Latent Abstraction for Retrieval-Augmented Generation

Ha Lan N.T*, **Minh-Anh Nguyen***, **Dung D. Le**
Center for AI Research, VinUniversity, Vietnam
{lan.nth, minh.na2, dung.ld}@vinuni.edu.vn
**Equal contribution*

Abstract

Retrieval-Augmented Generation (RAG) has become a standard approach for enhancing large language models (LLMs) with external knowledge, mitigating hallucinations, and improving factuality. However, existing systems rely on generating natural language queries at each hop and maintaining a strict architectural separation between retriever and generator, preventing them from leveraging the full representational capacity of the LLM. We propose **LAnR** (Latent Abstraction for RAG), a unified framework in which a single LLM jointly performs encoding, retrieval, and generation entirely within its own latent space. Rather than generating textual queries, LAnR produces dense retrieval vectors from the hidden states of a designated `[PRED]` token and uses them to match against encoded document representations from the same model. Furthermore, LAnR adaptively decides when sufficient evidence has been retrieved using a lightweight MLP control head over those same hidden states, eliminating explicit token-level stopping reasoning. Extensive experiments on five QA benchmarks spanning single-hop and multi-hop settings demonstrate that LAnR outperforms existing RAG methods, while achieving improved inference efficiency through reduced number of retrieval calls and tighter model integration.

1 Introduction

Retrieval-Augmented Generation (RAG) has emerged as a standard paradigm for enhancing LLMs with external knowledge, improving factuality, and mitigating hallucinations [2, 13, 30]. By retrieving relevant documents from external corpora and conditioning generation on this information, RAG systems enable LLMs to access up-to-date and domain-specific knowledge beyond their parametric memory. Despite these advantages, existing RAG frameworks exhibit several limitations. First, they rely heavily on explicit text-based retrieval, where the model must generate natural language queries to interact with a separate retrieval module. Second, they enforce a strict architectural separation between the retriever and the generator, often requiring independently trained components or additional fine-tuning via reinforcement learning (RL) or supervised fine-tuning (SFT) LLMs to work with retrieval modules [34, 20, 38, 3]. These design choices introduce substantial computational overhead, limit the full utilization of the LLM's shared representational capacity, and increase inference latency due to the need for explicit token-level generation at each step, illustrated in Figure 1.

Recent advances in latent reasoning suggest an alternative paradigm, where LLMs perform reasoning directly in hidden representation space rather than through fully verbalized intermediate steps [52, 33, 9, 45]. Such approaches demonstrate that latent trajectories in hidden states can encode rich reasoning processes without explicit token generation. However, this paradigm remains largely unexplored in the context of retrieval-augmented systems [9]. Extending latent reasoning to RAG is, however, non-trivial. First, unlike standard reasoning tasks, RAG lacks high-quality chain-of-thought supervision for constructing retrieval queries, making it difficult to distill text-based reasoning

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

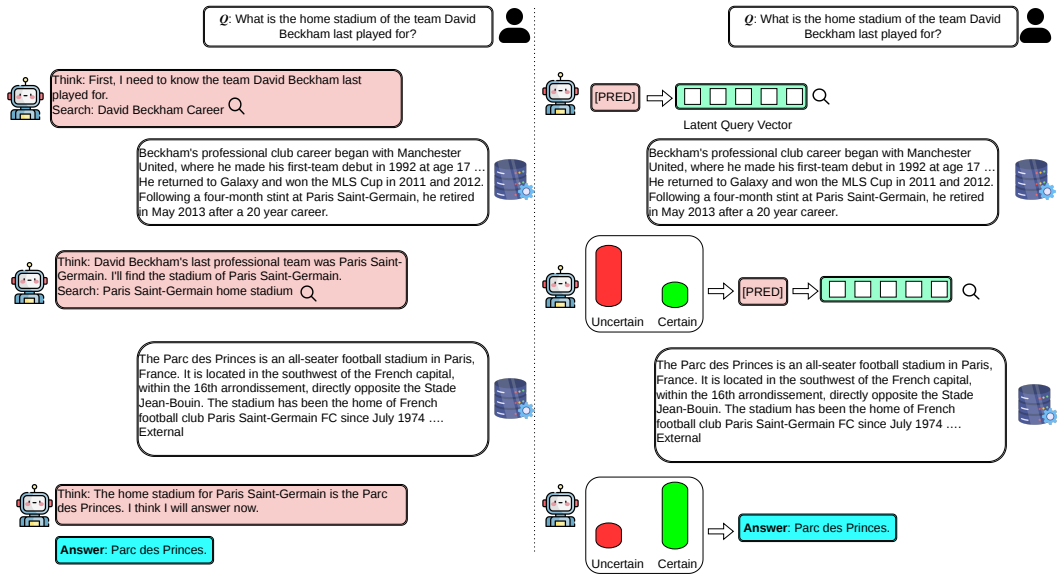


Figure 1: **Comparison between conventional RAG and LAnR for multi-hop QA.** Conventional RAG performs explicit reasoning at each hop, including generating intermediate text, forming search queries, and deciding whether to continue retrieval. In contrast, LAnR operates in latent space: a special token `[PRED]` produces query vectors from hidden states, while a lightweight MLP controls the retrieval process, enabling more efficient and integrated reasoning.

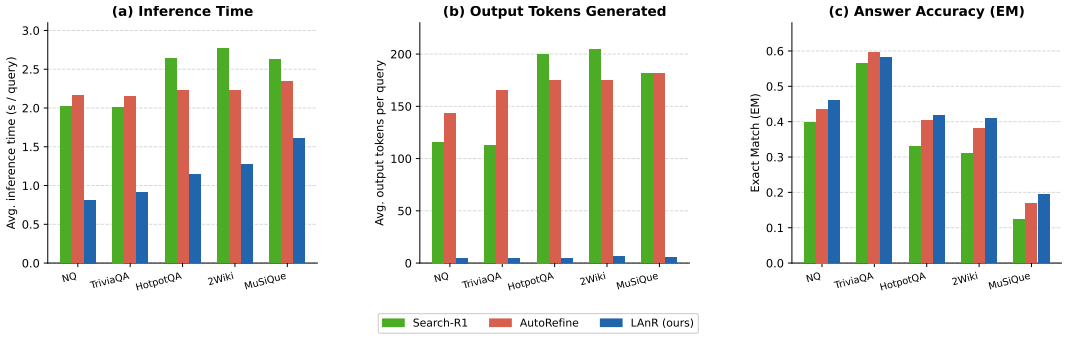


Figure 2: Comparison of inference time, generated tokens, and Exact Match accuracy between prior RAG methods and LAnR. Latent retrieval reduces latency and token generation while maintaining strong performance. All methods use Qwen2.5-3B-Instruct as backbone model to ensure a fair comparison.

into latent representations [15, 44]. Second, existing RAG architectures typically rely on separate embedding models for retrieval, which necessitate explicit natural language queries. Consequently, integrating both reasoning traces and retrieval queries into a shared latent space requires a unified model that simultaneously supports retrieval and generation.

To address these challenges, we propose a unified **latent abstraction retrieval-augmented generation** (LAnR) framework, wherein retrieval is conducted directly within the LLM's internal representation space. Instead of generating textual queries, the model produces latent query vectors derived from the hidden states of designated special tokens `[PRED]`. These vectors are used to retrieve documents from a vector database constructed using representations from the same LLM, enabling tight integration between retrieval and generation, detailed in Section 3. To support multi-hop reasoning, we further introduce a lightweight multi-layer perceptron (MLP) [29] head that predicts whether additional retrieval steps are necessary. This design is motivated by empirical observations that uncertainty signals, such as entropy in the token distribution, correlate with the need for further information acquisition, illustrated in Appendix C.

Extensive experiments across multiple benchmarks demonstrate that our approach achieves superior performance compared to state-of-the-art RAG methods, while significantly reducing inference latency due to minimal text generated, illustrated in Figure 2 and eliminating the need for serving a separately trained dense retriever. We provide our code at the repository `https://github.com/akBear23/LAnR`. We summarize our main contributions as follows:

1. We propose a novel latent RAG framework that leverages the internal representation space of an LLM to jointly perform document encoding, retrieval, and generation, thereby unifying these components within a single architecture.
2. We design an implicit retrieval control mechanism, implemented as a lightweight auxiliary head, which uses the LLM's hidden representation to adaptively determine whether additional retrieval is needed, without relying on explicit text-based reasoning.
3. We introduce a new approach for constructing retrieval query vectors directly from the LLM's latent representations, eliminating the need for intermediate natural language query generation.

2 Latent Query Construction

We begin by investigating latent query construction, where the LLM bypasses explicit textual query generation and instead produces a dense retrieval vector directly from its hidden representations. To enable this capability, the LLM is trained to internalize query formulation, allowing it to infer search intent from the input and map it directly into a retrieval vector. Throughout the paper, $\mathcal{M}$ is a Qwen2.5 backbone [46] (3B or 7B, Base or Instruct variant), which we extend with a special `[PRED]` token and a lightweight retrieval control head (Section 3); all parameters of the backbone are fine-tuned, and further implementation details are given in Appendix B. Formally, let $x = (x_1, \ldots, x_T)$ denote an input query token sequence and $\mathcal{M}$ an autoregressive language model with last-layer hidden states $h_t \in \mathbb{R}^d$. We append a designated token `[PRED]` to the input, forming $\tilde{x} = (x_1, \ldots, x_T, \texttt{[PRED]})$. Through causal self-attention, its hidden state attends to the full preceding context, yielding a latent query vector:

$$q = h_{\texttt{[PRED]}} \in \mathbb{R}^d. \tag{1}$$

Optionally, $N \geq 1$ consecutive `[PRED]` tokens can be appended, allowing the model to construct the query representation over multiple latent steps, which promotes higher-level abstraction and extends the LLM's latent reasoning process; we default to $N{=}1$ and study this design choice in Appendix E.

