ShiftRAG: Bypassing the Textual Bottleneck via Decoupled Learning and Continuous Soft Tokens

Youmin Ko[1], **Jihong Jeong**[1], **Hyunjoon Kim**[1,2*]
[1]Department of Artificial Intelligence, Hanyang University
[2]Department of Data Science, Hanyang University
{youminkk0213, jjh103, hyunjoonkim}@hanyang.ac.kr

Abstract

Despite advances in retrieval-augmented generation (RAG), suppressing hallucinations during complex reasoning remains a persistent challenge. We identify that existing RAG pipelines suffer from a textual bottleneck: by passing only raw text to the generator, they discard discriminative signals of the retriever. Furthermore, recent attempts to unify retrieval and generation within a single LLM induce severe objective conflict and exacerbate the rank collapse of decoder-only architectures, degrading performance on both tasks. To address these limitations, we introduce ShiftRAG, a decoupled RAG framework that employs a single LLM backbone but isolates discriminative retrieval learning from autoregressive generation. To bridge these decoupled modes, we project retrieval alignments into continuous soft tokens, establishing a high-bandwidth interface that infuses the generator with retrieval-side signals. Concurrently, our soft orthogonality objective mitigates rank collapse, while layer-wise relevance dynamics preserve the fine-grained latent space required for highly discriminative evidence separation. Extensive evaluations demonstrate that ShiftRAG significantly improves retrieval and question answering performance over state-of-the-art baselines. ShiftRAG preserves foundational capabilities of the backbone while maintaining a favorable efficiency-performance trade-off on general text encoding and generative tasks. Our code is available[2].

1 Introduction

Retrieval-augmented generation (RAG) [37] has emerged as the definitive framework for grounding LLMs with external knowledge across diverse knowledge-intensive domains, including multi-hop question answering (MHQA) [17, 43, 69] and knowledge graph reasoning [39, 42, 49, 57]. Current research has advanced RAG through three avenues: (i) learning dense representations specifically for retrieval [30, 31, 36, 52], (ii) adapting decoder-only LLMs into text encoders [2, 35, 41], and (iii) supporting both retrieval and generation within a single LLM backbone [45, 68]. Meanwhile, researchers have proposed to augment reasoning over retrieved evidence through explicit Chain-of-Thought [40, 61] and latent reasoning [55, 64]. Despite these sophisticated architectural enhancements, suppressing hallucinations [4, 25, 27, 28, 51] during retrieval-based reasoning remains a persistent challenge. We claim that this limitation stems from three factors.

First, overcoming the rank collapse inherent to decoder-only LLMs [12, 53] while preserving generative capacity remains a fundamental bottleneck. In deep layers, token representations naturally concentrate into a low-dimensional subspace, blurring the fine-grained semantic distinctions necessary to separate true evidence from distractors. Figures 1(a) and 1(b) diagnose this collapse, revealing

*Corresponding author.
[2]https://github.com/meaningful96/ShiftRAG

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

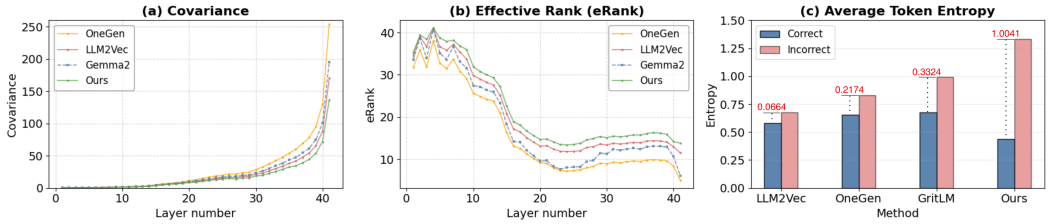

Figure 1: Motivating observations on layer-wise rank collapse and evidence-sensitive generation. (a) Layer-wise token covariance. (b) Effective rank (eRank) of token representations. (c) Average token entropy on 2WikiMultihopQA by answer correctness.

high token covariance[3] and low effective rank (eRank)[4] [71]. Current paradigms fail to resolve this trade-off: while contrastive adaptation in deep encoders [2, 35, 41] mitigates rank collapse, it severely degrades autoregressive generation. Unified LLMs [45, 68] preserve generation via backbone sharing, but inadvertently exacerbate rank collapse, exhibiting higher covariance and lower eRank than the base model (Gemma 2)[5]. Sharing a backbone or relying on standard attention mechanisms is insufficient to maintain both generative fluency and the discriminative representations for retrieval.

Second, standard RAG frameworks [5, 36, 52] suffer from a textual bottleneck, i.e., they transfer only raw document text to the LLM, discarding the implicit signals (the semantic alignment between question and document embeddings) from the retriever. Consequently, the generator is deprived of the retriever's underlying discriminative confidence. Figure 1(c) highlights this limitation: while an evidence-sensitive generator [63] should exhibit distinct predictive entropy gaps between correct and incorrect outputs, existing state-of-the-art baselines show small entropy differences[6]. This indicates that raw text alone fails to convey the strength of the supporting evidence. While Chain-of-Thought [40, 61] and continuous latent reasoning [55, 64] attempt to extract deeper rationales, they still derive these representations exclusively from explicit text. Lacking a mechanism to directly inject the embedding-level alignment of the retriever, these methods remain constrained by the textual interface.

Third, unifying retrieval and generation under shared adaptation parameters induces severe objective conflict. Discriminative retrieval requires a latent topology optimized for contrastive separation and fine-grained diversity by aligning the representation of a question with its supporting documents and separating it from distractors, whereas autoregressive generation organizes the representation space for sequential next-token prediction. In unified architectures [45, 68], forcing these fundamentally divergent tasks to share parameters creates competing optimization pressures, leading to representation instability and negative transfer. Even if adaptation parameters are decoupled, simultaneous joint training remains suboptimal; the generator is forced to optimize against a continuously shifting retrieval-conditioned interface, destabilizing retrieval convergence and downstream reasoning.

To address these issues, we introduce **ShiftRAG**, a novel RAG framework that employs a single LLM backbone across decoupled encoder and decoder modes. ShiftRAG transcends the standard textual interface by translating high-dimensional retrieval knowledge directly into latent reasoning signals. Our contributions are threefold. First, we cultivate a highly discriminative embedding space by exploiting layer-wise relevance dynamics, effectively mitigating the deep-layer rank collapse typical of decoder-only LLMs. Second, we project question–document embeddings and relevance scores into generation-friendly soft knowledge tokens. This latent bridge allows implicit retrieval alignments and explicit document text to jointly guide the reasoning of the generator. Third, by training the encoder and decoder modes independently and bridging them only at inference, we circumvent the severe objective conflict that plagues joint parameter optimization.

Extensive evaluations demonstrate that ShiftRAG achieves state-of-the-art retrieval and end-to-end QA performance across multi-hop (HotpotQA, 2WikiMultihopQA, MuSiQue) and single-hop (NaturalQuestions) benchmarks. Furthermore, auxiliary evaluations on text encoding and general generative tasks confirm that our decoupled approach preserves foundational capabilities of the backbone while maintaining a favorable efficiency-performance trade-off, establishing a robust new paradigm for retrieval-augmented reasoning.

[3] Covariance between hidden representations of tokens, where larger values indicate stronger pairwise directional alignment.
[4] Smaller values indicate fewer effective independent directions.
[5] More detailed analyses based on other LLMs are provided in Appendix E.9.
[6] Appendix E.6 shows the same trend for other datasets.

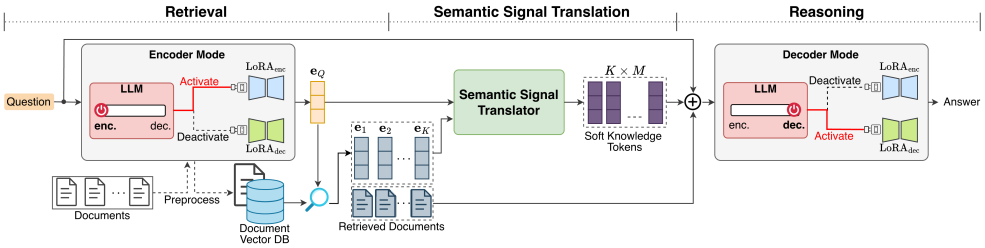

Figure 2: Illustration of the inference pipeline of ShiftRAG.

2 Related Work

RAG. Traditional dense retrieval maps queries and documents into unified vector spaces [30, 36, 46], while subsequent methods capture finer-grained interactions [31, 52] or exploit layer-wise representations [33]. Recently, LLMs have been adapted as powerful text encoders [2, 35, 41] and unified models capable of both retrieval and generation [32, 45, 58, 68]. However, despite these architectural advances, the RAG pipeline remains fundamentally disjointed. Because both LLM-based encoders and unified models interface with the generator exclusively via raw retrieved text, they discard the rich and latent discriminative signals formed during the retrieval process, limiting the ability of the generator to weight evidence effectively.

Latent Chain-of-Thought. While Chain-of-Thought (CoT) methods [10, 40, 61, 67] enhance reasoning transparency, they suffer from high generation latency and sensitivity to retrieved noise. Latent-CoT approaches [18, 55, 64] address this by compressing intermediate reasoning into continuous latent vectors. However, because these continuous representations are generated only from autoregressive prediction, they are structurally isolated from external knowledge modules, heavily restricting their integration into RAG systems.

Rank Collapse of LLMs. As LLMs increase in depth, hidden token representations tend to converge directionally, concentrating within a narrow subspace. This phenomenon, known as rank collapse, reduces the effective rank and representational diversity of deep-layer tokens, impairing fine-grained semantic discrimination. Theoretical studies [11, 47] demonstrate that this contraction accumulates across repeated self-attention layers. While architectural components like attention masks and LayerNorm can influence the geometry of this collapse [65], the degradation is particularly detrimental in decoder-only Transformers, where it severely weakens token-level expressivity [1, 6, 12, 53]. Despite this established limitation, recent LLM-based text encoders [2, 35] optimize for retrieval without explicitly mitigating the rank collapse of their pretrained backbones, hindering the formation of a semantically well-separated embedding space. This bottleneck becomes even more acute in unified LLMs [45, 68], where forcing generation and representation learning to share the same collapsed hidden space creates severe objective conflict.

3 Method

To enable a single LLM to master both retrieval and reasoning, we fine-tune a pre-trained causal LLM with two task-specific LoRA adapters. In **encoder mode**, we relax the causal mask to allow bidirectional self-attention, adapting the model to capture deep question-document semantic relevance. In **decoder mode**, the LLM reverts to its native autoregressive decoding to generate answers, leveraging the discriminative insights gained from the encoder mode.

Figure 2 illustrates the overall inference framework of ShiftRAG, which consists of three stages: retrieval, semantic signal translation, and reasoning. For offline indexing, the encoder-mode LLM maps every document into an embedding space. In the retrieval stage at inference, the encoder mode embeds the input question into this shared space to retrieve top-K relevant documents. In the semantic signal translation stage, we then translate the question-document embedding pairs and their corresponding relevance scores into soft knowledge tokens. These latent guidance vectors bridge the gap between the discriminative encoder and the generative decoder by mapping retrieval signals directly into the token embedding space of the decoder. Finally, in the reasoning stage, the LLM

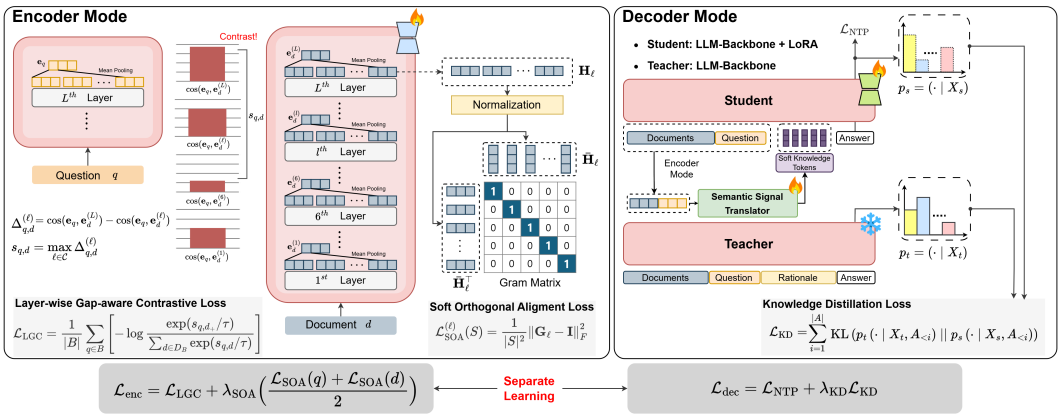

Figure 3: Overall framework of ShiftRAG which separately trains the encoder and decoder modes.

is switched to decoder mode, generating a grounded answer based on the question, the retrieved document text, and the injected soft knowledge tokens.

3.1 Gap-aware and Collapse-resilient Deep Encoder

To repurpose the decoder-only LLM as an encoder, we enable bidirectional self-attention (as illustrated in Figure 3), allowing the model to aggregate global context. To capture deep question-document relevance, we exploit the layer-wise representation gap (Section 3.1.1), which captures the semantic maturation of the input from shallow to deep layers. To mitigate rank collapse, we introduce a soft orthogonality constraint that regularizes the hidden representations of all tokens within a sequence (Section 3.1.2). This ensures that the model preserves discriminative feature diversity, preventing the 'narrow cone' effect often seen in deep transformer layers.

3.1.1 Layer-wise Gap-aware Contrastive Learning

Prior research on intermediate-layer representations in LLM has revealed that layers encode representations hierarchically: lower layers extract lexical and syntactic surface features, while higher layers capture abstract context-driven semantics. DoLa [7], DSAS [38], and RaLa [33] show that such layer-wise representational differences are closely related to reasoning and retrieval quality. From these observations, we propose layer-wise gap-aware contrastive (LGC) learning to fine-tune an LLM-based retriever.

Formally, let q and d denote a question and a document, respectively. Following RaLa [33] and DSAS [38], we partition the first half of the L layers of the LLM into five buckets, and randomly sample one layer from each. Let $\mathcal{C}$ denote the set of these selected candidate layers. For a given question q and document d, we define the relevance gap between the final layer L and any candidate layer $\ell \in \mathcal{C}$:

$$\Delta_{q,d}^{(\ell)} = \cos\left(\mathbf{e}_q, \mathbf{e}_d^{(L)}\right) - \cos\left(\mathbf{e}_q, \mathbf{e}_d^{(\ell)}\right) \tag{1}$$

where $\mathbf{e}_q$ is the final-layer mean-pooled representation of q, and $\mathbf{e}_d^{(\ell)}$ is the ℓ-th layer mean-pooled representation of d. We define the relevance score between q and d as the maximum relevance gap across all candidate layers:

$$s_{q,d} = \max_{\ell \in \mathcal{C}} \Delta_{q,d}^{(\ell)} \tag{2}$$

The choice of the layer with the highest gap ensures the encoder focuses on the maximum representational shift. By doing so, we capture the critical point where the encoder transitions from processing lexical surface features to encoding the abstract relevance necessary to resolve the question. This allows the model to adaptively select the lower layer that best reveals the relevance signal for each pair, rather than using a fixed contrasting layer for all pairs.

For each mini-batch B of questions during fine-tuning, let D_B be a set of positive and hard negative documents for every question $q \in B$. The layer-wise gap-aware contrastive loss is defined below:

$$\mathcal{L}_{\text{LGC}} = \frac{1}{|B|} \sum_{q \in B} \left[-\log \frac{\exp\left(s_{q,d_+}/\tau\right)}{\sum_{d \in D_B} \exp(s_{q,d}/\tau)} \right] \tag{3}$$

where d_+ is the positive document for question q, and τ is a temperature hyperparameter. This objective encourages the positive document to have a larger layer-wise gap score than negative documents, thereby forming a retrieval-oriented embedding space that reflects depth-wise semantic refinement rather than absolute similarity alone. [7]

For inference, we adopt the final-layer cosine similarity score, i.e., $s_{q,d} = \cos(\mathbf{e}_q, \mathbf{e}_d^{(L)})$, to minimize computational overhead. This allows the model to benefit from the discriminative representations formed during gap-aware training without the latency of multi-layer computation. Further analysis of this trade-off is available in Appendix D.3.

3.1.2 Soft Orthogonal Alignment Loss

LLM-based encoders may suffer from rank collapse [6, 12] in deeper layers, where token representations shrink into a low-dimensional subspace. When this tendency becomes severe, embeddings of different tokens in the same sequence become aligned in similar directions, reducing representation diversity. As a result, the encoder may fail to preserve the fine-grained semantic differences required for retrieval, hindering the latent separation of relevant and irrelevant documents.

To alleviate rank collapse [20, 24], we introduce soft orthogonal alignment (SOA) loss, which penalizes excessive correlation among token representations. For an input token sequence S, let $\mathbf{H}_\ell \in \mathbb{R}^{|S|\times h}$ denote the hidden representations of all tokens in S at layer ℓ. We first apply L_2 normalization to the matrix row-wise, denoted as $\bar{\mathbf{H}}_\ell$. We then compute the token-level Gram matrix $\mathbf{G}_\ell = \bar{\mathbf{H}}_\ell \bar{\mathbf{H}}_\ell^\top$, which serves as a pairwise cosine similarity matrix. In an ideal representation space in which tokens are mutually orthogonal, $\mathbf{G}_\ell$ approaches the identity matrix $\mathbf{I}$, with off-diagonal entries vanishing to 0 and diagonal entries remaining at 1. The SOA loss at layer ℓ is defined as the normalized squared Frobenius distance between $\mathbf{G}_\ell$ and $\mathbf{I}$:

$$\mathcal{L}_{\text{SOA}}^{(\ell)}(S) = \tfrac{1}{|S|^2}\|\mathbf{G}_\ell - \mathbf{I}\|_F^2 \tag{4}$$

This loss softly prevents different tokens from collapsing into overly similar directions while preserving useful semantic correlations. We apply this regularization to the last n layers, in which rank collapse is more likely to affect the sequence representations. The question-side and document-side multi-layer SOA losses are defined as:

$$\mathcal{L}_{\text{SOA}}(q) = \tfrac{1}{|B|}\sum_{q\in B}\left(\frac{1}{n}\sum_{\ell=L-n+1}^{L}\mathcal{L}_{\text{SOA}}^{(\ell)}(q)\right), \quad \mathcal{L}_{\text{SOA}}(d) = \frac{1}{|B|}\sum_{d\in D_B}\left(\frac{1}{n}\sum_{\ell=L-n+1}^{L}\mathcal{L}_{\text{SOA}}^{(\ell)}(d)\right) \tag{5}$$

While existing regularizers [20, 24] act on encoder weights or representation dimensionality, SOA regularizes the token representations to preserve representation diversity and retrieval-critical semantic discrimination.

