

DuoPRAG: Rank-Disentangled and Gated Knowledge-Reasoning Fusion

Nakhyun Choi
Jeonbuk National University
South Korea
skrgusrnt@jbnu.ac.kr

Cheoneum Park[†]
Hanbat National University
South Korea
parkce@hanbat.ac.kr

Seung-hoon Na[†]
UNIST
South Korea
nash@unist.ac.kr

Abstract

We rethink parametric retrieval-augmented generation with *DuoPRAG*, which integrates knowledge and reasoning capabilities through rank-disentangled and gated Low-Rank Adaptation (LoRA) composition. Existing parametric RAG methods encode document knowledge into LoRA parameters while leaving reasoning to be supplied through lexical chain-of-thought (CoT) prompting. This introduces three failure modes in outputs and contradicts the prompt-independent inference that parametric RAG aims to achieve. DuoPRAG parameterizes knowledge and reasoning into disentangled LoRA adapters, fused through a learnable mask and rank-specific gates. We achieve state-of-the-art performance on multi-hop QA benchmarks, outperforming the strongest parametric RAG baseline while preserving the parametric inference framework. Our results demonstrate that jointly parameterizing knowledge and reasoning is a natural and effective extension of the parametric RAG paradigm. Our code is available at <https://github.com/naknak-choi/DuoPRAG>.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Chowdhery et al., 2023; Rae et al., 2021; Chung et al., 2024) have shown remarkable performance on knowledge-intensive tasks such as QA benchmarks (Kwiatkowski et al., 2019; Joshi et al., 2017; Yang et al., 2018; Ho et al., 2020; Talmor and Berant, 2018), but retraining them to incorporate up-to-date knowledge incurs prohibitive cost. To address this, recent work proposes efficient alternatives that inject new knowledge into LLMs without full retraining, such as Retrieval-Augmented Generation (RAG; Lewis et al. (2020)) and low-rank adapter (LoRA) fine-tuning (Hu et al., 2022). Despite these advantages,

[†]Corresponding authors.

(a) Training prompt structure

Instruction: “You should reference the knowledge provided below...”
Few-shot CoT exemplars (8):
Q: When did the director of film *Hypocrite* die?
A: The film *Hypocrite* was directed by **Miguel Morayta**. Miguel Morayta died on 19 June 2013. So the answer is 19 June 2013.
... (7 more exemplars)
Retrieved passage + CoT trigger: “Passage 1: ... Let’s think step by step.”
Q: [Test question]
A: [Model generation]

(b) Observed failure modes

Over-generation. Q: Which film has the director born first, A or B?
A: ... So the answer is A. **Q: Who is the director of film...? A: ...**
Exemplar entity leak. Q: Which film has the director died first, ... or *The Working Class Goes To Heaven*?
A: The Working Class Goes To Heaven was directed by **Miguel Morayta**[†].
Format mismatch. Q: Which film has the director died earlier...? (expects film name)
A: ... So the answer is **1 February 1983** (date).
[†] Miguel Morayta appears only in the few-shot exemplars, not in the retrieved passage.

Figure 1: Parametric RAG jointly trains a lexical CoT prompt into a single LoRA (a), leading to three failure modes in outputs (b).

RAG suffers from a well-known issue known as *knowledge conflict*, where the LLM’s parametric knowledge contradicts the retrieved passages (Xu et al., 2024). To mitigate this, Parametric RAG has emerged as an active line of research, where retrieved documents are transformed into low-rank adapters and injected directly into the model’s parameters (Su et al., 2025a; Tan et al., 2025; Charakorn et al., 2026).

However, existing parametric RAG methods (Su et al., 2025a; Tan et al., 2025; Su et al., 2025b; Charakorn et al., 2026) train each per-document LoRA on a prompt that contains not

only the retrieved passages but also lexical CoT: chain-of-thought supplied as surface text, namely a trigger phrase (“Let’s think step by step”) and few-shot CoT exemplars (Fig. 1(a)). Since the loss is computed over the entire CoT-answer sequence, the same low-rank parameters are optimized to support both document-specific factual knowledge and procedural reasoning behavior. This lexical CoT prompting induces *CoT demonstration sensitivity* (Fu et al., 2023; Wang et al., 2023), causing substantial performance degradation when post-processing is not applied to the model’s raw outputs (see Table 1). This indicates that parametric knowledge injection alone cannot address CoT demonstration sensitivity. Beyond this performance degradation, the same lexical CoT prompts can also leak exemplar entities into the output (Fig. 1(b)), inducing conflicts with the injected or internal parametric knowledge (Kortukov et al., 2024). Taken together, these limitations motivate equipping reasoning capability with its own parametric module, trained separately from injected knowledge.

In this work, we present DuoPRAG, a method that integrates knowledge and reasoning capabilities through two separately trained LoRA modules: a knowledge LoRA derived from retrieved passages and a reasoning LoRA that operates over K latent reasoning tokens. Inspired by DuoLoRA (Roy et al., 2025), we fuse the two modules through a rank-wise gating mechanism beyond simple summation, allowing each rank dimension to be routed toward knowledge or reasoning as needed. This design enables query-only inference without placing retrieved passages and lexical CoT in the prompt, retaining multi-step reasoning through latent reasoning tokens. Our contributions are summarized as follows:

- We empirically revisit Parametric RAG, identifying three failure modes of lexical CoT reasoning entangled with knowledge in a single LoRA.
- We propose DuoPRAG, which removes the reliance on lexical CoT at inference and remains robust across prompt protocols, mitigating CoT demonstration sensitivity.
- Extensive experiments across three backbones and QA benchmarks show that DuoPRAG achieves higher average performance than prior parametric RAG methods on

reasoning-intensive QA while revealing limitations on knowledge-intensive OOD tasks.

2 Related Work

Hypernetwork-based parametric RAG. Parametric RAG (PRAG) (Su et al., 2025a) encodes retrieved documents as LoRA-form parameter deltas in the LLM’s feed-forward layers, but requires per-document offline adapter training. Recent work amortizes this cost via hypernetworks that map documents to LoRA weights at inference time (Tan et al., 2025; Charakorn et al., 2026). However, these approaches entangle reasoning and knowledge within a single LoRA and elicit reasoning externally through lexical CoT prompts, making multi-hop performance depend on prompt formatting rather than the adapter itself. We instead introduce a dedicated parametric reasoning module that composes with the knowledge adapter through learned fusion, removing the need for lexical CoT prompting.

Latent reasoning. Lexical CoT verbalizes intermediate reasoning steps, increasing generation length and sensitivity to prompt demonstrations. Latent-reasoning methods such as SIM-CoT (Wei et al., 2026) instead represent intermediate computation through continuous hidden states. DuoPRAG adopts this general principle to construct a reusable reasoning LoRA, but focuses on separating and fusing it with dynamically generated document-specific knowledge adapters in parametric RAG.

Rank-level LoRA composition. For combining two LoRAs with distinct functional roles without interference, DuoLoRA (Roy et al., 2025) merges content and style adapters in text-to-image diffusion. It shows that LoRA’s rank dimension can be partitioned to allocate distinct functional roles, and proposes a learnable rank-axis mask (ZipRank) that suppresses interference between the two attributes. Inspired by this insight of role separation along the rank dimension, we extend the idea to LLM-based knowledge-intensive QA. Our method decomposes the adapter into a per-document knowledge LoRA W_{knw} and a shared reasoning LoRA W_{rsn} , and augments the learnable mask with a token-conditional, rank-specific gate that modulates each rank’s contribution based on the input.

3 Analysis of PRAG’s Lexical Reasoning Supervision

Prompt Structure and Failure Modes. PRAG trains a single per-document LoRA using CoT-conditioned QA examples. Its prompt contains few-shot CoT exemplars, a retrieved passage, and a lexical trigger (Fig. 1(a)), while the target contains the rationale and final answer. The same adapter must therefore support both document-specific knowledge and procedural reasoning. Separately, exposing reasoning as lexical output introduces over-generation, exemplar entity leakage, format mismatch, and dependence on answer extraction (Fig. 1(b)).

From lexical to latent reasoning. Table 1 quantifies these output failures. Since PRAG verbalizes its reasoning as a lexical CoT, it depends heavily on answer-extraction post-processing: removing it sharply lowers PRAG’s average F1. These observations motivate us to relocate reasoning away from the lexical output space. Inspired by SIM-CoT, DuoPRAG shifts reasoning to the latent space via K latent reasoning tokens and generates only the final answer, completely eliminating these failure modes.

Why learned fusion? Separate knowledge and reasoning LoRAs do not guarantee distinct representations. Using linear Centered Kernel Alignment (CKA) (Kornblith et al., 2019) to compare token-to-token similarity structures, we find that under naive addition of independently trained LoRAs, the two branches exhibit high final-layer overlap (0.91 averaged across four datasets) despite near-zero parameter cosine similarity: the modules are parametrically orthogonal yet functionally redundant. This redundancy indicates that simple summation is insufficient and fusion must be learned. Indeed, DuoPRAG reduces the CKA to 0.031 while achieving the highest average F1, although we treat this correlation as diagnostic evidence rather than a causal claim.

4 Method

4.1 Preliminary

Notation. We build on the parametric RAG methods (Su et al., 2025a; Tan et al., 2025; Su et al., 2025b), which encodes retrieved passages into LoRA adapters attached to a frozen LLM. Let q denote the input question, $a = (a_1, \dots, a_{|a|})$ the answer token sequence, and $P = \{p_1, \dots, p_N\}$

	2WQA	HQA	PQA	CWQ	Avg.
<i>F1 with vs. without answer-extraction post-processing</i>					
PRAG (w/ post-proc.)	22.89	30.37	21.56	37.04	27.97
PRAG (w/o post-proc.)	4.95	4.01	16.12	31.85	14.23
DuoPRAG (w/ post-proc.)	37.17	29.26	30.67	46.08	35.79
DuoPRAG (w/o post-proc.)	38.44	29.66	30.67	46.54	36.33
<i>Failure mode prevalence (%) on raw outputs</i>					
Over-generation					
PRAG	90.50	97.67	—	—	94.09
DuoPRAG	0.00	0.00	—	—	0.00
Exemplar entity leak					
PRAG	58.25	5.17	—	—	31.71
DuoPRAG	0.17	0.00	—	—	0.08
Format mismatch					
PRAG	13.00	0.50	—	—	6.75
DuoPRAG	0.25	0.50	—	—	0.38

Table 1: Failure-mode analysis on Llama-3.2-1B. Only 2WQA/HQA use CoT exemplars, and only by PRAG.

the set of retrieved passages. The frozen LLM $\mathcal{M}$ is parameterized by θ . We write $(l, t) \in \mathcal{T}$ for an MLP projection module of $\mathcal{M}$, where l indexes a transformer layer and $t \in \{\text{gate}, \text{up}, \text{down}\}$ the projection type; $x_\tau \in \mathbb{R}^{d_t^{\text{in}}}$ denotes the hidden states entering module (l, t) at token position j .

A LoRA adapter is the low-rank additive update with rank r :

$$\Delta W = BA, \quad A \in \mathbb{R}^{r \times d_{\text{in}}}, \quad B \in \mathbb{R}^{d_{\text{out}} \times r}, \quad (1)$$

Under this formulation, parametric RAG decodes the answer as

$$p(a \mid q, P) = p_{\theta + \Delta W_P}(a \mid q), \quad \Delta W_P = \Phi(P), \quad (2)$$

where ΔW_P is a passage-conditioned LoRA adapter, and Φ denotes any parametric RAG.

DuoPRAG. DuoPRAG inserts *two* adapters, indexed by $b \in \{\text{knw}, \text{rsn}\}$, into every MLP target (l, t) . Each LoRA is kept in factored form of Eq. (1), $\Delta W_b^{(l,t)} = B_b^{(l,t)} A_b^{(l,t)}$, with rank r_b , scale $\alpha_b \in \mathbb{R}_{>0}$, and module-specific input/output dimensions $d_t^{\text{in}}, d_t^{\text{out}}$. The *knowledge adapter* W_{knw} is obtained from $\Phi(P)$ and injects retrieved evidence at the parameter level (§4.2); the *reasoning adapter* W_{rsn} is independent of P and is trained to inject an implicit reasoning trajectory through a few latent tokens (§4.3). The two adapters are composed by a rank-level mask-gate mechanism (§4.4) and added to the frozen base output. The number of latent reasoning tokens used by the reasoning LoRA is K , with $K = 4$ in our setting. For clarity, the indices (l, t) are made explicit only when needed.

