BRIDGECOMP: PROMPT COMPRESSION VIA REASONING-CONDITIONED ATTENTION AND ANCHOR-CONDITIONED POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The proliferation of large language models in long-context applications has made efficient inference paramount, and prompt compression has emerged as a prominent paradigm to condense context while preserving task-relevant information. However, contemporary compression frameworks introduce semantic distortion and hallucinations by underestimating reasoning-critical tokens or inflating token importance through superficial word-matching. Moreover, answer-level supervision alone provides only indirect guidance for preserving reasoning-critical evidence. To address these challenges, we introduce BridgeComp, a prompt compression framework that estimates token importance from latent tokens that serve as implicit reasoning states. We append learnable latent tokens to the question and use upper-layer attention to isolate deep relational reasoning signals from shallow lexical noise. To align token selection with both answer prediction and compression quality, we train the compressor with reinforcement learning that jointly rewards answer correctness and supporting-evidence preservation. Experiments on question-answering and long-context benchmarks show that BridgeComp achieves state-of-the-art performance. Under a compression setting retaining only 10% of tokens, BridgeComp improves Exact Match by up to 62.44% over the strongest baseline. For long-context compression, BridgeComp achieves up to an 11.75× compression speedup and reduces peak memory usage by up to 39.14%. Our code is available[1].

1 INTRODUCTION

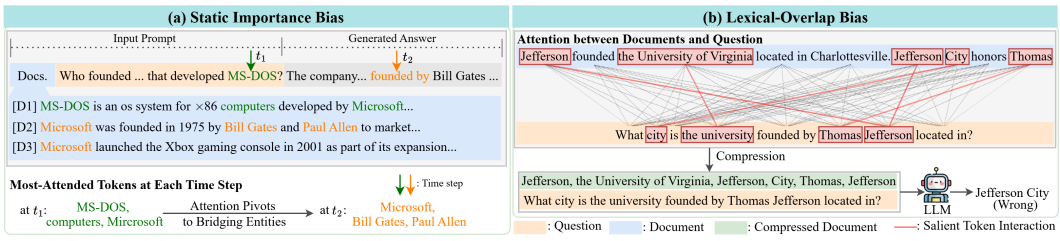


Figure 1: Motivation for ours. Existing prompt compression methods suffer from (a) static importance bias and (b) lexical-overlap bias, which can remove bridging information during compression.

The proliferation of large language models (LLMs) in long-context applications has made efficient inference paramount for tasks such as question answering (QA) (Java et al., 2026; Shi et al., 2026), document summarization (Kuwahara et al., 2026; Chang et al., 2024), and multi-turn dialogue (Laban et al., 2026; Naous et al., 2026). To mitigate the computational overhead in processing massive inputs, prompt compression has emerged as a prominent paradigm to condense context while preserving task-relevant information. However, existing compression frameworks introduce semantic distortion or discard subtle contextual cues, rendering LLMs prone to hallucinate (Ferrando et al., 2025; Orgad et al., 2025). We hypothesize that these limitations stem from three reasons.

[1] https://anonymous.4open.science/r/BridgeComp-0F35

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

First, existing methods suffer from static importance bias, assigning low importance scores to tokens that are essential for reasoning but less salient in the input prompt. They may overemphasize immediate, surface-level salience by evaluating token importance within the prompt. However, the relevance of vital intermediate clues is not always apparent in the initial prompt; these clues are needed to link question-related information in one document to answer-supporting evidence in another. As illustrated in Figure 1(a), attention shifts from prompt keywords to evidence needed for answer formation. At step t_1, attention is concentrated locally on keywords like "MS-DOS" and "computers", but by step t_2, as the model formulates the phrase "founded by", the attention pivots to distant bridging entities like "Microsoft" and "Bill Gates". Contemporary compressors do not treat token importance as a reasoning-conditioned quantity, thus prematurely purging superficially low-scoring but reasoning-critical tokens.

Second, existing attention-based compressors suffer from lexical-overlap bias, wherein token importance is inflated by superficial word-matching between a question and retrieved documents. This flaw is heavily exacerbated by treating all transformer layers uniformly. Consequently, these methods overemphasize the shallow lexical-matching cues dominant in early layers while diluting the nuanced, relational signals computed in deep layers. As shown in the question "What city is the university founded by Thomas Jefferson located in?" of Figure 1(b), question-to-document attention over-allocates budget to overlapping entities like "Thomas Jefferson" and "University". Meanwhile, the answer-bearing bridging context, i.e., "located in Charlottesville", is erroneously pruned. This reliance on lexical salience strips the compressed prompt of its vital reasoning paths.

Third, training a compressor only to predict the correct answer provides indirect guidance for preserving reasoning-critical evidence. This objective does not explicitly assess whether the compressed prompt retains the bridging information required for multi-hop reasoning.

To address these challenges, we introduce BridgeComp, a prompt compression framework that estimates token importance from soft tokens that serve as implicit reasoning states. By leveraging the attention distributions of these tokens specifically within the upper transformer layers, our framework successfully isolates deep relational reasoning signals from shallow lexical noise. We train the compressor with reinforcement learning (RL) that jointly rewards answer quality and supporting-evidence preservation, aligning token selection with the retention of reasoning-critical information.

Our contributions are fourfold: (1) we shift the prompt compression paradigm from preserving statistically salient tokens to retaining reasoning-critical bridges by formalizing token importance as a quantity conditioned on implicit reasoning states; (2) by scoring tokens through latent tokens and selectively exploiting upper-layer attention, we actively suppress lower-layer lexical distractions while preserving crucial downstream relational signals; (3) we introduce an RL-based training strategy that refines token selection using joint feedback on answer quality and supporting-evidence preservation; (4) we demonstrate state-of-the-art performance across QA and long-context benchmarks. Under an aggressive compression setting that retains only 10% of tokens, BridgeComp improves Exact Match by up to 62.44% over the strongest baseline. For long-context compression, BridgeComp achieves up to an 11.75$\times$ speedup and reduces peak memory by up to 39.14%.

2 Related Work

Prompt compression methods are categorized into two approaches. First, a perplexity- and relevance-based approach aims to preserve task-relevant information under constrained context budgets. Selective-Context (Li et al., 2023) and LLMLingua (Jiang et al., 2023) prune tokens using perplexity or entropy salience, while LLMLingua2 (Pan et al., 2024) and LongLLMLingua (Jiang et al., 2024) train distilled token selectors for task-agnostic compression. CPC (Liskavets et al., 2025) performs sentence-level compression by scoring the relevance of each sentence within its surrounding context. Second, an attention-based approach leverages the attention maps of LLMs as token-selection signals. DAC (Zhao et al., 2025a) scores document tokens by combining information entropy with the sum of self-attention weights received from other tokens. AttnComp (Zhao et al., 2025b) clusters document tokens into semantic units via a mixture of self-attention and query-to-document cross-attention. While recent studies (Ko et al., 2026; Chuang et al., 2024; Li et al., 2026) have shown that lower layers in transformers capture surface syntax whereas upper layers resolve abstract semantics, the attention-based approach neglects these layer-wise transitions by accumulating attention scores across all layers over document-to-document or question-to-document ranges, thus being susceptible

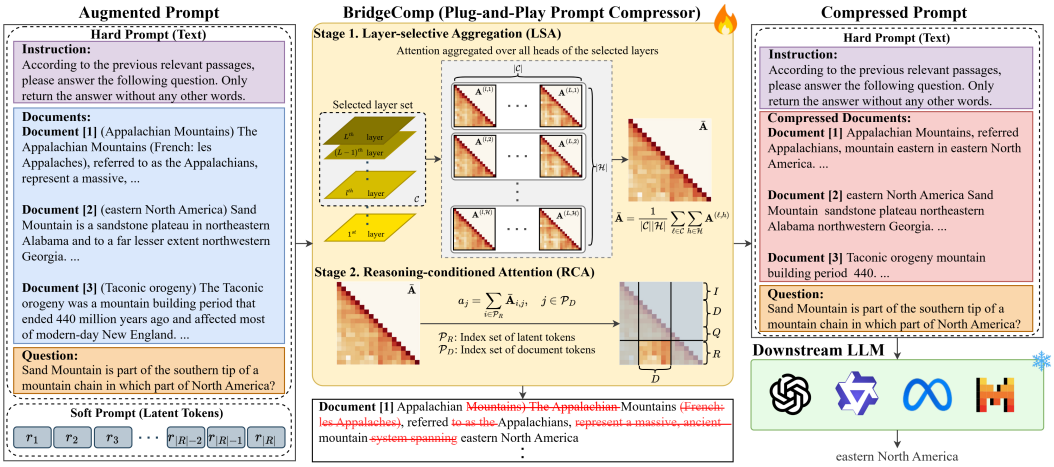


Figure 2: Illustration of the inference pipeline of BridgeComp.

to lexical-overlap bias. Both approaches compute token importance only from the input prompt. Consequently, they are vulnerable to static importance bias, failing to preserve bridging information.

Recently, reinforcement learning (RL) using downstream task rewards has been employed in the relevance-based approach. TACO-RL (Shandilya et al., 2025) and PCRL (Jung & Kim, 2024) learn token-level compression with task selectors. CoT-Influx (Huang et al., 2024) applies RL to CoT compression. However, they optimize compression decisions without modeling how evidence relevance evolves during reasoning, leaving reasoning-critical bridges vulnerable to pruning.

3 Problem Formulation

We represent a standard input prompt as $S = (I, D, Q)$, where I, D, and Q are an instruction, retrieved documents, and a question, respectively. Given a target compression rate $\tau \in (0, 1]$, the objective of prompt compression is to produce a compressed prompt $\widetilde{S}_\tau = (I, \widetilde{D}_\tau, Q)$, where $\widetilde{D}_\tau$ is a subset of tokens in D. This task is formulated as a constrained optimization problem that minimizes the discrepancy between the output distributions induced by the original and compressed prompts:

$$\min_{\widetilde{D}_\tau} \mathbb{D}\left(P(\cdot \mid S), P(\cdot \mid \widetilde{S}_\tau)\right) \text{ s.t. } \frac{\mid \widetilde{D}_\tau \mid}{\mid D \mid} \leq \tau \tag{1}$$

where $P(\cdot \mid \cdot)$ represents the output token distribution generated by the downstream LLM, and $\mathbb{D}(\cdot, \cdot)$ is a distance metric (e.g., Kullback-Leibler divergence) between the two distributions.

4 Method

4.1 Overview

Figure 2 illustrates the overview of our novel prompt compression framework BridgeComp. Initially, we construct an augmented prompt by concatenating the hard prompt S with a soft prompt R. The framework compresses the context through a two-stage pipeline. First, we aggregate attention signals from specified upper layers, filtering out early-stage lexical noise dominant in lower layers (Section 4.3). Next, we aggregate attention weights from the soft prompt R to each document token, in order to compute the latent reasoning-conditioned importance score of that token. All document tokens are ranked by these scores, and the lowest-ranked tokens are pruned to satisfy the compression rate τ (Section 4.4). Our parameter-efficient training procedure jointly optimizes the latent tokens in R and LoRA adapters through supervised learning and RL (Section 4.5).

4.2 Latent Token Instantiation

First, we construct a hybrid prompt that concatenates discrete text and continuous latent representations. We instantiate the abstract rationale $R = (r_1, r_2, \ldots, r_{|R|})$ as a sequence of learnable latent tokens, where $|R|$ is a tunable hyperparameter. The augmented prompt $S_R = (I, D, Q, R)$ is fed to the LLM, which will then contextualize the latent tokens over the entire preceding hard context during a forward pass. The impact of different rationale representations is analyzed in Appendix F.7.

4.3 Layer-selective Aggregation (LSA)

To isolate high-level semantics from shallow syntactic noise, BridgeComp aggregates attention signals selectively across specific representational stages rather than treating all transformer layers uniformly. Let $\mathcal{C}$ denote the set of selected upper layers, corresponding to the top $\rho\%$ of layers of the LLM. Let $\mathcal{H}$ denote the set of all attention heads per layer. Passing the augmented prompt S_R through the LLM yields the causal self-attention matrix $\mathbf{A}^{(\ell,h)} \in \mathbb{R}^{|S_R|\times|S_R|}$ for each layer $\ell \in \mathcal{C}$ and head $h \in \mathcal{H}$, where rows and columns correspond to query and key positions, respectively. We compute the aggregated attention matrix $\bar{\mathbf{A}} \in \mathbb{R}^{|S_R|\times|S_R|}$ by averaging over all heads within the upper layers:

$$\bar{\mathbf{A}} = \frac{1}{|\mathcal{C}||\mathcal{H}|} \sum_{\ell\in\mathcal{C}} \sum_{h\in\mathcal{H}} \mathbf{A}^{(\ell,h)} . \tag{2}$$

By omitting the lower layers, this aggregation actively suppresses early-stage surface lexical signals, ensuring that the downstream token scoring is driven primarily by late-stage semantic integration and relational dependencies. The impact of $|\mathcal{C}|$ on performance is analyzed in Appendix G.3.

4.4 Reasoning-conditioned Attention (RCA)

Attention within the input prompt alone can over-inflate the importance of document tokens that exhibit high document-internal frequency or superficial matches with the question. As illustrated in Figure 3, existing attention-based methods estimate token importance from the spatial attention regions produced by the hard prompt S, e.g., internal document-to-document attention tracking ($D\to D$) in DAC (Zhao et al., 2025a), and question-to-document cross-attention ($Q\to D$) in AttnComp (Zhao et al., 2025b). In contrast, BridgeComp isolates the scoring space to the attention directed from the continuous soft prompt to the discrete document tokens ($R\to D$). The causal attention masking scheme of decoder-only LLMs guarantees that these latent states in R attend to all preceding context during latent reasoning.