3.1.3 Training LLM-based Encoder

We employ a set of LoRA parameters LoRA_{enc} and optimize them using the joint objective:

$$\mathcal{L}_{\text{enc}} = \mathcal{L}_{\text{LGC}} + \lambda_{\text{SOA}}\left(\tfrac{\mathcal{L}_{\text{SOA}}(q)+\mathcal{L}_{\text{SOA}}(d)}{2}\right) \tag{6}$$

where λ_{SOA} is a hyperparameter. Following LLM2Vec, we initially perform a short MNTP warm-up to adapt LoRA_{enc} to bidirectional attention. More details are provided in Appendix B.1.

3.2 Semantic Signal Translation

Conventional RAG frameworks provide the generator with only raw document text while discarding semantic signals generated by the encoder. Consequently, the burden of determining question-document relevance falls entirely on autoregressive decoding of the generator. By exposing only explicit textual evidence, these methods prevent the LLM from utilizing implicit retrieval priors, e.g.,

[7] While RaLa [33] applies layer contrast as a multiplicative weight over token-level average maximum similarity (MaxSim) for reranking, we directly use the gap-aware relevance score in encoder-mode training. This allows layer contrast to enter the retrieval objective as the score itself, rather than as an auxiliary reweighting term, while avoiding the time/memory overhead of MaxSim, as analyzed in Appendix D.2.

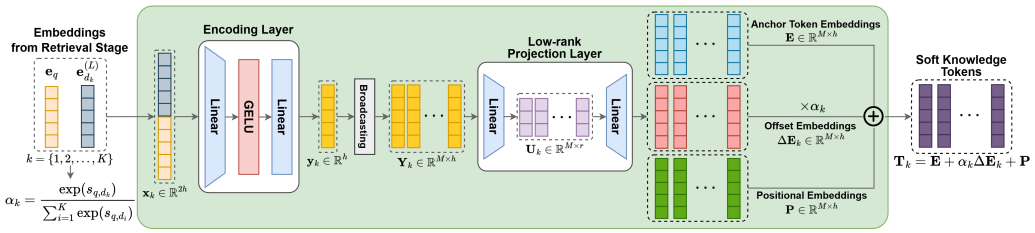

Figure 4: Illustration of Semantic Signal Translator (SST).

the degree of semantic alignment or relative document importance. To bridge this gap, the semantic signal translator (SST) projects retrieval-stage embeddings into the token embedding space of the generator, effectively injecting discriminative knowledge into reasoning.

As shown in Figure 4, SST feeds the concatenation of $\mathbf{e}_q$ and $\mathbf{e}_{d_k}$ into the learnable encoding layer to obtain its hidden representation $\mathbf{y}_k \in \mathbb{R}^h$, which is then broadcast into $\mathbf{Y}_k \in \mathbb{R}^{M\times h}$. A learnable low-rank projection layer maps $\mathbf{Y}_k$ to a low-rank matrix $\mathbf{U}_k \in \mathbb{R}^{M\times r}$, where r is the bottleneck dimension for low-rank signal translation, enabling efficient transfer of retrieval-side discriminative knowledge into the token embedding space. Next, we compute offset embeddings $\Delta\mathbf{E}_k \in \mathbb{R}^{M\times h}$ and scale them by the normalized relevance weight α_k, computed from similarities over the top-K documents. For each document d_k, we compute soft knowledge tokens $\mathbf{T}_k = \mathbf{E}+\alpha_k\Delta\mathbf{E}_k+\mathbf{P}$, where $\mathbf{P} \in \mathbb{R}^{M\times h}$ denotes learnable positional embeddings, and $\mathbf{E} \in \mathbb{R}^{M\times h}$ represents learnable anchor embeddings initialized as pretrained latent vectors [72]. This initialization is detailed in Appendix B.2 and analyzed in Appendix E.2. The anchor embeddings provide vocabulary-aligned references to transform the question-document representations from the LLM encoder into generation-friendly embeddings. Finally, we concatenate the soft knowledge tokens $\mathbf{T}_k$ for top-K documents to form $\mathbf{T}_{1:K}$.

3.3 Soft Knowledge-guided Reasoning

In the reasoning stage, the LLM generates a grounded answer by integrating retrieved document texts with their corresponding soft knowledge tokens. To bridge the gap between retrieval-side discriminative signals and generative reasoning, we optimize the decode-mode adapters LoRA_{dec} via a dual objective: (1) rationale-conditioned self-distillation, where a student LLM with LoRA_{dec} learns to mimic the output distribution of the same frozen LLM backbone conditioned on explicit rationales; and (2) next-token prediction (NTP) on the target answer to preserve linguistic coherence and factual grounding. As illustrated in Figure 3, this configuration ensures that the generator treats the soft knowledge tokens as a latent blueprint for the final response.

For objective (1), we employ knowledge distillation (KD) to encode explicit reasoning signals into the soft knowledge tokens $\mathbf{T}_{1:K}$. We formulate the teacher and student contexts prior to answer generation:

$$X_t = [d_{1:K}, q, \textit{rationale}], \quad X_s = [d_{1:K}, q, \mathbf{T}_{1:K}] \tag{7}$$

where *rationale* denotes a single-sentence textual rationale (see Appendix H.1) pre-constructed for teacher-side supervision. While the teacher utilizes explicit document-level reasoning signals, the student learns to approximate these in $\mathbf{T}_{1:K}$. For a ground-truth answer A, the KD loss aligns the student and teacher distributions at each token position i:

$$\mathcal{L}_{\text{KD}} = \sum_{i=1}^{|A|} \text{KL}\left(p_t\left(\cdot \mid X_t, A_{<i}\right) \,||\, p_s\left(\cdot \mid X_s, A_{<i}\right)\right) \tag{8}$$

where $A_{<i}$ is the answer prefix. Consequently, the student internalizes rationale-guided reasoning without taking text rationales as input. We show the impact of soft token placement in Appendix E.7.

The final objective for the decoder is a weighted sum of the standard NTP loss and the KD loss:

$$\mathcal{L}_{\text{dec}} = \mathcal{L}_{\text{NTP}} + \lambda_{\text{KD}}\mathcal{L}_{\text{KD}} \tag{9}$$

where $\mathcal{L}_{\text{KD}}$ transfers rationale-induced reasoning signals into $\mathbf{T}_{1:K}$, $\mathcal{L}_{\text{NTP}}$ supervises generation of the ground-truth answer, and λ_{KD} is a hyperparameter. The prompt used for decoder-mode training is provided in Appendix H.3.

4 Experiments

4.1 Experimental Setup

We evaluate our framework mainly on five datasets: NaturalQuestions [34] for single-hop QA, HotpotQA [66], 2WikiMultihopQA [22], and MuSiQue [59] for multi-hop QA, and SimpleQA [62] for factual QA. Each dataset contains 1,000 test questions. For the multi-hop QA benchmarks, we follow the test split configuration of RaLa [33] and compare all retrieval-based baselines under identical conditions. We optimize the encoder-mode LoRA adapter ($r = 16$) on the E5 [60] dataset by following LLM2Vec [2]. The decoder-mode LoRA adapter is trained with a higher rank ($r = 64$) on a hop-aware mixture constructed from both single- and multi-hop QA benchmarks. We use Gemma2-9B [14], LlaMA3.1-8B [16], Mistral-7B [29], and Qwen2.5-7B [50] as backbones, apply the SOA loss to the last four layers, and use five soft knowledge tokens for each question–document pair. Further details regarding the experimental setup and baselines are described in Appendix A.

4.2 Main Results

Table 1: Retrieval performance. The best and second-best performances are presented in bold and underlined, respectively. LLM2Vec, OneGen, and ShiftRAG use Gemma2-9B as the backbone.

Methods	**HotpotQA**			**2WikiMultihopQA**			**MuSiQue**		
	R@2	R@5	R@10	R@2	R@5	R@10	R@2	R@5	R@10
LLM2Vec	0.6130	0.7854	0.8612	0.5845	0.6451	0.6829	0.3598	0.5126	0.5977
NV-Embed-v2	<u>0.6318</u>	<u>0.7982</u>	<u>0.8736</u>	<u>0.5956</u>	<u>0.6632</u>	<u>0.7202</u>	<u>0.3781</u>	<u>0.5290</u>	<u>0.6236</u>
OneGen	0.5792	0.7450	0.8402	0.5584	0.6077	0.6392	0.3320	0.4777	0.5658
GritLM	0.5890	0.7487	0.8085	0.5558	0.6311	0.6977	0.3760	0.4990	0.5983
ShiftRAG	**0.6598**	**0.8214**	**0.8806**	**0.6080**	**0.7109**	**0.7530**	**0.4186**	**0.5634**	**0.6516**

Table 2: Question answering performance of ShiftRAG and baselines. The best and second-best performances are presented in bold and underlined, respectively.

Methods	**HotpotQA**		**2WikiMultihopQA**		**MuSiQue**	
	EM	F1	EM	F1	EM	F1
LLM2Vec	0.5030	0.6106	0.4400	0.4914	0.2430	0.3670
NV-Embed-v2	<u>0.5140</u>	<u>0.6220</u>	<u>0.4580</u>	<u>0.5064</u>	<u>0.2480</u>	<u>0.3783</u>
GritLM	0.4430	0.5258	0.3020	0.3665	0.1870	0.2872
OneGen	0.4670	0.5869	0.4220	0.4785	0.2200	0.3342
ShiftRAG	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**

Retrieval Performance. Table 1 compares the retrieval performance of ShiftRAG with state-of-the-art LLM-based encoder and unified LLM baselines. ShiftRAG consistently achieves the best performance across all datasets and all metrics (full results for all baselines are provided in Appendix C.1). Compared to NV-Embed-v2, i.e., the strongest baseline, ShiftRAG improves Recall@2 by 4.43% on HotpotQA (0.6318 $\rightarrow$ 0.6598) and 10.71% on MuSiQue (0.3781 $\rightarrow$ 0.4186). Compared against GritLM, i.e., a strong unified LLM baseline, ShiftRAG improves Recall@2 by 12.02% on HotpotQA (0.5890 $\rightarrow$ 0.6598) and 11.33% on MuSiQue (0.3760 $\rightarrow$ 0.4186). These results show that encoder-mode training forms a more discriminative embedding space than existing LLM-based encoders and unified models. We provide generalization results on NaturalQuestions and MTEB in Appendices C.3 and C.5, respectively, and analyze scalability in Appendix E.1.

Question Answering Performance. Table 2 summarizes the QA performance of ShiftRAG against baselines. To ensure a controlled evaluation of generation capacity, all retrieval-only baselines are paired with Gemma2-9B, the strongest generator among the backbone LLMs. ShiftRAG achieves the highest EM and F1 scores across all datasets (the complete QA results are provided in Appendix C.2). Compared to NV-Embed-v2, ShiftRAG improves EM by 9.14% on HotpotQA (0.5140 $\rightarrow$ 0.5610) and 22.98% on MuSiQue (0.2480 $\rightarrow$ 0.3050). Compared with OneGen under the same backbone setting, ShiftRAG improves EM by 20.13% on HotpotQA (0.4670 $\rightarrow$ 0.5610) and 38.64% on MuSiQue (0.2200 $\rightarrow$ 0.3050). These results demonstrate that translating retrieval-side discriminative knowledge into soft knowledge tokens provides effective reasoning signals that raw text alone cannot provide. Additional generalization experiments on MMLU [21], GSM8K [9], and TyDiQA [8] follow the

protocol of GritLM [45]. The complete results on NaturalQuestions, SimpleQA, and general tasks are provided in Appendices C.3, C.4, and C.5, respectively.

4.3 Ablation Study

Table 3: Ablation study on encoder-mode training losses. The best and second-best performances are presented in bold and underlined, respectively.

Category	HotpotQA			2WikiMultihopQA			MuSiQue		
	R@2	R@5	R@10	R@2	R@5	R@10	R@2	R@5	R@10
ShiftRAG	**0.6598**	**0.8214**	**0.8806**	**0.6080**	**0.7109**	**0.7530**	**0.4186**	**0.5634**	**0.6516**
w/o SOA	<u>0.6535</u>	<u>0.8021</u>	<u>0.8678</u>	<u>0.5998</u>	<u>0.6984</u>	<u>0.7425</u>	<u>0.4045</u>	<u>0.5511</u>	<u>0.6493</u>
w/o LGC	0.6360	0.7892	0.8445	0.5816	0.6917	0.7084	0.4008	0.5457	0.6140
w/o Both	0.6120	0.7800	0.8352	0.5711	0.6824	0.7005	0.3916	0.5382	0.6051

We conduct an ablation study to evaluate the impact of the encoder-mode training objectives. Table 3 reports the retrieval performance for three training setups: (a) *w/o* SOA removes SOA loss while retaining LGC; (b) *w/o* LGC replaces the LGC loss with standard InfoNCE loss [48] while retaining SOA, and (c) *w/o* Both optimizes the encoder solely using standard InfoNCE. *w/o* Both yields the largest degradation. Between the two components, the LGC loss drives the majority of the performance gains; *w/o* LGC reduces Recall@10 on MuSiQue by 5.77%. Removing SOA results in a marginal but consistent reduction. As a result, while exploiting layer-wise semantic progression (LGC) provides a powerful relevance signal, suppressing excessive token correlations (SOA) acts as an effective complementary regularizer to preserve fine-grained semantic discrimination.

Table 4: Ablation study on decoder-mode training components. The best and second-best performances are presented in bold and underlined, respectively.

Category	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
ShiftRAG	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**
w/o SST	0.5230	0.6468	0.4630	0.5194	0.2450	0.3639
w/o LoRA_{dec}	<u>0.5300</u>	<u>0.6570</u>	<u>0.4720</u>	<u>0.5212</u>	<u>0.2650</u>	<u>0.3883</u>
w/o Both	0.5020	0.6253	0.4400	0.4980	0.1970	0.2979

We investigate the impact of individual components for decoder mode. Table 4 shows QA performance under four settings: *w/o* SST removes SST and trains only LoRA_{dec}; *w/o* LoRA_{dec} retains SST but removes generator-side LoRA tuning; *w/o* Both uses the backbone without decoder-mode training. Between the two components, SST contributes more to QA performance: *w/o* SST reduces MuSiQue EM by 19.67% ($0.3050 \rightarrow 0.2450$), whereas *w/o* LoRA_{dec} reduces it by 13.11% ($0.3050 \rightarrow 0.2650$). These results indicate that translating retrieval-side signals into soft knowledge tokens is more effective than fine-tuning the generator only with retrieved documents. We further conduct a detailed SST component ablation in Appendix C.7. We analyze how SST affects confidence calibration and generator-side LoRA adaptation in Appendix E.3.

4.4 Impact of Learning Strategies

Table 5: Performance and training time comparisons across different learning strategies and parameter sharing configurations. The best and second-best scores are bolded and underlined, respectively.

Training method	Parameter	HotpotQA				2WikiMultihopQA				Time (h)
		Retrieval		QA		Retrieval		QA		
		R@2	R@5	EM	F1	R@2	R@5	EM	F1	
Joint learning	shared	0.4896	0.6298	0.1300	0.3430	0.3230	0.4666	0.1270	0.3432	<u>21</u>
	separate	<u>0.5264</u>	<u>0.6865</u>	<u>0.4730</u>	<u>0.5885</u>	<u>0.5159</u>	<u>0.6184</u>	<u>0.4360</u>	<u>0.5115</u>	23
Separate learning	separate	**0.6598**	**0.8214**	**0.5610**	**0.6790**	**0.6080**	**0.7109**	**0.5080**	**0.5528**	**10**

Table 5 shows the impact of joint learning versus separate learning strategies. Joint learning with shared parameters exhibits severe optimization instability, leading to a large degradation in QA

performance, e.g., EM drops precipitously to 0.1300 on HotpotQA and 0.1270 on 2WikiMultihopQA. Our analysis reveals that the encoder and decoder objectives produce negatively aligned gradients on shared LoRA parameters in 84% of updates. This frequent gradient conflict indicates that the retrieval task and the reasoning task possess fundamentally competing optimization directions. While utilizing separate parameters for each mode mitigates this degradation, simultaneous joint optimization remains suboptimal and computationally expensive. In contrast, separate learning using separate LoRA parameters demonstrates superior efficacy and efficiency. These results support our hypothesis that decoupling discriminative retrieval learning from autoregressive generation allows each mode to converge toward its respective optimum without mutual interference.

4.5 Impact of Semantic Signal Translator

Table 6: Impact of SST-generated soft knowledge tokens on QA performance. The best and second-best performances are presented in bold and underlined, respectively.