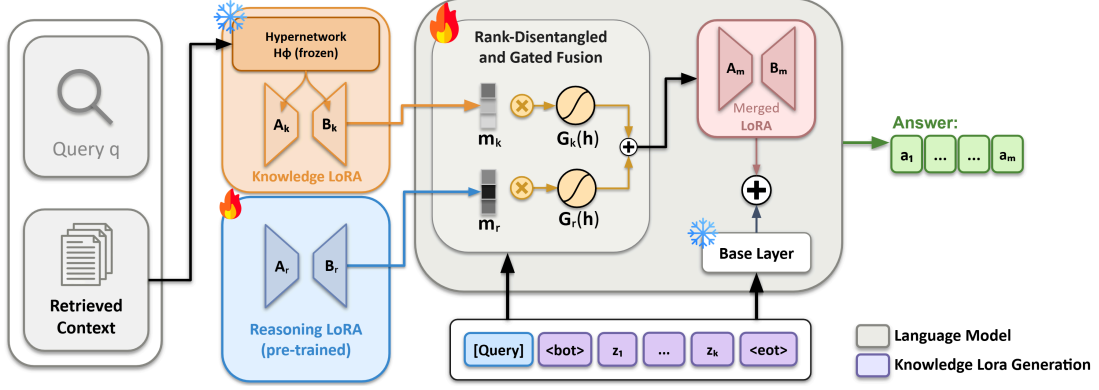


Figure 2: DuoPRAG overall architecture. Retrieved passages are converted into Knowledge LoRA parameters, while the Reasoning LoRA conditions the backbone to generate latent reasoning tokens for answer decoding without inserting passage text into the prompt.

4.2 Knowledge LoRA Branch

The knowledge LoRA W_{knw} encodes retrieved passages P into the MLP modules of the base LLM. DuoPRAG is agnostic to Φ : any parametric RAG encoder that produces LoRA factors in Eq. (1) can serve as Φ . At each module $(l, t) \in \mathcal{T}$, Φ maps each passage p_i to a factor pair

$$(A_{\text{knw},i}, B_{\text{knw},i}) = \Phi(p_i), \quad i = 1, \dots, N. \quad (3)$$

The N passage factor pairs are then merged at each module by factor-wise averaging, which preserves the factored form required by the rank-level composition (§4.4).

$$A_{\text{knw}} = \frac{1}{N} \sum_{i=1}^N A_{\text{knw},i}, \quad B_{\text{knw}} = \frac{1}{N} \sum_{i=1}^N B_{\text{knw},i}, \quad (4)$$

Different instantiations of Φ follow Eq. (3); in this work, we adopt Tan et al. (2025) as the default instantiation and evaluate the plug-in property in Table 4.

4.3 Reasoning LoRA Branch

The reasoning adapter W_{rsn} is independent of P and aims to inject an implicit reasoning trajectory through the same MLP modules. Drawing insight from Wei et al. (2026), we represent reasoning not as discrete chain-of-thought tokens but as K continuous latent vectors inserted between the question and the answer.

Latent reasoning sequence. Given a question-answer pair (q, a) , the training sequence is constructed as

$$\text{seq}(q, a) = [q; \langle \text{bot} \rangle; z_1; \dots; z_K; \langle \text{eot} \rangle; a], \quad (5)$$

where $\langle \text{bot} \rangle, \langle \text{eot} \rangle$ are learnable boundary tokens and $z_1, \dots, z_K \in \mathbb{R}^d$ are continuous latent vectors, which are not vocabulary tokens, produced autoregressively by the W_{rsn} -adapted base LLM. At step k , define the prefix $\text{pre}_k = [q; \langle \text{bot} \rangle; z_1; \dots; z_{k-1}]$. The k -th latent token is then generated as

$$z_k = \text{prj}(h_{k-1}), \quad h_{k-1} = \mathcal{M}_{W_{\text{rsn}}}(\text{pre}_{k-1})[-1], \quad (6)$$

for $k = 1, \dots, K$, where $[-1]$ denotes the *last-position* hidden state. After the K latents, $\langle \text{eot} \rangle$ is appended to form $[q; \langle \text{bot} \rangle; z_1; \dots; z_K; \langle \text{eot} \rangle]$, from which the answer is decoded. The projector $\text{prj}(h) = W_2 \text{GELU}(W_1 h)$ is a 2-layer MLP with $W_1, W_2 \in \mathbb{R}^{d \times d}$. In this way, W_{rsn} aligns the base LLM’s MLP outputs to serve as implicit reasoning steps rather than learning a direct $q \rightarrow a$ mapping.

Step-level supervision. To align each latent token z_k with a semantically meaningful reasoning step, we train an auxiliary decoder D_η that takes z_k as a *soft-prompt prefix* and autoregressively generates the corresponding k -th reasoning step text. D_η is initialized from $\mathcal{M}$ but trained with *independent weights*: only a PEFT LoRA (Mangrulkar et al., 2022) attached to $\{q_{\text{proj}}, v_{\text{proj}}\}$ is updated, ensuring that gradients on z_k do not propagate back into the frozen base θ . D_η is used solely during training and is discarded at inference.

4.4 Dual-LoRA Fusion: Rank Duo Mask $\times$ Rank-Specific Gate

When both W_{knw} and W_{rsn} are attached to the same MLP module and share its output, DuoPRAG composes them through two complementary mechanisms operating in the rank space of

each branch $b \in \{\text{knw}, \text{rsn}\}$.

Rank Duo Mask. For each layer l , we attach two learnable mask vectors $M_b^{(l)} \in \mathbb{R}^{r_b}$, each initialized to one. The mask is shared across MLP targets $t \in \mathcal{T}$ and across token positions, providing a static, branch-aware allocation over rank dimensions.

Rank-Specific Gate. In contrast to the mask, the gate is attached per MLP module: For each $(l, t) \in \mathcal{T}$, we further attach a linear gate head $W_{g,b}^{(l,t)} \in \mathbb{R}^{r_b \times d_t^{\text{in}}}$ that emits a token-dependent rank-wise score

$$g_b^{(l,t)}(x_\tau) = \sigma(W_{g,b}^{(l,t)} x_\tau) \in \mathbb{R}^{r_b}, \quad (7)$$

where σ is element-wise sigmoid. Unlike the static mask, the gate varies per token, so the same rank dimension can be activated at different strengths depending on the input context.

Composition. The mask and gate are combined into a single rank-wise routing vector

$$R_{b,\tau}^{(l,t)} = M_b^{(l)} \odot g_b^{(l,t)}(x_\tau), \quad (8)$$

which modulates the rank activation $A_b^{(l,t)} x_\tau$ before the up-projection $B_b^{(l,t)}$. The gated contribution of branch b is

$$\Delta_b^{(l,t)}(x_\tau) = \alpha_b B_b^{(l,t)}(R_{b,\tau}^{(l,t)} \odot A_b^{(l,t)} x_\tau), \quad (9)$$

and the final MLP output sums the frozen base with both branches:

$$\text{out}(x_\tau) = \text{base}(x_\tau) + \Delta_{\text{knw}}^{(l,t)}(x_\tau) + \Delta_{\text{rsn}}^{(l,t)}(x_\tau). \quad (10)$$

4.5 Training

DuoPRAG is trained in two stages: (1) reasoning-only pretraining without retrieval, and (2) joint LoRA training with both branches activated. Full details of the pipeline are provided in Appendix A.2.

Stage 1: reasoning-only pretraining. We attach only the reasoning branch W_{rsn} and operate under the W_{rsn} -only forward; the knowledge branch is not injected. The trainable parameters are W_{rsn} , the projector prj , the boundary token embeddings $\langle \text{bot} \rangle$, $\langle \text{eot} \rangle$, and the PEFT LoRA of auxiliary decoder D_η , whose output distribution we denote by $p_\eta(\cdot | \cdot)$.

Stage 2: joint LoRA training. Initializing W_{rsn} , prj , the boundary tokens, and D_η from the Stage 1

checkpoint, we attach the knowledge branch $W_{\text{knw}} = \Phi(P)$ and switch to the joint LoRA forward (Eq. (10)). At each branch, the retrieved passages P are encoded by Φ into per-module LoRA factors and merged via Eq. (4); the latent tokens z_k continue to be produced by the W_{rsn} -only forward to preserve their reasoning-only semantics. Trainable parameters in this stage are W_{rsn} , the masks $\{M_b^{(l)}\}$, the gate heads $\{W_{g,b}^{(l,t)}\}$, and D_η 's LoRA; Φ and the base LLM θ remain frozen.

Training objective. Both stages share the same composite loss combining answer-token supervision and step-level supervision. The answer-token loss is the autoregressive cross-entropy over the answer tokens following the latent prefix:

$$L_{\text{ans}} = - \sum_{i=1}^{|a|} \log p_{\theta+\Delta W}(a_i | q; \langle \text{bot} \rangle; z_1; \dots; z_K; \langle \text{eot} \rangle; a_{<i}), \quad (11)$$

where $\Delta W = \Delta W_{\text{rsn}}$ in Stage 1 and $\Delta W = \Delta W_{\text{rsn}} + \Delta W_{\text{knw}}(P)$ in Stage 2. The step-level loss trains D_η to generate each reasoning step text $s_k = (y_{k,1}, \dots, y_{k,L_k})$ from the latent prefix z_k :

$$L_{\text{step}} = - \sum_{k=1}^{\min(K, N_{\text{step}})} \sum_{t=1}^{L_k} \log p_\eta(y_{k,t} | z_k, y_{k,<t}), \quad (12)$$

where N_{step} is the number of annotated reasoning steps in the training example. The final objective is the weighted sum with $\lambda_{\text{ans}} = \lambda_{\text{step}} = 1$:

$$\mathcal{L} = \lambda_{\text{ans}} L_{\text{ans}} + \lambda_{\text{step}} L_{\text{step}}, \quad (13)$$

5 Experiments

5.1 Experimental Setup

Datasets and Protocol. We adopt the DyPRAG evaluation setting and evaluate on four multi-passage QA datasets that cover both multi-hop reasoning and single-hop QA: 2WikiMHQA (Ho et al., 2020), HotpotQA (Yang et al., 2018), PopQA (Mallen et al., 2023), and ComplexWebQuestions (CWQ) (Talmor and Berant, 2018) (details in Appendix A.1). Motivated by Section 3, for DuoPRAG we report main results under a *raw* prompt mode matching its training format, where the input consists only of the question followed by latent reasoning tokens. To verify that our gains are not tied to this choice, we additionally evaluate under the original PRAG *chat* protocol. Prompt formats for both modes are de-