Unlike prior LLM-based encoders [40, 4], where the query vector encodes a fully-formed textual query, here q is produced by the model's own reasoning over context. This distinction becomes essential in the multi-turn setting, detailed in Section 3, where intermediate sub-queries have no canonical textual form and must be inferred from the running context of previously retrieved evidence. Each representation of the document D_i are obtained from the same model $\mathcal{M}$ via last-token pooling [10, 13]:

$$d_i = h_{D_i}^{\text{last}} \in \mathbb{R}^d. \tag{2}$$

Embedding queries and documents within a shared representation space induced by the same parameters eliminates the need for a separate retriever and preserves the model's generative capability for downstream answer production. We train the `[PRED]` representation with a standard contrastive objective [39, 5]:

$$\mathcal{L}_{\text{CL}} = -\log \frac{\exp\big(\text{sim}(q, d^+)/\tau\big)}{\exp\big(\text{sim}(q, d^+)/\tau\big) + \sum_{j=1}^{N^-} \exp\big(\text{sim}(q, d_j^-)/\tau\big)}, \tag{3}$$

where d^+ is a positive document, $\{d_j^-\}_{j=1}^{N^-}$ are hard negatives mined from the model's own retrieval errors and periodically refreshed following ANCE [43], $\text{sim}(\cdot,\cdot)$ is cosine similarity, and τ is a temperature parameter. This setup gives us a query vector at any point in the model's forward pass. Building on this foundation, Section 3 introduces a multi-turn training objective designed to leverage both properties effectively.

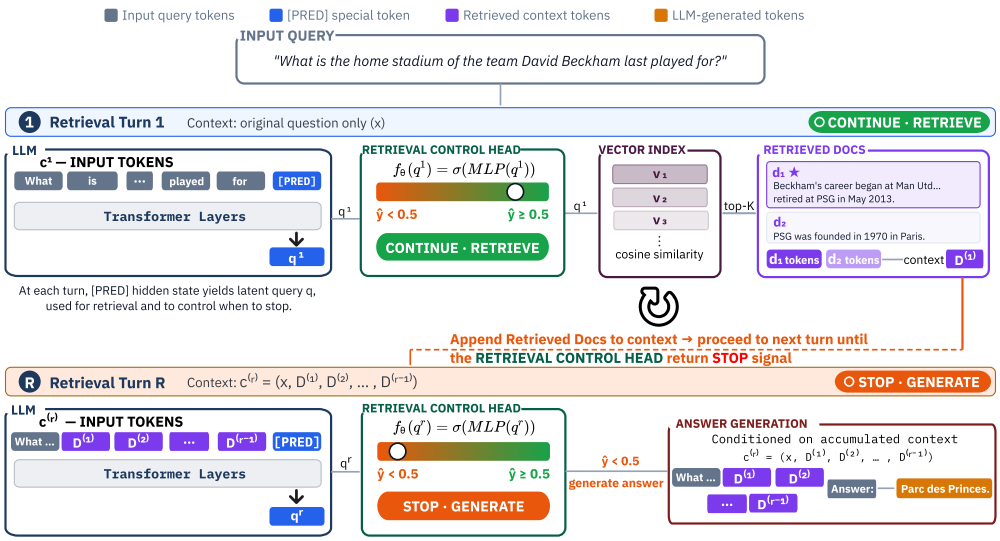


Figure 3: **Overview of LAnR.** Queries are injected into the LLM and combined with a [PRED] token to form a latent query from hidden representations. This latent query is used for retrieval and to decide whether further retrieval is needed via a lightweight MLP Retrieval Control Head. The LLM then generates the answer from the retrieved context.

3 End-to-End Framework and Training Objective

Building on the latent query construction in Section 2, we extend the framework from single-step implicit retrieval to a multi-turn setting. The central challenge is to determine ***when*** to trigger additional retrieval and ***what*** to retrieve at each step, while relying solely on latent reasoning in hidden space without generating intermediate text tokens. We address both through: (1) an MLP-based retrieval control head that decides whether further retrieval is necessary, and (2) an adaptive contrastive target mechanism that dynamically updates the retrieval objective based on remaining unretrieved evidence.

At each retrieval turn $r = 1, 2, \ldots, R$, the model constructs a latent query from the current context, which now includes both the original question and any previously retrieved documents, and decides whether to retrieve again or to proceed with answer generation. Let $c^{(r)}$ denote the accumulated context at turn r:

$$c^{(r)} = \big(x,\ D^{(1)},\ D^{(2)},\ \ldots,\ D^{(r-1)}\big),$$

where $D^{(s)}$ denotes the set of top-K documents retrieved at turn s. At each turn, we append [PRED] to $c^{(r)}$ and extract the latent query vector: $q^{(r)} = h^{(r)}_{\texttt{[PRED]}} \in \mathbb{R}^d$, which is used both for retrieval via similarity matching against the document index, detailed in Equation 2 and as input to the retrieval control head described below. The process repeats until the control head signals termination or a maximum number of turns R is reached, after which the model generates the final answer conditioned on the full accumulated context $c^{(R+1)}$.

MLP-based Retrieval Control Head. An additional component of our framework is a lightweight MLP head that determines whether the currently retrieved evidence is sufficient to answer the query, or whether additional retrieval is required. At each turn r, the Retrieval Control Head f_θ takes $q^{(r)}$ as input and produces a binary decision:

$$\hat{y}^{(r)} = f_\theta\big(q^{(r)}\big) = \sigma\big(\mathrm{MLP}(q^{(r)})\big) \in [0, 1], \tag{4}$$

where σ is the sigmoid function. A prediction $\hat{y}^{(r)} < 0.5$ indicates that all necessary evidence has been retrieved and the model should proceed to generation; $\hat{y}^{(r)} \geq 0.5$ signals that additional retrieval is needed. The training label is derived from the model's own retrieval state. Let $\mathcal{P}$ denote the full set of positive (gold) documents required to answer the query, and $\mathcal{R}^{(r)} = D^{(1)} \cup \cdots \cup D^{(r)}$ the

documents retrieved up to turn r. The ground-truth label is:

$$y^{(r)} = \begin{cases} 0, & \text{if } \mathcal{P} \subseteq \mathcal{R}^{(r)}, \\ 1, & \text{otherwise,} \end{cases} \tag{5}$$

i.e., the label is 0 (stop) when all positive documents have been successfully retrieved, and 1 (continue) otherwise. This label requires no additional stopping supervision: it is computed automatically from the model's own retrieval trajectory by checking whether the retrieved set covers $\mathcal{P}$, reusing the same evidence annotations already required to train the retriever (Eq. 7). No manually annotated stopping decisions, retrieval trajectories, or external LLM judges are needed. For datasets without annotated supporting passages, $\mathcal{P}$ is constructed by the same distant-supervision procedure used for retriever training (Appendix D). The control head is trained with binary cross-entropy:

$$\mathcal{L}_{\text{ctrl}} = -\sum_{r=1}^{R} \Big[y^{(r)} \log \hat{y}^{(r)} + (1 - y^{(r)}) \log(1 - \hat{y}^{(r)}) \Big]. \tag{6}$$

In the multi-turn setting, naively using the same contrastive target at every turn is suboptimal: once a positive document has been retrieved, it should no longer serve as the target for subsequent queries. At turn r, the set of leftover positive documents is: $\mathcal{P}^{(r)} = \mathcal{P} \setminus \mathcal{R}^{(r-1)}$, where $\mathcal{R}^{(0)} = \emptyset$. The contrastive loss at turn r uses a positive document sampled from $\mathcal{P}^{(r)}$:

$$\mathcal{L}_{\text{CL}}^{(r)} = -\log \frac{\exp\big(\text{sim}(q^{(r)}, d^{+(r)})/\tau\big)}{\exp\big(\text{sim}(q^{(r)}, d^{+(r)})/\tau\big) + \sum_{j=1}^{N^-} \exp\big(\text{sim}(q^{(r)}, d_j^-)/\tau\big)}, \tag{7}$$

where $d^{+(r)}$ is the embedding of a document drawn from $\mathcal{P}^{(r)}$. This steers each successive turn toward the missing evidence, avoiding redundant retrieval. The full training loss combines next-token prediction, multi-turn contrastive retrieval, and the control head objective:

$$\mathcal{L} = \sum_{r=1}^{R} \mathcal{L}_{\text{CL}}^{(r)} + \lambda \, \mathcal{L}_{\text{NTP}} + \mu \, \mathcal{L}_{\text{ctrl}}, \tag{8}$$

where λ and μ are hyperparameters balancing the three objectives. $\mathcal{L}_{\text{NTP}}$ denotes the standard next-token prediction loss, which maintains the generative capability of the LLM. We apply loss masking to retrieved tokens and `[PRED]` tokens, so that the optimization objective is computed only over tokens generated by the LLM, excluding retrieved content from gradient updates, following prior works [20, 34].

Inference. At inference time, the contrastive and control losses are not used; the trained model directly produces latent queries and stopping decisions. Retrieval starts from the input question x. At each turn r, the model appends a `[PRED]` token to the current context $c^{(r)}$ and runs a single forward pass to obtain $q^{(r)} = h^{(r)}_{\texttt{[PRED]}}$. The retrieval control head then computes $\hat{y}^{(r)} = f_\theta(q^{(r)})$. If $\hat{y}^{(r)} \geq 0.5$, $q^{(r)}$ is used as the query for nearest-neighbour search against the document index built with Eq. 2; the top-K documents are appended to the context, and the next turn begins. If $\hat{y}^{(r)} < 0.5$, retrieval terminates. Importantly, no text is generated during the retrieval phase: the model produces neither intermediate reasoning tokens nor textual search queries, only latent `[PRED]` states. After retrieval terminates (or the maximum of $R = 4$ rounds is reached), the prompt `Answer:` is appended to the accumulated context, and the model decodes the final answer autoregressively in a single generation pass. Figure 3 illustrates this procedure, and Appendix H provides hop-by-hop qualitative examples.

4 Experiments

In this section, we investigate the following research questions (RQs):

RQ1. How effective is LAnR compared to conventional agentic RAG systems in answer quality across single-hop and multi-hop benchmarks, and can its implicit retriever match text embedding models in retrieval accuracy?

RQ2. Does LAnR's retrieval control head adaptively allocate search hops based on query complexity, and does each retrieval hop find relevant evidence more effectively than text-based iterative search?

Table 1: **(RQ1)** Accuracy comparison of LAnR and baselines on QA benchmarks. **Bold** and underline denote best and second-best results, respectively. "-Instruct" and "-Base" indicate the corresponding Qwen2.5-3B backbone variants.