Category	**HotpotQA**		**2WikiMultihopQA**		**MuSiQue**	
	EM	F1	EM	F1	EM	F1
ShiftRAG	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**
Textual Evidence	<u>0.5330</u>	<u>0.6574</u>	<u>0.4790</u>	<u>0.5226</u>	<u>0.2650</u>	<u>0.3814</u>
w/o SST	0.5230	0.6468	0.4630	0.5194	0.2450	0.3639

We investigate whether the semantic information encapsulated in our soft knowledge tokens can be reduced to simple textualized relevance signals. Table 6 compares three settings: (1) ShiftRAG; (2) *Textual Evidence*, which replaces soft tokens from SST with an explicit hard prompt verbalizing the relevance score of the retrieved documents; and (3) *w/o* SST, which removes the soft tokens. While *Textual Evidence* consistently outperforms the *w/o* SST setting, confirming that the generator benefits from knowing retrieval-side relevance, ShiftRAG achieves higher performance on all datasets, improving MuSiQue EM by 15.09% (0.2650 $\rightarrow$ 0.3050) and F1 by 8.26% (0.3814 $\rightarrow$ 0.4129) over the textual baseline. These results demonstrate that soft tokens do not merely act as verbalized scores; instead, they serve as an effective interface that transfers complex, high-dimensional discriminative signals that are lost during textualization.

4.6 Geometric Intuition for Efficacy of Semantic Signal Translator

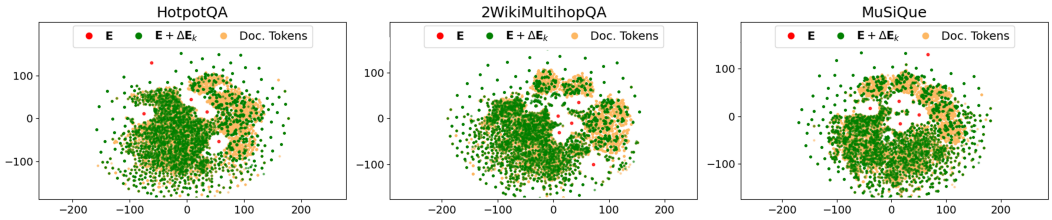

Figure 5: t-SNE visualization of anchor token embeddings (red), anchor token embeddings + offset embeddings (green), and original document token embeddings (orange) for each benchmark.

Figure 5 visualizes the anchor token embeddings $\mathbf{E}$, the translated embeddings $\mathbf{E} + \Delta\mathbf{E}_k$, and the decoder-mode document token embeddings (*Doc. Tokens*) in a shared t-SNE space. The anchor embeddings $\mathbf{E}$ form a localized set of reference points that are poorly aligned with the distribution of document tokens. However, upon incorporating the offset embeddings $\Delta\mathbf{E}_k$, the resultant vectors migrate toward the document token clusters, exhibiting a substantial distributional overlap across all datasets. This shift suggests that $\Delta\mathbf{E}_k$ effectively projects the anchor-token basis into a vocabulary-aligned region that is semantically congruent with the retrieved evidence. Consequently, the soft knowledge tokens enable SST to infuse the generator with retrieval-side knowledge effectively. This visualization confirms that our anchor-plus-translation design serves as a robust bridge between the retriever's latent space and the generator's reasoning manifold.

4.7 Inference Latency

We measure inference latency to evaluate the runtime cost of SST-based generation. Table 7 shows end-to-end latency, peak VRAM usage, and QA performance on HotpotQA. ShiftRAG requires 2.79s

Table 7: End-to-end inference latency, peak VRAM usage, and QA performance on HotpotQA.

Methods	Inference Latency		
	Time (s)	VRAM (GiB)	EM
NV-Embed-v2	8.72	38.82	0.514
GritLM	2.42	20.25	0.443
ShiftRAG	2.79	24.14	0.561

per example and 24.14 GiB of VRAM, while achieving the best EM of 0.561. The latency breakdown is 0.13s for question encoding, 0.02s for Faiss retrieval, 0.0007s for soft knowledge token generation, and 2.64s for answer generation. This breakdown confirms that the primary computational bottleneck resides in the LLM decoding process, while the overhead introduced by retrieval and SST remains negligible. Compared with GritLM [45], ShiftRAG is slightly slower (2.42s $\rightarrow$ 2.79s) and uses more VRAM (20.25 $\rightarrow$ 24.14 GiB), but improves EM by 26.64% (0.443 $\rightarrow$ 0.561). Compared with NV-Embed-v2 [35], ShiftRAG is faster (8.72s $\rightarrow$ 2.79s), uses less VRAM (38.82 $\rightarrow$ 24.14 GiB), and improves EM by 9.14% (0.514 $\rightarrow$ 0.561). These results indicate that ShiftRAG provides a favorable inference-time trade-off between efficiency and QA accuracy. The same trend is observed on 2WikiMultihopQA and MuSiQue. Training latency analysis is provided in Appendix D.1.

4.8 Impact of the Number of Soft Knowledge Tokens

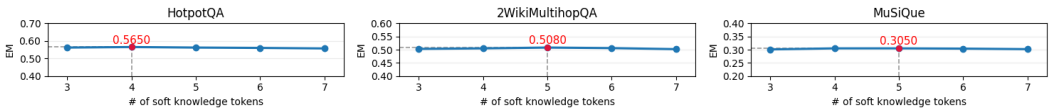

Figure 6: Impact of the number of soft knowledge tokens on QA performance.

We analyze the sensitivity of QA performance to the number M of soft knowledge tokens per document. As shown in Figure 6, the performance remains stable across the range of 3 to 7 tokens. While the optimal configuration varies slightly across datasets, the maximum variation is a negligible 1.43%. Hence, we fix M to 5 for all experiments. Appendix F shows other hyperparameter studies.

5 Conclusion and Limitation

We introduce ShiftRAG, a decoupled RAG framework that isolates discriminative retrieval learning from autoregressive generation. By proposing the semantic signal translator (SST), we successfully shift implicit high-dimensional retrieval alignments into latent soft knowledge tokens, providing an effective bridge between the retriever and generator. Furthermore, the soft orthogonality objective effectively mitigates rank collapse, while layer-wise gap-aware contrastive learning preserves the fine-grained latent topology required for robust evidence discrimination. Empirical evaluations demonstrate that ShiftRAG achieves superior retrieval accuracy and downstream QA quality.

Despite its strong empirical performance, our framework presents two limitations. Unlike explicit CoT pipelines, the soft knowledge tokens are continuous and inherently uninterpretable. Because our architecture relies on extracting and translating internal embedding states, it requires white-box access to the model weights; it cannot be applied to black-box LLMs.

Acknowledgments and Disclosure of Funding

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II201373, Artificial Intelligence Graduate School Program(Hanyang University)), by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT, Ministry of Science and ICT) (No. RS-2026-25548925), and by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. RS-2025-25424679).

References

[1] Federico Barbero, Andrea Banino, Steven Kapturowski, Dharshan Kumaran, João Guilherme Madeira Araújo, Alex Vitvitskyi, Razvan Pascanu, and Petar Veličković. Transformers need glasses! information over-squashing in language tasks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=93HCE8vTye.

[2] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. LLM2vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=IW1PR7vEBf.

[3] Abhra Chaudhuri, Anjan Dutta, Tu Bui, and Serban Georgescu. A closer look at multimodal representation collapse. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Vf9f7eNX6T.

[4] Lu Chen, Ruqing Zhang, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. Controlling risk of retrieval-augmented generation: A counterfactual prompting framework. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2380–2393, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.133. URL https://aclanthology.org/2024.findings-emnlp.133/.

[5] Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and Dong Yu. Dense X retrieval: What retrieval granularity should we use? In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.845. URL https://aclanthology.org/2024.emnlp-main.845/.

[6] Zheng-An Chen and Tao Luo. From condensation to rank collapse: A two-stage analysis of transformer training dynamics. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=gm5mkiTGOy.

[7] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Th6NyL07na.

[8] Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020. doi: 10.1162/tacl_a_00317. URL https://aclanthology.org/2020.tacl-1.30/.

[9] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[10] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3563–3578, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.212. URL https://aclanthology.org/2024.findings-acl.212/.

[11] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International conference on machine learning*, pages 2793–2803. PMLR, 2021.

[12] Yihe Dong, Lorenzo Noci, Mikhail Khodak, and Mufan Li. Is random attention sufficient for sequence modeling? disentangling trainable components in the transformer, 2025. URL https://arxiv.org/abs/2506.01115.

[13] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George J. Pappas. *Efficient and accurate estimation of lipschitz constants for deep neural networks*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[14] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, et al. Gemma 2: Improving Open Language Models at a Practical Size, 2024. URL https://arxiv.org/abs/2408.00118.

[15] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

[16] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 Herd of Models, 2024. URL https://arxiv.org/abs/2407.21783.

[17] Kyubeen Han, Youngjin Jang, and Harksoo Kim. Explainable multi-hop question answering: A rationale-based approach. *Big Data and Cognitive Computing*, 9(11), 2025. ISSN 2504-2289. doi: 10.3390/bdcc9110273. URL https://www.mdpi.com/2504-2289/9/11/273.

[18] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language model to reason in a continuous latent space, 2025. URL https://openreview.net/forum?id=tG4SgayTtk.

[19] Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: stability of stochastic gradient descent. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1225–1234. JMLR.org, 2016.

[20] Junlin He, Jinxiao Du, and Wei Ma. Preventing dimensional collapse in self-supervised learning via orthogonality regularization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=Y3FjKSsfmy.

[21] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

[22] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

[23] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[24] Hanxun Huang, Ricardo J. G. B. Campello, Sarah Monazam Erfani, Xingjun Ma, Michael E. Houle, and James Bailey. LDReg: Local dimensionality regularized self-supervised learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=oZyAqjAjJW.

[25] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, January 2025. ISSN 1558-2868. doi: 10.1145/3703155. URL http://dx.doi.org/10.1145/3703155.

[26] Yu Huang, Junyang Lin, Chang Zhou, Hongxia Yang, and Longbo Huang. Modality competition: What makes joint training of multi-modal network fail in deep learning? (Provably). In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9226–9259. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/huang22e.html.

[27] Andrew Jesson, Nicolas Beltran-Velez, Quentin Chu, Sweta Karlekar, Jannik Kossen, Yarin Gal, John P. Cunningham, and David Blei. Estimating the hallucination rate of generative ai. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 31154–31201. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3791f5fc0e8e43730466afd2bcdb7493-Paper-Conference.pdf.

[28] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), March 2023. ISSN 0360-0300. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

[29] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

[30] 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 Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

[31] Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 39–48, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450380164. doi: 10.1145/3397271.3401075. URL https://doi.org/10.1145/3397271.3401075.

[32] Savya Khosla, Aditi Tiwari, Kushal Kafle, Simon Jenni, Handong Zhao, John Collomosse, and Jing Shi. MAGNET: Augmenting generative decoders with representation learning and infilling capabilities. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27328–27346, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1325. URL https://aclanthology.org/2025.acl-long.1325/.

[33] Youmin Ko, Sung Jong Seo, and Hyunjoon Kim. Cooperative retrieval-augmented generation for question answering: Mutual information exchange and ranking by contrasting layers. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=D2XdJf1tXW.

[34] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

[35] Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=lgsyLSsDRe.

[36] Yibin Lei, Liang Ding, Yu Cao, Changtong Zan, Andrew Yates, and Dacheng Tao. Unsupervised dense retrieval with relevance-aware contrastive pre-training. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10932–10940, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.695. URL https://aclanthology.org/2023.findings-acl.695/.

[37] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[38] Jiakai Li, Rongzheng Wang, Yizhuo Ma, Shuang Liang, Guangchun Luo, and Ke Qin. DSAS: A universal plug-and-play framework for attention optimization in multi-document question answering. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=y68Q09Vc4K.

[39] Mufei Li, Siqi Miao, and Pan Li. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=JvkuZZ04O7.

[40] Yanyang Li, Shuo Liang, Michael Lyu, and Liwei Wang. Making long-context language models better multi-hop reasoners. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2462–2475, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.135. URL https://aclanthology.org/2024.acl-long.135/.

[41] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning, 2023. URL https://arxiv.org/abs/2308.03281.

[42] Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=oFBu7qaZpS.

[43] Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. Multi-hop question answering. *Foundations and Trends® in Information Retrieval*, 17(4):457–586, 2024.

[44] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.148. URL https://aclanthology.org/2023.eacl-main.148/.

[45] Niklas Muennighoff, Hongjin SU, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=BC4lIvfSzv.

[46] Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. Large dual encoders are generalizable retrievers. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.669. URL https://aclanthology.org/2022.emnlp-main.669/.

[47] Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: theoretical perspectives and the role of rank collapse. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

[48] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[49] Minbae Park, Hyemin Yang, Jeonghyun Kim, Kunsoo Park, and Hyunjoon Kim. Prograg: Hallucination-resistant progressive retrieval and reasoning over knowledge graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(39):32674–32682, Mar. 2026. doi: 10.1609/aaai.v40i39.40545. URL https://ojs.aaai.org/index.php/AAAI/article/view/40545.

[50] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

[51] Pranab Sahoo, Prabhash Meharia, Akash Ghosh, Sriparna Saha, Vinija Jain, and Aman Chadha. A comprehensive survey of hallucination in large language, image, video and audio foundation models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11709–11724, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.685. URL https://aclanthology.org/2024.findings-emnlp.685/.

[52] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.272. URL https://aclanthology.org/2022.naacl-main.272/.

[53] Sunny Sanyal, Ravid Shwartz-Ziv, Alex Dimakis, and sujay sanghavi. When attention collapses: How degenerate layers in LLMs enable smaller, stronger models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=2zQn0bUoPf.

[54] Kevin Scaman and Aladin Virmaux. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 3839–3848, Red Hook, NY, USA, 2018. Curran Associates Inc.

[55] Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.36. URL https://aclanthology.org/2025.emnlp-main.36/.

[56] Aaditya Singh, Adam Fry, Adam Perelman, et al. OpenAI GPT-5 System Card, 2025. URL https://arxiv.org/abs/2601.03267.

[57] Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nnVO1PvbTv.

[58] Jiashuo Sun, Shixuan Liu, Zhaochen Su, Xianrui Zhong, Pengcheng Jiang, Bowen Jin, Peiran Li, Weijia Shi, and Jiawei Han. GRACE: Generative representation learning via contrastive policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=hs9lwjH1bJ.

[59] 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. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

[60] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.642. URL https://aclanthology.org/2024.acl-long.642/.

[61] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

[62] Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models, 2024. URL https://arxiv.org/abs/2411.04368.

[63] Jiaqi Wei, Hao Zhou, Xiang Zhang, Di Zhang, Zijie Qiu, Noah Wei, Jinzhe Li, Wanli Ouyang, and Siqi Sun. Retrieval is not enough: Enhancing RAG through test-time critique and optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=cnUq7GkS6d.

[64] Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. SIM-cot: Supervised implicit chain-of-thought. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=6YRJ4jmVQl.

[65] Xinyi Wu, Amir Ajorlou, Yifei Wang, Stefanie Jegelka, and Ali Jadbabaie. On the role of attention masks and layernorm in transformers. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

[66] 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 Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

[67] Wenhao Yu, Hongming Zhang, Xiaoman Pan, Peixin Cao, Kaixin Ma, Jian Li, Hongwei Wang, and Dong Yu. Chain-of-note: Enhancing robustness in retrieval-augmented language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14672–14685, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.813. URL https://aclanthology.org/2024.emnlp-main.813/.

[68] Jintian Zhang, Cheng Peng, Mengshu Sun, Xiang Chen, Lei Liang, Zhiqiang Zhang, Jun Zhou, Huajun Chen, and Ningyu Zhang. OneGen: Efficient one-pass unified generation and retrieval for LLMs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4088–4119, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.237. URL https://aclanthology.org/2024.findings-emnlp.237/.

[69] Qin Zhang, Shangsi Chen, Dongkuan Xu, Qingqing Cao, Xiaojun Chen, Trevor Cohn, and Meng Fang. A survey for efficient open domain question answering. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14447–14465, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.808. URL https://aclanthology.org/2023.acl-long.808/.

[70] XiaoHui Zhang, Jaehong Yoon, Mohit Bansal, and Huaxiu Yao. Multimodal representation learning by alternating unimodal adaptation, 2024. URL https://openreview.net/forum?id=JFEFORc7fP.

[71] Yifan Zhang, Zhiquan Tan, Jingqin Yang, Weiran Huang, and Yang Yuan. Matrix information theory for self-supervised learning. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=wleAlsklEh.

[72] Yutao Zhu, Zhaoheng Huang, Zhicheng Dou, and Ji-Rong Wen. One token can help! learning scalable and pluggable virtual tokens for retrieval-augmented large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):26166–26174, Apr. 2025. doi: 10.1609/aaai.v39i24.34813. URL https://ojs.aaai.org/index.php/AAAI/article/view/34813.