Backbone Model	Method	2WikiMHQA EM / F1	HotpotQA EM / F1	PopQA EM / F1	CWQ EM / F1	Avg EM / F1
<i>Query-only approaches</i>						
LLaMA3.2-1B	Vanilla [†]	17.47 / 22.87	18.56 / 24.10	0.67 / 2.26	23.67 / 34.94	16.74 / 21.04
	PRAG-woC [†]	20.13 / 25.92	19.00 / 25.35	12.00 / 23.58	26.00 / 35.86	19.57 / 26.51
	DyPRAG-woC [†]	24.27 / 29.91	19.56 / 25.97	7.33 / 11.33	28.33 / 36.86	21.57 / 27.57
	DuoPRAG-woC	29.93 / 37.17	21.22 / 29.26	30.67 / 30.67	36.00 / 46.08	29.46 / 35.79
Qwen2.5-1.5B	Vanilla [†]	20.87 / 27.20	14.78 / 23.13	0.67 / 2.87	18.00 / 26.47	16.74 / 25.79
	PRAG-woC [†]	<u>21.93 / 29.38</u>	16.00 / 24.04	1.33 / 3.87	22.31 / 30.82	18.13 / 25.37
	DyPRAG-woC [†]	21.87 / 28.46	17.11 / 24.93	3.00 / 6.64	22.67 / 31.94	18.64 / 25.56
	DuoPRAG-woC	29.47 / 36.39	21.22 / 28.76	30.33 / 30.33	36.67 / 45.95	29.42 / 35.36
LLaMA3-8B	Vanilla [†]	30.00 / 36.43	19.89 / 28.64	4.67 / 7.96	30.00 / 42.44	24.43 / 31.85
	PRAG-woC [†]	<u>33.20 / 40.54</u>	35.55 / 45.88	<u>20.33 / 26.13</u>	<u>32.67 / 43.54</u>	<u>32.57 / 41.00</u>
	DyPRAG-woC [†]	32.07 / 39.17	24.67 / 37.33	11.00 / 13.60	<u>32.67 / 41.87</u>	27.80 / 36.23
	DuoPRAG-woC	38.40 / 45.00	<u>31.78 / 39.71</u>	42.33 / 42.56	39.67 / 49.62	38.04 / 44.22
<i>In-context Approaches</i>						
LLaMA3.2-1B	Standard RAG [†]	17.93 / 24.77	21.44 / 30.33	9.67 / 17.65	25.67 / 37.39	18.93 / 26.99
	PRAG [†]	20.60 / 26.94	<u>23.33 / 30.81</u>	<u>20.33 / 31.07</u>	28.33 / 39.63	22.17 / 29.78
	DyPRAG [†]	<u>26.33 / 32.53</u>	<u>23.33 / 30.80</u>	<u>21.33 / 29.93</u>	29.33 / 38.96	<u>25.23 / 31.80</u>
	DuoPRAG	32.07 / 40.06	27.45 / 35.71	53.33 / 54.17	39.00 / 47.58	37.96 / 44.38
Qwen2.5-1.5B	Standard RAG [†]	16.33 / 23.89	14.89 / 24.68	0.67 / 9.97	18.64 / 28.23	14.56 / 23.17
	PRAG [†]	19.07 / 27.29	19.33 / 26.15	2.67 / 12.61	21.67 / 32.13	17.77 / 25.96
	DyPRAG [†]	<u>18.87 / 25.87</u>	<u>20.67 / 30.13</u>	<u>7.33 / 22.69</u>	<u>23.67 / 33.57</u>	<u>18.74 / 27.60</u>
	DuoPRAG	28.73 / 36.28	31.22 / 40.93	51.33 / 52.00	41.33 / 49.44	38.15 / 44.66
LLaMA3-8B	Standard RAG [†]	28.40 / 34.20	19.13 / 28.67	5.67 / 16.13	25.33 / 35.45	23.04 / 30.86
	PRAG [†]	34.47 / 42.20	40.11 / 50.82	11.33 / 26.23	28.00 / 36.41	33.20 / 42.61
	DyPRAG [†]	<u>36.33 / 47.68</u>	33.22 / 43.22	21.00 / 32.86	29.67 / 39.07	<u>33.20 / 43.69</u>
	DuoPRAG	38.93 / 46.82	<u>38.44 / 49.52</u>	52.00 / 53.02	43.67 / 54.18	43.26 / 50.89

Table 2: In-domain experimental results. All metrics are reported as EM (%) and F1 (%). **Bold** and underline denote best and second-best performance within each base LM. [†] indicates results taken from DyPRAG (Tan et al., 2025). -woC: no retrieved passages in the prompt.

tailed in Appendix A.3. Following prior work (Su et al., 2025a), the suffix -woC denotes that retrieved passages are excluded from the prompt; methods without this suffix include the retrieved passages.

Backbone Models. We experiment with three instruction-tuned causal LMs of varying scales: LLaMA-3.2-1B-Instruct (Meta, 2024a), Qwen2.5-1.5B-Instruct (Qwen Team, 2024), and LLaMA-3-8B-Instruct (Meta, 2024b). The backbone is kept frozen in all experiments.

Baselines. Following Tan et al. (2025), we compare DuoPRAG against four baselines covering both standard and parametric RAG: Vanilla (no retrieval), Standard RAG (Lewis et al., 2020) (passages concatenated into the prompt), PRAG (Su et al., 2025a) (passages encoded into per-document LoRAs offline), and DyPRAG (Tan et al., 2025) (LoRA parameters generated at inference via a hypernetwork).

Evaluation Metrics. We report Exact Match for strict string-level correctness and F1 for token-level overlap that credits partial matches.

Implementation Details. Unless otherwise stated, we use a single training configuration across all backbones. All experiments are conducted on 8 NVIDIA RTX A6000 GPUs. Hyperparameters and hardware environment are

summarized in Appendix A.

5.2 Main Results

Table 2 reports EM and F1 across four QA benchmarks under both inference settings. DuoPRAG consistently outperforms all baselines on average across the three backbones, both with retrieved passages and without (DuoPRAG-woC). The without-passages result is particularly notable: by isolating the parametric modules, it shows that DuoPRAG internalizes retrieved knowledge into the Knowledge LoRA and reasons over it through K latent tokens, *without any lexical CoT prompting*. These in-domain gains carry over to out-of-distribution settings (§5.3) and translate into reduced CoT demonstration sensitivity (Output Length under CoT Prompts).

5.3 Out-Of-Distribution Performance

To assess whether DuoPRAG’s gains generalize beyond the training distribution, we evaluate on four unseen QA datasets (See Table 3): IIRC (Ferguson et al., 2020), StrategyQA (SQA) (Geva et al., 2021), OpenBookQA (OBQA) (Mihaylov et al., 2018), and MedMCQA (Pal et al., 2022); details are provided in Appendix A.1. We organize the analysis around the query-only (-woC) regime. On the reasoning-oriented SQA and IIRC, DuoPRAG-woC consistently outperforms

Backbone Model	Method	IIRC [†]	SQA [†]	OBQA	MedMCQA
Query-only approaches					
LLaMA3.2-1B	Vanilla	10.99	21.67	61.50	41.69
	DyPRAG-woC	14.04	39.67	62.17	41.52
	DuoPRAG-woC (Ours)	14.97	57.33	60.87	36.51
Qwen2.5-1.5B	Vanilla	8.78	1.00	72.11	42.97
	DyPRAG-woC	10.23	15.67	71.50	44.42
	DuoPRAG-woC (Ours)	12.88	56.67	70.33	44.35
LLaMA3-8B	Vanilla	13.23	33.33	74.68	59.67
	DyPRAG-woC	18.16	45.67	75.33	59.92
	DuoPRAG-woC (Ours)	18.59	67.33	75.67	60.67
In-context Approaches					
LLaMA3.2-1B	Standard RAG	40.38	27.67	76.17	61.83
	DyPRAG	41.91	50.33	74.50	62.50
	DuoPRAG (Ours)	55.11	59.00	72.43	56.80
Qwen2.5-1.5B	Standard RAG	30.52	39.00	82.17	68.42
	DyPRAG	38.25	43.33	83.17	66.79
	DuoPRAG (Ours)	52.88	59.67	83.67	68.60
LLaMA3-8B	Standard RAG	43.27	45.67	83.26	75.42
	DyPRAG	57.90	58.67	86.55	73.69
	DuoPRAG (Ours)	65.15	72.67	86.53	75.83

Table 3: Out-of-distribution performance. IIRC: F1, SQA: Acc, OBQA/MedMCQA: Recall (%). IIRC and SQA baselines[†] are taken from Tan et al. (2025); all other results, including OBQA and MedMCQA, are obtained under our setup.

DyPRAG-woC and Vanilla across all three backbones, and even exceeds Standard RAG without access to passages at inference time. In contrast, on the knowledge-intensive OBQA and MedMCQA, which require specialized science or medical facts, DuoPRAG-woC remains competitive among query-only methods but falls below Standard RAG, indicating that W_{knw} alone does not yet capture sufficiently fine-grained factual evidence.

5.4 Ablation Study

Knowledge LoRA Generator Variant. Table 4 tests whether DuoPRAG’s fusion framework transfers across knowledge-LoRA generators, using D2L (Charakorn et al., 2026) and DyPRAG (Tan et al., 2025). Under both generators, DuoPRAG improves over the corresponding parametric baseline on every task and backbone, confirming that the gain comes from the composition rather than a specific generator. Standard RAG remains strongest on these single-passage extractive tasks when the backbone has strong in-context capability (Qwen3-4B, LLaMA-3-8B), while the picture flips on Mistral-7B, where Standard RAG lags behind its scores on other backbones and DuoPRAG_{D2L}-woC overtakes it on all three tasks-suggesting that parametric injection becomes more competitive when the backbone exploits raw context less effectively. This is consistent with recent findings that hypernetwork-based parametric injection provides high-level semantic guidance but loses

Base LLM	Method	SQuAD	DROP	ROPES
Mistral-7B-v0.2-Inst.	Standard RAG	45.60	20.40	24.50
	D2L-woC	31.10	14.40	19.00
	DuoPRAG _{D2L} woC	51.50	32.60	55.90
Qwen3-4B	Standard RAG	73.00	40.60	73.30
	D2L-woC	30.40	14.00	21.20
	DuoPRAG _{D2L} woC	56.10	29.30	62.40
LLaMA-3.2-1B-Inst.	Standard RAG	57.30	23.00	52.60
	DyPRAG-woC	11.00	11.00	40.70
	DuoPRAG _{DyPRAG} woC	13.10	19.10	43.90
Qwen-2.5-1.5B-Inst.	Standard RAG	56.00	24.30	43.30
	DyPRAG-woC	8.80	5.50	27.20
	DuoPRAG _{DyPRAG} woC	17.00	16.70	47.10
LLaMA-3-8B-Inst.	Standard RAG	70.20	36.40	74.20
	DyPRAG-woC	13.20	9.80	39.80
	DuoPRAG _{DyPRAG} woC	23.10	20.60	57.30

Table 4: Results on the Doc-to-LoRA dataset (Charakorn et al., 2026) (F1, %).

Setting	LLaMA-3.2-1B EM / F1	Qwen-2.5-1.5B EM / F1	LLaMA-3-8B EM / F1
DuoPRAG-woC	29.46 / 35.79	29.42 / 35.36	38.04 / 44.22
<i>w/o (M, g)</i>			
$\Phi_{\text{knw}} : \text{DyPRAG}$	16.95 / 23.08	18.01 / 24.03	23.29 / 30.69
$\Phi_{\text{knw}} : \text{PRAG}$	16.16 / 23.09	19.71 / 26.55	9.73 / 19.49
<i>Branch choice $\Phi_{\text{knw}} : \text{DyPRAG}$</i>			
w W_{Direct}	24.77 / 30.59	26.08 / 31.36	26.78 / 32.27
w W_{Explicit}	20.52 / 26.27	21.17 / 25.98	28.25 / 34.51
only W_{rsn} , no Φ_{knw}	17.60 / 23.26	18.31 / 24.42	23.79 / 31.10

Table 5: Fusion and branch-choice ablation across base LLMs (Avg EM / F1, %).

fine-grained passage details (Tang et al., 2025), while in-context retrieval preserves token-level evidence that stronger backbones exploit more effectively.

Fusion Component Ablation. Table 5 summarizes per-backbone averages over the four benchmarks; full per-benchmark results are reported in Appendix B.1. We compare DuoPRAG-woC against three variants: *reasoning-only* (no knowledge branch), and *direct-answer* and *explicit-CoT*, which replace the latent reasoning branch with a plain LoRA trained (Stage 1/2) to emit only the answer or a verbalized reasoning chain. DuoPRAG-woC achieves the best average score, while every variant without learned fusion drops sharply. The direct-answer and explicit-CoT variants outperform the reasoning-only setting on some datasets but still fall far below DuoPRAG-woC on average. This supports the central claim that performance depends not only on the presence of reasoning and knowledge LoRAs, but also on the learned rank-level and token-level routing.

Fusion Attribution Variants. Retraining only Stage 2 from the same Stage-1 checkpoint with identical data and hyperparameters, we compare

Setup	2WikiMHQA	HotpotQA	PopQA	CWQ	Avg.
DuoPRAG (full)	29.93 / 37.17	21.22 / 29.26	30.67 / 30.67	36.00 / 46.08	29.46 / 35.79
Mask only	24.40 / 31.63	21.33 / 28.61	15.33 / 16.39	32.67 / 40.97	23.43 / 29.40
Gate only	27.47 / 35.19	20.22 / 27.43	25.00 / 25.89	35.33 / 42.56	27.00 / 32.77
Scalar gate	28.07 / 35.82	20.89 / 27.58	24.00 / 24.89	35.00 / 42.05	26.99 / 32.58
Token-independent gate	26.40 / 34.13	20.55 / 27.77	23.33 / 24.22	35.33 / 42.37	26.40 / 32.12
Parameter-controlled additive	34.00 / 39.63	21.11 / 26.99	20.33 / 20.33	32.67 / 42.42	27.03 / 32.34

Table 6: Fusion-attribution Variants on Llama3.2-1B (EM / F1, %).