	Single-Hop QA		Multi-Hop QA			
Methods	NQ	TriviaQA	HotpotQA	2Wiki	Musique	Avg.
w/o Retrieval						
Direct Generation	0.106	0.288	0.149	0.244	0.020	0.161
SFT	0.249	0.292	0.186	0.248	0.044	0.204
w/ Single-Hop Retrieval						
Naive RAG [25]	0.348	0.544	0.255	0.226	0.047	0.284
w/ Multi-Hop Retrieval						
Search-o1 [27]	0.238	0.472	0.221	0.218	0.054	0.241
IRCoT [38]	0.111	0.312	0.164	0.171	0.067	0.165
ReSearch-Instruct [7]	0.365	0.571	0.351	0.272	0.095	0.331
ReSearch-Base [7]	0.427	0.597	0.305	0.272	0.074	0.335
Search-R1-Instruct [20]	0.397	0.565	0.331	0.310	0.124	0.345
Search-R1-Base [20]	0.421	0.583	0.297	0.274	0.066	0.328
AutoRefine-Instruct [34]	0.436	0.597	0.404	0.380	0.169	0.397
AutoRefine-Base [34]	**0.467**	**0.620**	0.405	0.393	0.157	0.408
LAnR-Instruct	0.460	0.613	**0.419**	**0.408**	**0.193**	**0.419**
LAnR-Base	0.455	0.610	0.417	0.402	0.187	0.414

RQ3. Can LAnR reduce inference time by minimizing retrieval iterations and improving efficiency through more abstract query representations?

RQ4. What is the individual contribution of each training objective ($\mathcal{L}_{\text{NTP}}$, $\mathcal{L}_{\text{CL}}$, $\mathcal{L}_{\text{ctrl}}$) and the retrieval control head to LAnR's end-to-end performance?

4.1 Experimental Setup

Datasets. We evaluate our method on five open-domain QA benchmarks, including two single-hop datasets, Natural Questions (NQ) [24] and TriviaQA [21], and three multi-hop datasets: HotpotQA [47], 2WikiMultihopQA (2Wiki) [16], and MuSiQue [37]. We report Exact Match (EM) as the primary metric for answer generation. For retrieval evaluation on datasets with ground-truth documents, we use Recall as the main metric.

Baselines. We compare LAnR against three categories of methods: (1) generation without retrieval, including direct LLM generation and supervised fine-tuning (SFT) without retrieval; (2) single-hop retrieval methods, such as naive RAG [25] that retrieves once using the input query; and (3) multi-hop retrieval approaches, including agentic and iterative systems such as Search-o1 [27], IRCoT [38], Search-R1 [20], and AutoRefine [34].

Implementation Details. To simulate a realistic retrieval setting, we remove ground-truth context from the QA datasets and instead use the 2018 Wikipedia dump [22] as the external knowledge source. By default, retrieval returns the top-k documents at each step, with $k = 3$ following the setting of Search-R1 for fair comparison. All models are trained on the combined NQ and HotpotQA dataset, following the setup of prior works [20, 34]. Unless otherwise specified, all baselines and LAnR use the same Qwen2.5-3B models to ensure a fair comparison. Detailed training configurations are described in Appendix B.

4.2 Main Results

Overall Performance (RQ1). Table 1 reports EM accuracy across five QA benchmarks covering both single-hop and multi-hop reasoning. Retrieval-free methods perform poorly, while Naive RAG improves single-hop QA but remains ineffective on compositional tasks such as HotpotQA, 2Wiki, and MuSiQue. Among multi-hop approaches, Search-R1 and AutoRefine provide strong baselines, but LAnR consistently achieves the best overall performance, especially on challenging multi-hop benchmarks. LAnR-Instruct reaches 0.419 EM on HotpotQA, 0.408 on 2Wiki, and 0.193 on MuSiQue, outperforming AutoRefine-Instruct across all three datasets. Although AutoRefine

remains competitive on single-hop benchmarks, LAnR achieves the highest average EM overall, indicating a stronger balance between retrieval and reasoning. Despite being trained only on NQ and HotpotQA, LAnR also generalizes effectively to unseen datasets in a zero-shot setting.

Table 2: **Retrieval recall at varying document budgets.** BM25 is an unsupervised lexical retriever; BGE and E5 are dense retrievers trained with retrieval supervision. LAnR-Instruct-static uses the single-step [PRED] embedding as a fixed query vector (Section 2); LAnR-Instruct uses adaptive multi-hop retrieval with the learned controller, and the last column reports its recall together with the average number of retrieved documents. Despite joint training with generation, LAnR-Instruct exceeds the R@5 of all retrievers using a comparable number of documents. Matched-budget comparisons are given in Table 3.

Dataset	Method	R@1	R@3	R@5	R@10	Recall (budget)
HotpotQA	BM25 [31]	0.401	0.603	0.670	0.751	—
	BGE [5]	0.477	0.782	0.832	0.880	—
	E5 [39]	0.462	0.773	0.826	0.875	—
	LAnR-Instruct-static	0.451	0.769	0.800	0.830	—
	LAnR-Instruct	—	—	—	—	0.840±0.0012 (~5.5 docs)
2WikiMQA	BM25 [31]	0.360	0.555	0.605	0.656	—
	BGE [5]	0.411	0.643	0.680	0.715	—
	E5 [39]	0.415	0.656	0.687	0.717	—
	LAnR-Instruct-static	0.407	0.641	0.684	0.713	—
	LAnR-Instruct	—	—	—	—	0.715±0.0009 (~5.1 docs)
MuSiQue	BM25 [31]	0.216	0.310	0.350	0.400	—
	BGE [5]	0.280	0.435	0.500	0.573	—
	E5 [39]	0.290	0.417	0.454	0.503	—
	LAnR-Instruct-static	0.265	0.393	0.449	0.487	—
	LAnR-Instruct	—	—	—	—	0.516±0.0011 (~7.9 docs)

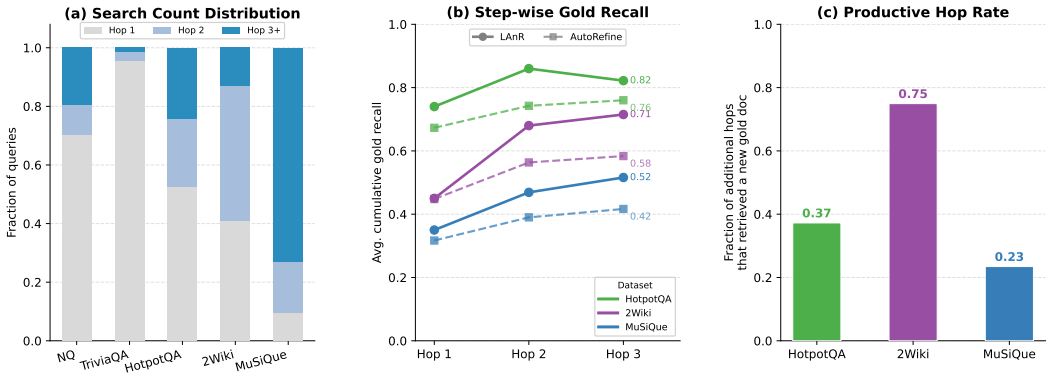


Figure 4: **(RQ2) LAnR adaptive search behaviour. (a)** Distribution of search counts per query across all five benchmarks. Single-hop datasets are dominated by a single search call, while compositional benchmarks require progressively more, showing the control head adapts to query complexity rather than applying a fixed budget. **(b)** Step-wise cumulative gold recall at each hop for LAnR (solid) and AutoRefine (dashed) on HotpotQA, 2Wiki, and MuSiQue. LAnR consistently outperforms AutoRefine at every hop on all three datasets. **(c)** Productive hop rate: fraction of additional search calls ($\geq$ Hop 2) that retrieved at least one new gold document per dataset.

Retrieval Quality vs. Text Embedding Retrievers. Table 2 compares LAnR-Instruct with sparse and dense retrievers across three multi-hop QA benchmarks. While BM25, BGE, and E5 rely on fixed top-K retrieval and are trained solely for retrieval, LAnR-Instruct jointly performs adaptive retrieval and generation. Despite this joint objective, LAnR-Instruct achieves competitive retrieval recall using substantially fewer retrieved documents. On HotpotQA and 2WikiMQA, it attains recalls of 0.840 and 0.715 with only $\sim$5 retrieved documents, matching or exceeding the Recall@5–10 performance of specialized retrievers. On the more challenging MuSiQue benchmark, LAnR-Instruct

achieves 0.516 recall with ∼7.9 documents, outperforming BM25 and remaining competitive with dense retrievers. In contrast, the single-step LAnR-Instruct-static variant consistently underperforms the adaptive version, highlighting the benefit of iterative retrieval with learned control.

Latent RAG Effectiveness (RQ2). To understand *how* LAnR retrieves rather than just *how well*, we analyse the search-level behaviour of the retrieval control head across all five evaluation benchmarks.

Adaptive search count. Figure 4(a) shows the distribution of search calls per query across all five datasets. On single-hop benchmarks (NQ, TriviaQA), the vast majority of queries are resolved in a single search call, reflecting the control head's ability to recognise when sufficient evidence has been gathered early. On compositional multi-hop benchmarks the distribution shifts substantially: 2Wiki and MuSiQue queries spread across two and three search calls, with MuSiQue requiring the most, consistent with its longer evidence chains.

Step-wise retrieval quality. Figure 4(b) compares cumulative gold recall across retrieval hops between LAnR and AutoRefine on the three multi-hop benchmarks. LAnR consistently achieves higher recall at every step, with the advantage appearing from the first hop and widening on more compositional datasets such as 2Wiki and MuSiQue. This suggests that latent query vectors better capture residual information needs, enabling subsequent retrieval steps to target missing evidence more effectively than text-based queries.

Control head precision. Figure 4(c) reports the productive hop rate: the fraction of additional search calls triggered by a "continue" decision that successfully retrieved at least one new gold document. On 2Wiki, 75% of additional calls are productive, confirming that the control head issues follow-up searches only when evidence is genuinely incomplete. On MuSiQue, the rate is lower at 23%, reflecting the difficulty of locating the precise gold documents in highly compositional three-hop and four-hop chains where missing any single query may fail to bridge the full reasoning gap.