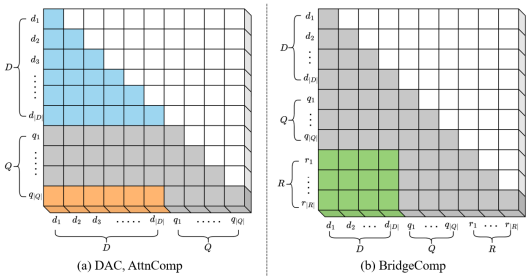


Figure 3: Comparison of targeted attention regions. Unlike DAC (blue) and AttnComp (orange), BridgeComp (green) scores document tokens using latent-token-to-document attention.

Formally, let $\mathcal{P}_D$ and $\mathcal{P}_R$ denote sets of positions of all document tokens and all latent tokens, respectively, within the augmented sequence S_R. We formulate the reasoning-conditioned attention score for a document token at each position $j \in \mathcal{P}_D$ as:

$$a_j = \sum_{i\in\mathcal{P}_R} \bar{\mathbf{A}}_{i,j} . \tag{3}$$

A larger a_j indicates that a document token at j receives stronger attention from contextualized latent reasoning states. Finally, BridgeComp ranks all document tokens in descending order of a_j, and prunes the lowest-scoring elements until the target compression rate τ is satisfied. The remaining tokens are preserved in their original relative positions to construct the compressed context $\widetilde{D}_\tau$.

By deriving token scores directly from contextualized latent states rather than fixed input-side salience or broad word-level matches, BridgeComp bypasses both static importance and lexical-overlap biases, preserving the exact bridging paths required for accurate answer formation. Further analyses of

lexical overlap, attention range, and robustness across different multi-hop problem types are provided in Appendices F.5, F.8, and F.9, respectively.

4.5 TRAINING

To train our compressor, we optimize the soft prompt embeddings θ_R and the compressor LoRA parameters θ_{LoRA} while keeping the underlying LLM backbone Φ completely frozen. First, we optimize only θ_R for 1,000 warm-up steps using the next-token prediction (NTP) objective to align the soft prompt tokens with the pretrained embedding space (Section 4.5.1). Following warm-up, we generate compression rollouts, evaluate them using a composite reward function (Section 4.5.2), and then jointly optimize $\theta = \{\theta_R,\ \theta_{\text{LoRA}}\}$ by combining NTP and RL objectives (Section 4.5.3).

4.5.1 NEXT-TOKEN PREDICTION (NTP)

Initially, we train BridgeComp on the augmented prompt S_R using the NTP loss: $\mathcal{L}_{\text{NTP}} = -\sum_{t=1}^{T} \log P\left(y_t \mid S_R, y_{<t}\right)$, where $y_{<t}$ denotes the answer tokens preceding y_t, and $Y = (y_1, \ldots, y_T)$ denotes the gold answer sequence. This loss forces R to encode document-question interactions and reasoning transitions required for answer formation.

4.5.2 COMPRESSION ROLLOUT AND REWARD

For each document token $d_j \in D$ (where $j \in \{1, \ldots, |D|\}$), we normalize the latent-to-document attention scores a_j (Section 4.4), and compute sampling logits u_j:

$$p_j = \frac{a_j + \epsilon}{\sum_{j'=1}^{|D|}(a_{j'} + \epsilon)}, \qquad u_j = \frac{\log p_j}{T_\pi}, \tag{4}$$

where $\epsilon > 0$ is a numerical-stability constant and T_π is the sampling temperature. Given a target compression rate τ, the token budget is $k = \lfloor \tau |D| \rfloor$. We construct an anchor mask $\mathbf{m} \in \{0,\ 1\}^{|D|}$ by selecting the top-k tokens with the highest p_j : $\mathbf{m}_j = 1$ if $j \in \text{top-k}\left(\{p_{j\prime}\}_{j\prime=1}^{|D|}\right)$; 0 otherwise.

Next, by exploring the local neighborhood while maintaining the budget constraint k, we sample rollouts to generate a set $\mathcal{M}$ of masks $\widetilde{\mathbf{m}}$ perturbed from $\mathbf{m}$. Each rollout swaps $2b$ tokens from $\mathbf{m}$: b tokens with $\mathbf{m}_j = 1$ are masked out ($1 \to 0$), and b tokens with $\mathbf{m}_j = 0$ become unmasked ($0 \to 1$).

Formally, let $J^- = (j_1^-, \ldots, j_b^-)$ and $J^+ = (j_1^+, \ldots, j_b^+)$ denote the ordered sequences of pruned and added token indices, respectively. Starting from $\mathbf{m}$, we generate a rollout mask $\tilde{\mathbf{m}}$ by iteratively masking out indices along J^- and unmasking indices along J^+, respectively. Within each rollout sampling pass, the token score logits u_j remain fixed, while selection probabilities are renormalized sequentially without replacement over the pools of remaining indices. Specifically, at each iteration $v \in \{1, \ldots, b\}$, pruning probability P_θ^- is obtained via softmax $\left(-u_{j_v^-}\right)$ over currently unmasked token indices (favoring low-scoring tokens for eviction), while unmasking probability P_θ^+ is obtained via softmax $\left(u_{j_v^+}\right)$ over token indices currently masked out (favoring high-scoring tokens for retention). The joint log-probability across all $2b$ swap decisions is:

$$\ell_{\tilde{\mathbf{m}}}(\theta) = \sum_{v=1}^{b} \log P_\theta^-\left(j_v^- \mid S_R, \mathbf{m}, j_{<v}^-\right) + \sum_{v=1}^{b} \log P_\theta^+\left(j_v^+ \mid S_R, \mathbf{m}, j_{<v}^+\right). \tag{5}$$

This formulation tracks the exact likelihood of the sampling execution order while preserving the token budget k. Unlike unconstrained token-level mask-generation in TACO-RL (Shandilya et al., 2025), this exact-budget swap exploration operates on a localized action space centered around the latent-to-document attention-based anchor. Applying $\tilde{\mathbf{m}}$ yields the compressed document $\widehat{D}$. Each rollout is evaluated by a composite reward function with hyperparameters λ_{attn} and λ_{miss}:

$$r(\tilde{\mathbf{m}}) = r_{\text{F1}}(\tilde{\mathbf{m}}) + \lambda_{\text{attn}} r_{\text{attn}}(\tilde{\mathbf{m}}) - \lambda_{\text{miss}} p_{\text{miss}}(\tilde{\mathbf{m}}), \tag{6}$$

where three crucial factors $r_{\text{F1}}(\tilde{\mathbf{m}})$, $r_{\text{attn}}(\tilde{\mathbf{m}})$, and $p_{\text{miss}}(\tilde{\mathbf{m}})$ are defined below.

Task Reward. Given the compressed hard prompt, a frozen LLM autoregressively generates the output: $\widehat{Y} = \mathrm{LLM}\left(I, \widehat{D}, Q\right)$. For a gold answer set $\mathcal{A}_Q$ for question Q, the task reward is defined as the F1-score between $\widehat{Y}$ and $\mathcal{A}_Q$, i.e., $r_{F1}(\widetilde{\mathbf{m}}) = \max_{Y \in \mathcal{A}_Q} \mathrm{F1}\left(\widehat{Y}, Y\right)$.

Attention Quality Reward. To encourage the latent tokens to attend to the document tokens relevant to Q, we split all tokens d_j in the retrieved documents into (1) the set D^+ of the tokens contained in the gold supporting documents, and (2) the set D^- of the remaining tokens. Attention quality reward is defined by using the retained latent-attention mass over supporting and non-supporting documents:

$$r_{\mathrm{attn}}(\tilde{\mathbf{m}}) = \frac{\sum_{d_j \in D^+} \tilde{\mathbf{m}}_j p_j}{\sum_{d_j \in D^+} p_j + \epsilon} - \beta_{\mathrm{rest}} \frac{\sum_{d_j \in D^-} \tilde{\mathbf{m}}_j p_j}{\sum_{d_j \in D^-} p_j + \epsilon}. \tag{7}$$

Document Coverage Penalty. Let $\mathcal{T}(D_i)$ denote the set of tokens in document D_i. Given a set $\mathcal{D}^+$ of gold documents, we penalize the sampled masks that fully purge all tokens in any gold document:

$$p_{\mathrm{miss}}(\tilde{\mathbf{m}}) = \frac{1}{|\mathcal{D}^+|} \sum_{D_i \in \mathcal{D}^+} \mathbb{I}\left[\sum_{d_j \in \mathcal{T}(D_i)} \tilde{\mathbf{m}}_j = 0\right]. \tag{8}$$

4.5.3 Anchor-conditioned Group Policy Gradient

We evaluate the quality of the anchor and perturbed masks using the reward function in Equation 6. The mean reward within $\mathcal{M}$ is $\bar{r} = \frac{1}{|\mathcal{M}|} \sum_{\widetilde{\mathbf{m}} \in \mathcal{M}} r\left(\widetilde{\mathbf{m}}\right)$. Following GPG (Chu et al., 2026), we use the group-centered improvement as $r\left(\widetilde{\mathbf{m}}\right) - \bar{r}$ for each sampled mask $\widetilde{\mathbf{m}}$. However, this signal compares sampled masks with one another but does not indicate whether they improve upon the anchor. To improve training stability by refining reward-derived advantages (Salmani-Zarchi et al., 2026), we incorporate an anchor-relative correction term into the advantage in the GPG loss. For each sampled mask $\widetilde{\mathbf{m}}$, the final advantage is computed by measuring (i) its relative reward improvement within $\mathcal{M}$ and (ii) its reward improvement over the anchor mask:

$$A_{\tilde{\mathbf{m}}} = \left(r(\tilde{\mathbf{m}}) - \bar{r}\right) + \eta \left(r(\tilde{\mathbf{m}}) - r(\mathbf{m})\right), \tag{9}$$

where η is a fixed coefficient throughout training. We use the policy loss below by adapting the advantage-weighted log-probability objective of GPG to the mask sampling process:

$$\mathcal{L}_{\mathrm{AGPG}} = -\frac{1}{|\mathcal{M}|} \sum_{\tilde{\mathbf{m}} \in \mathcal{M}} \mathrm{sg}\left(A_{\tilde{\mathbf{m}}}\right) \ell_{\tilde{\mathbf{m}}}(\theta), \tag{10}$$

where $\mathrm{sg}\left(\cdot\right)$ denotes the stop-gradient operator. Each rollout receives one advantage, shared across its $2b$ internal token selections through their joint log-probability. During the policy update, the anchor mask and all sampled masks are treated as fixed, and gradients flow through the selection log-probabilities rather than through the rewards. After the warm-up stage, we train θ with the complete training objective $\mathcal{L} = \mathcal{L}_{\mathrm{NTP}} + \lambda_{\mathrm{RL}} \mathcal{L}_{\mathrm{AGPG}}$, where λ_{RL} controls the contribution of compression policy optimization.

5 Experiments

5.1 Experimental Setup

We evaluate BridgeComp on seven benchmarks for single- and multi-hop QA, long-context tasks, and multi-agent systems. The QA and long-context datasets are detailed in Appendices A.1 and A.2, respectively. For a controlled plug-and-play comparison, all methods use the same downstream generators and the same top-10 documents retrieved by RaLa for each instance. To prevent data leakage, we exclude evaluation questions and documents from training; details are described in Appendix F.12. Unless otherwise stated, we use LLaMA3.2-3B as the compressor backbone, and GPT-5.6 Luna and Qwen3.5-9B as downstream generators. BridgeComp uses $|R| = 50$ latent tokens and the top 40% of compressor layers for layer-selective aggregation. Implementation details, baseline descriptions, and hyperparameter sensitivity analyses are provided in Appendices A.3, B, and G, respectively. Appendix D.6 presents comparisons with matched compressor and generator backbones. Appendix D.7 presents experiments applying BridgeComp to multi-agent systems. Appendix G.2 shows that the benefit of the anchor-relative correction is robust to its weight η. The token swap budget analysis is described in Appendix G.1.