Appendices

In this supplementary material, we provide further details on the following topics:

- **Appendix A: Experimental Setup**
- **Appendix B: Warm-up Training**
  - B.1: Encoder mode warm-up training
  - B.2: Anchor Token Initialization
- **Appendix C: Overall Performance**
  - C.1: Retrieval Performance
  - C.2: Question Answering Performance
  - C.3: Retrieval and Question Answering Performance on Single-hop QA Benchmark
  - C.4: Question Answering Performance on SimpleQA
  - C.5: Embedding and Generative Generalization
  - C.6: Performance Comparison with Latent-CoT Baselines
  - C.7: Ablation Study: Semantic Signal Translator
  - C.8: Performance Comparison with Simple Projection Baselines
  - C.9: Impact of Causal-Mask Relaxation
- **Appendix D: Latency**
  - D.1: Training Latency
  - D.2: Complexity Comparison with RaLa
  - D.3: Effectiveness-Efficiency Trade-off of Different Relevance Scoring Methods
- **Appendix E: Analysis**
  - E.1: Scalability Analysis
  - E.2: Impact of Anchor Token Initialization
  - E.3: Impact of SST on Confidence Calibration and LoRA Adaptation
  - E.4: Impact of SST with Fixed External Retrievers
  - E.5: Theoretical Interpretation of Learning Strategy
  - E.6: Overconfidence Analysis on Correct and Incorrect Cases
  - E.7: Effect of Soft Knowledge Token Placement
  - E.8: t-SNE Visualization Before and After Training with Bidirectional and Unidirectional Attention
  - E.9: Effective Rank Before and After Encoder-mode Training across LLM Backbones
  - E.10: SST Effectiveness on Modern Large LLMs
- **Appendix F: Hyperparameter Sensitivity**
  - F.1: Impact of Batch Size
  - F.2: Impact of LoRA Rank
  - F.3: Impact of λ_{SOA}
  - F.4: Impact of λ_{KD}
  - F.5: Impact of τ in $\mathcal{L}_{\text{LGC}}$
  - F.6: Impact of Candidate Layers in $\mathcal{L}_{\text{LGC}}$
  - F.7: Impact of Number of Layers in λ_{SOA}
  - F.8: Impact of LLM Backbones
- **Appendix G: Case Study**
  - G.1: Knowledge Comparison across Different Attention Policies
- **Appendix H: Prompt**
  - H.1: Prompt for Rationale Generation
  - H.2: Prompt for Anchor Token Initialization
  - H.3: Prompt for Decoder-Mode Training

A Experimental Setup

Table 8: Training-set statistics for decoder mode.

	1-hop	2-hop	3-hop	4-hop	Total
Examples	2,000	6,000	500	1,500	10,000

Table 9: Dataset statistics (test sets)

Category	Single-hop QA	Multi-hop QA			Factual QA
	NQ	HotpotQA	2WikiMultihopQA	MuSiQue	SimpleQA
Questions	1,000	1,000	1,000	1,000	1,000
Documents	9,633	9,811	6,119	11,656	20,686

Datasets. We train encoder mode on the E5 [60] dataset following LLM2Vec [2]. For decoder mode, we construct a hop-aware mixed training set by sampling instances from the training splits of NaturalQuestions [34], HotpotQA [66], 2WikiMultihopQA [22], and MuSiQue [59] according to hop count. Table 8 shows the training-set statistics. The final training set contains 10,000 examples: 2,000 1-hop, 6,000 2-hop, 500 3-hop, and 1,500 4-hop examples. Table 9 shows the test-set statistics for HotpotQA, 2WikiMultihopQA, MuSiQue, and SimpleQA [62]. Each benchmark contains 1,000 questions, and *Documents* denotes the size of the retrieval pool. We follow the test split configuration of RaLa [33] and compare all baselines under identical conditions.

Baselines. We evaluate ShiftRAG against four categories of baselines. Encoder-based retrievers include Contriever [36], GTR [46], ColBERTv2 [52], and RaLa [33]. LLM-based retrievers include GTE-Qwen2-7B-Instruct [41], LLM2Vec [2], and NV-Embed-v2 [35]. Unified LLM baselines include OneGen [68] and GritLM [45]. Latent-CoT baselines include Coconut [18], CODI [55], and Sim-CoT [64]. For fair comparison, we fine-tune LLM2Vec and OneGen using the same three backbones as ShiftRAG: Gemma2-9B [14], LlaMA3.1-8B [16], and Mistral-7B [29]. We use Gemma2-9B as the default backbone because it achieves the strongest retrieval and QA performance for both ShiftRAG and the fine-tuned baselines among the tested LLM backbones. Unless otherwise specified, we report results using Gemma2-9B. Since GritLM requires full fine-tuning, we use the open-weight checkpoint released on Hugging Face without additional training. Latent-CoT methods are not originally designed for retrieval-based QA; therefore, their adapted results are reported separately in Appendix C.6.

Evaluation Metrics. For evaluating ShiftRAG, we select the most commonly used retrieval, QA, embedding and generative benchmarks. For the retrieval task, to measure the ability to identify relevant information within a large-scale corpus, we use **Recall@**K (R@K) as the primary metric, where $K \in \{2, 5, 10\}$. It measures the proportion of instances where the ground-truth document is ranked within the top-K retrieved results. This metric focuses on the capacity of the model to narrow down the search space accurately.

For the question answering task, to measure the precision of the model in generating correct answers based on provided contexts or internal knowledge, we employ two standard metrics: **Exact Match (EM)** measures the percentage of predictions that exactly match the ground-truth answers. **F1-score** evaluates the harmonic mean of precision and recall based on the word-level overlap between the predicted and ground truth answers, providing a broader assessment of semantic alignment.

Following the setup of GritLM [45] and the MTEB [44] benchmark, we also evaluate the quality of the learned embedding representations through three different tasks:

(1) **Clustering (Clust.)**: We perform K-means clustering on the text embeddings and assess the alignment between the cluster assignments and ground-truth labels using the **V-measure**.

(2) **Semantic Textual Similarity (STS)**: We compute the cosine similarity between sentence embeddings and report the **Spearman correlation** with human-annotated similarity scores to verify how well the embeddings capture semantic relatedness.

(3) **Summarization (Summ.)**: We measure the cosine similarity between the embeddings of original texts and their summaries. The performance is scored via **Spearman correlation** against human quality ratings to evaluate the model's ability to encode core information.

To assess reasoning and multilingual proficiency, we follow the generative evaluation framework of GritLM [45], utilizing the following benchmarks:

(1) **MMLU**: We evaluate the model's broad world knowledge across various disciplines (humanities, social sciences, etc.) in a Zero-shot (0 FS) setting, using **Exact Match (EM)** as the evaluation metric.

(2) **GSM8K**: To test multi-step mathematical reasoning, we apply 8-shot (8 FS) prompting with Chain-of-Thought (CoT). The performance is measured by the **Exact Match (EM)** of the final numerical answer.

(3) **TyDiQA**: We evaluate multilingual QA capabilities using the Gold Passage (GP) and zero-shot (0 FS) setup. Given the linguistic diversity of the tasks, we report the **F1-score** to assess the accuracy and robustness of the generated responses across different languages.

Implementation Details. We train ShiftRAG on four NVIDIA RTX A6000 GPUs and run inference on a single GPU. Both encoder mode and decoder mode are trained with LoRA [23] adapters. For the encoder, we employ a global batch size of 16 and a LoRA rank of $r = 16$; for the decoder, we use a global batch size of 8 and $r = 64$. The LGC loss is calculated with a bucket size of four layers and a temperature of $\tau = 0.05$. The SOA loss is applied to the final $n = 4$ candidate layers with a coefficient of $\lambda_{\text{SOA}} = 1 \times 10^{-3}$. In decoder mode, the knowledge distillation weight is set to $\lambda_{\text{KD}} = 0.75$. SST is configured to generate five soft knowledge tokens per question–document pair. Experiments on MTEB [44], MMLU [21], GSM8K [9], and TyDiQA [8] follow the protocol of GritLM [45]. We evaluate our approach across several backbone LLMs, including Gemma2-9B [14], LlaMA3.1-8B [16], Mistral-7B [29], and Qwen2.5-7B [50].

B Warm-up Training

B.1 Encoder mode warm-up training

Before optimizing the encoder-specific LoRA adapters, LoRA_{enc} with the primary objective $\mathcal{L}_{\text{enc}}$, we perform a brief warm-up stage by using masked next-token prediction (MNTP), following LLM2Vec. Let $S = (x_1, \ldots, x_{|S|})$ be an input token sequence, $\tilde{S}$ its masked version, and M_S the set of masked positions. Under bidirectional attention, MNTP predicts each masked token x_i using the final-layer token representation at the previous position $i - 1$. This design preserves the autoregressive next-token prediction format inherent to decoder-only LLM pre-training:

$$\mathcal{L}_{\text{MNTP}}(S) = -\frac{1}{|M_S|} \sum_{i \in M_S} \log \Big[\text{softmax}\big(W_{\text{LM}} \mathbf{z}^{(L)}_{\tilde{S}, i-1}\big) \Big] \tag{10}$$

where $\mathbf{z}^{(L)}_{\tilde{S}, i-1} \in \mathbb{R}^h$ denotes the token representation of the final layer at position $i - 1$ computed from $\tilde{S}$, and W_{LM} represents the language modeling head. For $i = 1$, the previous position corresponds to the `[BOS]` token. For a mini-batch B of questions and the corresponding document set D_B, we average MNTP over both questions and documents:

$$\mathcal{L}_{\text{MNTP}}(B, D_B) = \frac{1}{2} \Big(\frac{1}{|B|} \sum_{q \in B} \mathcal{L}_{\text{MNTP}}(q) + \frac{1}{|D_B|} \sum_{d \in D_B} \mathcal{L}_{\text{MNTP}}(d) \Big) \tag{11}$$

The warm-up objective integrates this with the batch-averaged SOA penalty:

$$\mathcal{L}_{\text{warm}} = \mathcal{L}_{\text{MNTP}}(B, D_B) + \lambda_{\text{SOA}} \frac{\mathcal{L}_{\text{SOA}}(q) + \mathcal{L}_{\text{SOA}}(d)}{2} \tag{12}$$

In our experimental setup, out of a total training budget of 3,000 steps, only the initial 1,000 steps utilize MNTP in place of the contrastive loss $\mathcal{L}_{\text{LGC}}$. In practice, the warm-up stage takes less than one hour.

B.2 Anchor Token Initialization

In our decoder-mode training, we initialize the anchor token embeddings as the pretrained latent vectors [72] to align soft knowledge tokens with the token embedding space of the generator. For a question q and a retrieved document d_k, SST constructs soft knowledge tokens $\mathbf{T}_k \in \mathbb{R}^{M \times h}$ by combining anchor token embeddings $\mathbf{E} \in \mathbb{R}^{M \times h}$ with document-conditioned translation embeddings and positional embeddings, as described in Section 3.3. Since the document-conditioned translation embeddings are derived from encoder-mode representations, the resulting soft knowledge tokens may not be well aligned with the token embedding space used by the generator. Following SPRING [72], we freeze the student LLM and optimize only the anchor token embeddings $\mathbf{E}$, which serve as the trainable basis of $\mathbf{T}_k$.

For each training question-document pair (q, d_k), we use an offline-generated target rationale that explains whether d_k helps answer the question and how it contributes to the answer. The rationale construction procedure and the prompt for anchor token initialization are provided in Appendices H.1 and H.2, respectively. Given d_k, q, and $\mathbf{E}$, we form the initialization context as

$$X^{\text{ATI}} = [\text{Instruction}, d_k, q, \mathbf{E}] \tag{13}$$

The frozen student LLM then autoregressively predicts the target rationale from X^{ATI} using the standard next-token prediction (NTP) objective. During this stage, gradients are back-propagated only to $\mathbf{E}$, while all other

parameters remain fixed. This procedure therefore does not tune the student LLM itself; it only learns a small set of anchor embeddings that can be plugged into the generator's token space.

This initialization is performed once before the decoder-mode training. By optimizing $\mathbf{E}$ with rationale supervision, this stage provides a vocabulary-aligned basis for SST-generated soft knowledge tokens and stabilizes the later translation of retrieval-side signals into the decoder. The initialized anchor embeddings do not encode document-specific knowledge by themselves; rather, they act as reference points that allow the document-conditioned translation embeddings to produce generation-friendly soft knowledge tokens. The target rationales are strictly reserved for the training pretraining anchor token embeddings and for teacher-side supervision employed by knowledge distillation. Empirically, the optimization of anchor token embeddings $\mathbf{E}$ takes less than one hour.

C Overall Performance

C.1 Retrieval Performance

Table 10: Complete retrieval performance of ShiftRAG and baselines. The best and second-best performances are presented in bold and underlined, respectively. We conduct five runs with different random seeds and report the mean; the maximum standard deviation is ±0.05.

Methods	**HotpotQA**			**2WikiMultihopQA**			**MuSiQue**		
	R@2	R@5	R@10	R@2	R@5	R@10	R@2	R@5	R@10
Encoder-based Retrieval									
Contriever	0.4789	0.6185	0.7214	0.5060	0.6349	0.6386	0.3319	0.4646	0.5648
GTR (T5-base)	0.4913	0.6409	0.7379	0.5154	0.6256	0.6487	0.3314	0.4799	0.5754
ColBERTv2	0.5034	0.6971	0.7561	0.5460	0.6355	0.6565	0.3392	0.4823	0.5859
RaLa	0.5485	0.6800	0.7415	0.5510	0.6563	0.7007	0.3530	0.5047	0.5882
LLM-based Retrieval									
GTE-Qwen2-7B-Instruct	0.5892	0.7636	0.8296	0.5474	0.6222	0.6586	0.3408	0.4630	0.5757
LLM2Vec [Mistral-7B]	0.6085	0.7674	0.8403	0.5782	0.6393	0.6756	0.3571	0.4827	0.5901
LLM2Vec [LlaMA3.1-8B]	0.6096	0.7642	0.8532	0.5985	0.6762	0.7080	0.3740	0.5005	0.5832
LLM2Vec [Qwen2.5-7B]	0.6008	0.7660	0.8388	0.5778	0.6342	0.6780	0.3327	0.4580	0.5860
LLM2Vec [Gemma2-9B]	0.6130	0.7854	0.8612	0.5845	0.6451	0.6829	0.3598	0.5126	0.5977
NV-Embed-v2	0.6318	0.7982	0.8736	0.5956	0.6632	0.7202	0.3781	0.5290	0.6236
Unified LLM									
OneGen [Mistral-7B]	0.5855	0.7442	0.8037	0.5525	0.6273	0.6935	0.3738	0.4960	0.5947
OneGen [LlaMA3.1-8B]	0.5950	0.7553	0.8352	0.5950	0.6580	0.6964	0.3458	0.4772	0.5786
OneGen [Qwen2.5-7B]	0.5898	0.7634	0.8360	0.5760	0.6326	0.6690	0.3221	0.4468	0.5770
OneGen [Gemma2-9B]	0.5792	0.7450	0.8402	0.5584	0.6077	0.6392	0.3320	0.4777	0.5658
GritLM [Mistral-7B]	0.5890	0.7487	0.8085	0.5558	0.6311	0.6977	0.3760	0.4990	0.5983
ShiftRAG [Mistral-7B]	0.6235	0.7890	0.8536	0.5837	0.6683	0.7304	0.4019	0.5296	0.6320
ShiftRAG [LlaMA3.1-8B]	**0.6685**	0.8180	**0.8825**	0.6025	0.6960	0.7438	**0.4338**	0.5593	0.6485
ShiftRAG [Qwen2.5-7B]	0.6505	0.8020	0.8795	0.5837	0.6757	0.7188	0.3753	0.5039	0.5999
ShiftRAG [Gemma2-9B]	0.6598	**0.8214**	0.8806	**0.6080**	**0.7109**	**0.7530**	0.4186	**0.5634**	**0.6516**

Table 10 shows the complete retrieval results, extending Table 1 with encoder-based retrievers and backbone-wise results. ShiftRAG consistently outperforms encoder-based baselines by a large margin. Compared with RaLa [33], the strongest encoder-based retriever, ShiftRAG with Gemma2-9B improves HotpotQA R@2 by 20.29% (0.5485→0.6598), 2WikiMultihopQA R@10 by 7.46% (0.7007→0.7530), and MuSiQue R@2 by 18.58% (0.3530→0.4186). These results indicate that encoder-mode training with LLM backbones learns a stronger semantic embedding space than conventional encoder-based retrievers. In backbone-matched comparisons, every ShiftRAG variant outperforms the corresponding LLM2Vec and OneGen variants across all datasets and metrics. With LlaMA3.1-8B, ShiftRAG improves HotpotQA R@2 by 9.66% over LLM2Vec [2] (0.6096→0.6685) and MuSiQue R@2 by 25.45% over OneGen (0.3458→0.4338). With Gemma2-9B, ShiftRAG improves 2WikiMultihopQA R@10 by 10.27% over LLM2Vec (0.6829→0.7530) and 17.80% over OneGen (0.6392→0.7530). These results show that the retrieval gains are robust across backbone choices. Among ShiftRAG variants, LlaMA3.1-8B achieves the best R@2 on HotpotQA and MuSiQue, while Gemma2-9B achieves the best performance on all 2WikiMultihopQA metrics and the strongest R@5/R@10 on MuSiQue. Overall, these results show that separating retrieval-side discriminative learning from autoregressive answer generation and connecting them through SST-generated soft knowledge tokens improves end-to-end QA more effectively than retrieval-only or unified LLM baselines.

Table 11: Complete QA performance of ShiftRAG and baselines. The best and second-best performances are presented in bold and underlined, respectively. We conduct five runs with different random seeds and report the mean; the maximum standard deviation is ±0.05.