Table 3: **Retrieval recall under matched document budgets. (a)** 6-document budget: BM25, BGE, E5, and LAnR-static each retrieve the top-6 documents in a single call, while LAnR-adaptive performs two latent retrieval calls of top-3 documents each. **(b)** 10-document budget: retrievers use top-10, while LAnR-adaptive retrieves exactly 10 unique documents through three latent calls (top-3, top-3, top-4). The total number of retrieved documents is identical within each panel; the number of retrieval calls differs. **Bold** marks the best result per row.

Dataset	BM25	BGE	E5	LAnR-static	LAnR-adaptive
(a) 6-document budget (LAnR-adaptive: 2×top-3)					
NQ	0.638	0.798	**0.818**	0.812	0.795
TriviaQA	0.641	0.751	**0.811**	0.805	0.798
HotpotQA	0.692	0.847	0.841	0.847	**0.865**
2Wiki	0.620	0.690	0.696	0.693	**0.724**
MuSiQue	0.362	0.509	0.468	0.460	**0.520**
(b) 10-document budget (LAnR-adaptive: top-3 + top-3 + top-4)					
HotpotQA	0.751	0.880	0.875	0.830	**0.905**
2Wiki	0.656	0.715	0.717	0.713	**0.828**
MuSiQue	0.400	0.573	0.503	0.487	**0.601**

Matched-budget retrieval (RQ2). Table 3 controls for retrieval volume by fixing the total number of documents. With a budget of six documents, LAnR-adaptive (2×top-3) outperforms LAnR-static (top-6) and conventional retrievers on all three multi-hop benchmarks, achieving 0.520 on MuSiQue versus 0.460 and 0.509 for LAnR-static and BGE, respectively. With ten documents, it reaches 0.828 on 2Wiki, surpassing E5 (0.717). These results highlight the benefit of iterative latent queries in targeting residual evidence. On single-hop tasks, a forced second retrieval slightly reduces recall, as it may introduce irrelevant documents after sufficient evidence has been found. The default controller mitigates this by typically stopping after one retrieval call (Figure 4a). Note that document budgets, but not retrieval-call counts, are matched; latency is evaluated separately in Figure 2.

Inference Efficiency of Latent Retrieval (RQ3). Figure 5 directly tests the core efficiency claim of LAnR: each subplot shows the EM distribution for all three methods on a single benchmark, with

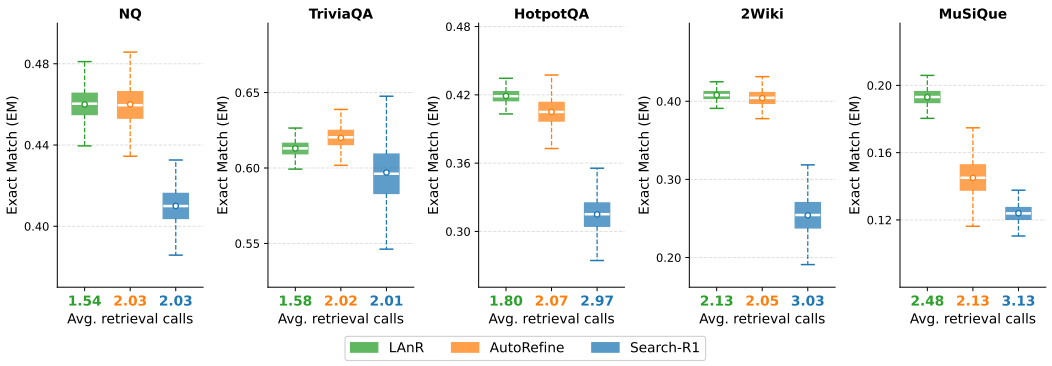


Figure 5: **(RQ3)** Per-dataset EM distributions for LAnR, AutoRefine, and Search-R1. LAnR achieves competitive or higher EM with the fewest retrieval calls and consistently narrow variance.

the colored x-tick labels reporting each method's average number of retrieval calls. Across all five datasets, LAnR issues the fewest retrieval calls ($1.54 - 2.47$), yet its median EM matches or exceeds AutoRefine ($2.02 - 2.13$ calls) and Search-R1 ($2.01 - 3.13$ calls). The narrow interquartile ranges confirm that LAnR's accuracy is stable across evaluation instances rather than driven by a subset of easy queries. Notably, on the multihop benchmarks, LAnR's gap over Search-R1 widens substantially despite Search-R1 issuing nearly twice as many retrievals, indicating that latent query vectors retrieve more relevant evidence per call than token-based queries.

The token and latency costs are quantified in Figure 2. LAnR generates on average only $\sim$5 output tokens per query, roughly $30\times$ fewer than Search-R1 ($\sim$163) or AutoRefine ($\sim$168), because the retrieval signal is encoded as a latent vector rather than verbalized text. This reduction translates directly to wall-clock savings: LAnR completes a query in $0.81 - 1.61$ s compared to $2.01 - 2.77$ s for Search-R1 and $2.15 - 2.34$ s for AutoRefine, a $1.5 - 2.7\times$ speedup across all benchmarks.

Key Takeaway. By replacing explicit query generation with compact latent vectors and an adaptive retrieval controller, LAnR performs fewer but more effective search calls, achieves higher cumulative gold recall, and maintains stable EM performance with lower computational cost.

4.3 Ablation Studies

Component ablation (RQ4). Table 4 isolates the contribution of three design choices: the MLP retrieval control head, NTP loss masking, and the NTP loss weight λ_{NTP}. Removing the control head and replacing it with a fixed 5-document budget leaves single-hop performance essentially unchanged but causes large drops on multi-hop tasks, confirming that adaptive retrieval is the primary driver of multi-hop gains. Removing NTP loss masking introduces a moderate but consistent degradation on all three multi-hop benchmarks, suggesting the masking prevents the model from conflating retrieval-query generation with next-token prediction over retrieved content. Reducing λ_{NTP} to 0.5 incurs a small performance drop, while setting it to 0 collapses performance to near zero, demonstrating that the NTP objective is indispensable for answer generation.

Table 4: Component ablation study (**RQ4**). ΔAvg denotes the absolute decrease in average performance relative to the full LAnR model.

Variant	NQ	TriviaQA	HotpotQA	2Wiki	MuSiQue	Avg	ΔAvg
LAnR-Instruct	**0.460**	**0.613**	**0.419**	**0.408**	**0.193**	**0.419**	—
w/o ctrl head	0.459	0.592	0.342	0.266	0.095	0.351	-0.068
w/o NTP loss masking	0.458	0.582	0.395	0.371	0.182	0.398	-0.021
$\lambda_{\text{NTP}} = 0.5$	0.455	0.578	0.411	0.402	0.189	0.407	-0.012
$\lambda_{\text{NTP}} = 0$	0.005	0.007	0.004	0.000	0.000	0.003	-0.416

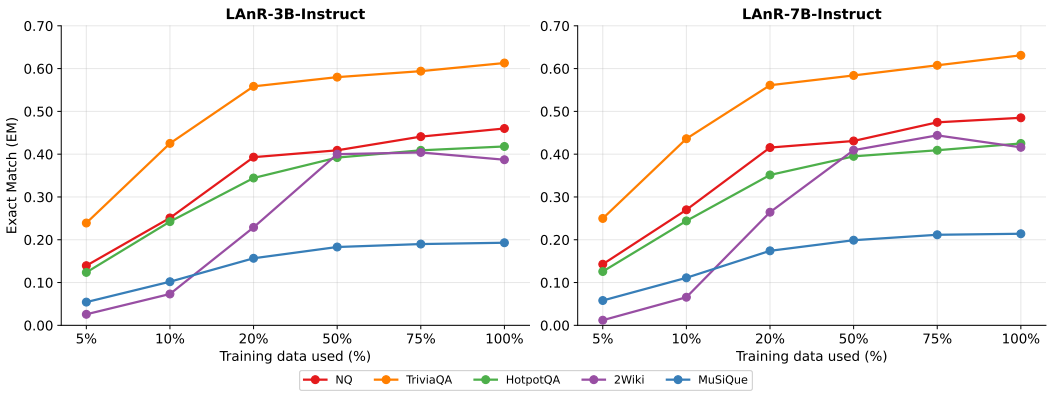


Figure 6: EM at various fractions of training data used (5 % to 100 %). Most of the performance gain is captured by the first 50 % of the data.

Additional Experimental Results. Figure 6 shows EM as a function of training-set fraction for both LAnR-3B and LAnR-7B across all five benchmarks. The learning curve rises steeply in the first 10-20% of data, where the four-dataset average EM climbs from 0.13 to 0.38. Notably, 2WikiMultihopQA starts near zero at 5% and gains most rapidly between 10-50%, reflecting the higher sample complexity of multi-hop compositional reasoning compared to single-hop benchmarks (NQ, TriviaQA), which plateau earlier. Beyond 50% gains flatten to within 3-5% of the full-data ceiling, suggesting that LAnR's latent retrieval mechanism generalises effectively with moderate training budgets. The 7B variant yields a consistent improvement over 3B across all data fractions, indicating that model scale and training data contribute largely independently to final performance.

Additional analyses are provided to further examine LAnR from multiple perspectives, including: (i) a detailed evaluation of the Retrieval Control Head, Appendix E.1; (ii) model scaling experiments spanning 3B to 7B backbones, Appendix E.2; (iii) statistical significance analysis across multiple random seeds, Appendix E.3; and (iv) the effect of varying the number of `[PRED]` tokens, where larger token counts improve abstraction capacity, Appendix E.4. Together, these findings further demonstrate the robustness, scalability, and retrieval effectiveness of LAnR across diverse QA benchmarks.

5 Conclusion

We introduced **latent abstraction retrieval-augmented generation** (LAnR), a unified framework that performs retrieval and reasoning directly in the latent space of a single LLM. By replacing explicit text-based query generation with latent query vectors derived from hidden states, LAnR eliminates the need for a separate retriever and reduces reliance on token-level reasoning. We further proposed a lightweight retrieval control head that adaptively determines when additional retrieval is required, enabling efficient multi-hop reasoning without explicit intermediate text. Empirical results across multiple benchmarks demonstrate that LAnR achieves competitive performance compared to existing RAG systems while significantly reducing inference latency. These findings highlight the potential of latent-space reasoning as a scalable alternative to conventional RAG pipelines.