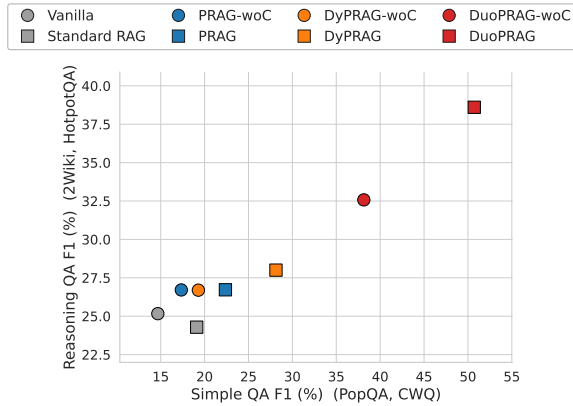


Figure 3: Simple QA vs. Reasoning QA on Qwen2.5-1.5B.

fusion attribution variants in Table 6. The results yield three findings. (1) Neither component alone suffices: mask-only and gate-only both fall short of the full fusion; the combination is what works. (2) Gating granularity matters: the scalar gate and the sample-level token-independent gate also underperform, so a fixed coefficient or per-sample routing cannot substitute for token-conditional, rank-wise modulation. (3) The gain does not come from parameter count: the additive variant, despite slightly more parameters than DuoPRAG, trails on average. We note it does surpass the full model on 2WikiMHQA. However, it is lower on the other three benchmarks and notably unstable on PopQA. The multiplicative token-conditional gate maintains a consistent average advantage. Full Details of the variants are provided in Table 13 and Appendix C.

5.5 Analysis

CoT Demonstration Sensitivity. Under an identical generation budget (128 new tokens for 2WikiMHQA/HotpotQA, 20 for PopQA/CWQ) and the *same few-shot CoT prompt* (refer to Appendix A.3, no passages), DuoPRAG answers in only ~ 4 tokens, over $20\times$ shorter than PRAG and DyPRAG, which echo the lexical CoT pat-

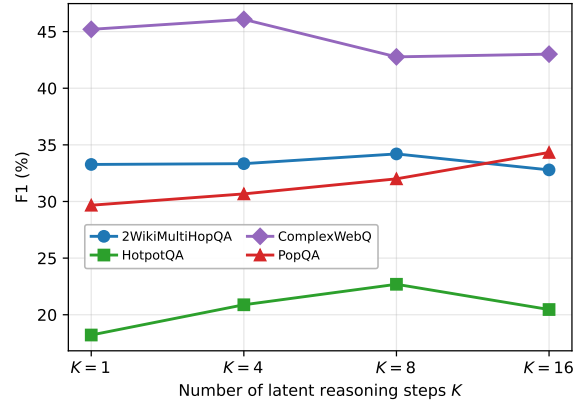


Figure 4: We retrain DuoPRAG-woC end-to-end for $K \in \{1, 4, 8, 16\}$ on LLaMA3.2-1B, varying *only* the number of latent reasoning steps.

tern of the exemplars before producing an answer (Fig. 5). This brevity reflects a design choice: DuoPRAG decouples reasoning from lexical CoT and carries it out in K latent tokens, so the surface output collapses to the final answer span. Consistent with this, under the original PRAG *chat* protocol (Table 11), DuoPRAG-woC retains its lead on LLaMA-3.2-1B and Qwen-2.5-1.5B, and on LLaMA-3-8B it achieves the highest EM while trailing PRAG-woC by 0.7 F1, an expected trade-off since larger backbones benefit more from CoT exemplars that DuoPRAG omits by design.

Latent Count K Analysis. We retrain DuoPRAG end-to-end for $K \in \{1, 4, 8, 16\}$ on LLaMA-1B, varying *only* the number of latent reasoning steps (Fig. 4). Performance is stable across the entire range, indicating that DuoPRAG is robust to the choice of K and that gains do not depend on a specific budget of latent steps. We adopt $K=4$ for its favorable accuracy-cost trade-off.

Output Length under CoT Prompts. PRAG and DyPRAG compute loss over the entire CoT-answer sequence, tightly coupling their outputs to the lexical CoT format seen during training. DuoPRAG instead performs reasoning through K la-

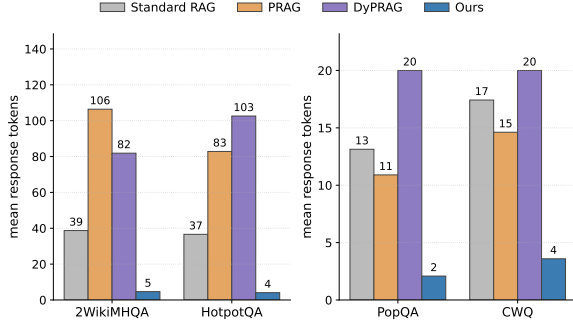


Figure 5: Comparison of response length across datasets, the backbone model is Llama3.2-1B.

Variant	CKA ($\downarrow$, avg. over 4 benchmarks)	Avg. F1
Full (DuoPRAG)	0.031	35.79
Gate only	0.105	32.77
Mask only	0.917	29.40
Independently trained LoRAs (naive addition)	0.910	23.08

Table 7: Linear CKA and average F1 score for fusion variants on LLaMA-3.2-1B. CKA is averaged over four benchmarks.

tokens integrated with the knowledge LoRA via rank-wise fusion (§4.4). Under the same few-shot CoT prompt at inference (Appendix A.3, no passages, identical generation budget), this design manifests in a sharp behavioral gap: PRAG and DyPRAG keep producing verbose CoT-style sequences even after reaching the answer, while DuoPRAG outputs only the answer (Fig. 5). This gap is not a side-effect to be controlled away by aligning DuoPRAG’s training objective with PRAG/DyPRAG. Doing so would replace DuoPRAG’s latent reasoning supervision with surface CoT supervision, collapsing the very architectural distinction we evaluate. The gap is the contribution: lexical CoT triggers no longer steer output.

Linear CKA Analysis. Table 7 reports linear CKA between the two branches for each fusion variant. The motivation for using CKA and its mathematical details are given in Appendix D. High CKA indicates that the two branches organize token-to-token similarity in a similar way and thus encode redundant information, whereas low CKA indicates that they capture distinct, complementary information. Independently trained LoRAs with naive addition and the mask-only variant both show high CKA and low average F1. In contrast, full DuoPRAG and the gate-only variant show low CKA and higher average F1. These results suggest that the rank-specific gate is the component that drives the two branches to encode distinct and complementary information. Notably,

Method	Knowledge injection (ms)	Latent reasoning (ms)	Answer generation (ms)	Total (ms)	Avg. output tokens	F1
Q-only	–	–	1,750	1,750	128	22.87
PRAG	26	–	1,506	1,533	72	25.92
DyPRAG	126	–	2,380	2,506	128	29.91
DuoPRAG	141	114	120	375	4	37.17

Table 8: Efficiency decomposition on 2WikiMulti-hopQA with LLaMA-3.2-1B. All methods share the same generation configuration with a 128 token cap.

the mask-only variant improves F1 over naive addition without reducing CKA, indicating that the mask contributes through a different mechanism than representational separation. The two components thus play complementary roles, consistent with our fusion attribution ablation above.

Efficiency Decomposition. Table 8 decomposes per-query latency. All F1 scores are computed after the official PRAG answer extraction post-processing. Under an identical generation configuration, PRAG and DyPRAG overgenerate due to their lexical CoT, with DyPRAG reaching the 128 token cap. In contrast, DuoPRAG spends 255 ms on pre-answer latent computation (knowledge injection and latent reasoning), which is more than offset by its 4 token answer generation, yielding 4.1–6.7 $\times$ lower total latency at the highest F1, demonstrating that it is not only more accurate but also substantially more efficient than the compared parametric RAG methods.

6 Conclusion

We empirically identified important limitations of existing parametric RAG framework and presented Duo Parametric RAG (DuoPRAG), a framework that disentangles parametric knowledge from reasoning by inserting two specialized LoRA branches and integrating them effectively through a rank duo mask and a rank-specific gate. Unlike prior parametric RAG methods that rely on lexical CoT for reasoning, DuoPRAG employs a dedicated reasoning LoRA with latent reasoning tokens, mitigating knowledge conflict and removing the need for lexical CoT at training and inference time. Across open-domain QA benchmarks including both single-hop and multi-hop reasoning, DuoPRAG improves over existing parametric RAG approaches and is particularly effective on reasoning-heavy QA, although it does not uniformly replace standard RAG on knowledge-intensive tasks. Out-of-domain evaluation further shows that our method remains competitive beyond the training distribution.

Limitations

DuoPRAG effectively adapts the base LM’s reasoning capability and avoids the three failure modes in outputs. Like other parametric RAG methods, its competitiveness against in-context retrieval is bounded by current hypernetwork-based encoders, which capture document-level semantics but lose fine-grained details (Tang et al., 2025); improving passage-to-parameter compression is a natural direction for future work. We also observe degradation on out-of-domain tasks that demand very specific factual knowledge from the retrieved passages. Finally, our evaluation is limited to open-domain QA. Whether the proposed two-branch parameter-level fusion generalizes to other reasoning-intensive settings, such as multi-step mathematical reasoning, code reasoning, and decision-driven agentic pipelines, remains an open question for future work.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2026-25471023). This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II201336, Artificial Intelligence Graduate School Support (UNIST)) and by an IITP grant funded by the Korea government (MSIT) (No. RS-2023-00216011, Development of Artificial Complex Intelligence for Conceptually Understanding and Inferring like Human). We used AI assistants (e.g., ChatGPT, Claude) to polish the writing of this paper; all research ideas, experiments, and analyses are the authors’ own.

References

- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.
- Rujikorn Charakorn, Edoardo Cetin, Shinnosuke Uesaka, and Robert Tjarko Lange. 2026. Doc-to-LoRA: Learning to instantly internalize contexts. *arXiv preprint arXiv:2602.15902*.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2023. [PaLM: Scaling language modeling with pathways](#). *Journal of Machine Learning Research*, 24(240):1–113.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. [Scaling instruction-finetuned language models](#). *Journal of Machine Learning Research*, 25(70):1–53.
- James Ferguson, Matt Gardner, Hannaneh Hajishirzi, Tushar Khot, and Pradeep Dasigi. 2020. [IIIRC: A dataset of incomplete information reading comprehension questions](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1137–1147. Association for Computational Linguistics.
- Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. [Complexity-based prompting for multi-step reasoning](#). In *International Conference on Learning Representations*.
- Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. [Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies](#). *Transactions of the Association for Computational Linguistics (TACL)*, 9:346–361.
- Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. 2005. Measuring statistical dependence with hilbert-schmidt norms. In *Algorithmic Learning Theory*, pages 63–77. Springer.
- 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, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *The Tenth International Conference on Learning Representations (ICLR)*.
- Peter Jansen, Elizabeth Wainwright, Steven Mar-morstein, and Clayton Morrison. 2018. [WorldTree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference](#). In

- Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. *Mistral 7b*. Preprint, arXiv:2310.06825.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. *TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension*. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.
- Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. *Similarity of neural network representations revisited*. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529. PMLR.
- Evgenii Kortukov, Alexander Rubinstein, Elisa Nguyen, and Seong Joon Oh. 2024. *Studying large language model behaviors under context-memory conflicts with real documents*. In *First Conference on Language Modeling*.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. *Natural questions: A benchmark for question answering research*. *Transactions of the Association for Computational Linguistics*, 7:453–466.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich K  ttler, Mike Lewis, Wen-tau Yih, Tim Rockt  schel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474.
- Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32100–32121. PMLR.
- 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 (ACL)*, pages 9802–9822.
- Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, and Sayak Paul. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. <https://github.com/huggingface/peft>.
- Meta. 2024a. Llama-3.2-1b-instruct. <https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct>. Model card, accessed 2024-09.
- Meta. 2024b. Meta-llama-3-8b-instruct. <https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct>. Model card, accessed 2024-04.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. *Can a suit of armor conduct electricity? a new dataset for open book question answering*. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2381–2391. Association for Computational Linguistics.
- Sewon Min, Xinxin Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. *Rethinking the role of demonstrations: What makes in-context learning work?* In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Ankit Pal, Logesh Kumar Umapathi, and Malaikanan Sankarasubbu. 2022. *MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering*. In *Proceedings of the Conference on Health, Inference, and Learning (CHIL)*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR.
- Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Qwen Team. 2025. Qwen3-4b-instruct-2507. <https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507>. Model card, accessed 2026-05-26.
- Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susanah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, and 60 others. 2021. *Scaling language models: Methods, analysis & insights from training Gopher*. *arXiv preprint arXiv:2112.11446*.