5.2 MAIN RESULTS

Methods	GPT-5.6 Luna						Qwen3.5-9B					
	HotpotQA		2WikiMultihopQA		MuSiQue		HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Compression rate τ = 0.5												
DAC	45.1	57.1	49.0	55.3	31.8	43.1	40.8	51.4	43.0	49.1	26.3	34.1
LLMLingua2	45.9	58.1	46.8	53.1	38.9	51.0	41.5	52.3	41.1	47.1	32.0	40.3
CPC	55.1	65.9	50.6	62.8	38.0	49.5	49.6	59.6	44.9	57.7	31.3	39.6
AttnComp	60.7	75.4	61.5	69.4	38.7	54.4	54.9	67.8	54.1	61.6	32.2	43.0
BridgeComp	**66.4**	**80.0**	**64.3**	**72.8**	**46.3**	**60.2**	**60.1**	**72.0**	**56.5**	**64.6**	**38.3**	**47.6**
Compression rate τ = 0.2												
DAC	33.2	40.5	34.4	39.5	21.9	29.4	26.3	32.7	27.8	31.3	17.0	21.2
LLMLingua2	35.7	47.3	40.9	43.8	24.2	40.2	28.2	38.2	33.0	34.7	18.9	29.0
CPC	54.1	63.7	48.1	55.2	28.3	45.5	42.4	50.6	39.5	43.3	22.1	33.0
AttnComp	58.8	71.2	55.6	63.2	33.8	47.7	46.5	57.5	44.9	50.1	26.3	34.5
BridgeComp	**64.9**	**78.5**	**58.3**	**67.6**	**41.9**	**56.7**	**51.3**	**63.4**	**47.1**	**53.5**	**32.6**	**41.0**
Original Prompt	65.2	79.9	64.0	72.0	47.1	61.0	63.2	76.0	58.7	66.7	40.8	53.2
Zero-shot	26.8	39.7	28.6	37.8	7.4	19.1	21.8	29.1	28.1	33.2	4.3	12.0

Table 1: Performance on multi-hop QA benchmarks. We standardize the compressor backbone to LLaMA3.2-3B for all methods except LLMLingua2, for which we use the publicly released checkpoint. Original Prompt and Zero-shot are reported as reference settings. The best and second-best results are highlighted in bold and underlined, respectively.

Table 1 presents performance on multi-hop QA benchmarks using GPT-5.6 Luna and Qwen3.5-9B as downstream generators. BridgeComp achieves the highest Exact Match (EM) and F1-score among all methods across all datasets, compression rates, and generators. The gains are most pronounced on MuSiQue. With GPT-5.6 Luna, BridgeComp improves EM over AttnComp by 19.64% (38.7→46.3) at $\tau = 0.5$ and 23.96% (33.8→41.9) at $\tau = 0.2$. Similar gains are observed with Qwen3.5-9B, where BridgeComp improves EM by 18.94% (32.2→38.3) and 23.95% (26.3→32.6) across the respective compression rates. These consistent improvements across open-weight and proprietary generators demonstrate that BridgeComp preserves bridging information under heavy context compression. LLM-as-a-Judge analyses in Appendix F.1 show that BridgeComp improves complete-chain preservation by 28.25% and reduces evidence omission by 47.33% relative to AttnComp. Standard deviations, performance stratified by question types, and compression rate sweeps are provided in Appendices D.1, F.9, and F.11, respectively.

5.3 PERFORMANCE COMPARISON WITH FINE-TUNED AGGRESSIVE COMPRESSION METHODS

Methods	HotpotQA		2WikiMultihopQA		MuSiQue		Time (s)
	EM	F1	EM	F1	EM	F1	
AutoCompressors	18.5	25.1	8.8	17.2	6.8	11.6	4.87
xRAG	28.0	35.3	11.9	18.9	9.0	14.5	1.16
ICAE	40.4	46.3	22.0	27.4	12.2	20.6	1.36
500xCompressor	41.8	60.3	23.2	27.8	11.5	19.0	1.62
SAC	43.1	58.9	27.8	36.7	18.7	28.2	1.58
BridgeComp	**50.7**	**63.7**	**37.0**	**44.8**	**27.6**	**39.6**	**0.18**

Table 2: QA performance and compression latency compared with fine-tuned aggressive compression methods on multi-hop QA benchmarks. The best and second-best results are highlighted in bold and underlined, respectively.

Table 2 compares BridgeComp with fine-tuned aggressive compression methods on multi-hop QA benchmarks. "Time" denotes the average compression time during inference. Since these baselines retain approximately 10% of document tokens, we evaluate BridgeComp at $\tau = 0.1$. They compress contexts into latent representations consumed by the generator, so we standardize both the compressor and generator backbones to LLaMA3.2-3B across all methods. BridgeComp achieves the highest EM and the lowest latency across all benchmarks. The efficiency gains stem from architectural differences:

whereas baseline aggressive compressors incur heavy computational overhead by autoregressively generating textual summaries or latent summary states, BridgeComp produces the compressed prompt via a single, non-generative latent-to-document attention pass. Consequently, BridgeComp achieves a superior efficiency-accuracy trade-off under extreme budget constraints. Comparisons against abstractive compression baselines are provided in Appendix D.2.

5.4 Performance Comparison with RL-based Prompt Compression Methods

Methods	HotpotQA		2WikiMultihopQA		MuSiQue		Time (s)
	EM	F1	EM	F1	EM	F1	
PCRL	41.1	52.4	43.2	54.9	32.6	39.6	0.67
CoT-Influx	43.3	54.5	44.8	52.3	33.2	40.0	0.89
BridgeComp	**66.4**	**80.0**	**64.3**	**72.8**	**46.3**	**60.2**	**0.18**

Table 3: QA performance and compression latency of RL-based methods on multi-hop QA benchmarks at $\tau = 0.5$. LLaMA3.2-3B and GPT-5.6 Luna are used as the compressor backbone and generator, respectively. The best results are highlighted in bold.

Table 3 compares BridgeComp with RL-based prompt compression methods[2] on multi-hop QA benchmarks at $\tau = 0.5$. BridgeComp consistently achieves the highest EM and F1 across all benchmarks while requiring the lowest compression latency. On HotpotQA, BridgeComp improves over CoT-Influx, the strongest baseline, by 53.35% EM (43.3→66.4) and 46.79% F1 (54.5→80.0). On MuSiQue, BridgeComp improves F1 by 50.50% (40.0→60.2). BridgeComp also achieves a 3.72× speedup over the fastest baseline PCRL (0.67s→0.18s). These results demonstrate that BridgeComp provides a better performance-efficiency trade-off than prior RL-based methods.

5.5 Evaluation on Long-Context Scenarios

Methods	SingleDoc	FewShot	Summarization	Code Completion	Synthetic	Avg.
DAC	34.1	55.4	25.5	32.5	49.5	39.4
LLMLingua2	34.9	58.4	<u>27.1</u>	33.7	53.5	41.5
CPC	34.4	58.5	25.2	<u>36.7</u>	<u>59.0</u>	42.8
AttnComp	<u>35.2</u>	<u>60.1</u>	25.2	35.8	58.0	<u>42.9</u>
BridgeComp	**37.9**	**63.6**	**29.1**	**38.1**	**61.0**	**45.9**

Table 4: Task-level performance on LongBench at $\tau = 0.5$. Each task score is averaged over all datasets within the corresponding task category. Avg. denotes the macro-average across task categories. The best and second-best results are highlighted in bold and underlined, respectively.

Table 4 compares prompt compression methods on LongBench at $\tau = 0.5$. Following LLMLingua2, we report task-level averages by averaging the dataset-level scores within each task category. BridgeComp achieves the best performance across all five task categories and the highest overall average. The largest gains are observed on Single-Document QA and Summarization, where BridgeComp improves over the strongest baselines by 7.67% (35.2→37.9) and 7.38% (27.1→29.1), respectively. Overall, BridgeComp improves the macro-average by 6.99% (42.9→45.9). These results demonstrate that BridgeComp generalizes beyond multi-hop QA and preserves task-relevant information across diverse long-context tasks, including few-shot learning, summarization, code completion, and synthetic tasks. Appendix E further analyzes compression latency and peak memory usage for inputs up to 40K tokens. At 40K tokens, BridgeComp achieves an 11.75× compression speedup over AttnComp and reduces peak memory usage by 39.1% relative to CPC. The complete LongBench results and a lost-in-the-middle analysis on NaturalQuestions are provided in Appendices D.3 and F.3, respectively. Evaluations on GSM8K and BBH are reported in Appendix D.4.

5.6 Ablation Study

As shown in Table 5, we conduct ablation studies for BridgeComp. *w/o* RCA replaces latent-token-to-document attention with all-token-to-document attention, and *w/o* LSA aggregates attention over all

[2]We do not include TACO-RL (Shandilya et al., 2025) in the experimental comparison because its implementation is not publicly available.

Configuration	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
Component Ablation						
w/o RCA	56.4	70.3	53.9	67.6	34.8	46.2
w/o LSA	58.9	72.1	54.6	68.1	39.6	53.1
Training Objective Ablation						
NTP	61.2	74.9	55.2	64.8	38.4	50.5
NTP + KL Distill.	63.8	77.6	59.4	69.2	42.3	54.9
NTP + GRPO	64.0	77.3	60.2	70.0	43.0	55.9
NTP + GPG	64.2	77.8	61.8	70.9	43.3	55.5
BridgeComp	**66.4**	**80.0**	**64.3**	**72.8**	**46.3**	**60.2**

Table 5: Ablation studies of BridgeComp at $\tau = 0.5$. Bold and underlined values indicate BridgeComp and the best alternative within each ablation setting, respectively.

transformer layers. Removing RCA causes the largest performance degradation, reducing EM by up to 24.84% on MuSiQue, while removing LSA also consistently degrades performance. These results demonstrate that RCA provides the primary reasoning-conditioned signal for preserving bridging information, while LSA suppresses early-layer lexical noise. Experiments applying LSA to existing attention-based compressors are provided in Appendix F.2. For training objectives, NTP + GRPO uses the GRPO objective without the anchor-relative correction ($\eta = 0$), while NTP + GPG uses only group-centered advantages. BridgeComp, i.e., NTP + anchor-conditioned GPG, achieves the best performance across all benchmarks, improving EM over NTP and NTP + GPG by up to 20.57% and 6.93%, respectively, on MuSiQue. Formulations of the GRPO and KL-distillation objectives are provided in Appendix C. An ablation study on reward components is provided in Appendix D.5. We examine how compressing the training documents affects QA performance in Appendix F.10.

5.7 Representation Alignment of Latent Tokens

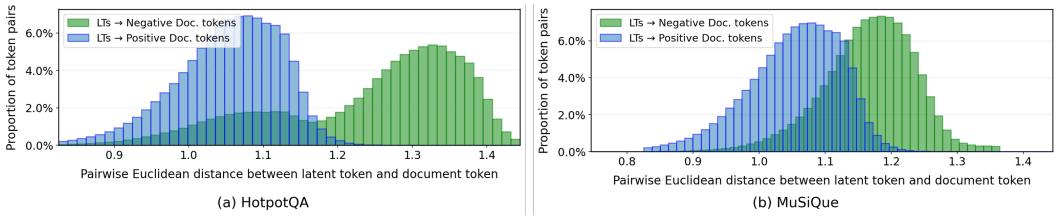


Figure 4: Token-level pairwise Euclidean distance distributions from latent tokens (LTs) to positive and negative document tokens on HotpotQA and MuSiQue. Distances are computed in the embedding space of the final hidden states.

We investigate the geometric alignment of latent tokens (LTs) within the representation space of the final hidden states of the model to verify their semantic affinity. For each test instance, we compute pairwise Euclidean distances between the hidden states of the LTs and those of both positive (ground truth) and negative document tokens. Figure 4 shows the normalized distribution of the pairwise distances. The distances between LTs and positive document tokens are significantly left-shifted toward smaller values, whereas those between LTs and negative document tokens display a right-shift toward larger intervals. This spatial proximity empirically validates the capacity of LTs to serve as high-fidelity, reasoning-conditioned signals for document-token scoring. We further analyze how latent tokens distinguish supporting from negative evidence in Appendix F.4.

6 Conclusion

In this paper, we introduce BridgeComp, a novel prompt compression framework. Through layer-selective aggregation and reasoning-conditioned attention, BridgeComp leverages upper-layer latent-to-document attention to preserve only the semantic bridging information necessary for reasoning. We optimize token selection through anchor-conditioned RL that jointly rewards answer quality and supporting-evidence preservation. Extensive empirical evaluations demonstrate that BridgeComp consistently outperforms existing methods in downstream task performance, while achieving lower compression latency through a single latent-to-document scoring pass.

AI USE STATEMENT

We used generative AI tools to generate textual summaries used as teacher-side supervision in the KL-distillation ablation (Section 5.6 and Appendix C.2; see Appendix I.2 for the prompt), and as LLM-based evaluators for auxiliary analyses of compression quality and semantic faithfulness (Appendices F.1 and F.6; see Appendices I.3–I.4 for the prompts). Generative AI tools were also used for grammatical correction and language editing of the manuscript. We did not use generative AI tools to formulate the research hypothesis, design the proposed method, or interpret the main experimental results. All AI-generated experimental artifacts and AI-assisted manuscript edits were reviewed by the authors. We take responsibility for the final content of this work.

ETHICS STATEMENT

This work does not involve human subjects, the collection of sensitive personal information, or other data requiring special ethical consideration. We do not identify any significant ethical concerns arising from the methodology or experiments presented in this work.

REPRODUCIBILITY STATEMENT

We provide detailed experimental settings and implementation information to facilitate reproducibility. The overall experimental setup is described in Section 5.1, while dataset construction and evaluation settings are documented in Appendices A.1 and A.2, and implementation details and hyperparameters are provided in Appendix A.3. Our source code is publicly available, and we plan to release the processed training data and trained model parameters to support reproduction and further research. Additional experimental details, prompts, and analysis settings are provided throughout the appendices.

REFERENCES

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. Booookscore: A systematic exploration of book-length summarization in the era of llms. In *International Conference on Learning Representations*, volume 2024, pp. 56059–56093, 2024.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, Chen Sun, Han Hou, Hui Yang, James Pan, Jianan Lou, Jiayi Mao, Jizheng Liu, Jinpeng Li, Kangyi Liu, Kenkun Liu, Rui Wang, Run Li, Tong Niu, Wenlong Zhang, Wenqi Yan, Xuanzheng Wang, Yuchen Zhang, Yi-Hsin Hung, Yuan Jiang, Zexuan Liu, Zihan Yin, Zijian Ma, and Zhiwen Mo. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations, 2025. URL https://arxiv.org/abs/2506.13651.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. xrag: extreme context compression for retrieval-augmented generation with one token. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3829–3846, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.232. URL https://aclanthology.org/2023.emnlp-main.232/.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. Gpg: A simple and strong reinforcement learning baseline for model reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 59637–59659, 2026.

