mao, Weihang Su, Jia Chen, and Yiqun Liu. 2024. Scaling laws for dense retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1339–1349.

[14] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*. PMLR, 3929–3938.

[15] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060* (2020).

[16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* (2021).

[17] Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O Arik. 2024. Long-Context LLMs Meet RAG: Overcoming Challenges for Long Inputs in RAG. *arXiv preprint arXiv:2410.05983* (2024).

[18] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).

[19] Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. 2023. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv preprint arXiv:2312.07559* (2023).

[20] Quinn Leng, Jacob Portes, Sam Havens, Matei Zaharia, and Michael Carbin. 2024. Long Context RAG Performance of LLMs. https://www.databricks.com/blog/long-context-rag-performance-llms Published in Mosaic AI Research. Accessed: 2024-10-13.

[21] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115* (2017).

[22] M Lewis. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[23] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[24] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[25] Haitao Li, Qingyao Ai, Xinyan Han, Jia Chen, Qian Dong, Yiqun Liu, Chong Chen, and Qi Tian. 2024. DELTA: Pre-train a Discriminative Encoder for Legal Case Retrieval via Structural Word Alignment. *arXiv preprint arXiv:2403.18435* (2024).

[26] Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, et al. 2024. GraphReader: Building Graph-based Agent to Enhance Long-Context Abilities of Large Language Models. *arXiv preprint arXiv:2406.14550* (2024).

[27] 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.

[28] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[29] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research* 21, 140 (2020), 1–67.

[30] Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. *arXiv preprint arXiv:2110.07367* (2021).

[31] Ruiyang Ren, Yuhao Wang, Yingqi Qu, Wayne Xin Zhao, Jing Liu, Hao Tian, Hua Wu, Ji-Rong Wen, and Haifeng Wang. 2023. Investigating the factual knowledge boundary of large language models with retrieval augmentation. *arXiv preprint arXiv:2307.11019* (2023).

[32] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing* 568 (2024), 127063.

[33] Weihang Su, Qingyao Ai, Xiangsheng Li, Jia Chen, Yiqun Liu, Xiaolong Wu, and Shengluan Hou. 2023. Wikiformer: Pre-training with Structured Information of Wikipedia for Ad-hoc Retrieval. *arXiv preprint arXiv:2312.10661* (2023).

[34] Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. DRAGIN: Dynamic Retrieval Augmented Generation based on the Real-time Information Needs of Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 12991–13013. https://doi.org/10.18653/v1/2024.acl-long.702

[35] Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643* (2018).

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

[37] A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* (2017).

[38] Shitao Xiao, Zheng Liu, Yingxia Shao, and Zhao Cao. 2022. RetroMAE: Pre-training retrieval-oriented language models via masked auto-encoder. *arXiv preprint arXiv:2205.12035* (2022).

[39] Saber Zerhoudi and Michael Granitzer. 2024. PersonaRAG: Enhancing Retrieval-Augmented Generation Systems with User-Centric Agents. *arXiv preprint arXiv:2407.09394* (2024).

[40] Qinggang Zhang, Junnan Dong, Hao Chen, Xiao Huang, Daochen Zha, and Zailiang Yu. 2023. Knowgpt: Black-box knowledge injection for large language models. *arXiv preprint arXiv:2312.06185* (2023).

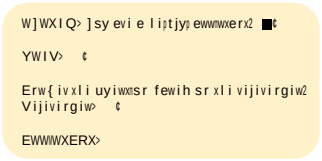


Figure 7: The template $\mathcal{T}$ used in our experiments.

Table 2: The task-specific instructions.

	Instruction
Zero-Shot RE	Please answer user question to the best of your ability. The answer MUST in ONE OR FEW WORDS.
T-REx	Please fill in the [MASK] in the sentence.
2WikiMultihopQA	Answer the user question that require reasoning over multiple Wikipedia articles. The answer MUST in ONE OR FEW WORDS.
ComplexWebQuestions	Answer the user question that require reasoning over multiple Wikipedia articles. The answer MUST in ONE OR FEW WORDS.
WoW	Integrating knowledge from Wikipedia to improve the informativeness of your answers.

A Failure Analysis for VanillaRAG

In this section, we present the failure analysis of VanillaRAG on the T-REx dataset using Llama-3-8B-Instruct. Specifically, we examine 400 randomly selected cases where the LLM provides correct answer with 1 injected document but fails when 20 documents are injected into its context. Through manual analysis, the failures can be categorized into three types:

- **Failure to Follow Instruction.** The T-REx instruction prompts the LLM to fill the `[MASK]` token in the given sentence based on the retrieved references. However, as the context length increases, the LLM often loses focus on the instruction and tends to either continue writing the context or summarize context. Among the 400 cases, 187 cases fall into this category.
- **Wrong Answer.** This issue arises because long context degrades the in-context learning capability of LLMs, even though the retrieved references are highly relevant to the question. LLM generates incorrect answers in 74 cases.
- **Meaningless Content.** Long contexts collapse the foundational capabilities of LLMs, resulting in the generation of random segments. This issue is observed in 139 cases.

From this analysis, it can be concluded that long contexts hinder the performance of VanillaRAG due to a decline in instruction-following, in-context learning, and other foundational capabilities in LLMs, further highlighting the limitations of directly injecting knowledge into the context.

B Template and Task-specific Instructions

The template $\mathcal{T}(\mathsf{T}, \mathsf{Q}, \mathcal{D})$ used in our experiments is presented in Figure 7. The task-specific instructions are shown in Table 2.

C Comparison with Long-Context RAG Methods

The most relevant work to ours is a recent study [17] published by Google, which provides an in-depth analysis of the challenges faced by RAG w.r.t. long-context. The study also proposes three potential solutions to address these issues.

Retrieval Reordering (RR). This is a training-free method proposed to address the *lost in the middle* issue. By reordering the knowledge documents based on their relevance scores to the question, the most relevant documents are placed at the beginning and end of the context.

RAG Fine-Tuning (RAG FT). This method aims to improve LLM robustness to hard negatives by training them with retrieved knowledge documents, which is identical to the settings of RAG FT in our experiments.

RAG FT with an intermediate reasoning (RAG FT w/. Int). RAG FT w/. Int explicitly trains the LLM to differentiate between relevant and irrelevant passages within the retrieved context by generating a reasoning paragraph. However, the inference time overhead is significantly scaling up due to the additional reasoning generation, making the comparison unfair. Therefore, this method is not included in our comparison.

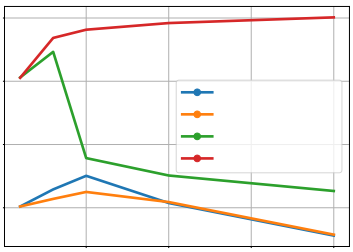


Figure 8: Comparison of different long-context RAG methods on the T-REx dataset.

The results are presented in Figure 8. Since LLMs are sensitive to the position of knowledge documents, RR exhibits unstable performance on the T-REx dataset, especially when the number of documents is limited, demonstrating that LLM performance is easily perturbed by the permutation of knowledge. RAG-oriented fine-tuning is an effective method for improving performance, but as the number of documents increases (e.g., greater than 5), performance declines sharply, which highlights the limitation of long context in LLMs. Our DecoupledRAG utilizes knowledge in a permutation-independent manner while avoiding excessively long contexts, thereby achieving superior performance.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009