References

[1] Muhammad Arslan, Hussam Ghanem, Saba Munawar, and Christophe Cruz. A survey on rag with llms. *Procedia computer science*, 246:3781–3790, 2024.

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

[3] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2023.

[4] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. Llm2vec: Large language models are secretly powerful text encoders, 2024. *URL https://arxiv. org/abs/2404.05961*, 2024.

[5] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 4(5), 2024.

[6] Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

[7] Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

[8] Xinghao Chen, Anhao Zhao, Heming Xia, Xuan Lu, Hanlin Wang, Yanjun Chen, Wei Zhang, Jian Wang, Wenjie Li, and Xiaoyu Shen. Reasoning beyond language: A comprehensive survey on latent chain-of-thought reasoning. *arXiv preprint arXiv:2505.16782*, 2025.

[9] Xinghao Chen, Anhao Zhao, Heming Xia, Xuan Lu, Hanlin Wang, Yanjun Chen, Wei Zhang, Jian Wang, Wenjie Li, and Xiaoyu Shen. Reasoning beyond language: A comprehensive survey on latent chain-of-thought reasoning. *arXiv preprint arXiv:2505.16782*, 2025.

[10] Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. xrag: Extreme context compression for retrieval-augmented generation with one token. *Advances in Neural Information Processing Systems*, 37:109487–109516, 2024.

[11] Debrup Das, Sam O'Nuallain, and Razieh Rahimi. Rader: Reasoning-aware dense retrieval models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19981–20008, 2025.

[12] Jingcheng Deng, Zhongtao Jiang, Liang Pang, Zihao Wei, Liwei Chen, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. Following the autoregressive nature of llm embeddings via compression and alignment. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12672–12688, 2025.

[13] Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*, 2023.

[14] Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. *arXiv preprint arXiv:2310.02226*, 2023.

[15] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[16] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 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, 2020.

[17] Liu Huanshuo, Hao Zhang, Zhijiang Guo, Jing Wang, Kuicai Dong, Xiangyang Li, Yi Quan Lee, Cong Zhang, and Yong Liu. Ctrla: Adaptive retrieval-augmented generation via inherent control. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 12592–12618, 2025.

[18] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 7969–7992, 2023.

[19] Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O Arik. Long-context llms meet rag: Overcoming challenges for long inputs in rag. *arXiv preprint arXiv:2410.05983*, 2024.

[20] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

[21] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 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, 2017.

[22] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 6769–6781, 2020.

[23] Chaeeun Kim and Seungone Kim. Freeson: Retriever-free retrieval-augmented reasoning via corpus-traversing mcts. *arXiv preprint arXiv:2505.16409*, 2025.

[24] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[25] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[26] Jindong Li, Yali Fu, Li Fan, Jiahong Liu, Yao Shu, Chengwei Qin, Menglin Yang, Irwin King, and Rex Ying. Implicit reasoning in large language models: A comprehensive survey. *arXiv preprint arXiv:2509.02350*, 2025.

[27] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, 2025.

[28] Zhijie Nie, Zhangchi Feng, Mingxin Li, Cunwang Zhang, Yanzhao Zhang, Dingkun Long, and Richong Zhang. When text embedding meets large language model: a comprehensive survey. *arXiv preprint arXiv:2412.09165*, 2024.

[29] Leonardo Noriega. Multilayer perceptron tutorial. *School of Computing. Staffordshire University*, 4(5):444, 2005.

[30] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.

[31] Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc, 2009.

[32] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36: 68539–68551, 2023.

[33] Dachuan Shi, Abedelkadir Asi, Keying Li, Xiangchi Yuan, Leyan Pan, Wenke Lee, and Wen Xiao. Swireasoning: Switch-thinking in latent and explicit for pareto-superior reasoning llms. *arXiv preprint arXiv:2510.05069*, 2025.

[34] Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. Search and refine during think: Facilitating knowledge refinement for improved retrieval-augmented reasoning. *arXiv preprint arXiv:2505.11277*, 2025.

[35] Levy Silva and Luciano Barbosa. Improving dense retrieval models with llm augmented data for dataset search. *Knowledge-based systems*, 294:111740, 2024.

[36] Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*, 2024.

[37] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[38] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 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, 2023.

[39] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

[40] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, 2024.

[41] Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. System-1.5 reasoning: Traversal in language and latent spaces with dynamic shortcuts. *arXiv preprint arXiv:2505.18962*, 2025.

[42] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[43] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*, 2020.

[44] Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23336–23351, 2025.

[45] Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot++: Test-time scaling with soft chain-of-thought reasoning. *arXiv preprint arXiv:2505.11484*, 2025.

[46] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[47] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 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, 2018.

[48] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

[49] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

[50] Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, Jie Xu, et al. The latent space: Foundation, evolution, mechanism, ability, and outlook. *arXiv preprint arXiv:2604.02029*, 2026.

[51] Zhenrui Yue, Honglei Zhuang, Aijun Bai, Kai Hui, Rolf Jagerman, Hansi Zeng, Zhen Qin, Dong Wang, Xuanhui Wang, and Michael Bendersky. Inference scaling for long-context retrieval augmented generation. *arXiv preprint arXiv:2410.04343*, 2024.

[52] Zhenrui Yue, Bowen Jin, Huimin Zeng, Honglei Zhuang, Zhen Qin, Jinsung Yoon, Lanyu Shang, Jiawei Han, and Dong Wang. Hybrid latent reasoning via reinforcement learning. *arXiv preprint arXiv:2505.18454*, 2025.

A Related Works

A.1 Latent Reasoning

A key limitation of dominant reasoning paradigms such as explicit chain-of-thought (CoT) [42, 49, 14, 50] lies in their reliance on discrete token generation during inference. In standard CoT decoding, the model commits to a single token at each step, sampled from the predicted distribution. While this process enhances interpretability by verbalizing intermediate reasoning steps, it inherently collapses the full probability distribution into a single trajectory, discarding uncertainty and eliminating alternative reasoning paths that may be informative. To address this limitation, recent work has explored latent reasoning [15, 44, 33, 52, 41], where reasoning unfolds directly in the continuous hidden-state space rather than through discrete text. This paradigm offers two primary advantages over CoT: (1) increased representational capacity per step, as continuous vectors can encode substantially richer information than individual tokens [8]; and (2) the ability to implicitly preserve multiple reasoning hypotheses without prematurely collapsing them into a single token sequence [26]. However, despite these advances, prior work on latent reasoning has primarily focused on improving reasoning quality in standalone language modeling settings. Its application to RAG remains largely unexplored. In particular, existing methods do not address how latent representations can be leveraged for retrieval itself, nor how they can guide adaptive retrieval decisions.

A.2 Retrieval Augmented Generation

RAG enhances LLMs by incorporating external knowledge sources to mitigate hallucinations and address knowledge gaps [25, 51, 1]. A central challenge in RAG systems lies in determining when and how to retrieve relevant information, as naive single-step retrieval often introduces irrelevant or insufficient context [18, 19]. Early approaches employ supervised fine-tuning (SFT) to train models for query generation and retrieval integration [3, 38, 48, 32]; however, these methods rely on high-quality annotated trajectories and often exhibit limited generalization to out-of-distribution settings. Leveraging the inherent structure of LLMs, an alternative direction is to fine-tune them as document encoders for text embedding [4, 40, 36, 12], or to use LLMs to generate synthetic data to support embedding training [11, 28, 35]. More recent work explores iterative and adaptive retrieval strategies, where models interleave reasoning and retrieval in a multi-step process, commonly described as "search-during-think" [20, 34, 6, 23]. However, existing approaches primarily operate in the discrete text space, relying on explicit query generation and token-level signals to control retrieval. Moreover, they often lack mechanisms for directly assessing retrieval sufficiency or refining retrieved evidence beyond surface-level interactions.

Table 5: Primary hyperparameters used by LAnR.

Hyper-parameter	**Value**
Training Batch Size	256
Chunk size	100 words
Micro Training Batch Size	32
Learning Rate	5×10^{-5}
Validation Batch Size	256
Total Training Steps	250
Max Search Actions	4
λ for NTP Loss	1
μ for Retrieval Control Head Loss	1

Table 6: Statistics of the datasets used in this paper.

	NQ	TriviaQA	HotpotQA	2Wiki	Musique
Train	79168	78785	90447	15,000	19,938
Dev	8757	8837	7405	12576	2417
Test	3610	11313	-	-	-

B More Implementation Details

Training Details We train LAnR using full-parameter fine-tuning on 2 NVIDIA H100 (80GB) GPUs. The training data is constructed by combining Natural Questions (NQ) [24] and HotpotQA [47], following a consistent setup across LAnR and all training-based baselines. We employ Fully Sharded Data Parallelism (FSDP) for distributed training and use bfloat16 precision for both training and evaluation. Table 5 summarizes the main hyperparameters. During inference, we sample with a temperature of 1.0 and allow up to 4 retrieval steps per query. Retrieved documents are concatenated and truncated to a maximum length of 512 tokens. For direct inference and supervised fine-tuning (SFT) baselines, we adopt Qwen2.5-3B-Instruct [46] as the backbone model.

Dataset Statistics. All datasets are obtained from the FlashRAG Datasets collection[1]. Detailed statistics are reported in Table 6. The LAnR training set is constructed from the training splits of NQ and HotpotQA, comprising 169,615 examples. For evaluation, we aggregate the test or development splits from seven benchmarks: test splits are used for datasets that provide them (e.g., NQ, TriviaQA), while development splits are used otherwise (e.g., HotpotQA, 2Wiki, MuSiQue).

Computational serving. Although LAnR-Instruct introduces higher indexing cost than conventional text embedding approaches, it reduces serving-time computational overhead by unifying retrieval and generation within a single model. Table 7 compares LAnR-Instruct with Auto-Refine-Instruct using the same Qwen2.5-3B backbone. Unlike Auto-Refine-Instruct, which requires an additional external embedding retriever during inference, LAnR-Instruct performs retrieval directly through latent representations inside the generation model. As a result, LAnR-Instruct requires lower VRAM usage and simplifies deployment by eliminating the need to maintain a separate text retrieval encoder.