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 *International Conference on Learning Representations*, volume 2024, pp. 54158–54183, 2024.

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.

Javier Ferrando, Oscar Balcells Obeso, Senthooran Rajamanoharan, and Neel Nanda. Do i know this entity? knowledge awareness and hallucinations in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WCRQFlji2q.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uREj4ZuGJE.

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

Jia-Chen Gu, Junyi Zhang, Di Wu, Yuankai Li, Kai-Wei Chang, and Nanyun Peng. BRIEF-pro: Universal context compression with short-to-long synthesis for fast and accurate multi-hop reasoning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 14221–14241, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.696. URL https://aclanthology.org/2026.findings-acl.696/.

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 (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 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/.

Xijie Huang, Li Lyna Zhang, Kwang-Ting Cheng, Fan Yang, and Mao Yang. Fewer is more: Boosting math reasoning with reinforced context pruning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13674–13695, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.758. URL https://aclanthology.org/2024.emnlp-main.758/.

Abhinav Java, Srivathsan Koundinyan, Nagarajan Natarajan, and Amit Sharma. FrugalRAG: Less is more in RL finetuning for multi-hop question answering. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=uQKtwdJN0o.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13358–13376, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.825. URL https://aclanthology.org/2023.emnlp-main.825/.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp.

1658–1677, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.91. URL https://aclanthology.org/2024.acl-long.91/.

Hoyoun Jung and Kyung-Joong Kim. Discrete prompt compression with reinforcement learning. *IEEE Access*, 12:72578–72587, 2024.

Youmin Ko, Sungjong Seo, and Hyunjoon Kim. Cooperative retrieval-augmented generation for question answering: Mutual information exchange and ranking by contrasting layers. *Advances in Neural Information Processing Systems*, 38:148795–148836, 2026.

Bruce Kuwahara, Chen-Yuan Lin, Xiao Shi Huang, Kin Kwan Leung, Jullian Yapeter, Ilya Stanevich, Felipe Perez, and Jesse Cresswell. Document summarization with conformal importance guarantees. *Advances in Neural Information Processing Systems*, 38:67107–67152, 2026.

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/.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=VKGTGGcwl6.

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. *Advances in Neural Information Processing Systems*, 38:174538–174564, 2026.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. Compressing context to enhance inference efficiency of large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6342–6353, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.391. URL https://aclanthology.org/2023.emnlp-main.391/.

Zongqian Li, Yixuan Su, and Nigel Collier. 500xCompressor: Generalized prompt compression for large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25081–25091, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1219. URL https://aclanthology.org/2025.acl-long.1219/.

Fulin Lin, Shaowen Chen, Ruishan Fang, Hongwei Wang, and Tao Lin. Stop wasting your tokens: Towards efficient runtime multi-agent systems. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=pzFhtpkabh.

Barys Liskavets, Maxim Ushakov, Shuvendu Roy, Mark Klibanov, Ali Etemad, and Shane K. Luke. Prompt compression with context-aware sentence encoding for fast and improved llm inference. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i23.34639. URL https://doi.org/10.1609/aaai.v39i23.34639.

Xin Liu, Runsong Zhao, Pengcheng Huang, Xinyu Liu, Junyi Xiao, Chunyang Xiao, Tong Xiao, Shengxiang Gao, Zhengtao Yu, and JingBo Zhu. Autoencoding-free context compression for LLMs via contextual semantic anchors. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=8Pi6Du0n7F.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *International Conference on Learning Representations*, volume 2024, pp. 9025–9049, 2024.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the dialogue: Training and evaluating user language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ykSmkVqzn4.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=KRnsX5Em3W.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 963–981, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.57. URL https://aclanthology.org/2024.findings-acl.57/.

Xianghe Pang, Shuo Tang, Rui Ye, Yuxin Xiong, Bolun Zhang, Yanfeng Wang, and Siheng Chen. Self-alignment of large language models via monopolylogue-based social scene simulation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Tianrui Qin, Qianben Chen, Sinuo Wang, He Xing, King Zhu, He Zhu, Dingfeng Shi, Xinxin Liu, Ge Zhang, Jiaheng Liu, Xitong Gao, Yuchen Eleanor Jiang, and Wangchunshu Zhou. Flashsearcher: Fast and effective web agents via DAG-based parallel execution. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=QuaJ6kJaBm.

Mohammad Mahdi Salmani-Zarchi, Zahra Rahimi, Heshaam Faili, and Mohammad Javad Dousti. MDP-GRPO: Stabilized group relative policy optimization for multi-constraint instruction following. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42784–42797, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1982. URL https://aclanthology.org/2026.acl-long.1982/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Shivam Shandilya, Menglin Xia, Supriyo Ghosh, Huiqiang Jiang, Jue Zhang, Qianhui Wu, Victor Rühle, and Saravan Rajmohan. Taco-rl: Task aware prompt compression optimization with reinforcement learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 1582–1597, 2025.

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 (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 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/.

Yuxin Shi, Han Fu, Zhuo Li, Chenghao Liu, Xiaoxue Ren, and Jianling Sun. Plan-answer-refine-on-graph: Structured planning and self-refinement for large language model reasoning on knowledge graphs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=g6XnP7Sgui.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks

and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Qwen Team. Qwen3.5-omni technical report, 2026. URL https://arxiv.org/abs/2604.15804.

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/.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mlJLVigNHp.

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 (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. CompAct: Compressing retrieved documents actively for question answering. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21424–21439, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1194. URL https://aclanthology.org/2024.emnlp-main.1194/.

Yi Zhao, Zuchao Li, Hai Zhao, Baoyuan Qi, and Liu Guoming. DAC: A dynamic attention-aware approach for task-agnostic prompt compression. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19395–19407, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.952. URL https://aclanthology.org/2025.acl-long.952/.

Yunlong Zhao, Haoran Wu, and Bo Xu. Leveraging attention to effectively compress prompts for long-context llms. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025b. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i24.34800. URL https://doi.org/10.1609/aaai.v39i24.34800.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Qihao Zhu Runxin Xu Junxiao Song Mingchuan Zhang Y.K. Li Y. Wu Daya Guo Zhihong Shao, Peiyi Wang. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

APPENDICES

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

- **Appendix A: Experimental Setup in Detail**
  - A.1: Details of Training and Evaluation Datasets for Multi-Hop QA
  - A.2: Details of Evaluation Datasets for LongBench
  - A.3: Implementation Details
- **Appendix B: Baselines**
- **Appendix C: Details of Other Training Objectives**
  - C.1: Training with GRPO Loss
  - C.2: Training with KL Distillation Loss
- **Appendix D: Overall Performance**
  - D.1: Main Results
  - D.2: Performance Comparison with Abstractive Compression Methods
  - D.3: Performance Comparison on LongBench
  - D.4: Performance Comparison on Mathematical and General Reasoning Tasks
  - D.5: Ablation Study of Reward for RL
  - D.6: Performance with Matched Compressor and Generator Backbones
  - D.7: Application to Multi-Agent Systems
- **Appendix E: Latency Analysis**
- **Appendix F: Analysis**
  - F.1: Semantic Faithfulness and Evidence Preservation Analysis
  - F.2: Impact of Layer-Selective Aggregation on Attention-Based Compressors
  - F.3: Lost-in-the-middle Analysis
  - F.4: Source-Representation Distance Analysis
  - F.5: Lexical Overlap Analysis
  - F.6: Pairwise LLM-as-a-Judge Evaluation of Compression Quality
  - F.7: Performance Comparison across Rationale Types
  - F.8: Impact of Attention Range
  - F.9: Performance Comparison Across Problem Types
  - F.10: Impact of Compressed Documents on Training
  - F.11: Performance Comparison across Compression Rates
  - F.12: Dataset Contamination Analysis
- **Appendix G: Hyperparameter Sensitivity**
  - G.1: Impact of the Number of Perturbations for Mask Sampling
  - G.2: Impact of the Anchor-Relative Weight η
  - G.3: Impact of the Number of Candidate Layers
  - G.4: Impact of the Number of Latent Tokens
  - G.5: Impact of the Number of Sampled Masks
  - G.6: Impact of LLM Backbones
  - G.7-G.15: Sensitivity to Other Hyperparameters
- **Appendix H: Case Study**
- **Appendix I: Prompt**

A Experimental Setup in Detail

A.1 Details of Training and Evaluation Datasets for Multi-Hop QA

Category	HotpotQA	2WikiMultihopQA	MuSiQue	NQ
Examples	1,000	1,000	1,000	1,000
Avg. Tokens	2,796.2	2,857.1	2,827.2	2,781.8

Table 6: Dataset statistics of the QA test sets. Avg. Tokens denotes the average number of input tokens.

Category	1-hop	2-hop	3-hop	4-hop	Total
# Examples	7,000	20,000	2,609	5,633	35,242

Table 7: Statistics of the hop-stratified mixed training set used for training BridgeComp.

Table 6 reports the statistics of the QA test sets used in our experiments, while Table 7 summarizes the hop-stratified mixed training set used to train BridgeComp. We construct the training set from subsets of NaturalQuestions (Kwiatkowski et al., 2019), HotpotQA (Yang et al., 2018), MuSiQue (Trivedi et al., 2022), and 2WikiMultihopQA (Ho et al., 2020), retaining only question–document pairs that do not appear in the corresponding evaluation test sets to prevent direct train–test contamination. The resulting training set contains 35,242 examples, consisting of 7,000 1-hop, 20,000 2-hop, 2,609 3-hop, and 5,633 4-hop examples. For all QA experiments, we use the same top-10 documents retrieved by RaLa (Ko et al., 2026) for each question. We additionally use xbench-DeepSearch (Chen et al., 2025) and GAIA (Mialon et al., 2024) exclusively for downstream multi-agent system evaluation; neither benchmark is included in the training set.

A.2 Details of Evaluation Datasets for LongBench

Task Category	Dataset	Metrics	Examples	Avg. Tokens	Max Tokens
Single-Document QA	NarrativeQA	F1	200	29,787	65,287
	Qasper	F1	200	4,930	21,118
	MultiFieldQA-en	F1	150	6,901	14,960
Few-shot Learning	TREC	Accuracy	200	6,768	11,379
	TriviaQA	F1	200	11,751	23,299
	SAMSum	ROUGE-L	200	9,148	17,973
Summarization	GovReport	ROUGE-L	200	10,240	51,392
	QMSum	ROUGE-L	200	13,867	30,389
	MultiNews	ROUGE-L	200	2,607	13,935
Synthetic Tasks	PassageCount	Accuracy	200	14,879	28,965
	PassageRetrieval-en	Accuracy	200	12,457	15,145
Code Completion	LCC	Edit Similarity	500	3,165	30,150
	RepoBench-P	Edit Similarity	500	10,805	39,124

Table 8: Statistics of the LongBench evaluation datasets. Avg. Tokens and Max Tokens denote the average and maximum number of input tokens, respectively.

Table 8 summarizes the LongBench (Bai et al., 2024) datasets used for long-context evaluation. We evaluate BridgeComp on all 13 datasets spanning five task categories: Single-Document QA, Few-shot Learning, Summarization, Synthetic, and Code Completion. Each dataset is evaluated using its task-specific metric, including F1, Accuracy, ROUGE-L, and Edit Similarity. The evaluation sets contain 150–500 examples per dataset, with average input lengths ranging from 2,607 tokens for MultiNews to 29,787 tokens for NarrativeQA. The maximum input length reaches 65,287 tokens on NarrativeQA, providing a challenging setting for evaluating prompt compression under long-context inputs.

A.3 Implementation Details

We use LLaMA3.2-3B (Grattafiori et al., 2024) as the compressor backbone and GPT-5.6 Luna (Singh et al., 2025) and Qwen3.5-9B (Team, 2026) as downstream generators. BridgeComp uses 50

Category	Hyperparameter	Setting
Model	Number of latent tokens	50
	LoRA rank r	32
	Candidate layer ratio ρ	40%
Training	Training steps	2,000
	Latent-token warm-up steps	1,000
	Per-device batch size	4
Compression	Training compression rate	0.5
	Main evaluation compression rates	0.2, 0.5
	Aggressive compression rate	0.1
Policy Rollout	Number of sampled masks n_m	4
	Number of token swaps b	5
	Policy temperature T_π	1
Reward	Attention reward weight λ_{attn}	0.2
	Negative-attention weight β_{rest}	1
	Missing-support penalty weight λ_{miss}	0.2
	Numerical stability ϵ	10^{-8}
GPG	Anchor-relative weight η	0.5
	RL loss weight λ_{RL}	0.5

Table 9: Hyperparameter settings of BridgeComp.

latent tokens (LTs), a LoRA rank of 32, the top 40% of layers for layer-selective aggregation, and a per-device batch size of 4. Unless otherwise specified, training consists of 2,000 optimization steps: a 1,000-step warm-up that optimizes only the LT embeddings using the NTP objective, followed by 1,000 steps of joint optimization of the LT embeddings and LoRA parameters using the combined NTP and AGPG objectives. The compressor backbone remains frozen throughout training. The impact of the number of joint-optimization steps is analyzed in Appendix G.7. Training takes approximately one hour and 40 minutes on four NVIDIA RTX A6000 GPUs. During compression inference, BridgeComp uses the trained LTs and LoRA adapters, whereas the downstream generator receives only the resulting compressed hard prompt without reusing these components, enabling plug-and-play generation. The LLaMA3.2-3B compressor requires 7.20 GiB peak memory during compression inference and can operate on a single GPU.