Methods	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
Encoder-based Retrieval						
Contriever	0.3660	0.4436	0.3310	0.3611	0.1990	0.2697
GTR (T5-base)	0.3810	0.4611	0.3450	0.3705	0.2070	0.2804
ColBERTv2	0.4030	0.4829	0.3650	0.3965	0.2190	0.2968
RaLa	0.4460	0.5400	0.4040	0.4397	0.2430	0.3284
LLM-based Retrieval						
GTE-Qwen2-7B	0.4790	0.5801	0.4340	0.4723	0.2320	0.3528
LLM2Vec [Mistral-7B]	0.4400	0.5220	0.3010	0.3598	0.1920	0.2916
LLM2Vec [LlaMA3.1-8B]	0.4240	0.5228	0.3260	0.4318	0.1800	0.2664
LLM2Vec [Qwen2.5-7B]	0.4400	0.5365	0.3430	0.3928	0.1770	0.2776
LLM2Vec [Gemma2-9B]	0.5030	0.6106	0.4400	0.4914	0.2430	0.3670
NV-Embed-v2	0.5140	0.6220	0.4580	0.5064	0.2480	0.3783
Unified LLM						
OneGen [Mistral-7B]	0.4200	0.5036	0.2820	0.3356	0.1770	0.2774
OneGen [LlaMA3.1-8B]	0.4120	0.5096	0.2850	0.3860	0.1660	0.2532
OneGen [Qwen2.5-7B]	0.4230	0.5082	0.2960	0.3478	0.1680	0.2596
OneGen [Gemma2-9B]	0.4670	0.5869	0.4220	0.4785	0.2200	0.3342
GritLM [Mistral-7B]	0.4430	0.5258	0.3020	0.3665	0.1870	0.2872
ShiftRAG [Mistral-7B]	0.4670	0.5882	0.3250	0.4007	0.2180	0.3146
ShiftRAG [LlaMA3.1-8B]	0.4630	0.5947	0.4200	0.5124	0.2360	0.3602
ShiftRAG [Qwen2.5-7B]	0.4840	0.6025	0.4210	0.4876	0.2050	0.3324
ShiftRAG [Gemma2-9B]	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**

C.2 Question Answering Performance

Table 11 shows the complete QA results, extending Table 2 with encoder-based retrievers and backbone-wise results. To ensure a controlled evaluation of generation capacity, all retrieval-only baselines are paired with Gemma2-9B, the strongest generator among the backbone LLMs. ShiftRAG with Gemma2-9B achieves the best overall QA performance across all datasets and metrics. Compared to RaLa [33], i.e., the strongest encoder-based retriever, ShiftRAG improves EM by 25.78% on HotpotQA (0.4460 → 0.5610), 25.74% on 2WikiMultihopQA (0.4040 → 0.5080), and 25.51% on MuSiQue (0.2430 → 0.3050). Compared with NV-Embed-v2 [35], i.e., the strongest LLM-based retrieval baseline, ShiftRAG improves EM by 9.14% (0.5140 → 0.5610), 10.92% (0.4580 → 0.5080), and 22.98% (0.2480 → 0.3050), respectively. The backbone-matched results show the same trend. Every variant of ShiftRAG outperforms the corresponding LLM2Vec [2] and OneGen [68] variants across all datasets and all metrics. For example, with LlaMA3.1-8B, ShiftRAG improves 2WikiMultihopQA EM by 28.83% over LLM2Vec (0.3260 → 0.4200) and 47.37% over OneGen (0.2850 → 0.4200). With Gemma2-9B, ShiftRAG improves HotpotQA EM by 11.53% over LLM2Vec (0.5030 → 0.5610) and 20.13% over OneGen (0.4670 → 0.5610). These results indicate that the QA gains are robust across varying backbone architectures. Among ShiftRAG variants, Gemma2-9B achieves the best QA performance on all datasets and all metrics. Collectively, our findings support the conclusion that decoupling retrieval-oriented discriminative learning from generative adaptation, and bridging them via SST-generated soft knowledge tokens, is more effective for end-to-end QA than either retrieval-only or unified LLM approaches.

C.3 Retrieval and Question Answering Performance on Single-hop QA Benchmark

Table 12: Retrieval and QA performance on NaturalQuestions. The best and second-best performances are presented in bold and underlined, respectively.

Methods	Retrieval			QA	
	R@2	R@5	R@10	EM	F1
RaLa	0.5923	0.8180	0.9145	0.3010	0.5036
LLM2Vec	0.6006	0.8217	0.9260	0.3150	0.5108
NV-Embed-v2	0.6078	0.8144	0.9318	0.3220	0.5207
OneGen	0.5648	0.7768	0.8746	0.2830	0.4983
GritLM	0.5776	0.7789	0.9090	0.2900	0.5062
ShiftRAG	**0.6290**	**0.8660**	**0.9780**	**0.3350**	**0.5256**

Table 12 shows retrieval and QA performance on NaturalQuestions [34], which evaluates whether ShiftRAG generalizes beyond multi-hop QA. ShiftRAG achieves the best performance across all retrieval and QA metrics. Compared with the strongest baseline for each metric, ShiftRAG improves Recall@2 by 3.49% over NV-Embed-v2 (0.6078 $\rightarrow$ 0.6290), Recall@5 by 5.39% over LLM2Vec [2] (0.8217 $\rightarrow$ 0.8660), and Recall@10 by 4.96% over NV-Embed-v2 [35] (0.9318 $\rightarrow$ 0.9780). The same trend holds for QA, where ShiftRAG improves EM by 4.04% (0.3220 $\rightarrow$ 0.3350) and F1 by 0.94% (0.5207 $\rightarrow$ 0.5256) over NV-Embed-v2. These results indicate that the proposed training methods for the encoder and decoder modes are not limited to multi-hop reasoning, but remain effective in the single-hop setting.

C.4 Question Answering Performance on SimpleQA

Table 13: QA performance on the recent factual QA benchmark SimpleQA. The best and second-best performances are presented in bold and underlined, respectively.

Methods	**SimpleQA**	
	EM	F1
RaLa	0.5450	0.6371
OneGen	0.4970	0.5826
GritLM	<u>0.5860</u>	<u>0.7182</u>
ShiftRAG	**0.6740**	**0.8005**

Table 13 shows QA performance on SimpleQA [62], which is excluded from training and used only at inference time. ShiftRAG achieves the best performance, reaching 0.6740 EM and 0.8005 F1-score. Compared with GritLM [45], i.e,g, the strongest baseline, ShiftRAG improves EM by 15.02% (0.5860 $\rightarrow$ 0.6740) and F1-score by 11.46% (0.7182 $\rightarrow$ 0.8005). These results indicate that ShiftRAG generalizes well to recent factual QA without task-specific training.

C.5 Embedding and Generative Generalization

Table 14: Embedding and generative generalization under the common Mistral-7B backbone. Left: performance of Clustering, STS, and Summarization on the MTEB benchmark. Right: generative performance on MMLU, GSM8K, and TyDiQA. The best and second-best performances are presented in bold and underlined, respectively.

MTEB Performance			
Task $\rightarrow$	Clust.	STS	Summ.
Metric $\rightarrow$	V-Meas.	Spear.	Spear.
# Dataset $\rightarrow$	11	10	1
LLM2Vec	0.4410	<u>0.8370</u>	0.3105
GritLM	<u>0.5082</u>	0.8303	<u>0.3565</u>
ShiftRAG	**0.5110**	**0.8414**	**0.3759**

Generative Performance			
Dataset $\rightarrow$	MMLU	GSM8K	TyDiQA
Metric $\rightarrow$	EM	EM	F1
Setup $\rightarrow$	0 FS	8FS, CoT	0FS, GP
Mistral-7B	0.6018	0.4450	<u>0.5590</u>
GritLM	<u>0.5768</u>	**0.5762**	0.5545
ShiftRAG	**0.6195**	0.5536	**0.5681**

Table 14 reports the embedding and generative generalization performance of ShiftRAG, utilizing the Mistral-7B backbone to ensure a fair comparison with GritLM [45]. To assess general embedding quality beyond retrieval, we evaluate ShiftRAG on MTEB [44] task groups that do not directly measure the retrieval performance: Clustering, STS, and Summarization. ShiftRAG achieves state-of-the-art results across all three categories, improving over the strongest competitor by 0.55% on Clustering (0.5082 $\rightarrow$ 0.5110), 0.53% on STS (0.8370 $\rightarrow$ 0.8414), and 5.44% on Summarization (0.3565 $\rightarrow$ 0.3759). These results indicate that our encoder mode preserves and enhances semantic representation quality across diverse text embedding tasks.

For generative generalization, we evaluate MMLU [21], GSM8K [9], and TyDiQA [8] under the same protocol as GritLM. ShiftRAG outperforms the strongest baselines on MMLU and TyDiQA, with gains of 2.94% (0.6018 $\rightarrow$ 0.6195) and 1.63% (0.5590 $\rightarrow$ 0.5681), respectively. While ShiftRAG trails GritLM on GSM8K, it improves the base Mistral-7B performance by 24.40% (0.4450 $\rightarrow$ 0.5536), despite the absence of math-intensive training data. To adapt SST for these diverse tasks, we use the following inputs: (a) for MMLU, we input candidate-answer embeddings; (b) for GSM8K, we split each question into two segments and use their embeddings; (c) for TyDiQA, we use question–paragraph embeddings. Overall, ShiftRAG successfully retains its generative intelligence while simultaneously excelling as a high-performance retriever under the same backbone.

Table 15: QA performance comparison with retrieval-adapted Latent-CoT baselines. The best and second-best performances are presented in bold and underlined, respectively.

Methods	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
Coconut	0.5160	0.6185	0.4120	0.4404	0.1895	0.2290
CODI	0.5270	0.6254	0.4210	0.4484	0.1970	0.2413
Sim-CoT	<u>0.5360</u>	<u>0.6416</u>	<u>0.4420</u>	<u>0.5146</u>	<u>0.2070</u>	<u>0.2659</u>
ShiftRAG	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**

C.6 Performance Comparison with Latent-CoT Baselines

Table 15 compares ShiftRAG with Latent-CoT baselines adapted to the retrieval-based QA setting. Since Coconut [18], CODI [55], and Sim-CoT [64] are originally designed to use only the question as input, we provide them with the same retrieved documents used by ShiftRAG during both training and inference. ShiftRAG achieves the best performance across all datasets and all metrics. Compared with Sim-CoT, i.e., the strongest Latent-CoT baseline, ShiftRAG improves EM by 14.93% on 2WikiMultihopQA ($0.4420 \rightarrow 0.5080$) and 47.34% on MuSiQue ($0.2070 \rightarrow 0.3050$). Since all methods use identical retrieved documents, the gains mainly come from how the retrieved knowledge is consumed during generation. These results indicate that soft knowledge tokens generated by SST provide a more effective interface for transferring retrieval-side discriminative knowledge than directly adapting Latent-CoT methods to retrieved text.

C.7 Ablation Study: Semantic Signal Translator

Table 16: Ablation study on SST components. The best and second-best performances are presented in bold and underlined, respectively.

Category	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
Ours	**0.5610**	**0.6790**	**0.5080**	**0.5506**	**0.2960**	**0.4067**
w/o relevance weight	<u>0.5570</u>	<u>0.6765</u>	<u>0.5060</u>	<u>0.5480</u>	<u>0.2910</u>	<u>0.4025</u>
w/o low-rank projection layer	0.5520	0.6754	0.4970	0.5421	0.2880	0.3976
w/o anchor tokens	0.5400	0.6533	0.4890	0.5306	0.2800	0.3705
w/o SST	0.5230	0.6468	0.4630	0.5194	0.2450	0.3639

Table 16 shows the effect of each SST component: *w/o* relevance weight fixes the retrieval-score-based relevance weight α_k to 1, *w/o* low-rank projection layer removes the low-rank projection layer and directly uses the output of the encoding layer as the offset embeddings, *w/o* anchor tokens removes the anchor token embeddings, and *w/o* SST removes the SST-generated soft knowledge tokens. Removing SST gives the largest degradation, reducing MuSiQue EM by 17.23% ($0.2960 \rightarrow 0.2450$). Within SST, anchor tokens have the largest effect, followed by the low-rank projection layer and relevance weighting: removing anchor tokens reduces MuSiQue EM by 5.41% ($0.2960 \rightarrow 0.2800$), removing the low-rank projection layer reduces it by 2.70% ($0.2960 \rightarrow 0.2880$), whereas removing the relevance weight reduces it by 1.69% ($0.2960 \rightarrow 0.2910$). These results indicate that anchor tokens provide a stable token-space basis for translating retrieval-oriented embeddings into generation-friendly soft knowledge tokens, the low-rank projection layer supports compact document-conditioned signal translation, and relevance weighting helps reflect the relative importance of retrieved documents during reasoning.

C.8 Performance Comparison with Simple Projection Baselines

Table 17: QA performance comparison between SST and simple embedding-to-soft-token projection baselines. All methods use the same question–document representations and number of soft knowledge tokens, with other training and inference settings fixed.

Methods	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
1-layer MLP	0.4505	0.5704	0.4008	0.4572	0.2376	0.3339
2-layer MLP	<u>0.4651</u>	<u>0.5894</u>	<u>0.4125</u>	<u>0.4595</u>	<u>0.2446</u>	<u>0.3355</u>
ShiftRAG	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**

We investigate whether the benefits of SST can be achieved by a simpler embedding-to-soft-token projection. Table 17 compares SST with 1- and 2-layer MLPs, which directly project the same question–document rep-

resentations into the same number of soft knowledge tokens. We replace only the translation module while keeping the remaining training and inference settings fixed. Unlike the component ablations in Appendix C.7, this comparison evaluates the SST design against alternative projection architectures.

ShiftRAG consistently achieves the best performance across all datasets and metrics. Although the 2-layer MLP outperforms the 1-layer MLP, both remain below SST. Compared with the 2-layer MLP, ShiftRAG improves EM by 20.62% on HotpotQA (0.4651 → 0.5610) and 24.69% on MuSiQue (0.2446 → 0.3050). These results indicate that directly projecting retrieval representations into soft tokens is less effective than the proposed SST design, supporting its role as an interface for translating retrieval-side discriminative signals into generation-friendly embeddings

C.9 Impact of Causal-Mask Relaxation

Table 18: Effect of causal-mask relaxation on retrieval performance. Uni. and Bi. denote unidirectional and bidirectional attention in encoder mode, respectively. The best and second-best performances are presented in bold and underlined, respectively.

Methods	**HotpotQA**			**2WikiMultihopQA**			**MuSiQue**		
	R@2	R@5	R@10	R@2	R@5	R@10	R@2	R@5	R@10
RaLa	0.5485	0.6800	0.7415	<u>0.5510</u>	<u>0.6563</u>	<u>0.7007</u>	0.3530	0.5047	0.5882
ShiftRAG (Uni.)	<u>0.5938</u>	<u>0.7475</u>	<u>0.8190</u>	0.5506	0.6440	0.6944	<u>0.3818</u>	<u>0.5127</u>	<u>0.5909</u>
ShiftRAG (Bi.)	**0.6598**	**0.8214**	**0.8806**	**0.6080**	**0.7109**	**0.7530**	**0.4186**	**0.5634**	**0.6516**

We analyze the effect of causal-mask relaxation on encoder-mode retrieval performance. Table 18 compares two ShiftRAG variants that differ only in their encoder-side attention policy: unidirectional attention retains the original causal mask, whereas bidirectional attention allows each token to attend to the entire input sequence. We also include RaLa as a dedicated bidirectional encoder baseline. The comparison between the two ShiftRAG variants isolates the contribution of attention policy within our encoder-mode training, complementing the qualitative analysis of their embedding distributions.

Bidirectional attention consistently improves retrieval performance across all datasets and metrics. Compared with unidirectional attention, it improves Recall@2 by 11.11% on HotpotQA (0.5938 → 0.6598) and 10.43% on 2WikiMultihopQA (0.5506 → 0.6080). ShiftRAG with bidirectional attention also outperforms RaLa across all reported metrics, improving HotpotQA Recall@5 by 20.79% (0.6800 → 0.8214) and MuSiQue Recall@2 by 18.58% (0.3530 → 0.4186). These results indicate that causal-mask relaxation contributes substantially to adapting the decoder-only backbone for evidence retrieval, supporting the use of bidirectional attention in encoder mode.

D Latency

D.1 Training Latency

Table 19: Training latency and data size comparison between ShiftRAG and baselines. Time denotes end-to-end training time measured.

Methods	**Time**	**Data Size**
LLM2Vec	5h 30m	96,000
OneGen	14h	174,500
GritLM	48h	2,886,912
ShiftRAG	10h 10m	106,000

We analyze training cost to examine whether separated dual-mode training introduces substantial overhead. Table 19 shows the measured end-to-end training time and the size of training data for ShiftRAG and baselines. LLM2Vec [2], OneGen [68], and ShiftRAG are trained with LoRA, whereas GritLM [45] requires full fine-tuning. ShiftRAG uses 96,000 retrieval examples and 10,000 generation examples, resulting in 106,000 training examples in total. Although ShiftRAG trains both encoder mode and decoder mode, it requires 10h 10m, which is shorter than OneGen (14h) and much shorter than GritLM (48h). Compared with LLM2Vec, ShiftRAG requires additional training time because it learns decoder mode as well, but yields stronger retrieval and QA performance. These results indicate that separate LoRA-based training provides a favorable trade-off between training efficiency and performance.

D.2 Complexity Comparison with RaLa

We analyze the computational complexity and training memory of RaLa-style contrasting compared with our gap-aware relevance scoring. Following the notation in Section 3.1, let $|q|$ and $|d|$ denote the sequence lengths of a question and a document, respectively, h denote the hidden dimension size, and $|\mathcal{C}|$ denote the number of candidate layers used for layer contrast. In the unoptimized RaLa formulation, layer contrast is computed over token-level MaxSim across all question–document token pairs and layers from each bucket. Therefore, its scoring complexity is $O(|\mathcal{C}| \cdot |q| \cdot |d| \cdot h)$. The optimized RaLa formulation reduces the layer-contrast computation to a `[CLS]`-based gap-aware weight, but its main matching score still relies on token-level average MaxSim. Thus, its asymptotic complexity becomes $O(|q| \cdot |d| \cdot h)$, which is asymptotically identical to ColBERT-style late interaction. In contrast, our gap-aware relevance scoring computes layer-wise semantic alignment gaps over mean-pooled question and document representations. Once the pooled representations are obtained, the scoring complexity is $O(|\mathcal{C}| \cdot h)$, and even when mean pooling is counted, the cost grows only linearly with the sequence length, rather than bilinearly with $|q||d|$. Thus, our method removes the token-token interaction term required by average MaxSim while using the layer-wise semantic alignment gap itself as the contrastive similarity score in the LGC loss.