B.1 Retrieval Control Head Model Design

The retrieval control head f_θ is a single affine projection followed by a sigmoid:

$$\hat{y}^{(r)} = \sigma\big(W\, q^{(r)} + b\big), \qquad W \in \mathbb{R}^{1\times d}, \quad b \in \mathbb{R}, \tag{9}$$

where $q^{(r)} = h^{(r)}_{\texttt{[PRED]}} \in \mathbb{R}^d$ is the last-layer hidden state of the final `[PRED]` token at retrieval turn r, and d is the LLM's hidden dimension (d=2048 for Qwen-2.5-3B, d=3584 for Qwen-2.5-7B). The head introduces d+1 trainable parameters, 2,049 for the 3B backbone and 3,585 for the 7B backbone,

[1] `https://huggingface.co/datasets/RUC-NLPIR/FlashRAG_datasets`

Table 7: **Serving efficiency comparison.** Both Auto-Refine-Instruct and LAnR-Instruct use the same Qwen2.5-3B backbone. Auto-Refine-Instruct requires an additional external embedding model for retrieval, whereas LAnR-Instruct performs retrieval and generation within a unified model.

Method	Retrieval Architecture	Extra Retriever	VRAM Usage (GB BF16)
Auto-Refine-Instruct	Text-based retrieval	Yes	6.48
LAnR-Instruct	Unified latent retrieval	No	5.85

adding negligible capacity (under 0.0002% of total model parameters) and requiring zero additional LLM forward passes at inference.

C Latent Retrieval Control

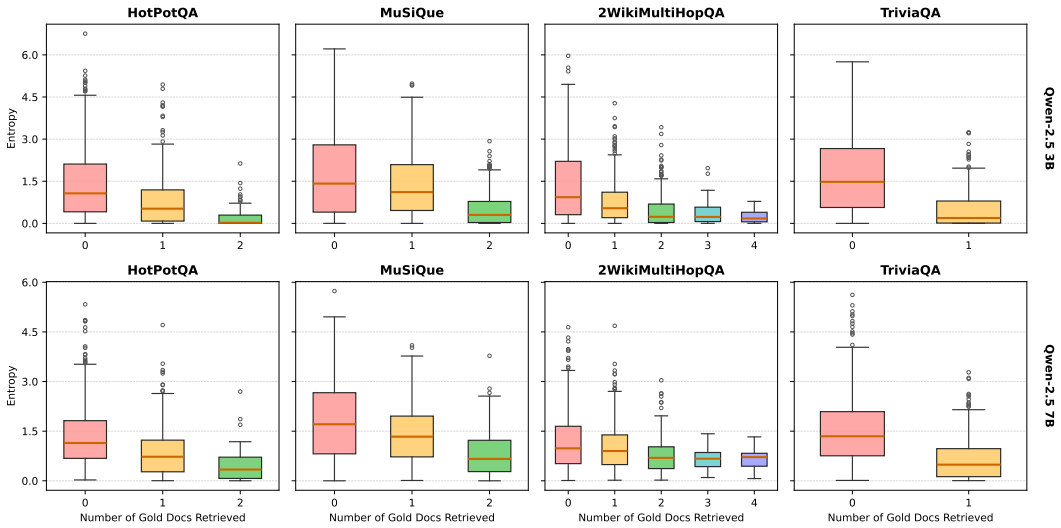


Figure 7: Entropy distribution on RAG benchmarks using Qwen models when answers are generated with gold documents provided in context. Increasing the number of gold documents leads to lower entropy, indicating more confident predictions.

Existing iterative RAG systems rely on explicit token-level signals to determine retrieval termination [38, 27, 20], entangling the stopping decision with surface-level text generation rather than the model's internal assessment of evidence sufficiency. This raises a natural question: *Do the hidden states of an LLM already encode a reliable signal of whether the retrieved context is adequate, even before any answer tokens are produced?*

Setup. Given an input query token sequence x with an instruction to produce an answer, and a subset $\mathcal{G}_k \subseteq \mathcal{P}$ of k gold supporting documents sampled from the full set $\mathcal{P}$, we construct a probe sequence that enforces answer generation conditioned on the provided context:

$$\tilde{x}_k = \big(x,\ \mathcal{G}_k,\ \texttt{[PRED]}\big), \tag{10}$$

By causal attention, the hidden state $h_k := h_{\texttt{[PRED]}}(\tilde{x}_k) \in \mathbb{R}^d$ attends to the full query and gold context. The LM head induces a distribution over the first answer token, $p_k(v) = \mathrm{softmax}(W h_k)_v$, with predictive entropy: $H_k = -\sum_{v\in\mathcal{V}} p_k(v) \log p_k(v)$, where $\mathcal{V}$ denotes the vocabulary.

Experimental Findings. We vary k from 0, as no gold evidence to $k_{\max} = |\mathcal{P}|$, as all gold evidence) and measure the resulting entropy distribution. Figure 7 reports this across four multi-hop QA benchmarks: HotPotQA, TriviaQA, MuSiQue, and 2WikiMultiHopQA, using model Qwen-2.5 Instruct at the 3B and 7B scales without any finetuning. Across all datasets and model combinations, H_k decreases monotonically as k increases. At k=0, the model exhibits high entropy, reflecting

substantial uncertainty over the answer space; at $k{=}k_{\max}$, the distribution concentrates near $H_k < 1.0$. This separation is most pronounced on compositional multi-hop benchmarks, such as MuSiQue and 2WikiMultiHopQA, where the absence of bridge documents prevents shortcuts via parametric recall.

Observation 1 (Entropy-Sufficiency Correlation) *H_k decreases monotonically in k; in particular,* $\mathbb{E}[H_k \mid \mathcal{P} \subseteq \mathcal{G}_k] < \mathbb{E}[H_k \mid \mathcal{P} \not\subseteq \mathcal{G}_k]$.

Implication for retrieval control. Observation 1 establishes that H_k is a reliable signal of evidence sufficiency. Moreover, H_k is a deterministic function of the same hidden state h_k used for retrieval, as in Eq. 1, so entropy-based stopping decisions can, in principle, be inferred from h_k alone, without requiring full text generation at each retrieval step. Building on this, Section 3 introduces a lightweight MLP-based Retrieval Control Head that operates directly on $h_{\texttt{[PRED]}}$ to predict whether further retrieval is needed.

D Control-Head Supervision under Varying Annotation Regimes

The control-head label $y^{(r)}$ defined in Eq. 5 requires access to a set of gold supporting documents $\mathcal{P}$. Such annotations are directly available in standard multi-hop QA datasets (e.g., HotpotQA, 2WikiMultihopQA, MuSiQue), and are the same kind of relevance annotations used to construct positive pairs when training supervised dense retrievers such as DPR, E5, and BGE [22, 39, 5]. LAnR uses these annotations both for the contrastive retrieval objective and to derive the stopping label, so the controller introduces no supervision signal beyond what retriever training already requires.

Span-containment labeling for single-hop datasets. For single-hop datasets (e.g., NQ, TriviaQA), passage-level annotations are not directly aligned with the retrieval corpus. We adopt a standard Dense Passage Retrieval (DPR) [22] strategy to construct $\mathcal{P}$.

Chunking. We first segment the Wikipedia corpus $\mathcal{C}$ into fixed-length passages (100-word chunks), which serve as the retrieval units. This preprocessing step is deterministic and independent of model training.

Positive passage assignment. We then construct a single positive passage per query using dataset-specific heuristics:

(i) Datasets with annotated contexts (e.g., NQ). We align the original annotated gold passage to the closest matching chunk in $\mathcal{C}$ based on answer span overlap. The matched chunk is treated as the positive passage. Examples for which no alignment is found are discarded.

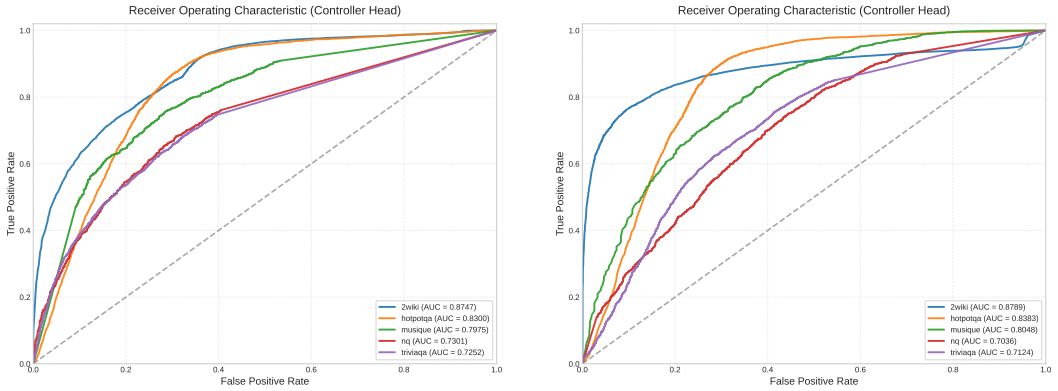
(ii) Datasets without annotated contexts (e.g., TriviaQA). We adopt distant supervision by retrieving passages using a sparse retriever (e.g., BM25, E5), and selecting the highest-ranked passage that contains the answer string as the positive passage. If no retrieved passage within the top-k results contains the answer, the example is removed.

This procedure yields weakly supervised "gold" passages that are not exhaustively annotated but have been shown to be effective in prior work. The resulting set $\mathcal{P}$ is used consistently for both retrieval training and control-head supervision, ensuring a unified notion of relevance without requiring additional manual annotation.

E More Experimental Results

E.1 Retrieval Control Head Analysis

Figure 8 shows ROC curves for the retrieval control head on both LAnR-3B-Base and LAnR-3B-Instruct across all five benchmarks. Both variants achieve strong AUC on multi-hop datasets, confirming that the [PRED] hidden state carries a reliable evidence-sufficiency signal when supporting documents are compositionally distributed. Performance of both model variants is lower on single-hop datasets (Base: 0.730, 0.725; Instruct: 0.704, 0.712), where the distinction between “enough” and “not enough” evidence is less clear because a single retrieved document often suffices. The consistent improvement of the Instruct variant across all five datasets and the clear stratification by task complexity confirm that the control head learns a genuine sufficiency signal rather than a dataset-specific heuristic.