For the main QA evaluation (Section 5.2), methods are compared using identical retrieved documents, target compression rates ($\tau = 0.2, 0.5$), and downstream generators without additional reader tuning. Results are averaged over five runs with different random seeds, with standard deviations reported in Appendix D.1. Original Prompt and Zero-shot use the full retrieved context and no retrieved documents, respectively. For each ablation setting in Section 5.6, we rerun the complete compressor training procedure, keeping the component configuration consistent between training and compression inference.

For the comparison with fine-tuned aggressive compression methods (Section 5.3), all methods use LLaMA3.2-3B as both the compressor and generator backbone, with the same retrieved documents, QA training data, and answer supervision. BridgeComp uses $\tau = 0.1$, whereas all compared baselines except xRAG and 500xCompressor—including ICAE with 256 memory tokens—use compressed-context budgets exceeding 10% of the input document-token count, favoring these baselines in terms of token budget. We follow xRAG's original setting of one soft token per document. AutoCompressors and 500xCompressor use full compressor fine-tuning. For 500xCompressor, increasing the number of compression tokens yields diminishing performance gains but sharply increases memory usage, with larger settings causing out-of-memory errors; we therefore use the maximum feasible token count under a comparable memory budget. Generation-based baselines incur higher compression latency. For the separate plug-and-play comparison with abstractive compression methods (Appendix D.2), we use LLaMA3.2-3B as the compressor backbone and GPT-5.6 Luna as the downstream generator without additional reader tuning.

B BASELINES

We compare BridgeComp with representative prompt compression baselines. These baselines can be grouped into four main categories: perplexity-based methods, attention-based methods, fine-

tuned aggressive compression methods and RL-based compression methods. We retrain baselines requiring fine-tuning, particularly aggressive compression and RL-based methods, using the same LLM backbone and training dataset as BridgeComp. All remaining training and inference settings follow the model-specific configurations provided in the respective official implementations.

Perplexity-based Prompt Compression Methods

- **Selective-Context** (Li et al., 2023): Selective-Context removes low-information lexical units based on entropy-based salience, thereby reducing the prompt length while retaining informative content.
- **LLMLingua** (Jiang et al., 2023): LLMLingua introduces budget control and iterative token-level compression based on perplexity, allowing the prompt to be compressed under a target token budget.
- **LLMLingua2** (Pan et al., 2024): LLMLingua2 trains a distilled classifier specialized for task-agnostic prompt compression, improving the efficiency and transferability of token selection.
- **LongLLMLingua** (Jiang et al., 2024): LongLLMLingua extends LLMLingua to long-context settings through question-aware coarse-to-fine compression, which first identifies relevant coarse contexts and then performs finer token-level compression.
- **CPC** (Liskavets et al., 2025): CPC trains a context-aware sentence encoder on a synthetically generated question-relevance dataset, and performs sentence-level compression by scoring the relevance of each sentence to the question within its context.

Attention-based Prompt Compression Methods

- **DAC** (Zhao et al., 2025a): DAC estimates token importance by combining accumulated attention and information entropy, and applies a dynamic compression procedure that updates token scores during compression.
- **AttnComp** (Zhao et al., 2025b): AttnComp constructs semantic units from document self-attention graphs and additionally incorporates limited question-document attention to guide compression.

Fine-tuned Aggressive Compression Methods

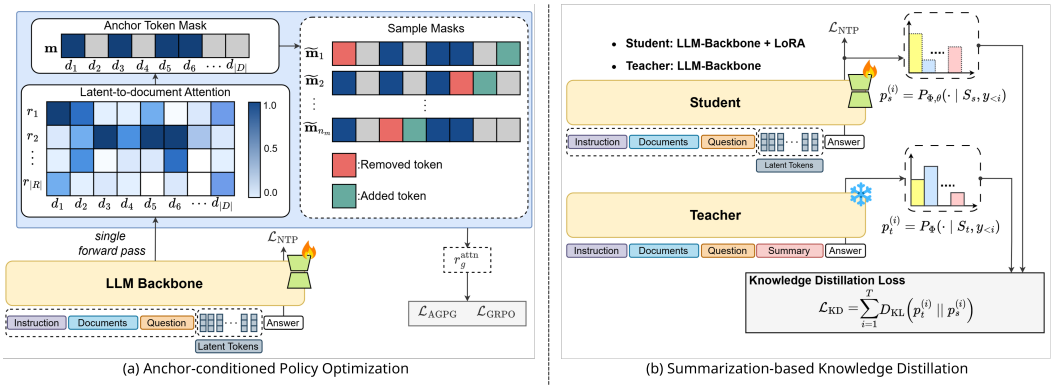
- **AutoCompressors** (Chevalier et al., 2023): AutoCompressors recursively compresses long document segments into latent summary vectors, which are accumulated and used as soft prompts for subsequent segments.
- **xRAG** (Cheng et al., 2024): xRAG compresses each retrieved document into a single soft token by mapping its dense retrieval embedding into the LLM representation space with a trained projector.
- **ICAE** (Ge et al., 2024): ICAE compresses a long context into a small set of soft memory slots, using the LLM with a LoRA adapter as the encoder, which is trained to reconstruct the original context from the slots.
- **500xCompressor** (Li et al., 2025): 500xCompressor compresses long contexts into a small set of special compressed tokens and stores contextual information in their hidden states and key-value caches.
- **RECOMP** (Xu et al., 2024): RECOMP follows a retrieve-compress-prepend framework, where retrieved documents are compressed into query-focused textual contexts through sentence extraction or abstractive summarization.
- **CompAct** (Yoon et al., 2024): CompAct performs iterative hard-prompt compression by repeatedly updating a query-relevant compressed summary over document segments until sufficient evidence is obtained.
- **BRIEF-Pro** (Gu et al., 2026): BRIEF-Pro uses a trained compressor to generate query-relevant textual summaries from retrieved documents, producing a compact hard prompt for the reader model.

- **SAC** (Liu et al., 2026): SAC selects existing context tokens as anchors, adds a shared learnable anchor embedding to their original token embeddings, and uses bidirectional attention to compress contextual information into their key–value representations without autoencoding pretraining.

RL-based Compression Methods

- **TACO-RL** (Shandilya et al., 2025): TACO-RL uses reinforcement learning to fine-tune an encoder-based token classifier for task-aware compression, with task-specific rewards measuring the consistency of downstream task results between original and compressed prompts.
- **PCRL** (Jung & Kim, 2024): PCRL trains a token-level MLP classifier on top of a frozen encoder with reinforcement learning for adaptive compression, rewarding greater compression while preserving downstream task results.
- **CoT-Influx** (Huang et al., 2024): CoT-Influx uses reinforcement learning for coarse-to-fine few-shot CoT compression, selecting useful demonstrations and pruning redundant tokens based on downstream performance rewards.

C Details of Other Training Objectives



(a) Anchor-conditioned Policy Optimization (b) Summarization-based Knowledge Distillation

Figure 5: Illustration of BridgeComp training with (a) RL-based exact-budget local exploration and (b) the KL distillation objective. In (a), from a single forward pass, BridgeComp computes latent-to-document attention, constructs a deterministic anchor mask, and generates multiple sampled rollout masks around the anchor for policy optimization.

C.1 Training with GRPO Loss

Figure 5(a) illustrates BridgeComp training with GRPO-based exact-budget local exploration. We use the rollout construction and reward function defined in Section 4.5.2. For each input, we construct a deterministic anchor mask $\mathbf{m}$ and generate n_m sampled masks $\widetilde{\mathbf{m}}_g$. Each sampled mask is obtained by applying one swap action $a_g = \left(S_g^-,\ S_g^+\right)$ to the same anchor, where S_g^- contains exactly b distinct indices selected by the anchor and S_g^+ contains exactly b distinct indices excluded by it. The policy selects which token indices to remove and add; the operation types and their counts are fixed.

We use the sampling record $\xi_g = \left(\boldsymbol{s}_g^-,\ \boldsymbol{s}_g^+\right)$ and the token-selection distributions P_θ^- and P_θ^+ defined in Section 4.5.2. For compact notation, let $o \in \{-,\ +\}$ index removal and addition, respectively. For $v \in \{1,\ \ldots,\ b\}$, $s_{g,v}^o$ is the token index chosen at the v-th selection in the corresponding pool, and $\boldsymbol{s}_{g,<v}^o$ records the preceding selections in that pool. We define the conditioning context as

$$c_{g,v}^o = \left(S,\ m,\ s_{g,<v}^o\right) \tag{11}$$

Each distribution excludes indices already selected within its own pool and renormalizes over the remaining candidates. Thus, although the logits remain fixed within a rollout, the selection probabilities depend on the same-pool history through sampling without replacement. Selections in one pool do not change the candidates in the other.

Standard GRPO (Zhihong Shao, 2024) optimizes a PPO-style (Schulman et al., 2017) clipped surrogate objective with a KL-divergence penalty relative to a fixed reference policy. To adapt this objective to our token-selection process, let q_θ^o denote the current token-selection distribution and $q_{\theta_{\text{old}}}^o$ the corresponding distribution used to collect the rollouts. Their probability ratio for the observed selection is

$$\omega_{g,v}^o(\theta) = \frac{q_\theta^o\left(s_{g,v}^o \mid c_{g,v}^o\right)}{q_{\theta_{\text{old}}}^o\left(s_{g,v}^o \mid c_{g,v}^o\right)}. \tag{12}$$

During the policy update, the anchor mask and sampled indices are held fixed. The numerator and denominator are evaluated using the same anchor and recorded history, and therefore the same remaining candidates. The rollout-policy probabilities in the denominator are treated as constants.

Let A_g^{GRPO} denote the group-normalized advantage assigned to the sampled mask $\widetilde{\mathbf{m}}_g$, ϵ_{clip} the clipping threshold, and $\text{sg}(\cdot)$ the stop-gradient operator. Let q_{ref}^o denote the token-selection distribution induced by a fixed reference policy over the same remaining candidates, and let β_{KL} control the KL penalty. The full objective adapted to our sampling process is

$$\mathcal{L}_{\text{GRPO}}^{\text{full}} = -\frac{1}{n_m}\sum_{g=1}^{n_m}\frac{1}{2b}\sum_{o\in\{-,+\}}\sum_{v=1}^{b}\Bigg[\min\Big(\omega_{g,v}^o(\theta)\,\text{sg}\big(A_g^{\text{GRPO}}\big),\ \text{clip}\big(\omega_{g,v}^o(\theta),1-\epsilon_{\text{clip}},1+\epsilon_{\text{clip}}\big)\,\text{sg}\big(A_g^{\text{GRPO}}\big)\Big) - \beta_{\text{KL}}D_{\text{KL}}\Big(q_\theta^o(\cdot \mid c_{g,v}^o)\,\|\,q_{\text{ref}}^o(\cdot \mid c_{g,v}^o)\Big)\Bigg]. \tag{13}$$

For the GRPO ablation, we retain the exact-budget local exploration and the clipped surrogate objective, but set $\beta_{\text{KL}} = 0$. Since our GPG and AGPG objectives omit reference-policy regularization, this setting avoids conflating differences in policy-update rules with the effects of an additional KL penalty. Accordingly, the GRPO loss used in this ablation is

$$\mathcal{L}_{\text{GRPO}} = -\frac{1}{n_m}\sum_{g=1}^{n_m}\frac{1}{2b}\sum_{o\in\{-,+\}}\sum_{v=1}^{b}\min\Big(\omega_{g,v}^o(\theta)\,\text{sg}\big(A_g^{\text{GRPO}}\big),\ \text{clip}\big(\omega_{g,v}^o(\theta),1-\epsilon_{\text{clip}},1+\epsilon_{\text{clip}}\big)\,\text{sg}\big(A_g^{\text{GRPO}}\big)\Big). \tag{14}$$

The probability ratio and clipping are evaluated separately for each token selection, while the same rollout-level advantage is shared across all $2b$ selections. These selections constitute one swap action and receive an advantage determined by its resulting mask; they are not separate swap actions. The factor $\frac{1}{2b}$ averages their contributions within each rollout.

Finally, as in AGPG, we combine the GRPO policy loss with the next-token prediction objective. After the 1,000-step warm-up stage, the training objective is

$$\mathcal{L} = \mathcal{L}_{\text{NTP}} + \lambda_{\text{RL}}\mathcal{L}_{\text{GRPO}}, \tag{15}$$

where λ_{RL} controls the contribution of the GRPO policy loss.

C.2 Training with KL Distillation Loss

As an alternative to reinforcement learning, we train BridgeComp using rationale-conditioned knowledge distillation, following CODI (Shen et al., 2025). Figure 5(b) illustrates the overall training procedure. Both the teacher and student are constructed from the compressor backbone Φ. The teacher uses the frozen compressor backbone without LoRA or latent tokens, whereas the student augments the same backbone with the trainable LoRA parameters and latent-token embeddings.