- Aniket Roy, Shubhankar Borse, Shreya Kadambi, Debasmit Das, Shweta Mahajan, Risheek Garrepalli, Hyojin Park, Ankita Nayak, Rama Chellappa, Munawar Hayat, and Fatih Porikli. 2025. DuoLoRA: Cycle-consistent and rank-disentangled content-style personalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15395–15404.
- Weihang Su, Yichen Tang, Qingyao Ai, Junxi Yan, Changyue Wang, Hongning Wang, Ziyi Ye, Yujia Zhou, and Yiqun Liu. 2025a. Parametric retrieval augmented generation. *arXiv preprint arXiv:2501.15915*.
- Zhan Su, Fengran Mo, Jinghan Zhang, Yuchen Hui, Jiaao Sun, and Jian-Yun Nie. 2025b. Parametric retrieval-augmented generation using latent routing of LoRA adapters. *arXiv preprint arXiv:2511.17044*.
- 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, New Orleans, Louisiana. Association for Computational Linguistics.
- Yuqiao Tan, Shizhu He, Huanxuan Liao, Jun Zhao, and Kang Liu. 2025. [Dynamic parametric retrieval augmented generation for test-time knowledge enhancement](#). Preprint, arXiv:2503.23895.
- Minghao Tang, Shiyu Ni, Jingtong Wu, Zengxin Han, and Keping Bi. 2025. Understanding parametric knowledge injection in retrieval-augmented generation. *arXiv preprint arXiv:2510.12668*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. [Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.
- Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023. [Towards understanding chain-of-thought prompting: An empirical study of what matters](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2717–2739, Toronto, Canada. Association for Computational Linguistics.
- Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. 2026. SIM-CoT: Supervised implicit chain-of-thought. In *The Fourteenth International Conference on Learning Representations (ICLR)*.
- Rongwu Xu, Zehan Qi, Zhijiang Guo, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. 2024. [Knowledge conflicts for LLMs: A survey](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8541–8565, Miami, Florida, USA. Association for Computational Linguistics.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A dataset for diverse, explainable multi-hop question answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.
- Yuchen Zeng and Kangwook Lee. 2024. The expressive power of low-rank adaptation. In *The Twelfth International Conference on Learning Representations*.
- Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*.

A Experimental Details

A.1 Evaluation Benchmarks and Training Data

In-Domain Benchmarks. Following prior work (Su et al., 2025a; Tan et al., 2025), we evaluate on four QA benchmarks that jointly assess multi-hop reasoning and fact-based question answering. (1) 2WikiMultiHopQA (2WQA) (Ho et al., 2020) requires combining information across two or more Wikipedia articles via inter-entity relations, with evidence triples and reasoning paths provided to verify multi-step inference. (2) HotpotQA (HQA) (Yang et al., 2018) tests multi-hop reasoning over distinct Wikipedia paragraphs, covering both *bridge* and *comparison* questions with supporting-sentence annotations. (3) PopQA (PQA) (Mallen et al., 2023) is a single-hop, fact-centric benchmark automatically built from Wikidata triples, emphasizing knowledge recall and entity disambiguation across a wide popularity range. (4) ComplexWebQuestions (CWQ) (Talmor and Berant, 2018) extends WebQuestionsSP (Yih et al., 2016) with compositional operators (conjunction, composition, comparatives, superlatives), evaluating multi-step reasoning together with web-scale retrieval.

Out-of-Domain Benchmarks. To examine generalization beyond the training distribution, we additionally report results on four out-of-domain (OOD) benchmarks spanning implicit reasoning, incomplete-context QA, commonsense reasoning, and domain-specific medical knowledge. (1) StrategyQA (SQA) (Geva et al., 2021) is a yes/no benchmark whose reasoning steps are not stated in the question and must be inferred via implicit decomposition, with human-annotated Wikipedia evidence. (2) IIRC (Ferguson et al., 2020) contains $\sim 13\text{K}$ Wikipedia-grounded questions whose context paragraph is only partially informative, requiring retrieval of missing evidence from linked documents. Answer types include extractive spans, numbers requiring discrete reasoning, binary answers, and unanswerable questions. (3) OpenBookQA (OBQA) (Mihaylov et al., 2018) is a multiple-choice dataset based on WorldTree (Jansen et al., 2018) science facts, probing commonsense reasoning by combining a given fact with broader world knowledge. (4) MedMCQA (MQA) (Pal et al., 2022) is a large-

Stage	Dataset	Used	Train (orig.)	%
Stage 1	SQuAD	7,933	87,599	9.1
	DROP	6,830	77,400	8.8
	PWC	21,739	241,564	9.0
	ROPES	963	10,924	8.8
	NaturalQuestions	7,200	79,168	9.1
	Total	44,665	496,655	9.0
Stage 2	2WikiMultiHopQA	900	167,454	0.5
	HotpotQA	540	90,447	0.6
	PopQA	180	14,267	1.3
	CWQ	179	27,639	0.6
	Total	1,799	299,807	0.6

Table 9: Training data composition. All samples are augmented with reasoning steps generated by Qwen3.5-72B using Figure 7. The Stage 2 datasets are taken from the official DyPRAG GitHub repository (Tan et al., 2025).

scale multiple-choice medical QA dataset from Indian medical entrance exams, testing generalization to a specialized professional domain far from training.

Training Data. We adopt a two-stage training strategy. Stage 1 uses single-passage QA data to pre-train implicit reasoning ability, while Stage 2 uses multi-hop QA data to jointly train the dual-LoRA architecture. Each training example is constructed by augmenting the original question-answer pair with a multi-step reasoning trajectory generated by Qwen3.5-72B*, which serves as the reference for the latent reasoning supervision term L_{step} . Per-dataset usage ratios are reported in Table 9. To rule out potential test-set leakage, we cross-check the original question IDs (qids) of all training examples against those of the evaluation sets and confirm no overlap.

A.2 Training Details

Component	Stage 1	Stage 2
Data	44,665 samples	1,799 samples
Initialization	from scratch	Stage 1 ckpt (step 2,792)
Trainable	W_{ran} , prj, embed, aux dec	+ mask M , gate g
Frozen	base LM	base LM, W_{knw} (DyPRAG)
LR (main)	1×10^{-4}	1×10^{-5}
LR (aux dec)	2×10^{-5}	1×10^{-5}
LR (M , g)	—	1×10^{-3}
Scheduler	cosine, 150 warmup	cosine, 50 warmup
Weight decay	0.01	0.0
Grad accum	4	4
Distributed data parallel	8	8

Table 10: Training configuration for Stage 1 (reasoning LoRA pretraining) and Stage 2 (DyPRAG joint training).

All experiments are conducted on $8 \times$ NVIDIA RTX A6000 GPUs, with the base LM kept

*<https://huggingface.co/Qwen/Qwen3.5-72B>

frozen throughout. For the main experiments, we use LLaMA-3.2-1B-Instruct, Qwen-2.5-1.5B-Instruct, and LLaMA-3-8B-Instruct. For the D2L generator experiments, we additionally use Mistral-7B-Instruct-v0.2 (Jiang et al., 2023) and Qwen3-4B-Instruct-2507 (Qwen Team, 2025).

Optimization. We use AdamW with the LR/scheduler/weight-decay settings in Table 10 and gradient clipping at max-norm 1.0. All training is performed with random seed 42.

Components. The reasoning LoRA W_{rsn} is applied to {gate-proj, up-proj, down-proj} with rank 2, $\alpha = 32$. The auxiliary decoder D_η is a copy of the base LM equipped with LoRA (rank 8, $\alpha = 16$) on {q-proj, v-proj}. We use $K = 4$ latent reasoning tokens and learnable embeddings for the three special tokens ($\langle \text{bot} \rangle$, $\langle \text{eot} \rangle$, latent). The knowledge LoRA W_{knw} is the DyPRAG (Tan et al., 2025) parameter translator (rank 2) and is frozen throughout.

Two-stage procedure. As shown in Figure 10, Stage 1 trains W_{rsn} , the latent projector, special-token embeddings, and D_η from scratch. Stage 2 resumes from the Stage 1 checkpoint, additionally introduces the rank-mask M and token-gate g , and trains them jointly with the Stage 1 components while keeping W_{knw} frozen. M is initialized to all-ones and the token-gate projections g are zero-initialized.

A.3 Inference Settings

All experiments use the Hugging Face implementations of LLaMA, Qwen, and Mistral. For a fair comparison, DuoPRAG and all baselines share the same prompt template (Fig. 11, 12, 13) following Tan et al. (2025), and we use greedy decoding for reproducibility. The maximum number of new tokens is set to 128 for multi-hop tasks and 20 for single-hop tasks.

Chat-mode evaluation. Table 11 evaluates the same DuoPRAG-woC model from our main results under the chat protocol (Fig. 13), whose prompt format differs substantially from the training format (Fig. 11). Despite this train-evaluation mismatch and without any retrieved passages at inference, DuoPRAG-woC attains the highest average EM on all three backbones, outperforming every prior baseline, including those evaluated *with* retrieved context. These results indicate that the gains of DuoPRAG are robust to the evaluation

prompt format, rather than an artifact of train-evaluation alignment.

A.4 Supervision is matched to each method’s own design, not withheld from baselines

PRAG and DyPRAG are trained and evaluated with the manually written IR-CoT few-shot demonstrations (Trivedi et al., 2023), exactly as in their official setups. DuoPRAG’s teacher-generated trajectories are not an extra distillation layered on top. They are module-specific supervision required to train the reasoning LoRA, a component the baselines do not have. We do not claim the two supervision forms are identical; rather, each method receives the guidance its own design prescribes, so neither baseline is disadvantaged relative to its original proposal.

B Additional Ablation Experiment Results

B.1 Fusion Component Ablation

Table 12 ablates the two design choices underlying the dual-LoRA composition of DuoPRAG-woC: (i) retaining the learned latent reasoning branch W_{rsn} alongside the knowledge branch Φ_{knw} , and (ii) fusing them via the learned rank-mask M and token-gate g rather than by naive concatenation.

Naive concatenation baselines. Rows under *w/o* (M, g) reuse the Stage 1 checkpoint of DuoPRAG with W_{rsn} frozen and set Φ_{knw} to either PRAG or DyPRAG. They combine W_{rsn} and Φ_{knw} via naive concatenation, without learning the rank-mask M or token-gate g . These variants isolate the contribution of the learned fusion.

Reasoning supervision baselines. Rows under *Branch choice* Φ_{knw} : DyPRAG fix Φ_{knw} to DyPRAG unless explicitly removed, and vary the reasoning branch. The W_{Direct} and $W_{Explicit}$ variants follow the same two-stage training schedule as DuoPRAG, but replace the latent reasoning branch with a plain LoRA of matched target modules, rank, and scaling, differing only in the Stage 1 and 2 supervision target. W_{Direct} is trained to emit the answer alone, carrying no reasoning and decoding only the answer tokens at inference. $W_{Explicit}$ is trained on a verbalized chain (“Let’s think step by step. [S1] . . . The answer is X”), exposing reasoning as surface text at the cost of roughly $30\times$ more decoded tokens.

Backbone	2WQA	HQA	PopQA	CWQ	Avg
LLaMA-3.2-1B	25.20 / 31.36	20.33 / 27.01	27.67 / 28.11	34.33 / 41.94	26.88 / 32.10
Qwen-2.5-1.5B	24.74 / 30.25	24.11 / 29.93	26.33 / 26.78	37.00 / 45.52	28.04 / 33.12
LLaMA-3-8B	29.80 / 35.83	28.56 / 36.81	39.00 / 39.39	39.33 / 49.19	34.17 / 40.30

Table 11: Performance of DuoPRAG-woC under the *chat* prompt mode (EM / F1, %). Following the original PRAG protocol, multi-hop tasks (2WQA, HQA) use a CoT few-shot prompt and single-hop tasks (PopQA, CWQ) use a direct prompt. Knowledge is still injected exclusively through W_{knw} (no passages in the prompt).