(a) LAnR-3B-Base (b) LAnR-3B-Instruct

Figure 8: ROC curves for the retrieval control head across five benchmarks. Instruction tuning yields consistent gains, with the largest improvement on MuSiQue

E.2 Effect of Model Size

Table 8: EM and F1 across methods and model sizes (Qwen-2.5-Base, 3B and 7B) on five QA benchmarks.

		Single-Hop		Multi-Hop			
Model	Metric	NQ	TriviaQA	HotpotQA	2Wiki	MuSiQue	Avg.
Qwen-2.5-3B-Base							
SearchR1-3B-Base	EM	0.421	0.583	0.297	0.274	0.066	0.328
	F1	0.476	0.650	0.380	0.322	0.123	0.390
AutoRefine-3B-Base	EM	0.467	0.620	0.405	0.393	0.157	0.408
	F1	0.534	0.689	0.503	0.453	0.233	0.482
LAnR-3B-Base (ours)	EM	0.455	0.610	0.417	0.402	0.183	0.410
	F1	0.535	0.654	0.589	0.603	0.256	0.527
Qwen-2.5-7B-Base							
SearchR1-7B-Base	EM	0.469	0.627	0.410	0.272	0.173	0.390
	F1	0.552	0.700	0.517	0.327	0.236	0.466
AutoRefine-7B-Base	EM	0.484	0.659	0.451	0.405	0.187	0.437
	F1	0.574	0.729	0.573	0.607	0.283	0.553
LAnR-7B-Base (ours)	EM	0.482	0.631	0.459	0.415	0.205	0.438
	F1	0.570	0.724	0.590	0.612	0.295	0.558

Table 9: Statistical analysis against search-during-think baselines. The p-value column represents the T-test result of LAnR-Instruct v.s. AutoRefine-Instruct and Search-R1-Instruct.

Model	NQ	TriviaQA	HotpotQA	2wiki	Musique	p-value
LAnR-Instruct	0.460 ± 0.008	0.613 ± 0.005	0.419 ± 0.006	0.408 ± 0.006	0.193 ± 0.005	–
AutoRefine-Instruct	0.461 ± 0.010	0.620 ± 0.007	0.405 ± 0.012	0.404 ± 0.010	0.145 ± 0.011	7.49×10^{-3}
Search-R1-Instruct	0.410 ± 0.009	0.597 ± 0.019	0.315 ± 0.016	0.254 ± 0.023	0.124 ± 0.005	1.31×10^{-39}

Table 8 compares Search-R1, AutoRefine, and LAnR across Qwen-2.5-Base 3B and 7B backbones using both EM and F1 metrics. Scaling from 3B to 7B consistently improves performance for all methods, with the largest gains observed on multi-hop benchmarks that require compositional reasoning. At both model scales, LAnR achieves the strongest overall multi-hop performance. With the 3B backbone, LAnR-3B-Base attains the highest EM on HotpotQA (0.417), 2Wiki (0.402), and MuSiQue (0.183), while also substantially outperforming prior methods in F1, particularly on HotpotQA (0.589 vs. 0.503 for AutoRefine) and 2Wiki (0.603 vs. 0.453). Similar trends hold at 7B scale, where LAnR-7B-Base achieves the best EM and F1 across all three multi-hop datasets,

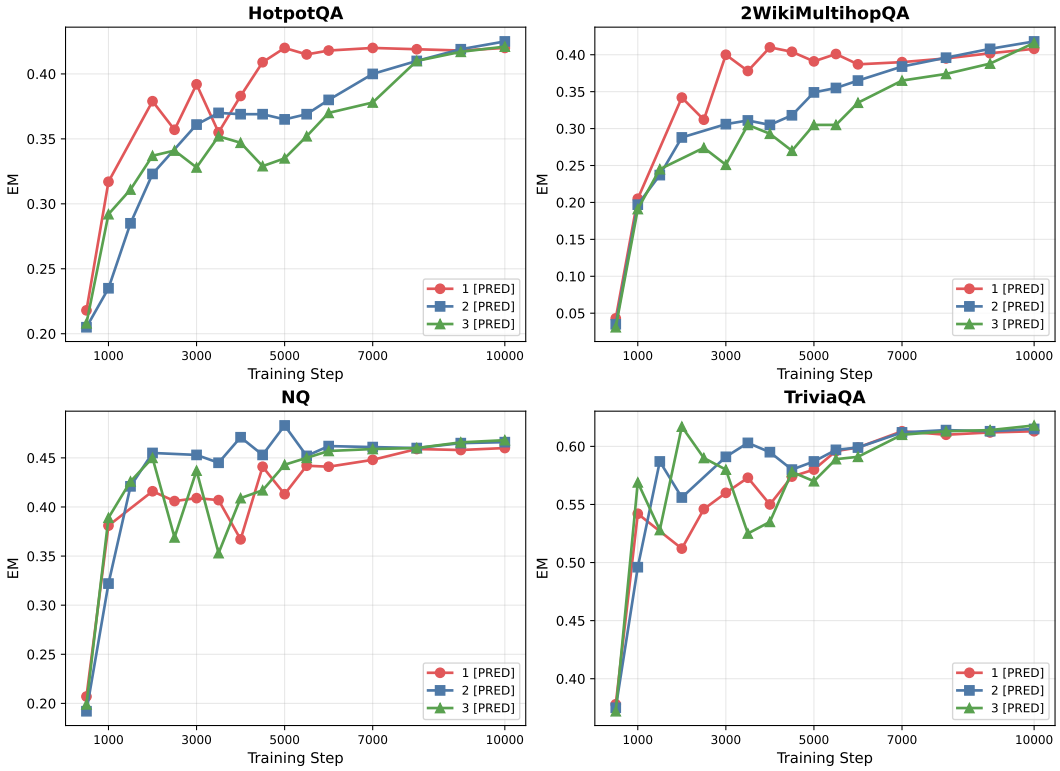


Figure 9: EM accuracy by training step for models trained with 1, 2, and 3 [PRED] tokens across four benchmarks. N=1 converges fastest and performs best on multi-hop datasets; additional tokens offer marginal gains only on single-hop TriviaQA.

reaching 0.459/0.590 on HotpotQA, 0.415/0.612 on 2Wiki, and 0.205/0.295 on MuSiQue. Although AutoRefine remains competitive on single-hop datasets such as NQ and TriviaQA, the results show that LAnR benefits more consistently from increased model capacity, suggesting that latent retrieval scales effectively with stronger backbone representations and is particularly advantageous for complex multi-step reasoning tasks.

E.3 Variance Analysis

Table 9 reports the mean and standard deviation across three runs with different random seeds to evaluate the robustness of search-during-think methods. LAnR-Instruct consistently achieves lower variance than competing baselines across most datasets, indicating more stable retrieval and reasoning behavior. To assess statistical significance, we perform paired T-tests between LAnR-Instruct and AutoRefine-Instruct. The resulting low p-values ($p \ll 0.01$) confirm that the performance differences are statistically significant, demonstrating that the gains of latent retrieval are reliable rather than arising from random variation.

E.4 Ablation on Numbers of [PRED] Tokens

We further analyze the effect of training with multiple [PRED] tokens, detailed in Figure 9. Increasing the number of [PRED] tokens encourages the model to construct more abstract latent queries. On single-hop benchmarks such as TriviaQA and NQ, LAnR consistently benefits from additional [PRED] tokens, with performance improvements emerging early in training. In contrast, on more challenging multi-hop benchmarks, models with multiple [PRED] tokens require substantially longer training before surpassing the performance of the single-[PRED] setting. These results suggest that while additional [PRED] tokens can improve abstraction capability, they also increase optimization difficulty for complex reasoning tasks. Therefore, we adopt $N = 1$ as the default configuration throughout our experiments.

E.5 Comparison with Adaptive RAG Methods

LAnR decides when to continue or stop retrieving, so we additionally compare it with two representative adaptive RAG methods that also learn retrieval timing: Self-RAG [3], and CtrlA [17]. Both baselines use the same E5 retriever over the 2018 Wikipedia corpus as our other baselines.

Metrics. In addition to Exact Match (EM), we report Contains-Match (CM), which considers a prediction correct if it contains the normalized gold answer as a span. Since Self-RAG and CtrlA often generate verbose, free-form responses instead of concise extractive answers, EM may substantially underestimate their performance. For instance, Self-RAG achieves an EM of 0.215 but a CM of 0.730 on TriviaQA. CM provides a more permissive evaluation criterion that partially mitigates this discrepancy in answer formats.

Table 10: **Comparison with adaptive RAG methods.** EM and Contains-Match (CM) on five QA benchmarks. Self-RAG and CtrlA use 7B backbones (Llama-2-7B and Mistral-7B-Instruct); LAnR uses Qwen2.5-3B and Qwen2.5-7B backbones. All methods retrieve from the same 2018 Wikipedia corpus.

		Single-Hop		Multi-Hop			
Model	Metric	NQ	TriviaQA	HotpotQA	2Wiki	MuSiQue	Avg.
Self-RAG (7B)	EM	0.230	0.215	0.145	0.015	0.025	0.126
	CM	0.455	**0.730**	0.385	0.350	0.090	0.402
CtrlA (7B)	EM	0.125	0.120	0.113	0.110	0.131	0.120
	CM	0.535	0.710	0.370	0.400	0.160	0.435
LAnR-Base-3B	EM	0.455	0.610	0.417	0.402	0.187	0.414
	CM	0.540	0.693	0.502	0.472	0.248	0.491
LAnR-Base-7B	EM	**0.482**	**0.631**	**0.459**	**0.415**	**0.205**	**0.438**
	CM	**0.597**	0.719	**0.514**	**0.494**	**0.266**	**0.518**

Results. Table 10 shows that both LAnR variants outperform Self-RAG and CtrlA on all benchmarks under EM, including LAnR-Base-3B, which uses fewer than half as many parameters. Under CM, LAnR-Base-7B achieves the best average score (0.518) and leads on four of five benchmarks. Self-RAG performs best on TriviaQA (0.730 vs. 0.719), where LAnR-Base-3B also slightly trails both baselines. LAnR's gains are particularly pronounced on multi-hop tasks, as shown on MuSiQue (0.248 vs. 0.090 for Self-RAG and 0.160 for CtrlA). These results suggest that the improvements stem not merely from model scale, but from jointly optimizing latent-space retrieval control and query formation with answer generation.