This difference is particularly important for LLM-based retrieval encoders. Since LLM hidden states have much larger hidden dimensions than conventional encoder backbones, applying average MaxSim to LLM hidden states substantially increases peak memory usage. In our preliminary training with a Gemma2-2B backbone, average MaxSim either caused out-of-memory errors or required reducing the per-device batch size from 8 to 2. To maintain the same global batch size, the gradient accumulation step had to be increased by $4\times$, resulting in roughly $16\times$ longer training time in our setup. Our mean-pooled gap-aware relevance scoring avoids this token-level MaxSim computation, allows training with the original per-device batch size, and more directly reflects layer-wise semantic alignment in the retrieval objective. We further speculate that token-level average MaxSim may be less suitable for decoder-only LLM encoders whose late-layer token representations already exhibit rank-collapse tendencies, because MaxSim assumes sufficiently discriminative token-level representations.

D.3 Effectiveness-Efficiency Trade-off of Different Relevance Scoring Methods

Table 20: Effectiveness-efficiency trade-off between cosine similarity and gap-aware relevance scoring. The best retrieval performances are presented in bold.

Scoring Methods	HotpotQA				2WikiMultihopQA				MuSiQue			
	R@2	R@5	R@10	Time (s)	R@2	R@5	R@10	Time (s)	R@2	R@5	R@10	Time (s)
cosine	0.6598	0.8214	0.8806	0.93	0.6080	0.7109	0.7530	0.96	0.4186	**0.5634**	0.6516	0.87
gap-aware	**0.6621**	**0.8275**	**0.8869**	3.17	**0.6099**	**0.7126**	**0.7571**	3.73	**0.4188**	0.5632	**0.6527**	3.12

We evaluate the trade-off between retrieval effectiveness and inference efficiency across different relevance scoring methods. Table 20 compares standard final-layer cosine similarity, i.e., $s_{q,d} = \cos(\mathbf{e}_q, \mathbf{e}_d^{(L)})$, against the gap-aware relevance score defined in Equation 2. The gap-aware metric yields marginal retrieval gains, improving HotpotQA R@10 by 0.72% (0.8806 $\rightarrow$ 0.8869) and 2WikiMultihopQA R@10 by 0.54% (0.7530 $\rightarrow$ 0.7571). However, it incurs a substantial 3.63× increase ($0.92s \rightarrow 3.34s$) in average latency. Given that the performance gains are disproportionate to the added computational cost, we prioritize inference efficiency and adopt final-layer cosine similarity during inference.

E Analysis

E.1 Scalability Analysis

$$\text{Relative Gain } (\%) \;=\; \left(\text{R@10}^{\text{ShiftRAG}}_{\text{candidate size}} - \text{R@10}^{\text{baseline}}_{\text{candidate size}}\right) \times 100 \tag{14}$$

We evaluate the scalability of ShiftRAG by increasing the candidate pool size from 10,000 to 60,000. Table 21 shows R@10 under each candidate pool size. Retrieval performance decreases for all methods as the search space becomes larger, but ShiftRAG consistently achieves the best performance at every size. The performance gap remains large under larger candidate pools. Compared with RaLa, ShiftRAG improves Recall@10 by 57.24% at 10,000 candidates (0.4789 $\rightarrow$ 0.7530) and 74.78% at 60,000 candidates (0.3235 $\rightarrow$ 0.5654). The gain over GritLM increases from 27.84% to 44.94%, and the gain over NV-Embed-v2 increases from 19.18% to 25.81%. These results indicate that our encoder mode learns robust discriminative embeddings and preserves stronger retrieval quality as the number of candidate documents increases.

Table 21: Scalability analysis on R@10 under increasing candidate pool sizes. Relative gain is computed against each baseline at the same candidate size. The best and second-best performances are presented in bold and underlined, respectively.

Methods	**Candidate Pool Size**					
	10,000	20,000	30,000	40,000	50,000	60,000
RaLa	0.4789	0.4574	0.4207	0.3781	0.3351	0.3235
GritLM	0.5890	0.5708	0.5307	0.4824	0.4341	0.3901
NV-Embed-v2	<u>0.6318</u>	<u>0.6165</u>	<u>0.5787</u>	<u>0.5339</u>	<u>0.4904</u>	<u>0.4494</u>
ShiftRAG	**0.7530**	**0.7410**	**0.6977**	**0.6516**	**0.6085**	**0.5654**
Relative Gain over RaLa (%)	57.24	62.00	65.84	72.34	81.59	74.78
Relative Gain over GritLM (%)	27.84	29.82	31.47	35.07	40.18	44.94
Relative Gain over NV-Embed-v2 (%)	19.18	20.19	20.56	22.05	24.08	25.81

Table 22: Impact of anchor token initialization on QA performance. The best performances are presented in bold.

Category	**HotpotQA**		**2WikiMultihopQA**		**MuSiQue**	
	EM	F1	EM	F1	EM	F1
Random initialization	0.5530	0.6635	0.4960	0.5412	0.2990	0.4020
Anchor token initialization	**0.5610**	**0.6790**	**0.5080**	**0.5528**	**0.3050**	**0.4129**

E.2 Impact of Anchor Token Initialization

We analyze the effect of anchor token initialization in SST. Table 22 compares random initialization with the anchor token initialization procedure described in Appendix B.2. Anchor token initialization consistently improves QA performance, increasing EM by 1.45% on HotpotQA (0.5530→0.5610), 2.42% on 2WikiMultihopQA (0.4960→0.5080), and 2.01% on MuSiQue (0.2990→0.3050). The model remains competitive even with random initialization, suggesting that anchor token initialization mainly serves as an auxiliary mechanism for improving optimization stability. These results indicate that pre-aligning anchor tokens with the generator's token space helps SST produce more stable soft knowledge tokens.

E.3 Impact of SST on Confidence Calibration and LoRA Adaptation

Table 23: QA performance and confidence statistics under different uses of SST. "Generate *w/o* SST" removes SST only at inference after SST-based training, whereas "Train *w/o* SST" trains the generator without SST. The best and second-best performances are presented in bold and underlined, respectively.

Methods	**HotpotQA**	**2WikiMultihopQA**	**MuSiQue**	**Correct**		**Incorrect**	
				Entropy	Top-1 Prob.	Entropy	Top-1 Prob.
ShiftRAG	**0.5610**	**0.5080**	**0.3050**	0.4187	0.8343	1.3430	0.5091
Generate *w/o* SST	<u>0.5490</u>	<u>0.4980</u>	<u>0.2890</u>	0.3869	0.8804	1.0063	0.6762
Train *w/o* SST	0.5230	0.4630	0.2450	0.2204	0.8831	0.4102	0.8455
Vanilla Backbone	0.5020	0.4400	0.1970	0.1492	0.9186	0.1899	0.8957

We analyze the impact of SST on QA performance and generator confidence. Motivated by the overconfidence observation in Figure 1, Table 23 compares four settings: Vanilla Backbone, Train *w/o* SST, Generate *w/o* SST, and ShiftRAG. QA performance consistently improves as SST is more fully incorporated, with ShiftRAG achieving the highest EM on all datasets. At the same time, SST makes the generator less overconfident on incorrect answers rather than uniformly uncertain. In particular, the top-1 probability on incorrect answers decreases from 0.8957 in Vanilla Backbone to 0.5091 in ShiftRAG, while the entropy increases from 0.1899 to 1.3430. This produces a clearer confidence separation between correct and incorrect generations: the entropy gap increases from 0.0407 in Vanilla Backbone and 0.1898 in Train *w/o* SST to 0.6194 in Generate *w/o* SST and 0.9243 in ShiftRAG. These results indicate that SST mitigates premature commitment to incorrect answers and encourages correctness-aware decoding. Moreover, Generate *w/o* SST still outperforms Train *w/o* SST on all datasets, even though neither setting uses SST at inference, suggesting that SST not only provides auxiliary soft knowledge tokens at test time but also reshapes the generator-side LoRA adaptation during training.

Table 24: QA performance with fixed external retrievers. Both inference settings use decoder-mode LoRA trained with SST and KD, while w/ SST additionally provides soft knowledge tokens at inference. The best performance within each retriever is presented in bold.

Retriever	Inference	HotpotQA		2WikiMultihopQA		MuSiQue	
		EM	F1	EM	F1	EM	F1
LLM2Vec	w/o SST	0.5030	0.6106	0.4400	0.4914	0.2430	0.3670
LLM2Vec	w/ SST	**0.5220**	**0.6281**	**0.4630**	**0.5202**	**0.2610**	**0.3902**
NV-Embed-v2	w/o SST	0.5140	0.6220	0.4580	0.5064	0.2480	0.3783
NV-Embed-v2	w/ SST	**0.5300**	**0.6392**	**0.4690**	**0.5251**	**0.2630**	**0.3913**

E.4 Impact of SST with Fixed External Retrievers

We investigate whether the effectiveness of SST is restricted to the representations learned by our encoder. Table 24 evaluates SST with two external retrievers, LLM2Vec and NV-Embed-v2. For each retriever, we compare generation with and without SST-generated soft knowledge tokens while keeping retrieval fixed. Both settings use decoder-mode LoRA trained with SST and KD; therefore, w/o SST removes the soft knowledge tokens only at inference, rather than excluding SST from training. This comparison examines whether retrieval-side signals remain useful when they originate from encoders other than the proposed encoder.

SST consistently improves EM and F1 across all three datasets with both external retrievers. On MuSiQue, SST improves EM by 7.41% with LLM2Vec (0.2430 $\rightarrow$ 0.2610) and 6.05% with NV-Embed-v2 (0.2480 $\rightarrow$ 0.2630). Since the retriever remains fixed within each comparison, these gains reflect the benefit of supplying retrieval-side latent signals during generation rather than improvements in document retrieval. These results indicate that SST can complement raw textual evidence using representations from different retrievers, extending its applicability beyond the embedding space learned by ShiftRAG.

E.5 Theoretical Interpretation of Learning Strategy

Table 25: Impact of learning strategies and parameter sharing on retrieval, QA, and training time. The best and second-best performances are presented in bold and underlined, respectively. The Parameter column indicates whether retrieval and generation adaptation parameters are shared or separate.

Training Methods	Parameter	HotpotQA				2WikiMultihopQA				Time (h)
		Retrieval		Generation		Retrieval		Generation		
		R@2	R@5	EM	F1	R@2	R@5	EM	F1	
Joint Learning	shared	0.4896	0.6298	0.1300	0.3430	0.3230	0.4666	0.1270	0.3432	21 h
	separate	0.5264	0.6865	0.4730	0.5885	0.5159	0.6184	0.4360	0.5115	23 h
Alternating Learning	shared	0.5166	0.6634	0.2200	0.4036	0.4923	0.5012	0.1860	0.4172	20 h
	separate	<u>0.5938</u>	<u>0.7665</u>	<u>0.5220</u>	<u>0.6451</u>	<u>0.5654</u>	<u>0.6682</u>	<u>0.4780</u>	<u>0.5409</u>	**10 h**
Separate Learning	separate	**0.6598**	**0.8214**	**0.5610**	**0.6790**	**0.6080**	**0.7109**	**0.5080**	**0.5528**	**10 h**

We analyze why Separate Learning provides a more stable optimization strategy than Joint Learning or Alternating Learning. Table 25 compares these strategies under shared and separate LoRA parameter settings. Joint Learning with shared parameters is the most unstable configuration, causing severe QA collapse with EM dropping to 0.1300 on HotpotQA and 0.1270 on 2WikiMultihopQA. Separating LoRA parameters mitigates this collapse, and Alternating Learning [3, 26, 70] further reduces simultaneous objective interference by updating only one mode at each step. However, both remain below Separate Learning. In particular, Separate Learning achieves the best retrieval and QA performance with the same 10 h average training time as Alternating Learning with separate parameters. These results suggest that stable dual-mode training requires two conditions: encoder and decoder modes should not compete over the same adaptation parameters, and decoder-mode training should not depend on a moving retrieval-conditioned latent interface.

We first formalize the effect of parameter sharing. Let $\mathbf{w}$ denote the vectorized shared LoRA parameters, and let $\mathcal{L}_{\mathrm{enc}}$ and $\mathcal{L}_{\mathrm{dec}}$ denote the encoder-mode and decoder-mode objectives, respectively. At step t, Joint Learning updates the shared parameters as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\big(\mathbf{g}_t^{\mathrm{enc}} + \lambda_J \mathbf{g}_t^{\mathrm{dec}}\big), \tag{15}$$

where

$$\mathbf{g}_t^{\mathrm{enc}} = \nabla_{\mathbf{w}} \mathcal{L}_{\mathrm{enc}}(\mathbf{w}_t), \qquad \mathbf{g}_t^{\mathrm{dec}} = \nabla_{\mathbf{w}} \mathcal{L}_{\mathrm{dec}}(\mathbf{w}_t). \tag{16}$$

Here, $\lambda_J > 0$ controls the relative contribution of the decoder-mode objective.

Lemma E.1 (Gradient conflict under smoothness)**.** *Assume that $\mathcal{L}_{\text{enc}}$ is β_{enc}-smooth with respect to $\mathbf{w}$. For the joint update in Equation 15, the one-step change of the encoder-mode objective is bounded by*

$$\mathcal{L}_{\text{enc}}(\mathbf{w}_{t+1}) \leq \mathcal{L}_{\text{enc}}(\mathbf{w}_t) - \eta\|\mathbf{g}_t^{\text{enc}}\|^2 - \eta\lambda_J\langle\mathbf{g}_t^{\text{enc}}, \mathbf{g}_t^{\text{dec}}\rangle + \frac{\beta_{\text{enc}}\eta^2}{2}\|\mathbf{g}_t^{\text{enc}} + \lambda_J\mathbf{g}_t^{\text{dec}}\|^2. \tag{17}$$

Proof. By β_{enc}-smoothness [15, 19],

$$\mathcal{L}_{\text{enc}}(\mathbf{w}_{t+1}) \leq \mathcal{L}_{\text{enc}}(\mathbf{w}_t) + \langle\nabla_{\mathbf{w}}\mathcal{L}_{\text{enc}}(\mathbf{w}_t), \mathbf{w}_{t+1} - \mathbf{w}_t\rangle + \frac{\beta_{\text{enc}}}{2}\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2. \tag{18}$$

Substituting $\nabla_{\mathbf{w}}\mathcal{L}_{\text{enc}}(\mathbf{w}_t) = \mathbf{g}_t^{\text{enc}}$ and $\mathbf{w}_{t+1} - \mathbf{w}_t = -\eta(\mathbf{g}_t^{\text{enc}} + \lambda_J\mathbf{g}_t^{\text{dec}})$ gives Equation 17.

Lemma E.1 does not assume that encoder-mode and decoder-mode gradients always conflict. Rather, it shows that conflict becomes harmful when the two gradients are negatively aligned. Let

$$\cos\theta_t = \frac{\langle\mathbf{g}_t^{\text{enc}}, \mathbf{g}_t^{\text{dec}}\rangle}{\|\mathbf{g}_t^{\text{enc}}\|\|\mathbf{g}_t^{\text{dec}}\|}. \tag{19}$$

Then the first-order term in Equation 17 can be written as

$$-\eta\|\mathbf{g}_t^{\text{enc}}\|\big(\|\mathbf{g}_t^{\text{enc}}\| + \lambda_J\|\mathbf{g}_t^{\text{dec}}\|\cos\theta_t\big). \tag{20}$$

When $\cos\theta_t < 0$, the decoder-mode gradient weakens the descent direction for the encoder-mode objective. If $\lambda_J\|\mathbf{g}_t^{\text{dec}}\||\cos\theta_t|$ becomes comparable to $\|\mathbf{g}_t^{\text{enc}}\|$, the first-order descent term can be largely canceled, while the smoothness term remains non-negative. The same argument applies symmetrically to $\mathcal{L}_{\text{dec}}$. Thus, shared-parameter Joint Learning can fail when discriminative retrieval adaptation and autoregressive generation adaptation push the same LoRA parameters in different directions.

We empirically verify this conflict by measuring the cosine similarity between $\mathbf{g}_t^{\text{enc}}$ and $\mathbf{g}_t^{\text{dec}}$ on the shared LoRA parameters. We summarize the conflict frequency using the negative-gradient ratio

$$\rho_{\text{neg}} = \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[\cos\theta_t < 0]. \tag{21}$$

Over the first 500 updates, the cosine similarity is negative in 84.2% of steps on average across three runs. This explains why Joint Learning with shared parameters shows the largest degradation in Table 25.

Parameter separation removes this direct gradient conflict, but it does not fully stabilize decoder-mode training. Even with separate LoRA_{enc} and LoRA_{dec}, the decoder-mode objective depends on SST-generated soft knowledge tokens, which are produced from encoder-side representations. Let

$$\mathbf{T}_t = \mathbf{T}_{1:K}(\mathbf{w}_t^{\text{enc}}) \tag{22}$$

denote the stacked soft knowledge tokens generated from the current encoder-mode parameters. For notational clarity, we fix the input examples and SST parameters, and isolate only the drift induced by encoder-mode updates. The decoder-mode objective can then be written as $\mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_t)$. Hence, if encoder mode keeps changing during joint or alternating training, the generator is optimized against a latent interface whose distribution also changes.