Base LLM	Setting	2WikiMHQA EM / F1	HotpotQA EM / F1	PopQA EM / F1	CWQ EM / F1	Avg EM / F1
LLaMA-3.2-1B	DuoPRAG-woC	29.93 / 37.17	21.22 / 29.26	30.67 / 30.67	36.00 / 46.08	29.46 / 35.79
	<i>w/o</i> (M, g)					
	Φ_{knw} : DyPRAG	21.13 / 27.69	16.67 / 22.87	0.00 / 2.76	30.00 / 39.00	16.95 / 23.08
	Φ_{knw} : PRAG	20.73 / 28.44	15.22 / 21.34	2.00 / 6.63	26.67 / 35.95	16.16 / 23.09
	<i>Branch choice</i> Φ_{knw} : DyPRAG					
	w W_{Direct}	34.20 / 39.66	16.89 / 22.74	17.33 / 17.67	30.67 / 42.29	24.77 / 30.59
	w $W_{Explicit}$	29.00 / 34.55	18.44 / 24.55	18.33 / 19.08	16.33 / 26.91	20.52 / 26.27
	only W_{rsn} , no Φ_{knw}	20.93 / 26.96	18.11 / 24.27	0.67 / 3.62	30.67 / 38.20	17.60 / 23.26
Qwen-2.5-1.5B	DuoPRAG-woC	29.47 / 36.39	21.22 / 28.76	30.33 / 30.33	36.67 / 45.95	29.42 / 35.36
	<i>w/o</i> (M, g)					
	Φ_{knw} : DyPRAG	22.27 / 29.04	17.44 / 24.41	0.67 / 2.92	31.67 / 39.76	18.01 / 24.03
	Φ_{knw} : PRAG	21.60 / 28.02	18.56 / 24.80	5.00 / 11.16	33.67 / 42.23	19.71 / 26.55
	<i>Branch choice</i> Φ_{knw} : DyPRAG					
	w W_{Direct}	33.33 / 38.52	18.00 / 22.70	22.67 / 22.67	30.33 / 41.53	26.08 / 31.36
	w $W_{Explicit}$	26.80 / 33.02	17.89 / 23.40	18.00 / 18.69	22.00 / 28.79	21.17 / 25.98
	only W_{rsn} , no Φ_{knw}	23.13 / 29.94	19.11 / 25.64	0.67 / 2.76	30.33 / 39.35	18.31 / 24.42
LLaMA-3-8B	DuoPRAG-woC	38.40 / 45.00	31.78 / 39.71	42.33 / 42.56	39.67 / 49.62	38.04 / 44.22
	<i>w/o</i> (M, g)					
	Φ_{knw} : DyPRAG	24.40 / 31.37	22.78 / 30.86	6.67 / 12.01	39.33 / 48.51	23.29 / 30.69
	Φ_{knw} : PRAG	7.13 / 20.27	4.11 / 11.75	0.67 / 5.42	27.00 / 40.51	9.73 / 19.49
	<i>Branch choice</i> Φ_{knw} : DyPRAG					
	w W_{Direct}	32.67 / 38.69	18.78 / 24.16	20.67 / 20.89	35.00 / 45.36	26.78 / 32.27
	w $W_{Explicit}$	35.80 / 42.72	23.55 / 31.24	19.33 / 20.23	34.33 / 43.84	28.25 / 34.51
	only W_{rsn} , no Φ_{knw}	23.73 / 29.90	26.11 / 34.69	4.33 / 8.58	41.00 / 51.24	23.79 / 31.10

Table 12: Fusion component ablation of DuoPRAG (EM / F1, %). Setting labels follow Table 5; rows under *w/o* (M, g) drop the learned fusion; Φ_{knw} denotes the knowledge branch choice; rows under *Branch choice* Φ_{knw} : DyPRAG replace or remove the reasoning/knowledge branch while fixing Φ_{knw} to DyPRAG unless noted.

The *only* W_{rsn} , no Φ_{knw} row keeps the latent reasoning branch but removes the knowledge branch.

Full DuoPRAG-woC is consistently best. DuoPRAG-woC achieves the highest average EM/F1 across all three backbones, confirming that W_{rsn} , Φ_{knw} , and the learned fusion (M, g) are complementary: removing or replacing any one of them degrades performance.

Learned fusion is critical. Replacing (M, g) with naive concatenation consistently degrades performance, even when Φ_{knw} is set to DyPRAG, showing that simply summing the two LoRA updates is suboptimal. The W_{Direct} and $W_{Explicit}$ variants further confirm that supervision format alone cannot compensate: despite following the

same Stage 1 and Stage 2 schedule, both remain below DuoPRAG-woC. Together, these results show that the rank-mask M and token-gate g are needed to align W_{rsn} and Φ_{knw} updates along the rank axis, not merely combine them.

Both branches are indispensable. Removing the knowledge branch also degrades performance substantially: the *only* W_{rsn} , no Φ_{knw} variant yields near-zero EM on PopQA across all base LMs, reflecting the absence of injected entity-level knowledge. Only when W_{rsn} and Φ_{knw} are fused through (M, g) does the model jointly recall factual knowledge and apply the reasoning needed to chain it.

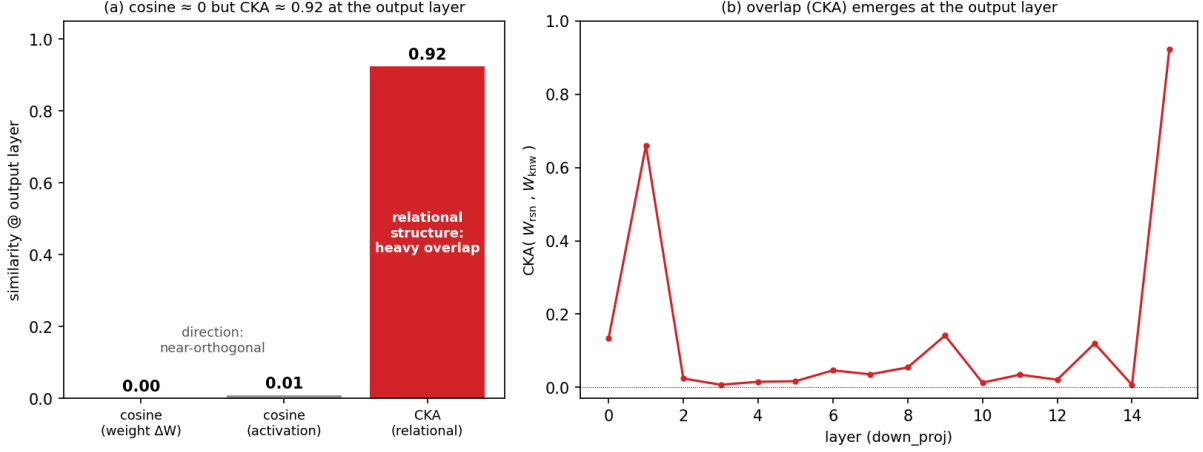


Figure 6: Similarity between independently constructed and trained reasoning (W_{rsn}) and knowledge (W_{knw}) LoRAs, measured at the down_proj projection of LLaMA-3.2-1B and averaged over four multi-hop/QA datasets: (a) cosine similarity vs. CKA at the output layer; (b) layer-wise CKA.

Variant	Description	Branch output $o_{m,j}$
Full DuoPRAG	mask + token-conditional gate, $g_{m,j} = \sigma(G_m h_j)$	$B_m(M_m \odot z_{m,j} \odot g_{m,j})$
Mask only	gate removed	$B_m(M_m \odot z_{m,j})$
Gate only	mask removed	$B_m(z_{m,j} \odot g_{m,j})$
Scalar gate	learnable scalar replaces the token-conditional gate	$B_m(M_m \odot z_{m,j}) \sigma(\alpha_m)$
Token-independent gate	one gate per sample, $\bar{g}_m = \sigma(G_m \text{Pool}(H_{\text{prompt}}))$, shared across tokens	$B_m(M_m \odot z_{m,j} \odot \bar{g}_m)$
Parameter-controlled additive	additive term replaces the multiplicative gate; 10.37M params vs. DuoPRAG’s 10.17M	$B_m(M_m \odot (z_{m,j} + P_m h_j))$

Table 13: Variant ablation of DuoPRAG. Each variant modifies the masking or gating mechanism applied to the rank-space representation $z_{m,j}$.

C Fusion Attribution Details

Variant definitions. Let $m \in \{\text{knw}, \text{rsn}\}$ index the branch, j denote the token position, and $M_m \in \mathbb{R}^r$ denote the rank-level mask. For each branch, the rank-space representation and aggregated output are defined as

$$z_{m,j} = A_m h_j, \quad \Delta h_j = \sum_{m \in \{\text{knw}, \text{rsn}\}} o_{m,j}. \quad (14)$$

We ran the ablation variants of DuoPRAG under identical conditions (See Table 13 for details): starting from the same Stage-1 checkpoint, with training data and hyperparameters held fixed, we re-trained only Stage 2 for each fusion variant and evaluated it with the same protocol as our main experiments (Table 2).

D Centered Kernel Alignment Analysis

Why CKA? A single LoRA that must jointly learn factual knowledge and reasoning behavior carries two distinct signals on the same limited

rank dimensions, and its expressivity is known to be constrained by rank (Zhang et al., 2023; Liu et al., 2024; Zeng and Lee, 2024). In high dimensional representation spaces, two branches can appear nearly orthogonal as individual directions while still sharing the same dimensions in how they organize token-to-token similarity. Direction-level metrics such as cosine similarity cannot detect this form of overlap, so we use Centered Kernel Alignment (CKA) (Gretton et al., 2005; Kornblith et al., 2019), which compares two representations by how much they share this token-to-token similarity structure.

Definition. For representation matrices X, Y centered along the token axis, linear CKA is

$$\text{CKA}(X, Y) = \frac{\|X^\top Y\|_F^2}{\|X^\top X\|_F \|Y^\top Y\|_F}, \quad (15)$$

which lies in $[0, 1]$: values near 1 indicate that the two representations organize token-to-token similarity in the same way, values near 0 indicate minimal shared structure. CKA makes it better suited

than cosine similarity for this comparison.

Observation. We separately construct a knowledge LoRA from retrieved passages and train a reasoning LoRA for latent reasoning tokens, without any differentiation constraint, and measure similarity between the two branches (Fig. 6). At the output layer, cosine similarity is ≈ 0 at both the weight ($\Delta W = BA$, 0.00) and activation (0.01) levels, yet CKA reaches 0.92: the two branches appear disentangled under cosine but organize token-to-token similarity almost identically. This overlap is negligible in intermediate layers and emerges almost entirely at the output layer. Such redundancy suggests that the two signals interfere when they share the same rank dimensions, motivating our explicit differentiation of the two branches via the rank duo mask and rank-specific gates.

E Non-causal CoT Sensitivity Analysis

Using LLaMA-3.2-1B-Instruct, we ran four controlled interventions (perturbing exemplars, triggers, stop rules, and answer extraction) on 2Wiki-MultihopQA, the benchmark where multi-hop CoT prompting matters most. All comparisons use a fully matched setup: fixed per-query passage and knowledge parameters, an identical chat template, unified greedy decoding and stop rules, and the official PRAG extractor for all methods.

Exemplars and triggers. We cross exemplar presence (8-shot CoT) $\times$ trigger presence ("Let's think step by step"). DuoPRAG's native condition is P00, PRAG/DyPRAG's is P11. See Table 14 for details. The baseline's benefit is attributable almost entirely to the exemplars, not the trigger. The trigger alone lengthens output while roughly halving F1, so longer lexical reasoning does not by itself recover the exemplar benefit. The matched-prompt cells rule out the alternative explanation that DuoPRAG merely benefits from a favorable native prompt. Results are also robust to trigger wording: two semantically equivalent paraphrases shift EM by ≤ 1.6 for all methods.

Perturbing exemplars content. Table 15 reports EM and F1 for each perturbation, with the trigger fixed to native. For the baselines, exemplar presence matters most. Removing exemplars costs PRAG 16.3 EM and removing rationales costs it 20.2 EM, with comparable drops of 15.5 and 16.9 EM for DyPRAG. Format mismatch has a smaller and less consistent effect. PRAG loses 3.1 EM un-

der mismatched rationale and 4.0 EM under mismatched CoT, while DyPRAG loses 2.5 EM under mismatched CoT but gains 1.9 EM under mismatched rationale. Corrupting facts or demonstration answers within a structurally coherent exemplar causes no degradation at all, and the wrong-answer row improves every system, which suggests a property of the condition rather than of any one model. Whether exemplars are present therefore matters far more than whether their format aligns with the task, and format alignment in turn matters far more than whether their content is correct. This ordering extends the finding of Min et al. (2022) to the parametric-RAG setting. Output length also stays near-constant at around 4.6 tokens, consistent with reasoning performed latently rather than from lexical CoT.