Let Z denote a textual summary that preserves the evidence required to answer the question. We construct the teacher and student contexts as

$$S_t = (I,\ D,\ Q,\ Z)\,,\ S_s = (I,\ D,\ Q,\ R)\,, \tag{16}$$

where the teacher receives the explicit textual summary Z, while the student receives the learnable latent-token sequence R. The summaries are generated offline using GPT-5.6 Sol, with the summarization prompt provided in Appendix I.2. They are used only as teacher-side supervision during training and are not required during compression inference.

Let $Y = (y_1, \ldots, y_T)$ denote the gold answer sequence, where $y_{<i} = (y_1, \ldots, y_{i-1})$ is the gold answer prefix preceding the i-th answer token. At each answer position, we define the teacher and student next-token distributions as

$$p_t^{(i)} = P_\Phi\left(\cdot \mid S_t, y_{<i}\right),\ p_s^{(i)} = P_{\Phi,\theta}\left(\cdot \mid S_s, y_{<i}\right), \tag{17}$$

where $\theta = \{\theta_R,\ \theta_{\mathrm{LoRA}}\}$ denotes the trainable latent-token embeddings and LoRA parameters of the student. We apply knowledge distillation only at the gold-answer token positions:

$$\mathcal{L}_{\mathrm{KD}} = \sum_{i=1}^{T} D_{\mathrm{KL}}\left(p_t^{(i)} \parallel p_s^{(i)}\right). \tag{18}$$

The KL divergence is not computed over instruction, document, question, summary, or latent-token positions. Instead, it aligns the teacher and student output distributions only when predicting the gold answer. Through this objective, the student learns to reproduce the answer-generation behavior induced by the explicit summary using only the latent tokens R. This distillation objective is applied entirely to the compressor during training and does not introduce any dependency on the downstream generator.

Finally, we combine the knowledge-distillation objective with the next-token prediction loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{NTP}} + \lambda_{\mathrm{KD}}\mathcal{L}_{\mathrm{KD}}, \tag{19}$$

where λ_{KD} controls the contribution of the knowledge-distillation loss.

D Overall Performance

D.1 Main Results

Methods	GPT-5.6 Luna						Qwen3.5-9B					
	HotpotQA		2WikiMultihopQA		MuSiQue		HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Compression rate $\tau = 0.5$												
Selective-Context	40.6±1.2	51.9±1.5	44.0±0.7	50.1±1.2	32.3±0.7	41.8±1.4	36.7±1.9	46.7±1.2	38.7±1.1	44.5±1.0	26.7±0.8	33.0±0.8
LLMLingua	42.5±0.9	53.9±1.1	43.5±1.1	49.9±1.2	33.6±0.7	39.5±0.8	38.4±1.4	48.5±1.5	38.2±1.1	44.3±1.5	27.8±0.8	31.2±0.9
DAC	45.1±1.4	57.1±1.7	49.0±1.1	55.3±1.1	31.8±0.7	43.1±1.0	40.8±1.4	51.4±2.0	43.0±1.1	49.1±0.8	26.3±0.7	34.1±1.1
LLMLingua2	45.9±1.4	58.1±1.5	46.8±1.4	53.1±1.4	38.9±0.9	51.0±1.5	41.5±1.1	52.3±2.3	41.1±1.5	47.1±1.0	32.0±0.8	40.3±1.0
LongLLMLingua	53.0±1.2	63.4±1.7	49.4±0.9	60.9±1.5	36.8±0.9	47.3±1.3	48.0±1.3	57.0±2.0	43.4±1.1	54.1±1.2	30.5±1.1	37.4±1.0
CPC	55.1±1.5	65.9±1.7	50.6±1.6	62.8±1.7	38.0±1.2	49.5±1.1	49.6±2.3	59.6±2.5	44.9±1.6	57.7±1.1	31.3±1.3	39.6±0.7
AttnComp	60.7±1.5	75.4±1.6	61.5±2.0	69.4±1.4	38.7±0.7	54.4±1.7	54.9±2.0	67.8±2.6	54.1±0.7	61.6±1.3	32.2±1.3	43.0±0.7
BridgeComp	**66.4±0.7**	**80.0±0.8**	**64.3±0.7**	**72.8±0.8**	**46.3±0.5**	**60.2±0.7**	**60.1±1.1**	**72.0±1.7**	**56.5±0.7**	**64.6±1.0**	**38.3±1.1**	**47.6±1.0**
Compression rate $\tau = 0.2$												
Selective-Context	30.7±0.7	39.8±1.0	32.7±0.9	35.6±1.0	17.2±0.9	28.3±0.7	24.3±2.3	32.1±1.6	26.4±1.0	28.2±1.7	13.4±1.1	20.5±1.3
LLMLingua	33.4±0.7	41.9±0.9	34.0±0.7	36.9±0.9	21.8±0.7	30.3±0.7	26.4±2.0	33.9±1.7	27.5±0.8	29.2±1.3	17.0±0.5	21.9±0.7
DAC	33.2±0.7	40.5±1.0	34.4±0.7	39.5±1.2	21.9±0.8	29.4±0.8	26.3±1.9	32.7±1.4	27.8±0.8	31.3±1.1	17.0±0.5	21.2±0.5
LLMLingua2	35.7±0.9	47.3±1.2	40.9±1.3	43.8±1.3	24.2±0.7	40.2±1.0	28.2±1.6	38.2±0.9	33.0±0.9	34.7±1.4	18.9±0.8	29.0±0.9
LongLLMLingua	51.4±1.5	59.7±1.2	46.4±1.1	51.7±0.9	27.0±0.9	43.7±1.1	40.6±1.7	48.2±0.5	37.5±0.9	41.0±1.2	21.0±0.5	31.6±0.9
CPC	54.1±1.1	63.7±1.9	48.1±0.9	55.2±1.3	28.3±0.7	45.5±1.0	42.4±1.2	50.6±1.6	39.5±0.7	43.3±1.2	22.1±0.5	33.0±1.3
AttnComp	58.8±1.0	71.2±1.7	55.6±1.1	63.2±1.4	33.8±0.9	47.7±1.3	46.5±1.0	57.5±1.6	44.9±0.9	50.1±1.1	26.3±0.8	34.5±0.7
BridgeComp	**64.9±0.7**	**78.5±0.8**	**58.3±0.9**	**67.6±0.8**	**41.9±0.5**	**56.7±0.5**	**51.3±0.8**	**63.4±0.7**	**47.1±0.6**	**53.5±1.0**	**32.6±0.7**	**41.0±1.0**
Original Prompt	65.2±1.6	79.9±1.9	64.0±2.0	72.0±1.9	47.1±1.7	61.0±1.9	63.2±2.4	76.0±2.3	58.7±1.9	66.7±2.7	40.8±0.8	53.2±2.3
Zero-shot	26.8±2.5	39.7±3.0	28.6±2.8	37.8±2.4	7.4±2.0	19.1±1.9	21.8±0.8	29.1±1.0	28.1±0.8	33.2±1.4	4.3±0.6	12.0±0.4

Table 10: Complete QA performance on multi-hop QA benchmarks under two compression rates ($\tau = 0.2$ and $\tau = 0.5$) with GPT-5.6 Luna and Qwen3.5-9B generators. We report the mean ± standard deviation over five runs with different random seeds. The best and second-best results among the compression methods are highlighted in bold and underlined, respectively.

Table 10 reports the complete QA performance on HotpotQA, 2WikiMultihopQA, and MuSiQue under two compression rates, $\tau = 0.2$ and $\tau = 0.5$, using GPT-5.6 Luna and Qwen3.5-9B as

downstream generators. Original Prompt and Zero-shot are included as reference settings, and we report the mean ± standard deviation over five runs with different random seeds. Across all benchmarks, compression rates, and generators, BridgeComp consistently achieves the highest EM and F1 among the compared compression methods. This consistent advantage indicates that the compressed contexts produced by BridgeComp preserve information that remains useful across different downstream LLMs.

The gains are most pronounced on MuSiQue. At $\tau = 0.5$, BridgeComp improves EM over AttnComp by 19.64% (38.7→46.3) with GPT-5.6 Luna and 18.94% (32.2→38.3) with Qwen3.5-9B, while F1 improves by 10.66% (54.4→60.2) and 10.70% (43.0→47.6), respectively. Under the more aggressive $\tau = 0.2$ setting, the EM gains further increase to 23.96% (33.8→41.9) and 23.95% (26.3→32.6). These results show that BridgeComp robustly preserves bridging information even under stringent compression budgets. The consistent improvements across both generators further support its plug-and-play applicability without requiring the compressor and generator to share the same backbone.

D.2 Performance Comparison with Abstractive Compression Methods

Methods	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
Abstractive Compression Methods						
RECOMP	33.9	40.1	20.2	37.8	12.3	21.9
CompAct	46.8	60.5	31.6	46.1	15.5	27.4
BRIEF-Pro	<u>52.8</u>	<u>64.9</u>	<u>42.5</u>	<u>57.9</u>	<u>21.3</u>	<u>36.7</u>
BridgeComp	**56.9**	**70.3**	**51.5**	**60.4**	**34.6**	**50.3**

Table 11: QA performance (EM and F1) of BridgeComp and abstractive compression methods on multi-hop QA benchmarks using GPT-5.6 Luna as the generator ($\tau = 0.1$). The best and second-best results are highlighted in bold and underlined, respectively.

Table 11 further compares BridgeComp with abstractive compression methods using GPT-5.6 Luna as the generator. BridgeComp consistently achieves the highest EM and F1 across all three multi-hop QA benchmarks. In particular, on MuSiQue, BridgeComp improves EM over BRIEF-Pro, the strongest baseline, by 62.44% (21.3→34.6) and F1 by 37.06% (36.7→50.3). The consistent improvements across HotpotQA, 2WikiMultihopQA, and MuSiQue demonstrate that BridgeComp effectively preserves information required for multi-hop reasoning even when paired with a stronger generator.

D.3 Performance Comparison on LongBench

Methods	Single-Doc QA			Few-shot Learning		
	Nar.QA (F1)	QASPER (F1)	Mul.QA (F1)	TREC (Acc.)	Tri-QA (F1)	SAMSum (ROUGE-L)
Compression rate $\tau = 0.5$						
DAC	27.2	36.8	38.4	43.2	81.5	41.4
LLMLingua2	<u>28.3</u>	37.0	39.4	45.3	87.6	<u>42.3</u>
LongLLMLingua	26.3	34.4	37.2	45.6	82.4	41.3
CPC	27.3	36.6	39.2	46.7	87.3	41.6
AttnComp	27.7	<u>37.7</u>	<u>40.2</u>	<u>47.8</u>	<u>90.3</u>	42.2
BridgeComp	**29.3**	**40.3**	**44.1**	**50.3**	**95.7**	**44.7**
Compression rate $\tau = 0.2$						
DAC	18.4	28.0	31.3	<u>33.0</u>	79.9	23.4
LLMLingua2	19.8	32.2	25.7	24.0	76.6	35.1
LongLLMLingua	20.3	33.1	27.0	26.3	77.0	35.3
CPC	22.3	34.5	32.7	27.6	79.3	36.4
AttnComp	<u>24.0</u>	<u>35.2</u>	<u>35.4</u>	30.8	<u>82.2</u>	<u>37.7</u>
BridgeComp	**28.6**	**39.3**	**41.1**	**35.2**	**89.7**	**41.8**

Table 12: Dataset-level performance on LongBench for single-document QA and few-shot learning under two compression rates ($\tau = 0.2$ and $\tau = 0.5$). Evaluation metrics are shown in parentheses. The best and second-best results are highlighted in bold and underlined, respectively.

Methods	Summarization			Code		Synthetic	
	GovReport (ROUGE-L)	QMSum (ROUGE-L)	MultiNews (ROUGE-L)	LCC (Edit Sim.)	RepoBench (Edit Sim.)	PassageCnt. (Acc.)	PassageRet. (Acc.)
Compression rate $\tau = 0.5$							
DAC	26.8	24.6	25.0	34.8	30.1	8.0	91.0
LLMLingua2	27.5	27.4	26.4	35.0	32.3	13.0	94.0
LongLLMLingua	27.0	25.6	25.4	38.9	34.2	12.0	96.0
CPC	24.3	25.4	26.0	38.8	34.5	18.0	**100.0**
AttnComp	23.7	25.4	26.5	37.9	33.6	16.0	**100.0**
BridgeComp	**28.6**	**29.5**	**29.1**	**40.6**	**35.6**	**22.0**	**100.0**
Compression rate $\tau = 0.2$							
DAC	25.6	23.9	26.1	30.4	23.9	6.0	88.0
LLMLingua2	22.2	25.3	25.6	31.5	24.3	6.0	90.0
LongLLMLingua	23.0	25.2	25.8	35.6	29.7	10.0	96.0
CPC	24.2	25.6	26.0	36.1	31.2	14.0	**100.0**
AttnComp	25.0	26.0	26.3	34.3	30.8	15.0	**100.0**
BridgeComp	**26.8**	**27.3**	**28.1**	**38.9**	**33.4**	**16.0**	**100.0**

Table 13: Dataset-level performance on LongBench for summarization, code completion, and synthetic tasks under two compression rates ($\tau = 0.2$ and $\tau = 0.5$). Evaluation metrics are shown in parentheses. The best and second-best results are highlighted in bold and underlined, respectively.