Lemma E.2 (Encoder-side updates induce decoder-objective drift)**.** *Assume that $\mathcal{L}_{\text{dec}}(\mathbf{w}^{\text{dec}}; \mathbf{T})$ is L_T-Lipschitz in $\mathbf{T}$, and that $\mathbf{T}_{1:K}(\mathbf{w}^{\text{enc}})$ is L_w-Lipschitz in $\mathbf{w}^{\text{enc}}$. Then the decoder-objective drift caused by an encoder-side update satisfies*

$$|\mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_{t+1}) - \mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_t)| \leq L_T L_w\|\mathbf{w}_{t+1}^{\text{enc}} - \mathbf{w}_t^{\text{enc}}\|. \tag{23}$$

Proof. Since $\mathcal{L}_{\text{dec}}$ is L_T-Lipschitz in $\mathbf{T}$ [13, 54],

$$|\mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_{t+1}) - \mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_t)| \leq L_T\|\mathbf{T}_{t+1} - \mathbf{T}_t\|. \tag{24}$$

Since $\mathbf{T}_{1:K}(\mathbf{w}^{\text{enc}})$ is L_w-Lipschitz in $\mathbf{w}^{\text{enc}}$, we have

$$\|\mathbf{T}_{t+1} - \mathbf{T}_t\| \leq L_w\|\mathbf{w}_{t+1}^{\text{enc}} - \mathbf{w}_t^{\text{enc}}\|. \tag{25}$$

Combining Equations 24 and 25 gives Equation 23.

Lemma E.2 shows that parameter separation alone does not guarantee a stationary decoder-mode objective. If the encoder-mode parameters are updated as

$$\mathbf{w}_{t+1}^{\text{enc}} = \mathbf{w}_t^{\text{enc}} - \eta_{\text{enc}}\mathbf{g}_t^{\text{enc}}, \tag{26}$$

then Equation 23 becomes

$$|\mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_{t+1}) - \mathcal{L}_{\text{dec}}(\mathbf{w}_t^{\text{dec}}; \mathbf{T}_t)| \leq \eta_{\text{enc}}L_T L_w\|\mathbf{g}_t^{\text{enc}}\|. \tag{27}$$

Thus, larger encoder-mode updates can induce larger changes in the decoder-mode objective, even when the two modes use separate LoRA parameters. Over multiple encoder-mode updates, this drift accumulates as

$$|\mathcal{L}_{\mathrm{dec}}(\mathbf{w}_t^{\mathrm{dec}}; \mathbf{T}_{t+m}) - \mathcal{L}_{\mathrm{dec}}(\mathbf{w}_t^{\mathrm{dec}}; \mathbf{T}_t)| \leq L_T L_w \sum_{i=t}^{t+m-1} \|\mathbf{w}_{i+1}^{\mathrm{enc}} - \mathbf{w}_i^{\mathrm{enc}}\|. \tag{28}$$

This explains why Joint Learning with separate parameters remains suboptimal. Although direct gradient conflict is reduced, the generator is still trained under a non-stationary SST interface. Alternating Learning mitigates simultaneous objective interference by updating only one mode at a time, but it does not remove the drift because $\mathbf{T}_{1:K}$ can still change after encoder-mode updates. Separate Learning removes this source of instability by fixing $\mathbf{w}^{\mathrm{enc}}$ before decoder-mode training. In this case, encoder-induced changes satisfy $\mathbf{w}_{t+1}^{\mathrm{enc}} = \mathbf{w}_t^{\mathrm{enc}}$, so $\mathbf{T}_{t+1} = \mathbf{T}_t$, and the drift term in Lemma E.2 becomes zero.

This interpretation matches the empirical trend in Table 25. Joint Learning with shared parameters performs worst, consistent with the gradient conflict in Lemma E.1 and the measured negative-gradient ratio. Joint Learning with separate parameters improves over the shared-parameter variant, but remains below Alternating Learning and Separate Learning because the generator still follows a moving soft-token interface. Alternating Learning with separate parameters reduces simultaneous updates, but still allows encoder-side drift between decoder-mode updates. In contrast, Separate Learning first learns stable encoder-mode representations and then trains decoder mode with a fixed retrieval-conditioned interface. Compared with Alternating Learning with separate parameters, Separate Learning improves HotpotQA R@2 by 11.11% (0.5938 → 0.6598) and EM by 7.53% (0.5220 → 0.5610), with the same 10 h average training time. Overall, these results support our design choice of using separate adaptation paths and stage-wise training for encoder and decoder modes.

E.6 Overconfidence Analysis on Correct and Incorrect Cases

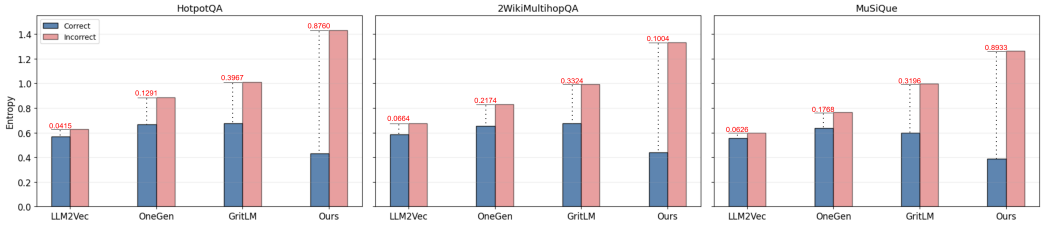

Figure 7: Average token entropy over the generated answer on MHQA benchmarks, grouped by answer correctness. Red values indicate the entropy gap between incorrect and correct cases.

As shown in Figure 7, for LLM2Vec[2], retrieval is performed using the LLM2Vec encoder, while answer generation is conducted by the backbone LLM. Across all three datasets, ShiftRAG exhibits the clearest entropy separation between correct and incorrect answers. On HotpotQA, the entropy gap is 0.8760 for ShiftRAG, compared with 0.3967 for GritLM[45], 0.1291 for OneGen[68], and 0.0415 for LLM2Vec. A similar trend is observed on 2WikiMultihopQA, where ShiftRAG again shows the largest entropy gap of 1.0004, substantially exceeding those of GritLM (0.3324), OneGen (0.2174), and LLM2Vec (0.0664). On MuSiQue, ShiftRAG also achieves the largest gap at 0.8933, compared with 0.3196 for GritLM, 0.1768 for OneGen, and 0.0626 for LLM2Vec. These results indicate that existing baselines remain relatively overconfident even when their answers are incorrect, as reflected by their limited entropy separation between correct and incorrect cases. In contrast, ShiftRAG assigns substantially higher entropy to incorrect decodings while maintaining lower entropy on correct ones, yielding a much clearer separation by answer correctness. This supports our claim that transferring retrieval-side knowledge through SST and soft knowledge tokens leads to more correctness-aware decoding behavior and better mitigates hallucination-prone overconfidence.

E.7 Effect of Soft Knowledge Token Placement

Table 26: Effect of different placements of soft knowledge tokens on QA performance and soft-token injection latency in ShiftRAG.

Category	HotpotQA			2WikiMultihopQA			MuSiQue		
	EM	F1	Time (ms)	EM	F1	Time (ms)	EM	F1	Time (ms)
Pairwise	**0.5630**	**0.6827**	1.65	0.5060	**0.5736**	1.75	0.2910	0.4032	1.35
Between	0.5580	0.6779	0.87	0.4980	0.5470	1.01	0.2880	0.4008	0.92
Behind	0.5610	0.6790	0.86	**0.5080**	0.5528	1.02	**0.3050**	**0.4129**	0.91

We compare three soft knowledge token placement strategies in generation. Pairwise inserts each set of question–document-specific soft knowledge tokens immediately after its corresponding document, Between places all soft knowledge tokens between the document texts and the question, and Behind places all soft knowledge tokens after the textual input, i.e., behind the documents and question. Time (ms) measures the additional latency required to arrange and inject the generated soft knowledge tokens into the input embeddings. As shown in Table 26, Pairwise is consistently the slowest because it requires document-wise iterative insertion, taking 1.65/1.75/1.35 ms on HotpotQA, 2WikiMultihopQA, and MuSiQue, respectively, which is about 1.70× slower on average than Behind. Although Pairwise achieves slightly better performance on HotpotQA and 2WikiMultihopQA in some metrics, Behind yields the best average EM and the strongest overall trade-off between accuracy and efficiency. In particular, Behind consistently outperforms Between on all datasets and metrics, and achieves the best MuSiQue performance with 0.3050 EM and 0.4129 F1-score. Therefore, we adopt Behind as the default soft knowledge token placement in all experiments.

E.8 t-SNE Visualization Before and After Training with Bidirectional and Unidirectional Attention

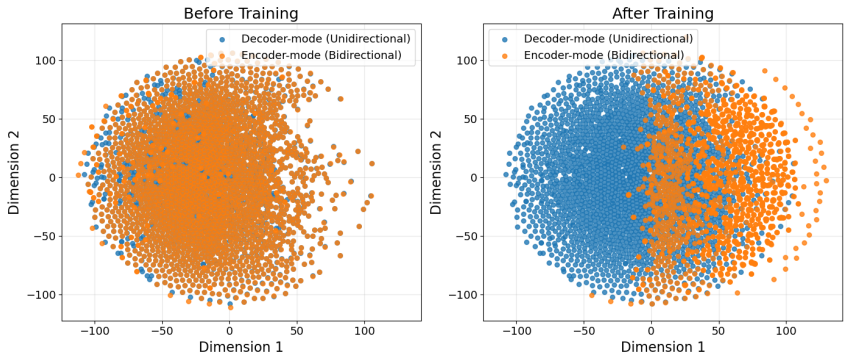

Figure 8: t-SNE visualization of document embeddings in the 2WikiMultihopQA dataset, comparing distributions before and after training under bi-directional and uni-directional attention mechanisms.

To leverage pre-trained Large Language Models (LLMs) as encoders for the retrieval tasks, the standard uni-directional attention mechanism of decoder-only models must be transitioned to bi-directional attention, followed by task-specific fine-tuning for document embeddings. As illustrated in Figure 8, the distributions of document embeddings derived from each attention mechanism after fine-tuning exhibit a distinct separation. This clear visualization indicates that the embedding space optimized for retrieval (bidirectional attention) is fundamentally different from the embedding space utilized for generation (unidirectional attention). These findings provide strong empirical justification for the necessity of separate training for encoder and decoder modes. Furthermore, experimental results on the HotpotQA and MuSiQue datasets show consistent trends, reinforcing the generalizability of this observation.

E.9 Effective Rank Before and After Encoder-mode Training across LLM Backbones

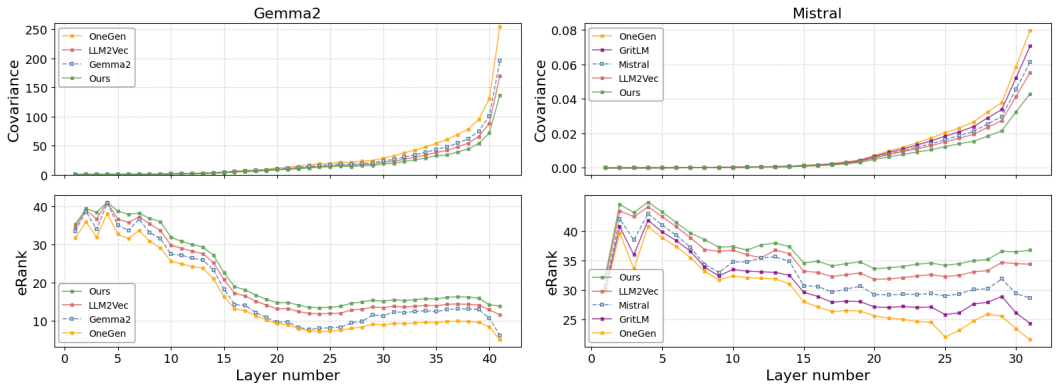

Figure 9: Layer-wise covariance between hidden representations of different tokens and effective rank (eRank) of token representations on Gemma2-9B (left) and Mistral-7B (right), respectively.

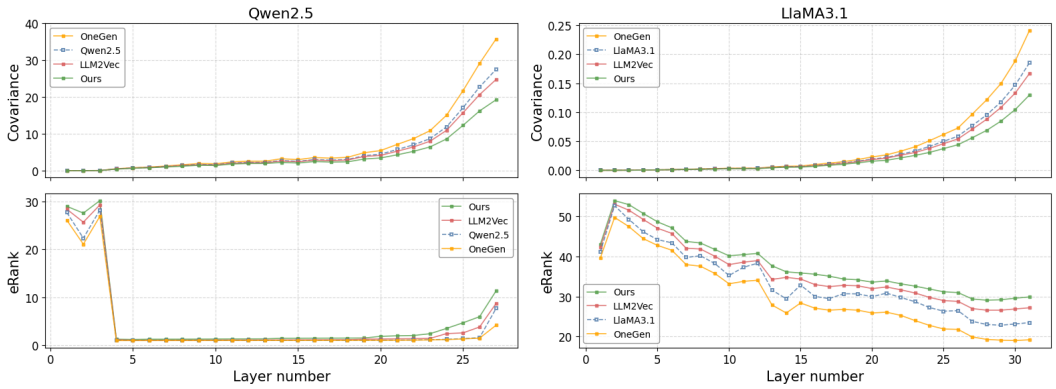

Figure 10: Layer-wise covariance between hidden representations of different tokens and effective rank (eRank) of token representations on Qwen2.5-7B (left) and LlaMA3.1-8B (right), respectively.

We further analyze the layer-wise rank collapse tendency using covariance between hidden representations of different tokens and eRank across four backbone families. Figures 9 and 10 show consistent patterns on 2WikiMultihopQA using Gemma2-9B [14], Mistral-7B [29], Qwen2.5-7B [29], and LlaMA3.1-8B [16]. As the layer depth increases, covariance generally increases, while eRank decreases or remains low in deeper layers. This indicates that token hidden representations become increasingly aligned with each other and are concentrated into fewer effective directions. The trend is particularly clear in late layers, supporting our claim that decoder-only LLM representations can suffer from rank collapse when used as retrieval encoders.

The same tendency is consistently observed in unified LLMs. Compared with backbone LLMs, unified LLMs such as OneGen [68] generally show higher covariance and lower eRank in deeper layers, indicating stronger directional alignment and reduced effective dimensionality. In Figure 9, GritLM [45] on Mistral-7B also follows the same tendency as OneGen, showing stronger late-layer collapse than the backbone model. Figure 10 further confirms that this behavior is not specific to Gemma2 or Mistral, but also appears on Qwen2.5 and LlaMA3.1. These results support our claim that unified LLMs tend to intensify rank collapse in deeper representations, which can make it difficult to preserve the token-level diversity needed for discriminative retrieval representations. We observe the same tendency on other MHQA benchmarks, as well as SimpleQA and NaturalQuestions.

E.10 SST Effectiveness on Modern Large LLMs

Table 27: QA performance of large LLM generators with a fixed NV-Embed-v2 retriever. For each generator, both settings use the same SST-trained checkpoint and differ only in whether SST-generated soft knowledge tokens are provided at inference. The best performance within each generator is presented in bold.

Generator	Inference	HotpotQA		2WikiMultihopQA		MuSiQue	
		EM	F1	EM	F1	EM	F1
Gemma4-31B	w/o SST	0.5920	0.7012	0.4960	0.5976	0.3120	0.4358
Gemma4-31B	w/ SST	**0.6140**	**0.7116**	**0.5280**	**0.6142**	**0.3410**	**0.4873**
Qwen3.6-35B	w/o SST	0.6050	0.7293	0.5070	0.5624	0.3290	0.4437
Qwen3.6-35B	w/ SST	**0.6310**	**0.7548**	**0.5330**	**0.5788**	**0.3660**	**0.5026**

We investigate whether SST remains effective with larger, more recent LLM generators. Table 27 compares Gemma4-31B and Qwen3.6-35B with and without SST-generated soft knowledge tokens at inference. We use NV-Embed-v2 as the fixed retriever and keep the retrieved documents, document order, relevance scores, generator checkpoint, and decoding configuration unchanged within each comparison. Both settings retain the decoder-mode LoRA learned with SST, while w/o SST removes the soft knowledge tokens only at inference. This controlled comparison isolates the inference-time contribution of retrieval-side latent signals beyond raw textual evidence.

SST consistently improves EM and F1 across all datasets for both generators. The gains are particularly pronounced on MuSiQue, where SST improves EM by 9.29% with Gemma4-31B (0.3120 $\rightarrow$ 0.3410) and 11.25% with Qwen3.6-35B (0.3290 $\rightarrow$ 0.3660). The corresponding F1 improvements are 11.82% (0.4358 $\rightarrow$ 0.4873) and 13.27% (0.4437 $\rightarrow$ 0.5026), respectively. These results indicate that the benefits of SST are not restricted to the smaller backbones used in our main experiments. Larger generators can reason over raw

retrieved text while still benefiting from explicitly transferred question–document alignment signals, suggesting that the latent interface provides complementary guidance for evidence-grounded generation.