Stop rules and answer extraction. To control for this factor, we use the same stopping criterion for all methods and conditions: generation terminates upon EOS token or when the maximum number of new tokens is reached. Across all 17 conditions, DuoPRAG generates only 4.5–4.8 visible tokens on average and never reaches the token limit, demonstrating that its results are invariant to the maximum-generation cap. Although the baselines reach the cap in some settings (up to 50–70% of generations in the most affected cells), the answer-bearing phrase "the answer is" is typically generated before truncation. Moreover, their performance under the native condition closely matches that obtained with their official evaluation setup. These observations indicate that truncation is not the binding constraint and does not explain DuoPRAG's performance gains.

F Parameter-matched Baseline

We counted the trainable parameters of all implemented methods and added parameter-budget-controlled comparisons (Table 16). We report both the injected LoRA update ΔW and the total trainable parameters, as the two differ across methods.

DuoPRAG's budget, bracketed from below and above. DuoPRAG uses 10.17M trainable parameters in total (two rank-2 LoRAs plus mask and gating). To control for this, we scaled PRAG to $r=16$ (7.86M, just below our budget) and $r=32$ (15.72M, 55% above it). Both fall substantially short of DuoPRAG: 29.72 and 29.36 avg. F1 vs. our 35.79, and this despite PRAG training a fresh adapter per document, while our reasoning components

Prompt condition	PRAG		DyPRAG		DuoPRAG	
	EM	F1	EM	F1	EM	F1
P00: no exemplar, no trigger	9.67	18.31	5.87	19.21	27.73	34.01
P01: trigger only	6.33	9.94	6.53	10.03	26.93	33.26
P10: exemplar only	22.20	27.96	22.20	27.78	25.20	31.21
P11: exemplar + trigger	22.66	28.42	22.07	27.57	25.60	31.80

Table 14: Exemplars and triggers: 2×2 factorial analysis of prompt composition (EM / F1). PRAG and DyPRAG depend heavily on in-context exemplars, whereas DuoPRAG remains stable across all four conditions.

Exemplar condition	Perturbs	PRAG		DyPRAG		DuoPRAG	
		EM	F1	EM	F1	EM	F1
correct (native)	(reference)	22.66	28.42	22.07	27.57	25.60	31.80
none	presence	6.33	9.94	6.53	10.03	26.93	33.26
answer-only	rationale presence	2.47	17.40	5.13	17.72	27.74	33.45
mismatched rationale	format alignment	19.53	25.31	24.00	29.47	25.00	31.33
mismatched CoT	format alignment	18.67	23.63	19.53	24.17	24.67	31.31
invalid rationale (wrong facts, correct answer)	factual validity	23.07	29.14	23.87	29.55	25.33	31.39
wrong answer (type-matched)	answer correctness	25.00	31.32	24.53	30.35	28.87	35.10
plausible-but-wrong CoT	both	25.73	31.46	23.80	29.32	28.07	33.81

Table 15: Exemplar content (EM / F1, trigger fixed to native). The *correct (native)* row is the unperturbed reference. Each subsequent row perturbs the property named in the second column.

are trained once and shared. Notably, performance does not improve from $r=16$ to $r=32$, indicating saturation rather than under-parameterization.

A competitor already exceeds our budget. DyPRAG at its default $r=2$ already trains 37.75M parameters ($3.7 \times$ DuoPRAG) for its shared hyper-network, yet reaches only 27.57 avg. F1; scaling it to $r=8$ (132.1M) further drops F1 to 24.53. Together, these results indicate that across both baselines, more trainable parameters do not translate into gains, and that DuoPRAG’s improvements stem from its architecture rather than from parameter count.

G Cross-benchmark score overview

We provide two visualizations to summarize DuoPRAG’s behavior across benchmarks.

Simple vs. multi-hop QA. Figure 8 contrasts simple, single-hop QA (x -axis) against task-matched multi-hop reasoning QA (y -axis), pairing PopQA with 2WikiMultiHopQA and CWQ with HotpotQA across the three backbones. DuoPRAG (red) consistently lies toward the upper-right, advancing simple and reasoning tasks jointly, whereas the baseline and prior parametric-RAG variants tend to improve along only one axis.

In-domain vs. OOD. Figure 9 gives a compact view of the overall score landscape across in-domain (§5.2) and OOD evaluations (§5.3), with points closer to the upper-right indicating stronger performance on both axes. The two axes are not intended to imply any predefined semantic pairing. They show how each method’s scores distribute across benchmark groups and backbones.

H Reasoning step augmentation prompt

We augmented the reasoning steps in the training data using the prompt template in Figure 7. Given a passage, question, and gold answer, the prompt instructs the model to generate passage-grounded step-by-step reasoning and to output the given answer verbatim in the final line.

I Case Study

To qualitatively examine DuoPRAG’s behavior, we present representative successful and failed examples on 2WikiMHQA in Tables 17 and 18. Passages are encoded into W_{knw} rather than placed in the prompt, and the model reasons over K latent tokens. The final answer is shown below, and the italicized line annotates the implicit chain.

The successful cases (S1–S3) span different reasoning structures and different degrees to which

Method	r	Inj. ΔW	Learnable	2WQA		HQA		PopQA		CWQ		Avg.	
				EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
PRAG-woC	2	0.98M	0.98M/doc [†]	20.13	25.92	19.00	25.35	12.00	23.58	26.00	35.86	19.57	26.51
PRAG-woC	16	7.86M	7.86M/doc [†]	23.87	29.84	22.44	28.85	21.33	24.38	25.33	35.81	23.24	29.72
PRAG-woC	32	15.72M	15.72M/doc [†]	23.47	29.44	21.11	28.02	22.67	25.20	24.33	34.80	22.89	29.36
DyPRAG-woC	2	0.98M	37.75M [‡]	24.27	29.91	19.56	25.97	7.33	11.33	28.33	36.86	21.57	27.57
DyPRAG-woC	8	3.93M	132.1M [‡]	21.60	27.13	17.34	24.24	8.33	9.43	28.00	37.31	18.82	24.53
DuoPRAG-woC	2+2	1.97M	10.17M	29.93	37.17	21.22	29.26	30.67	30.67	36.00	46.08	29.46	35.79

Table 16: Parameter-budget-controlled comparison on LLaMA-3.2-1B-Instruct. PRAG with $r = 16$ and $r = 32$ provides lower- and over-budget controls around DuoPRAG’s 10.17M trainable parameters, respectively. [†] One document-specific adapter trained per document; [‡] a single shared hypernetwork trained once.

the required facts appear in the internalized context: S2 and S3 are fully supported by W_{knw} , while S1 requires a leaf fact (the performer’s birthplace) absent from the context. In each case DuoPRAG’s two-branch composition, reasoning via W_{rsn} over knowledge in W_{knw} and the base prior, produces the correct answer.

The failure cases highlight two recurring patterns. In F2 and F3, the required fact is absent from W_{knw} and the answer must fall back on the base prior, which fails for low-frequency entities at either the bridge (F2) or leaf (F3) position. In F1, the relevant facts are present in W_{knw} , but the prediction returns the salient intermediate entity rather than completing the second hop. We present these as illustrative observations on a small set of examples rather than as systematic evidence.

PROMPT: Reasoning Step Augmentation

Task instruction. You are given a passage, a question about the passage, and the correct answer. Generate a chain-of-thought that uses the passage to derive the answer.

Generation rules.

- Each [S#] is one logical reasoning step.
- Each step should reference relevant evidence from the passage.
- The response must start with “Let’s think step by step.”
- The final line must be “The answer is <answer>” using the given answer verbatim.
- The model must output nothing else, including preambles or closing remarks.

Input fields.

```
Passage: {context}  
Question: {question}  
Answer: {answer}
```

Required output format.

```
Let’s think step by step.  
[S1] <reasoning step that uses information from the passage>  
[S2] <reasoning step>  
...  
The answer is <answer>
```

Figure 7: Prompt template used to augment training data with passage-grounded chain-of-thought reasoning.

PRAG Benchmark: Simple QA vs. Reasoning QA

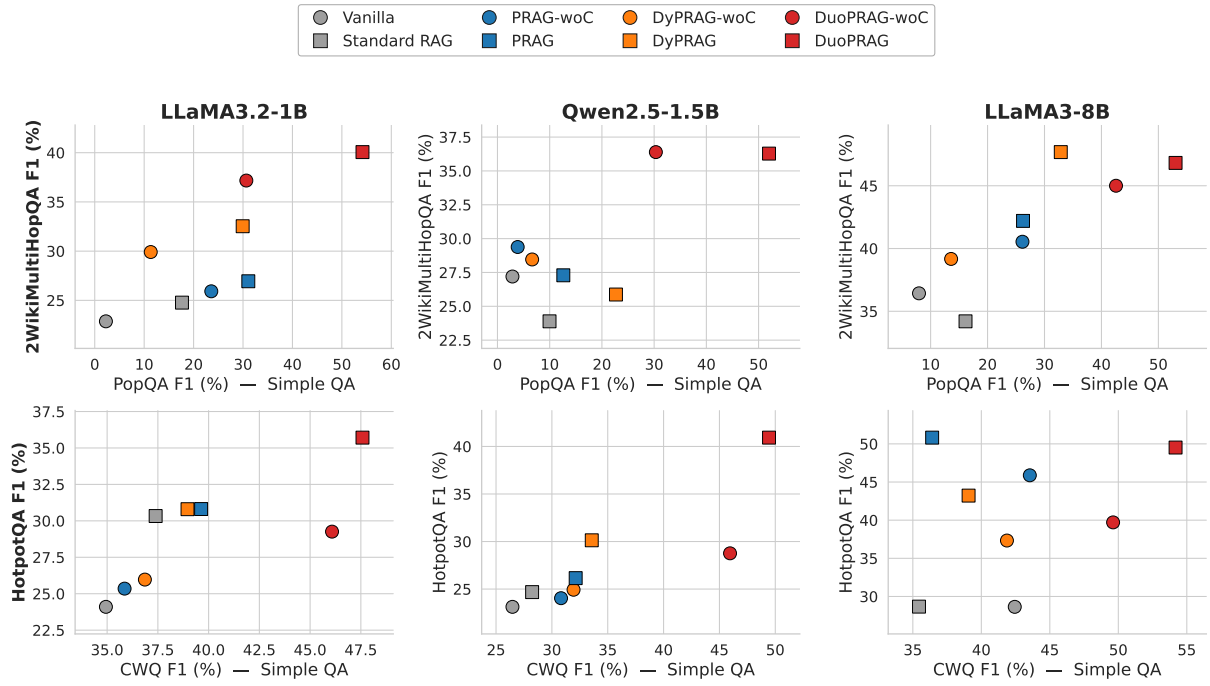


Figure 8: Each panel plots a simple question-answering task score (x -axis) against a task-matched multi-hop reasoning question-answering task score (y -axis), across three backbones (columns). Metrics are token-F1 for all tasks. Marker color denotes the method family, and marker shape denotes the prompting setting: $\bullet$ query-only inference without context, and $\blacksquare$ in-context inference with context prepended to the prompt. Upper-right is better.