Retrieval recall against iterative and adaptive methods. To compare retrieval quality directly, we evaluate all methods under a controlled budget of two sequential retrieval rounds with three documents per round (six documents in total) and report Recall@6 on the three multi-hop benchmarks. All baselines use the same E5 retriever and 2018 Wikipedia index, so they differ only in how they form retrieval queries and decide when to retrieve; LAnR retrieves over the same corpus with its own latent queries and document embeddings. We also include *Naive 2-hop E5*, a retrieval-only baseline without LLM involvement: after retrieving the top-3 passages for the question, snippets from these passages are appended to the question to form the second-round query.

Table 11 shows that LAnR achieves the highest recall on all three benchmarks under the same budget. Whereas text-based methods must first generate an intermediate query, which can introduce ambiguity or lexical bias, each LAnR query is optimized directly toward the remaining evidence set $\mathcal{P}^{(r)} = \mathcal{P} \setminus \mathcal{R}^{(r-1)}$ through the multi-turn contrastive objective (Eq. 7).

E.6 Query and Document Read-out Design

LAnR reads out the query from a dedicated `[PRED]` token and the document embedding from the last token of the document (Eqs. 1–2). The goal of this design is to let one model produce both representations while limiting interference between the retrieval and generation objectives. The

Table 11: **Retrieval recall (Recall@6) of adaptive and iterative methods** under a budget of two retrieval rounds $\times$ top-3 documents. All baselines use the same E5 retriever and Wikipedia index.

Method	HotpotQA	2Wiki	MuSiQue
Naive 2-hop E5 (reformulation)	0.805	0.662	0.446
Search-R1 [20]	0.828	0.674	0.468
CtrlA [17]	0.832	0.680	0.496
Self-RAG [3]	0.842	0.717	0.492
AutoRefine [34]	0.853	0.706	0.503
LAnR	**0.865**	**0.724**	**0.520**

asymmetry lies only in the read-out position: both embeddings come from the same LLM parameters and are optimized jointly with the same contrastive objective.

Table 12: EM over training steps when the last input token (no `[PRED]` token) is used as the retrieval query.

Steps	NQ	TriviaQA	HotpotQA	2Wiki
1000	0.103	0.247	0.186	0.145
3000	0.194	0.298	0.224	0.182
5000	0.236	0.314	0.259	0.213

Query side: dedicated `[PRED]` token vs. last input token. Our initial design used the hidden state of the last input token directly as the query, i.e., zero `[PRED]` tokens. Table 12 reports the EM of this variant over training. It trains substantially more slowly and stays well below the `[PRED]`-based model (Figure 9). We hypothesize that the last-token state is optimized simultaneously for next-token prediction and retrieval, which creates competing objectives, whereas a dedicated `[PRED]` position provides a separate read-out location where the model can encode retrieval intent without disturbing its generative representation.

F Limitations

Corpus encoding overhead. LAnR encodes the entire retrieval corpus using the same LLM used for generation. While this enables tight representational alignment, it is more computationally expensive than conventional retrieval approaches, including sparse lexical methods such as BM25 [31] and lightweight dense retrievers such as BGE [5]. Re-encoding is also required whenever the backbone model is updated, which may limit practicality in frequently updated knowledge bases.

Interpretability of latent queries. Unlike text-based RAG systems, where retrieval queries are human-readable and easily auditable, LAnR's latent query vectors are not directly interpretable. This makes it harder to diagnose retrieval failures or explain why specific documents were retrieved, which may be a concern in high-stakes deployments.

Evaluation scope. All experiments use English-language open-domain QA benchmarks with Wikipedia as the knowledge source. Performance on non-English, domain-specific, or long-document corpora remains untested, and results may not generalize to settings with significantly different document distributions or answer styles.

G Broader Impact

LAnR is foundational research on latent retrieval-augmented generation efficiency and does not target any specific deployment. Nevertheless, we briefly discuss potential societal implications.

Positive impacts. By reducing inference latency by up to $2.7\times$ and token generation by $\approx 30\times$ compared to existing RAG systems, LAnR lowers the computational cost of knowledge-grounded language model inference, improving accessibility for resource-constrained practitioners and reducing

energy consumption at scale. Improved factuality in question answering also has broad benefits in educational, medical, and scientific information access.

Potential negative impacts. As with any system that improves the fluency and factual grounding of LLM outputs, LAnR could lower the barrier to generating more convincing disinformation or fabricated content at scale. The efficiency gains in particular may make high-volume automated generation more feasible. Additionally, because LAnR retrieves from large external corpora, any biases present in those corpora may be silently propagated into generated answers without explicit query traces, potentially making such biases harder to audit than in text-based retrieval pipelines where queries are inspectable.

H Case Studies

The following examples trace LAnR's retrieval process hop by hop. At each search, the model runs one forward pass over a growing context ending with `[PRED]`. The hidden state at that position encodes all evidence seen so far, drives the next dense retrieval query, and is fed to the MLP control head to decide whether to continue. The context after h searches is:

$$\texttt{[Q]}\ \textit{question}\ \underbrace{[d_1^{(1)}]\cdots[d_K^{(1)}]}_{\text{search 1}}\ \cdots\ \underbrace{[d_1^{(r)}]\cdots[d_K^{(r)}]}_{\text{search } r}\ \texttt{[PRED]}$$

Gold supporting documents are underlined with *.

Example 1 - MuSiQue (3-hop)

2WikiMultihopQA - 2-hop compositional

Question: *"Where do Greyhound buses leave from in the city where the performer of Darlings formed?"*

Reasoning chain: Darlings → performer: Kevin Drew → formation city: Toronto → bus terminal

Turn 1 $p_{\text{ctrl}} = 0.9977 \rightarrow$ CONTINUE

Context: `[Q]` *Where do Greyhound buses leave from . . . formed?* `[PRED]`

Retrieved: Kevin Drew*, Spirit If. . ., Detropia

[Kevin Drew] Kevin Drew (born September 9, 1976) is a Canadian musician and songwriter who, together with Brendan Canning, founded the expansive Toronto baroque-pop collective Broken Social Scene.

Turn 2 $p_{\text{ctrl}} = 0.7922 \rightarrow$ CONTINUE

Context: `[Q] ... [Kevin Drew: ...Toronto baroque-pop collective...] [Spirit If: ···] [Detropia: ···] [PRED]`

Retrieved: Toronto Coach Terminal*, Alvarado Transportation Center, Dufferin St. Bridges

[Toronto Coach Terminal] The Toronto Coach Terminal is the central bus station for inter-city services in Toronto, Ontario, Canada. It is located at 610 Bay Street, in the city's Downtown.

Turn 3 $p_{\text{ctrl}} \approx 0 \rightarrow$ STOP

Context: `[Q] ... [Kevin Drew: ···] [Toronto Coach Terminal: ...Toronto...Bay St...] ··· [PRED]`

Retrieved: Darlings (Kevin Drew album)*, Peace Center, Darlington County

[Darlings (Kevin Drew album)] Darlings is the second solo album by Broken Social Scene co-founder Kevin Drew. It was released on March 18, 2014.

Answer: `The Toronto Coach Terminal` ✓ *Correct* * gold document

The `[PRED]` state at Search 1 already encodes *Toronto* implicitly through the Kevin Drew article, letting the model jump directly to the bus-terminal query at Search 2 without materialising "Toronto" as an explicit intermediate result. All three gold documents are retrieved across three latent searches, and the control head stops as soon as the evidence chain is complete.

Example 2 - 2WikiMultihopQA (2-hop)

2WikiMultihopQA - 2-hop compositional

Question: *"Where was the director of film* The Green Fog *born?"*
Reasoning chain: The Green Fog → director: Guy Maddin → birthplace: Winnipeg, Manitoba

Turn 1 $p_{\text{ctrl}} = 0.9999 \rightarrow$ CONTINUE

Context: `[Q]` *Where was the director of film The Green Fog born?* `[PRED]`

Retrieved: The Green Fog*, Walon Green, Adam Green (filmmaker)

[The Green Fog] The Green Fog is an experimental film directed by Guy Maddin, Evan Johnson, and Galen Johnson that loosely revisits the plot of Alfred Hitchcock's *Vertigo* through found footage.

Turn 2 $p_{\text{ctrl}} = 0.9992 \rightarrow$ CONTINUE

Context: `[Q] ... [The Green Fog: ...directed by Guy Maddin...] [Walon Green: ···] [Adam Green: ···] [PRED]`

Retrieved: Guy Maddin*, The Heart of the World, The Forbidden Room

[Guy Maddin] Guy Maddin (born February 28, 1956) is a Canadian screenwriter, director, author, cinematographer, and film editor of both features and short films, from **Winnipeg, Manitoba**.

Turn 3 $p_{\text{ctrl}} = 0.0000 \rightarrow$ STOP

Context: `[Q] ... [The Green Fog: ···] [Guy Maddin: ...Winnipeg, Manitoba...] ··· [PRED]`

(Control head halts; no new documents retrieved.)

Answer: `Winnipeg, Manitoba` ✓ *Correct* * gold document

This example shows the sharpest control-head transition across all three datasets: p_{ctrl} holds at 0.9999 and 0.9992 while evidence accumulates, then drops to 0.0000 the moment *Guy Maddin* and his birthplace are both in context. No text query names "Guy Maddin" at any step; the latent query at Search 2 is shaped entirely by the `[PRED]` representation formed over the *The Green Fog* article.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the main paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a dedicated limitation discussion in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: There are no theoretical results in this paper.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper fully discloses all the information needed to reproduce the main experimental results of the paper in Appendix B, and we provide the code to assist the reproducibility of our experimental results in `https://github.com/akBear23/LAnR`.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: This paper provides open access to the code, includes instructions for running the code in the link `https://github.com/akBear23/LAnR`, and we use public data to train in link `https://huggingface.co/datasets/RUC-NLPIR/FlashRAG_datasets`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix B contains all experimental details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in the appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide statistical significance analyses of different seeds in Appendix E.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include all experiments computed in the Appendix B, covering all resources used for generating model responses.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm the research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss broader impact in Appendix G.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The datasets are used via `https://huggingface.co/datasets/RUC-NLPIR/FlashRAG_datasets` under `CC-BY-SA-4.0` license. We use the Qwen-2.5 series models [46] that are under the Apache license 2.0.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.