F Hyperparameter Sensitivity

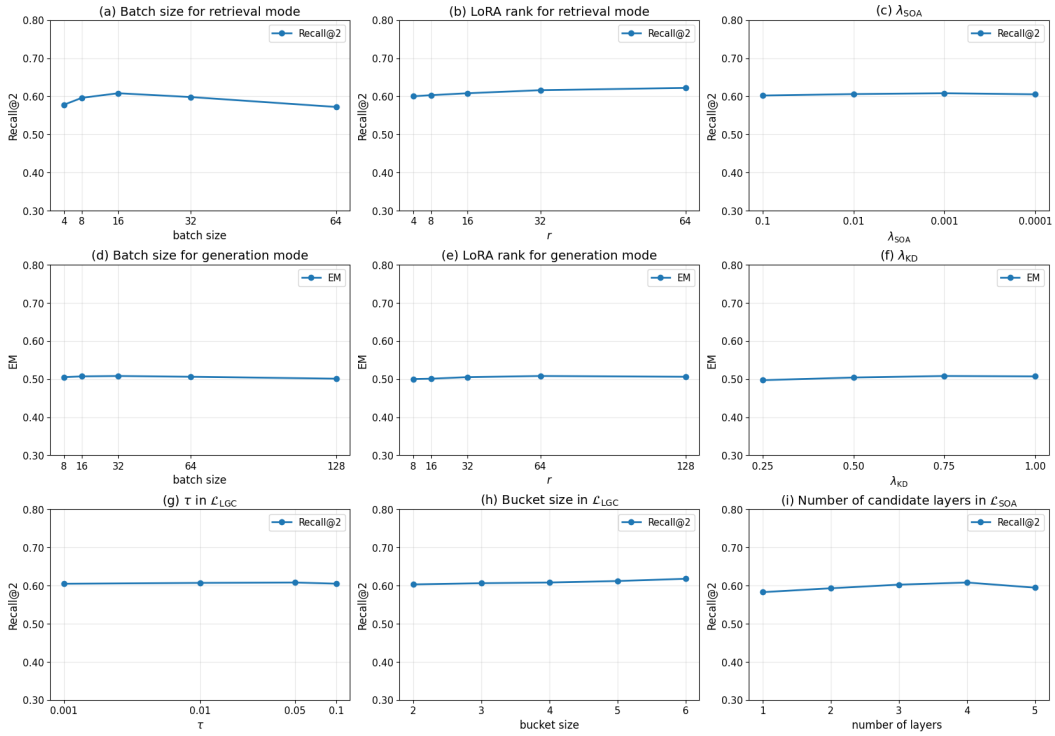

Figure 11: Hyperparameter sensitivity analysis on (a) batch size in encoder mode, (b) LoRA rank in encoder mode, (c) λ_{SOA} in encoder mode training, (d) batch size in decoder mode, (e) LoRA rank in decoder mode, (f) λ_{KD} in decoder mode training, (g) temperature τ in $\mathcal{L}_{\text{LGC}}$, (h) bucket size in $\mathcal{L}_{\text{LGC}}$. (i) number of candidate layers in $\mathcal{L}_{\text{SOA}}$.

F.1 Impact of Batch Size

We conduct experiments on encoder mode and decoder mode to evaluate the impact of batch size. Figure 11 (a) illustrates the R@2 metrics for batch sizes of 4, 8, 16, 32, and 64. The results show that a batch size of 16 yields the highest R@2, while performance drops sharply when the batch size exceeds 32. This trend can be attributed to the dynamics of training with in-batch negatives; smaller batch sizes increase the relative proportion of hard negatives within the batch, thereby facilitating the optimization of the layer-wise gap-aware contrastive (LGC) loss. However, when the batch size is reduced below 16, the absolute number of negative samples becomes insufficient, hindering the overall training process. In contrast to the retrieval performance, the decoder mode shows no significant variation in EM as the batch size increases, as illustrated in Figure 11 (d). This is because the objective functions for decoder mode do not depend on the number of negative samples, unlike the encoder mode objective. While the remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.2 Impact of LoRA Rank

We conduct experiments in both encoder and decoder modes to analyze the impact of the LoRA rank r. As illustrated in Figures 11 (b) and 11 (e), the model exhibits highly stable performance across various rank settings. Specifically, while R@2 and EM both peak at $r = 64$, the overall variance remains marginal. In encoder mode, R@2 shows only minor fluctuations as r increases from 4 to 64. Similarly, in decoder mode, EM remains relatively consistent across rank values, with only a negligible decline observed at $r = 128$. While remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.3 Impact of λ_{SOA}

We conduct experiments in encoder mode to analyze the effect of the hyper-parameter λ_{SOA}. Figure 11 (c) shows R@2 for λ_{SOA} values of 0.1, 0.01, 0.001, and 0.0001. The highest R@2 is achieved at λ_{SOA}= 0.001, although the performance remains remarkably stable across the tested range. While the remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.4 Impact of λ_{KD}

We conduct experiments in decoder mode to analyze the effect of the hyper-parameter λ_{KD}. Figure 11 (f) shows the EM for λ_{KD} values of 0.25, 0.5, 0.75, and 1. The highest EM is achieved at λ_{KD}= 0.75, although the performance exhibits a gradual increase until λ_{KD} reaches 0.75. While the remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.5 Impact of Temperature τ in $\mathcal{L}_{\text{LGC}}$

We conduct experiments in encoder mode to analyze the effect of the temperature hyper-parameter τ in the Layer-wise Gap-aware Contrastive (LGC) loss. Figure 11 (g) shows the R@2 scores for τ values of 0.001, 0.01, 0.05, and 0.1. The highest R@2 is attained at τ = 0.05, with the model demonstrating minimal sensitivity to variations in τ within the tested range. While the remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.6 Impact of Bucket Size in $\mathcal{L}_{\text{LGC}}$

We conduct experiments in encoder mode to analyze the effect of the bucket size in $\mathcal{L}_{\text{LGC}}$. Figure 11 (h) shows the R@2 scores for the bucket size ranging from 2 to 6. The highest R@2 is attained using 6 layers as a bucket, although the performance increases only marginally as the bucket size increases. While the remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.7 Impact of Number of Candidate Layers in $\mathcal{L}_{\text{SOA}}$

We conduct experiments in encoder mode to analyze the effect of the number of the final layers used to calculate the $\mathcal{L}_{\text{SOA}}$. Figure 11 (i) shows the R@2 scores for using the last 1 to 5 layers. The highest R@2 is attained using the last 4 layers. While the remaining figures focus on the 2WikiMultihopQA dataset, other datasets, including HotpotQA and MuSiQue, follow a similar trend.

F.8 Impact of LLM Backbones

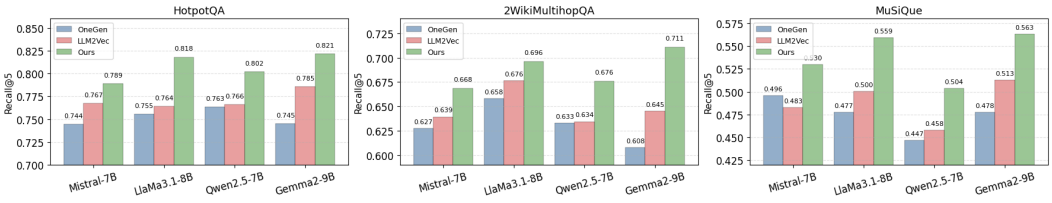

Figure 12: Retrieval performance of ShiftRAG and baselines across different LLM backbones.

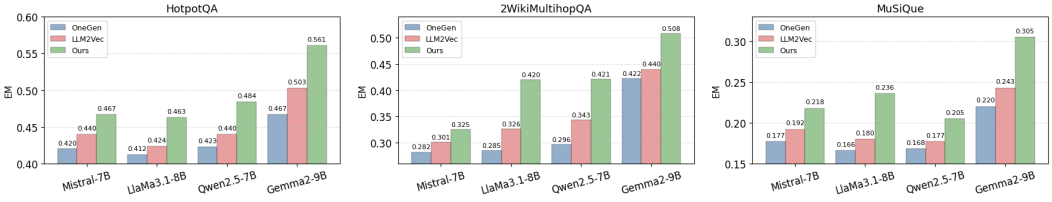

Figure 13: QA performance of ShiftRAG and baselines across different LLM backbones.

To demonstrate the generalizability of our approach beyond the 2WikiMultihopQA dataset discussed in the main text, we include retrieval and end-to-end Question Answering performance for the HotpotQA and MuSiQue datasets across various Large Language Model (LLM) backbones, including Mistral-7B[29], LlaMA3.1-8B[16], Qwen2.5-7B[50], and Gemma2-9B[14].

As illustrated in Figures 12 and 13, our experimental results indicate that ShiftRAG consistently outperforms LLM2Vec[2] and OneGen[68] across all evaluated datasets and backbones. Regarding retrieval performance, the most significant gain was observed on the MuSiQue dataset using the LlaMA3.1-8B backbone, where ShiftRAG achieved an 11.75% increase in R@5 compared to LLM2Vec. Furthermore, performance improvements were noted on HotpotQA with the LlaMA backbone at 7.04% and on 2WikiMultihopQA with Gemma at 4.02%. Regarding end-to-end QA performance, ShiftRAG achieved its most significant improvement on the MuSiQue dataset using LlaMA3.1-8B, recording a 31.11% higher EM than LLM2Vec. Notable gains were observed on 2WikiMultihopQA with the LlaMA backbone at 28.83% and on HotpotQA with the Gemma backbone at 11.53%.

The performance enhancements remained consistent across diverse model architectures, including Mistral, LlaMA, Gemma, and Qwen. This stability suggests that the methodology of ShiftRAG is not dependent on a specific model structure but serves as a universally applicable optimization framework for RAG-based systems. Notably, the Gemma2-9B model exhibited the most robust and consistent performance across both retrieval and reasoning tasks compared to other backbones. Based on these results, Gemma was selected as the primary backbone for ShiftRAG.

G Case Study

G.1 Knowledge Comparison across Different Attention Policies

Figure 14 illustrates how the attention policy used in encoder-mode training affects the retrieved evidence. For the question "Where was the place of death of Abdul-Aziz Bin Muhammad's father?", the bidirectionally trained retriever returns both Abdul-Aziz bin Muhammad and Muhammad bin Saud, where the latter provides the key evidence that the father's place of death was "Diriyah". In contrast, the unidirectionally trained retriever still retrieves "Abdul-Aziz bin Muhammad", but fails to retrieve Muhammad bin Saud and instead returns lexically similar yet non-supporting documents such as Abdul Aziz bin Fahd and Abdullah ibn Muhammad. This comparison shows that the two attention mechanisms rely on different signals. Bidirectional retrieval is better at capturing global relational semantics needed to connect father and place of death, whereas unidirectional retrieval is more prone to lexical-overlap bias over surface entity names. As a result, the bidirectionally trained retriever identifies more answer-supporting evidence, while the unidirectionally trained retriever tends to retrieve documents that are locally similar in wording but less useful for reasoning.

H Prompt

H.1 Prompt 1. Prompt for Rationale Generation

We generate pair-specific textual rationales using GPT-5-nano [56] with the prompt in Figure 15. For each question–document pair, the model produces a single concise sentence that explains whether the passage helps answer the question and how the passage contributes to the answer. If the passage is not useful, the prompt forces the model to first output a fixed rejection sentence and then briefly explain why the passage is irrelevant. The generated rationales are constructed offline and used only as teacher-side supervision for the knowledge distillation loss in decoder-mode training.

H.2 Prompt 2. Prompt for Anchor Token Initialization

We use the prompt in Figure 16 to initialize anchor token embeddings. Given a question, a passage, and a pre-generated textual rationale, the model is trained to reproduce the rationale in a concise sentence form. This procedure aligns anchor token embeddings with the generator's token space before decoder-mode training.

H.3 Prompt 3. Prompt for Decoder-Mode Training

We use the prompt in Figure 17 for decoder-mode training. The model receives the retrieved documents and the question as textual input, while $<$ `LRN_VIRTUAL_TOKENS` $>$ denotes the position where SST-generated soft knowledge tokens are inserted. The model is trained to generate only the final answer, without additional explanations or formatting.

Case Study: Knowledge Discrimination

Question: Where was the place of **death** of **Abdul-Aziz Bin Muhammad**'s **father**?
Answer: Diriyah

Train Retrieval Mode using Bidirectional Attention

Document [1] (Title: Abdul-Aziz Bin Muhammad) Imam Abdul Aziz bin Muhammad bin Saud d. 1803 was the second ruler of the First Saudi State and **son of Muhammad** bin Saud. He was also the son-in-law of Muhammad bin Abdul Wahhab. He ruled from 1765 until 1803
is supporting: True

Document [2] (Title: Saud bin Abdul-Aziz Bin Muhammad bin Saud) Imam Saud al Kabeer bin **Abdul-Aziz Bin Muhammad** bin Saud d. 1814 ruled the First Saudi State from 1803 to 1814. During his rule, Saud annexed Mecca and Medina from the Ottoman Empire.
is supporting: False

Document [3] (Title: Muhammad bin Saud) Muhammad ibn Saud ibn Muhammad ibn Muqrin Al-Maridi **died** 1765, also known as Ibn Saud, was the emir of Ad-Diriyyah and is considered the founder of the First Saudi State and the Saud dynasty, which are named for his father, Saud ibn Muhammad ibn Muqrin **died** 1725. Ibn Saud's family then known as the Al Muqrin traced its descent to the tribe of Banu Audi and Hanifa tribes but, despite popular misconceptions, Muhammad ibn Saud was neither a nomadic bedouin nor was he a tribal leader. Rather, he was the chief ëmiröf an agricultural settlement near modern-day Riyadh, called Diriyah. Furthermore, he was a competent and ambitious desert warrior.
is supporting: True

Train Retrieval Mode using Unidirectional Attention

Document [1] (Title: Abdul Aziz bin Fahd) Abdul Aziz bin Fahd is a Saudi Arabian Prince and member of the Royal House of Saud.
is supporting: False

Document [2] (Title: Abdul-Aziz Bin Muhammad) Imam Abdul Aziz bin Muhammad bin Saud d. 1803 was the second ruler of the First Saudi State and son of Muhammad bin Saud. He was also the son-in-law of Muhammad bin Abdul Wahhab. He ruled from 1765 until 1803. **is supporting**: True

Document [3] (Title: Abdullah ibn Muhammad) **Abdullah ibn Muhammad** also known as Tahir **ibn Muhammad** was one of the sons of **Muhammad** and Khadija. Qasim ibn **Muhammad** was his older brother. Muhammad gave him the name of his father.
is supporting: False

Figure 14: A qualitative case study showing that bidirectionally trained retrieval preserves the full evidence chain needed for the answer, including the surface entity cue (orange), the father relation (blue), and the death-related clue (red), whereas unidirectionally trained retrieval overemphasizes surface lexical overlap (orange) and retrieves non-supporting documents that miss the relation and death evidence (blue, red) required to infer the correct answer.

Prompt: Rationale Generation

Given a question and a passage, write a rationale explaining whether the passage helps answer the question. If the passage helps answer the question, write exactly one concise English sentence explaining which fact in the passage is useful and how it supports answering the question. The passage may provide a direct answer, a required attribute of an entity, or one necessary piece of evidence for a multi-hop question.

If the passage does not help answer the question, first output exactly:
- Start "This passage does not help answer the given question.". Then write exactly one concise English sentence explaining why the passage is not useful, such as an entity mismatch, missing required information, no relation to the question, or irrelevant topic.
- Do not answer the question directly unless it is necessary to explain the passage's usefulness. Do not add labels, bullet points, JSON, citations, or any extra text.

Passage:
(Title: {title}) {paragraph}

Question:
{question}

Rationale:

Figure 15: Prompt for generating pair-specific textual rationales.

Prompt: Anchor Token Initialization

Given a question and a passage: if helpful, write one English sentence why; otherwise output exactly "This passage does not help answer the given question." Then write one sentence why.

Passage:
(Title: {title}) {paragraph}

Question:
{question}

Rationale:
{rationale}

Figure 16: Prompt used for anchor token initialization with textual rationale supervision.

Prompt: Question Answering

According to the previous relevant passages, please answer the following question. Only return the answer without any other words.

Passage:
Document [1] (Title: {title$_1$}) {paragraph$_1$}
Document [2] (Title: {title$_2$}) {paragraph$_2$}
...
Document [K] (Title: {title$_K$}) {paragraph$_K$}

Question:
{question}

Answer:
{answer}

$<$ `Soft Knowledge Tokens` $>$

Figure 17: Prompt used for generation-mode training with retrieved documents and contextualized tokens.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

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 abstract and introduction lay out three main claims and explain each in detail. These claims concern the textual bottleneck in standard RAG pipelines, the rank collapse and objective conflict that arise when decoder-only LLMs are adapted for retrieval and generation, and the proposed decoupled framework that bridges retrieval and generation through continuous soft knowledge tokens, layer-wise gap-aware retrieval, and soft orthogonal signals. The methodology, experiments, ablation studies, and analyses substantiate these claims through the proposed decoupled encoder and decoder modes, semantic signal translator, retrieval and QA evaluations, component ablations, and learning-strategy comparisons.

   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: The limitations of the work are laid out in the discussion portion of the paper. See Section 5.

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: [Yes]

   Justification: We provide correct and complete proofs for each theoretical result, as illustrated in the experiments and appendix sections.

   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: Datasets, models, and hyperparameters used in implementing proposed algorithms are all described in detail. See Section 4.1 and Appendix A.

   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: We already provide the code and the preprocessed datasets for reproducing the experimental results in the paper.

   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: We provide the details of our experiments including hyperparameter settings and prompts for training and inference.

   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 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 performed training and inference five times for each setting, computed the mean and standard deviation, and added this information to the table caption. See Appendix C.1 and Appendix C.2.

   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: The paper provides the compute resources and runtime information needed to reproduce the experiments. Appendix A reports the hardware setup, including training on four NVIDIA RTX A6000 GPUs and inference on a single GPU. Section 4.4 compares training time with retrieval and QA performance across learning strategies, while Appendices D.1, 4.7, and D.2 provide additional training latency, end-to-end inference latency with peak VRAM usage, and computational complexity analyses, respectively.

   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 reviewed the NeurIPS Code of Ethics and made sure that our paper conforms to it in every respect.

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: [N/A]

Justification: We do not anticipate our work to have any meaningful positive or negative societal impacts.

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: No such models or datasets are involved.

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: We cite the original papers, such as those describing the datasets.

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: [Yes]

Justification: All of the code and data assets released alongside our paper are appropriately documented for reproducibility.

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: We did not employ crowdsourcing or involve any human subjects in our research.

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: We did not employ crowdsourcing or involve any human subjects in our research.

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: [Yes]

    Justification: The paper documents the use of LLMs that affects the methodology, beyond writing or formatting assistance. In particular, Appendix H.1 describes the offline use of GPT-5-nano to generate pair-specific textual rationales for question-document pairs. These generated rationales are used only as teacher-side supervision for decoder-mode training and anchor token initialization, as described in Sections 3.3 and Appendix B.2, and are not provided at inference time.

    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.