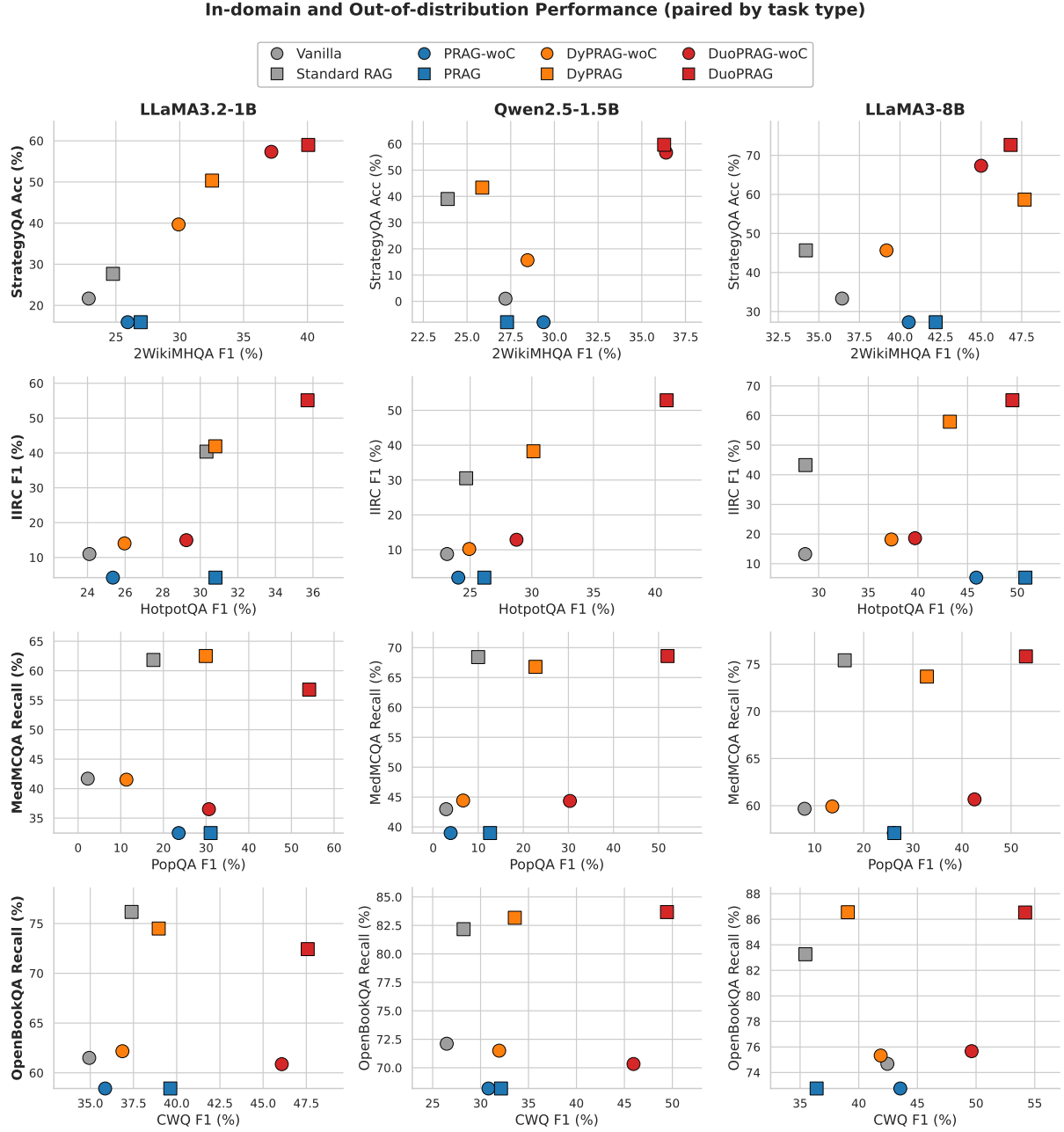


Figure 9: Each panel plots an in-domain score (x -axis) against a task-matched OOD score (y -axis), across three backbones (columns) and four task pairs (rows). Metrics are token-F1 for IIRC, accuracy for StrategyQA, and recall for OpenBookQA and MedMCQA. PRAG is placed on the x -axis because its OOD scores are unavailable. Upper-right is better.

S1 — Bridge, context entity + parametric leaf (2WikiMHQA, compositional)	
Q: Where was the performer of song <i>I See You (Theme from Avatar)</i> born?	
Context (W_{knw}): “I See You (Theme from Avatar)” ... the official theme song recorded by British singer <i>Leona Lewis</i> for the 2009 film <i>Avatar</i> [...]	
<i>hop 1: song → performer = Leona Lewis (from context); hop 2: Leona Lewis → birthplace (from parametric prior — not in context).</i>	
DuoPRAG: London, UK	Answer: London, UK

S2 — Comparison, both operands grounded (2WikiMHQA, comparison)	
Q: Which film was released more recently, <i>Toolbox Murders</i> or <i>I Met You in Naples</i> ?	
Context (W_{knw}): <i>Toolbox Murders</i> [... 2004 ... re-released as <i>Toolbox Murders 2</i> in 2015]; <i>I Met You in Naples</i> ... is a 1946 Italian musical melodrama film [...]	
<i>compare the two release years (2004 vs. 1946), return the more recent.</i>	
DuoPRAG: Toolbox Murders	Answer: Toolbox Murders

S3 — Two-hop, fully grounded (2WikiMHQA, compositional)	
Q: Where was the place of death of Tamatoa VI’s father?	
Context (W_{knw}): [Tamatoa VI’s father] married Tetua-nui Ha’amarurai ... <i>He died at Huahine in 1905.</i> Tamatoa VI ... reigned on ... Huahine [...]	
<i>hop 1: subject → father; hop 2: father → place of death = Huahine (both in context).</i>	
DuoPRAG: Huahine	Answer: Huahine

Table 17: Successful DuoPRAG cases (LLaMA-3-8B, closed-book). Above the dotted line is the input: the question and the supporting facts internalized as the knowledge LoRA W_{knw} (condensed; [. . .] marks elision).

F1 — Hop collapse: returns the intermediate entity (2WikiMHQA, inference)	
Q: Who is Gaston, Duke of Orléans’s paternal grandmother?	
Context (W_{knw}): Gaston, Duke of Orléans ... was the third son of King <i>Henry IV of France</i> and his wife <i>Marie de’ Medici</i> [...]	
<i>required: father (Henry IV) → his mother (= Jeanne d’Albret). Model stops one hop short and returns the salient name in context.</i>	
DuoPRAG: Marie de Medici	Answer: Jeanne d’Albret
Analysis: Returns Gaston’s <i>mother</i> (directly stated) instead of the <i>paternal grandmother</i> ; the two-hop kinship chain collapses to one hop.	

F2 — Parametric hallucination on a rare entity (2WikiMHQA, inference)	
Q: Who is the father-in-law of T D?	
Context (W_{knw}): T D ... was the wife of <i>Thiu Tr</i> and mother of T Đc [...]	
<i>required: husband (Thiu Tr) → his father (= Minh Mng). Husband’s father is not stated; the bridge hop is unsupported.</i>	
DuoPRAG: Cao Pi	Answer: Minh Mng
Analysis: The missing hop is filled from the parametric prior, emitting an unrelated East-Asian ruler (Cao Pi); knowledge for this low-frequency entity was not reliably internalized in W_{knw} .	

F3 — Correct bridge, wrong leaf fact (2WikiMHQA, compositional)	
Q: What is the place of birth of the performer of song <i>If You Want to Be My Woman</i> ?	
Context (W_{knw}): “If You Want to Be My Woman” ... written and recorded by American country artist <i>Merle Haggard</i> [...]	
<i>hop 1: song → performer = Merle Haggard (correct, from context); hop 2: birthplace (parametric) — recalled incorrectly.</i>	
DuoPRAG: Camas	Answer: Bakersfield, California
Analysis: Mirrors S1 in structure but the leaf parametric lookup fails: the bridge entity is right, yet the birthplace is wrong.	

Table 18: Failure cases for DuoPRAG (LLaMA-3-8B, closed-book). Same layout as Table 17.

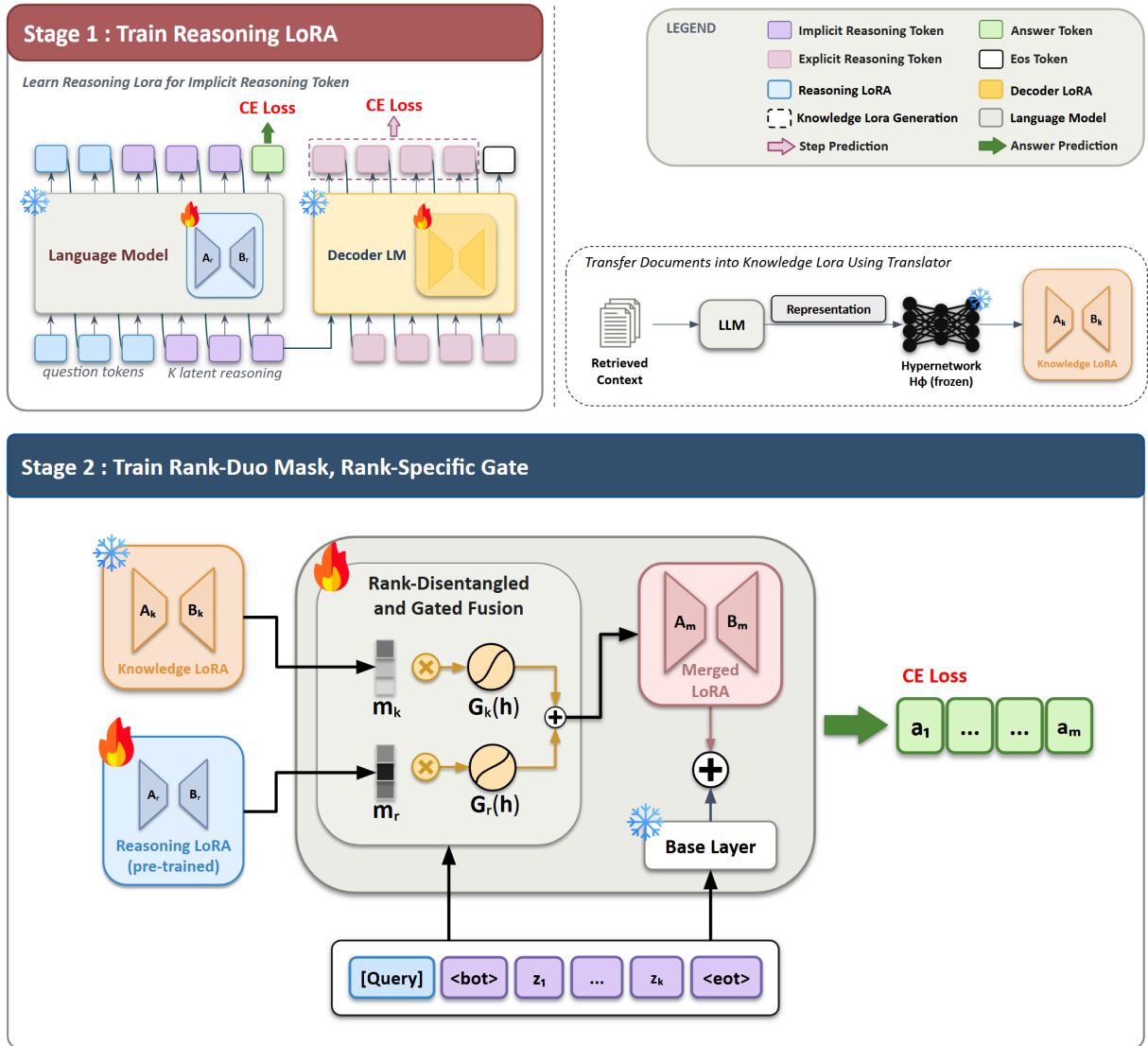


Figure 10: Two-stage training pipeline. Stage 1 learns a Reasoning LoRA over latent reasoning tokens with supervision from a decoder LM, while a frozen hypernetwork translates retrieved documents into a Knowledge LoRA. Stage 2 fuses the two LoRAs through rank-wise masks and input-dependent gates, producing a merged adapter for answer prediction.

Prompt Format of DuoPRAG

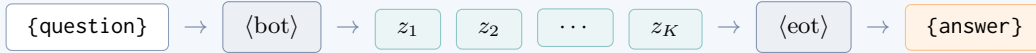


Figure 11: Prompt format of DuoPRAG in our experiments.

Prompt Format of No-CoT

You should answer the question by referring to the knowledge provided below and integrating your own knowledge.

Passage 1: {passages[0]}

Passage 2: {passages[1]}

Passage 3: {passages[2]}

Question: {question}

The answer is {answer}

Figure 12: Prompt format of No-CoT in our experiments.

Prompt Format of CoT

You should reference the knowledge provided below and combine it with your own knowledge to answer the question. Please follow the format of the examples about how to answer the questions.

Question: fewshot_q[0]

Answer: fewshot_a[0]

Question: fewshot_q[1]

Answer: fewshot_a[1]

Question: fewshot_q[2]

Answer: fewshot_a[2]

...

Here are some reference.

Passage 1: {passages[0]}

Passage 2: {passages[1]}

Passage 3: {passages[2]}

Let's think step by step. Answer the questions in the same format as above.

Question: {question}

Answer: {answer}

Figure 13: Prompt format of CoT in our experiments.