Tables 12 and 13 report the complete dataset-level results on LongBench under two compression rates. BridgeComp achieves the best or tied-best performance across all 13 datasets at both $\tau = 0.5$ and $\tau = 0.2$. At $\tau = 0.5$, the largest improvement is observed on PassageCount, where BridgeComp improves accuracy over the strongest baseline by 22.22% (18.0→22.0). Substantial gains are also observed on MultiNews and MultiFieldQA, with improvements of 9.81% (26.5→29.1) and 9.70% (40.2→44.1), respectively. Under the more aggressive $\tau = 0.2$ setting, BridgeComp maintains the same overall trend, including gains of 19.17% (24.0→28.6) on NarrativeQA and 16.10% (35.4→41.1) on MultiFieldQA. These results show that the task-level improvements reported in the main results are consistently supported by dataset-level gains across diverse long-context tasks.

D.4 Performance Comparison on Mathematical and General Reasoning Tasks

Methods	Compression rate $\tau = 0.5$		Compression rate $\tau = 0.2$	
	GSM8K	BBH	GSM8K	BBH
DAC	70.5	52.9	61.7	49.9
LLMLingua2	68.6	51.8	62.6	47.7
LongLLMLingua	71.2	50.3	62.5	49.3
CPC	72.6	54.2	63.7	56.8
AttnComp	74.9	55.1	65.1	60.2
BridgeComp	**78.2**	**59.6**	**68.6**	**64.7**

Table 14: EM performance on reasoning benchmarks, GSM8K and BBH, under two compression rates ($\tau = 0.2$ and $\tau = 0.5$). The best and second-best results are highlighted in bold and underlined, respectively.

Table 14 compares BridgeComp with prompt compression baselines on GSM8K (Cobbe et al., 2021) and BBH (Suzgun et al., 2023) under two compression rates. At $\tau = 0.5$, BridgeComp achieves the highest EM on both benchmarks, improving over AttnComp, the strongest baseline, by 4.41% (74.9→78.2) on GSM8K and 8.17% (55.1→59.6) on BBH. A consistent trend is also observed at the more aggressive compression rate of $\tau = 0.2$, where BridgeComp again outperforms all baselines on both datasets. These results indicate that BridgeComp effectively preserves information required for reasoning under different compression budgets.

D.5 Ablation Study of Reward for RL

We conduct an ablation study to evaluate the contribution of each component in the compression reward. Table 15 compares the complete reward with variants that remove the answer-level F1 reward r_{F1}, latent-attention reward r_{attn}, or supporting-document miss penalty p_{miss}. Removing r_{attn} yields the largest degradation, reducing F1 on 2WikiMultihopQA by 5.08% (72.8→69.1) and EM on HotpotQA by 4.82% (66.4→63.2). This indicates that latent-attention preservation provides

Methods	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
BridgeComp	**66.4**	**80.0**	**64.3**	**72.8**	**46.3**	**60.2**
- *w/o* r_{F1}	64.8	78.4	62.9	70.2	45.2	58.5
- *w/o* r_{attn}	63.2	77.3	61.8	69.1	44.1	58.0
- *w/o* p_{miss}	65.3	79.3	63.8	71.7	46.0	59.6

Table 15: Ablation study of the reward components used for RL training.

the primary reward signal for retaining reasoning-critical evidence during compression. Removing r_{F1} also consistently degrades performance, while p_{miss} provides a complementary safeguard against completely removing supporting documents.

D.6 Performance with Matched Compressor and Generator Backbones

Methods	LLaMA3.2-3B						Qwen3.5-9B					
	HotpotQA		2WikiMultihopQA		MuSiQue		HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Compression rate $\tau = 0.5$												
Selective-Context	32.0	42.2	26.2	31.9	19.2	26.9	39.2	49.8	41.0	51.1	28.5	36.5
LLMLingua	33.7	44.1	26.0	31.7	19.9	25.6	41.2	51.9	40.6	50.8	29.6	34.7
DAC	35.8	46.1	29.3	35.2	18.9	27.5	44.3	54.6	46.8	56.3	29.1	37.5
LLMLingua2	42.8	47.7	31.9	33.6	22.6	31.8	52.1	56.1	49.9	54.0	33.8	43.6
LongLLMLingua	44.4	51.2	33.0	38.6	23.0	32.7	54.4	60.6	51.7	61.9	33.6	43.3
CPC	45.6	53.7	33.9	40.3	23.7	34.0	55.9	62.8	53.3	65.2	34.1	44.1
AttnComp	48.5	61.6	35.4	40.7	24.2	34.4	59.2	72.7	56.1	67.1	35.2	45.4
BridgeComp	**53.1**	**65.6**	**38.7**	**47.0**	**27.8**	**39.6**	**64.6**	**77.2**	**62.2**	**75.9**	**41.1**	**53.6**
w/ LoRA	53.9	67.5	40.9	48.5	30.0	42.5	66.1	80.0	64.9	78.8	44.1	57.5
w/ LoRA + LTs	55.2	70.2	44.3	51.7	33.2	46.8	68.1	83.7	69.3	83.3	48.3	63.2
Compression rate $\tau = 0.2$												
Selective-Context	24.0	31.9	18.8	23.5	10.8	18.7	29.1	37.6	27.7	34.2	14.4	23.9
LLMLingua	26.3	33.5	19.3	24.4	13.6	19.9	31.8	39.6	28.7	35.4	18.2	25.4
DAC	26.1	32.7	19.8	26.4	13.7	19.5	31.2	38.2	29.1	36.2	18.3	24.8
LLMLingua2	28.1	38.2	23.4	29.3	15.2	26.6	33.9	44.8	34.2	42.1	20.1	33.7
LongLLMLingua	40.3	48.1	26.5	34.7	17.0	29.0	51.2	60.7	41.2	52.2	24.2	37.7
CPC	41.9	49.6	27.2	37.1	17.6	31.0	52.5	63.5	43.2	55.5	25.3	39.0
AttnComp	44.8	56.6	31.3	41.4	20.7	30.9	54.6	66.6	44.7	60.2	26.8	39.5
BridgeComp	**50.7**	**63.7**	**37.0**	**48.0**	**27.6**	**39.5**	**59.9**	**73.3**	**53.5**	**68.0**	**35.9**	**49.2**
w/ LoRA	51.7	66.0	39.6	49.9	30.2	42.9	61.8	76.7	56.8	71.5	39.5	53.9
w/ LoRA + LTs	53.3	69.3	43.7	53.7	34.1	48.0	64.3	81.2	62.1	76.9	44.5	60.7
Original Prompt	49.9	63.3	36.3	43.7	28.9	39.6	63.2	76.0	58.7	66.7	40.8	53.2
Zero-shot	15.0	20.2	17.6	22.1	5.4	11.1	21.8	29.1	28.1	33.2	4.3	12.0

Table 16: QA performance with matched compressor and generator backbones on multi-hop QA benchmarks under two compression rates ($\tau = 0.2$ and $\tau = 0.5$). "w/ LoRA" and "w/ LoRA + LTs" denote generation using the trained LoRA adapters and using both the LoRA adapters and latent tokens, respectively. The best and second-best results among compression methods, excluding the LoRA variants, are highlighted in bold and underlined, respectively.

Table 16 presents QA performance when the compressor and generator use the same backbone, evaluated with LLaMA3.2-3B and Qwen3.5-9B under two compression rates ($\tau = 0.2$ and $\tau = 0.5$). Here, BridgeComp denotes the standard plug-and-play setting, where the compressed prompt is passed directly to the backbone LLM for answer generation. "*w/* LoRA" enables the trained LoRA adapters during generation after BridgeComp compression, while "*w/* LoRA + LTs" additionally incorporates the learned latent tokens (LTs) during generation. BridgeComp consistently outperforms all competing compression methods across both matched backbones, datasets, and compression rates. The gains are most pronounced on MuSiQue under aggressive compression ($\tau = 0.2$), where BridgeComp improves EM over AttnComp, the strongest baseline, by 33.33% (20.7→27.6) with LLaMA3.2-3B and 33.96% (26.8→35.9) with Qwen3.5-9B. These results show that the advantage

of BridgeComp is maintained even when the same backbone is used for both compression and generation.

Enabling the generation-side components further improves performance. Compared with the standard BridgeComp setting, using LoRA (*w/* LoRA) consistently increases EM and F1, while using LoRA and latent tokens (*w/* LoRA + LTs) yields the strongest overall results. For example, on MuSiQue at $\tau = 0.2$, using LoRA (*w/* LoRA) improves EM by 9.42% (27.6→30.2) with LLaMA3.2-3B and by 10.03% (35.9→39.5) with Qwen3.5-9B. Using LoRA and latent tokens (*w/* LoRA + LTs) further improves EM over the standard BridgeComp setting by 23.55% (27.6→34.1) and 23.96% (35.9→44.5), respectively. These results indicate that BridgeComp already provides effective compression in the plug-and-play setting, while the learned LoRA adapters and latent tokens can be further exploited during generation when the compressor and generator share the same backbone.

D.7 Application to Multi-Agent Systems

Methods	DeepSearch	GAIA
Flash-Searcher	64.6	59.6
Flash-Searcher (+BridgeComp)	66.7	62.1
SupervisorAgent	58.5	50.9
SupervisorAgent (+BridgeComp)	60.3	52.6

Table 17: Pass@1 performance of multi-agent systems on DeepSearch and GAIA. BridgeComp replaces the original summarization module in Flash-Searcher and SupervisorAgent.

Table 17 evaluates the applicability of BridgeComp to multi-agent systems on DeepSearch (Chen et al., 2025) and GAIA (Mialon et al., 2024). We replace the original summarization module in Flash-Searcher (Qin et al., 2026) and SupervisorAgent (Lin et al., 2026) with BridgeComp while keeping the remaining agent pipeline unchanged. Incorporating BridgeComp consistently improves Pass@1 across both systems and benchmarks. In particular, Flash-Searcher improves from 64.6 to 66.7 on DeepSearch and from 59.6 to 62.1 on GAIA, while SupervisorAgent improves from 58.5 to 60.3 and from 50.9 to 52.6, respectively. These results demonstrate that BridgeComp can serve as a plug-and-play compression module beyond standalone QA pipelines and effectively preserve information exchanged within multi-agent reasoning systems.

E Latency Analysis

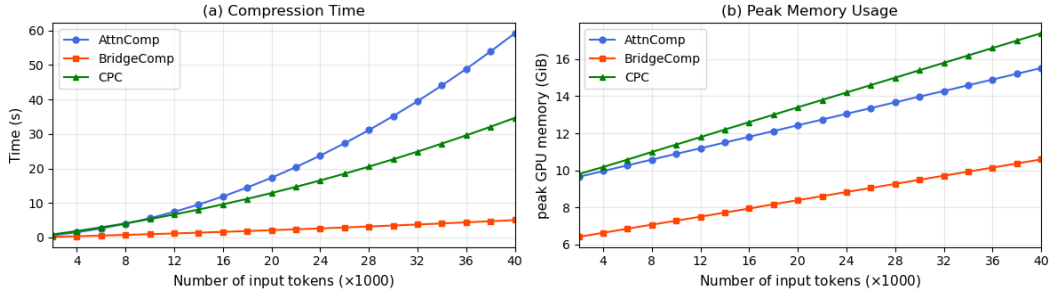


Figure 6: Inference-time compression latency and peak memory usage across different input lengths. (a) Compression time and (b) peak memory usage of AttnComp, CPC, and BridgeComp for inputs ranging from 2K to 40K tokens ($\tau = 0.5$).

We analyze how the computational cost of prompt compression scales as input context increases up to 40K tokens. Figure 6 compares BridgeComp with AttnComp and CPC [3] in terms of compression latency and peak VRAM usage. BridgeComp consistently incurs the least time and memory across all lengths. At 40K tokens, BridgeComp requires only 5.04 seconds, achieving speedups of 11.75× over AttnComp (59.22 seconds) and 6.88× over CPC (34.68 seconds). Peak memory consumption is similarly reduced to 10.59 GiB, representing a 31.8% reduction relative to AttnComp

[3] AttnComp and CPC represent the strongest competitive baselines in QA performance (Section 5.2).

(15.52 GiB) and 39.1% relative to CPC (17.40 GiB). Importantly, this efficiency is accompanied by greater robustness to increasing context length. With 40K tokens, BridgeComp achieves 41.3 EM, corresponding to a 37.8% reduction (66.4→41.3) from its performance at the 2K-token setting. In contrast, AttnComp and CPC suffer severe drops of 62.3% (60.7→22.9) and 70.6% (55.1→16.2), respectively. Although BridgeComp appends 50 latent tokens, its computational overhead remains limited because compression is performed through a single latent-to-document scoring pass.

F Analysis

F.1 Semantic Faithfulness and Evidence Preservation Analysis

Methods	(A) Complete chain preservation	(B) Semantic distortion	(C) Unfaithful claims	(D) Evidence omission	(E) Retrieval failure
CPC	40.2	6.8	1.4	27.9	23.7
AttnComp	44.6	6.1	1.3	24.3	23.7
BridgeComp	**57.2**	**5.0**	**1.3**	**12.8**	**23.7**

Table 18: LLM-as-a-Judge analysis on MuSiQue. Values are percentages; lower is better for B, C, D, and E, while higher is better for A. The best results are highlighted in bold.

To analyze how compression affects fine-grained context integrity, we conduct a semantic error analysis on MuSiQue, i.e., our most challenging multi-hop benchmark. Following established judge evaluation protocols (Pang et al., 2024), each compressed instance is categorized into one of five mutually exclusive outcomes: (A) complete-chain preservation measures whether all supporting facts and intermediate reasoning bridges required to derive the answer are faithfully retained; (B) semantic distortion denotes cases where token deletion alters the core semantics of a claim (e.g., flipping negations, modifying argument roles or relations); (C) unfaithful claims denote claims or relations that are unsupported by the original context; (D) evidence omission denotes cases where required evidence present in the original retrieved context is removed during compression, while the remaining tokens remain faithful; and (E) retrieval failure denotes cases where required evidence is missing from the initial retrieved context prior to compression.

As shown in Table 18, BridgeComp substantially shifts the outcome distribution toward *complete-chain preservation*, improving it by 28.25% (44.6→57.2) over AttnComp, while reducing evidence omission by 47.33% (24.3→12.8). *Semantic distortion* drops by 18.03% (6.1→5.0), whereas *unfaithful claims* remain unchanged at 1.3%. *Retrieval failure* remains identical across all methods at 23.7%, as all compressors operate on the same retrieved contexts. As a result, BridgeComp successfully preserves complete reasoning chains while maintaining semantic faithfulness.

F.2 Impact of Layer-Selective Aggregation on Attention-Based Compressors

Methods	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
DAC	45.1	57.1	49.0	55.3	31.8	43.1
DAC + LSA	48.6	59.7	52.1	57.9	35.6	46.7
Gain (%)	7.76	4.55	6.33	4.70	11.95	8.35
AttnComp	60.7	75.4	61.5	69.4	38.7	54.4
AttnComp + LSA	64.2	78.6	63.2	71.3	43.9	57.2
Gain (%)	5.77	4.24	2.76	2.74	13.44	5.15
BridgeComp	**66.4**	**80.0**	**64.3**	**72.8**	**46.3**	**60.2**

Table 19: Effect of applying layer-selective aggregation (LSA) to attention-based prompt compression methods at a compression rate $\tau = 0.5$. For + LSA, the original attention aggregation is restricted to the top 40% of transformer layers, while all other components and inference settings remain unchanged. Gain (%) denotes the relative improvement over the corresponding method without LSA, and the best results in each column are highlighted in bold.

We further investigate whether the benefit of layer-selective aggregation (LSA) generalizes beyond BridgeComp to existing attention-based prompt compression methods. To this end, we apply LSA to DAC and AttnComp by restricting their original attention aggregation to the top 40% of transformer layers, while preserving their original token-scoring mechanisms and all other inference settings. As shown in Table 19, applying LSA consistently improves both methods across all three multi-hop QA benchmarks. For DAC, the largest improvements are observed on MuSiQue and HotpotQA, where EM increases by 11.95% (31.8→35.6) and 7.76% (45.1→48.6), respectively. Similarly, AttnComp achieves its largest gains on MuSiQue and HotpotQA, improving EM by 13.44% (38.7→43.9) and 5.77% (60.7→64.2), respectively. These results demonstrate that selectively aggregating upper-layer attention provides more effective compression signals even when applied to different attention-based token-scoring mechanisms. Nevertheless, BridgeComp consistently outperforms the LSA-enhanced baselines, indicating that LSA and reasoning-conditioned latent-token scoring provide complementary benefits. This finding further supports our hypothesis that uniformly aggregating attention across all transformer layers can dilute late-stage semantic and relational signals with lower-layer lexical information, and shows that LSA provides a generally applicable mechanism for improving attention-based prompt compression.

F.3 Lost-in-the-middle Analysis

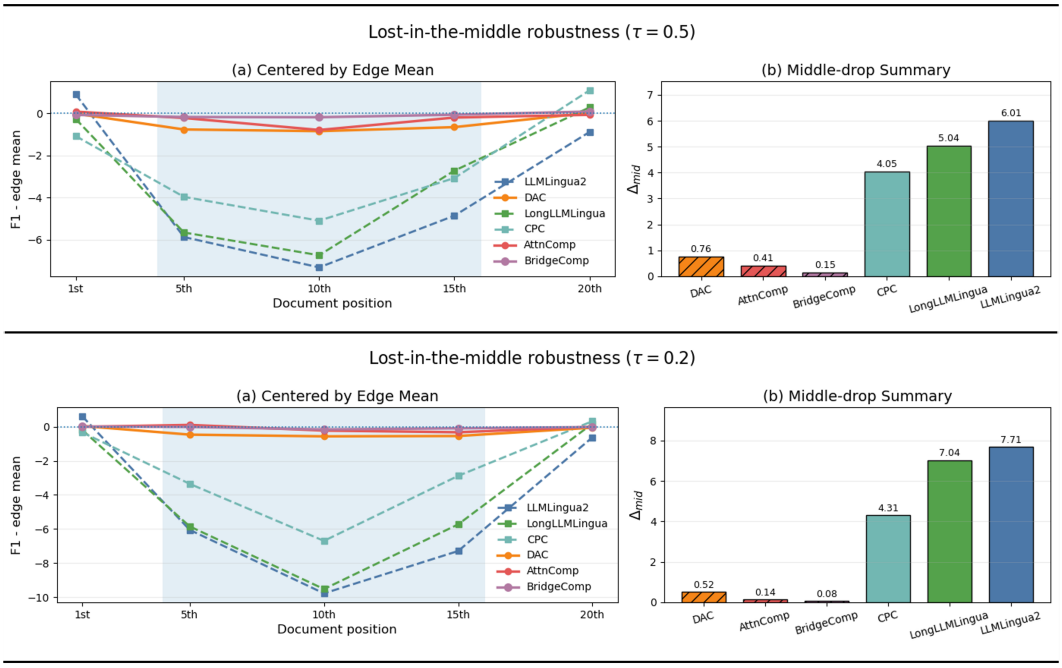


Figure 7: Lost-in-the-middle robustness on NaturalQuestions under two compression rates. (a) Position-wise F1 centered by the edge mean and (b) middle-position drop Δ_{mid}.

We analyze the lost-in-the-middle effect on NaturalQuestions by varying the position of the gold document across the 1st, 5th, 10th, 15th, and 20th positions under two compression rates ($\tau = 0.5$ and $\tau = 0.2$). We define $F1_{\text{edge}}$ as the average $F1$ at the 1st and 20th positions and $F1_{\text{mid}}$ as the average at the 5th, 10th, and 15th positions. The middle-position drop is defined as $\Delta_{\text{mid}} = F1_{\text{edge}} - F1_{\text{mid}}$, where a smaller Δ_{mid} indicates stronger positional robustness. As shown in Figure 7, we identify a consistent pattern across the evaluated methods: attention-based compressors (DAC, AttnComp, and BridgeComp) exhibit substantially smaller middle-position drops than relevance-based compressors (LLMLingua2, LongLLMLingua, and CPC) at both compression rates. This pattern extends beyond BridgeComp, highlighting the stronger positional robustness of attention-based compression in our evaluation. BridgeComp further achieves the smallest Δ_{mid}, reducing it relative to AttnComp by 63.41% (0.41→0.15) at $\tau = 0.5$ and 42.86% (0.14→0.08) at $\tau = 0.2$.

Methods	Source–d_{pos}	Source–d_{neg}	Δ
AttnComp	1.1268	1.1526	0.0258
BridgeComp	1.0542	1.2438	0.1896

Table 20: Source-representation distances to positive and negative document tokens where a larger distance gap Δ indicates stronger evidence discrimination.

F.4 Source-Representation Distance Analysis

We further analyze how source representations distinguish supporting evidence from negative document tokens. For AttnComp, following its original formulation, we use the final question-token representation as the source, while BridgeComp uses LTs. As shown in Table 20, BridgeComp achieves a substantially larger distance gap (Δ) than AttnComp, improving Δ by 7.35$\times$ (0.0258$\rightarrow$0.1896). This indicates that LTs provide stronger discrimination between supporting and negative evidence.

F.5 Lexical Overlap Analysis

Methods	HotpotQA	2WikiMultihopQA	MuSiQue
AttnComp	15.62	15.10	18.60
BridgeComp	8.76	9.06	10.50

Table 21: Exact question-token overlap ratio (%) between the question and compressed context, evaluated on 500 randomly selected samples from each dataset.

We analyze the question-token overlap ratio between the question and the compressed context as shown in Table 21. BridgeComp consistently exhibits substantially lower lexical overlap with the question than AttnComp, decreasing from 15.62 to 8.76 on HotpotQA, 15.10 to 9.06 on 2WikiMultihopQA, and 18.60 to 10.50 on MuSiQue. The lower overlap suggests that BridgeComp is less driven by surface-level lexical matching with the query when selecting tokens. This property is desirable for multi-hop QA, where essential bridge entities and intermediate facts may not explicitly appear in the question. Instead of preferentially retaining question-matching tokens, BridgeComp preserves evidence according to its contextual importance for the underlying reasoning chain.

F.6 Pairwise LLM-as-a-Judge Evaluation of Compression Quality

Methods	HotpotQA			2WikiMultihopQA			MuSiQue		
	Win	Tie	Loss	Win	Tie	Loss	Win	Tie	Loss
vs CPC	786	22	192	795	20	185	764	18	218
vs AttnComp	690	7	303	613	12	375	741	6	253

Table 22: Pairwise LLM-as-a-Judge evaluation of compression quality on multi-hop QA benchmarks. Wins, ties, and losses indicate whether BridgeComp preserves reasoning-critical evidence better than, as well as, or worse than each baseline.

We further evaluate compression quality using a pairwise LLM-as-a-Judge protocol (Zheng et al., 2023; Pang et al., 2024), comparing how well compressed contexts preserve supporting facts, intermediate bridge entities, answer-bearing evidence, and multi-hop reasoning chains. As shown in Table 22, BridgeComp consistently achieves substantially more wins than losses against both CPC and AttnComp across all benchmarks. In particular, BridgeComp records 786 wins versus 192 losses against CPC on HotpotQA and 741 wins versus 253 losses against AttnComp on MuSiQue. These results further demonstrate that BridgeComp preserves reasoning-critical evidence more effectively than prior compression methods.

F.7 Performance Comparison across Rationale Types

Table 23 evaluates how different rationale types affect prompt compression. We compare explicit summarization text with learnable latent tokens (LTs). LTs consistently outperform text rationales across

Rationale	HotpotQA	2WikiMultihopQA	MuSiQue
LTs	**66.4**	**64.3**	**46.3**
Text	64.2	62.5	42.2

Table 23: EM performance of BridgeComp with different rationale representations during training. LTs denote learnable latent tokens, while Text denotes explicit summarization text.

all benchmarks, with the largest gain on MuSiQue, improving EM by 9.72% (42.2→46.3). These results indicate that LTs provide a more effective implicit representation for reasoning-conditioned compression than explicit summarization text.

F.8 Impact of Attention Range

Source Token	HotpotQA		2WikiMultihopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
Latent Tokens	**66.4**	**80.0**	**64.3**	**72.8**	**46.3**	**60.2**
Question Tokens	<u>58.5</u>	<u>72.9</u>	<u>61.3</u>	<u>69.9</u>	<u>41.8</u>	<u>54.1</u>
All Tokens	56.4	70.3	53.9	67.6	34.8	46.2

Table 24: QA performance with different source tokens for document-token scoring at $\tau = 0.5$. Document tokens are fixed as targets; Latent Tokens, Question Tokens, and All Tokens indicate latent-to-document, question-to-document, and all-prompt-token-to-document attention. The best and second-best results are highlighted in bold and underlined, respectively.

We analyze how the choice of attention source tokens affects document-token scoring. Table 24 compares three settings where the target tokens are fixed to document tokens: latent tokens, question tokens, and all prompt tokens. The All Tokens setting is identical to the *w/o* RCA variant in the ablation study (Section 5.6). Latent-to-document attention achieves the best EM and F1 across all benchmarks. Compared with question-to-document attention, it improves EM by 13.50% (58.5→66.4), 4.89% (61.3→64.3), and 10.77% (41.8→46.3) on HotpotQA, 2WikiMultihopQA, and MuSiQue, respectively. Compared with all-prompt-token-to-document attention, it improves EM by 17.73% (56.4→66.4), 19.29% (53.9→64.3), and 33.05% (34.8→46.3), respectively.

These results indicate that using latent tokens as the attention source is more effective than using broader source ranges. While question-token scoring outperforms all-token scoring, confirming that question conditioning yields a superior compression signal, it ultimately underperforms our latent-to-document scoring approach. This performance gap arises because question tokens remain bounded by surface-level features, prioritizing superficial word-level matches while remaining blind to answer-critical, non-overlapping tokens within the documents. In contrast, BridgeComp leverages latent tokens to provide dynamic, reasoning-conditioned states. By anchoring document-token scoring within this latent space, our framework successfully isolates and preserves the deep semantic bridging information required for downstream answer formation.

F.9 Performance Comparison Across Problem Types

Methods	HotpotQA		2WikiMultihopQA			
	Comparison	Bridge	Comparison	Bridge	Inference	Compositional
LongLLMLingua	51.3	57.6	77.5	56.6	26.9	43.6
CPC	53.2	59.9	<u>80.3</u>	55.3	27.8	<u>47.7</u>
AttnComp	<u>60.1</u>	<u>62.5</u>	78.7	<u>60.4</u>	<u>30.6</u>	46.0
BridgeComp	**64.6**	**71.2**	**81.2**	**70.2**	**31.5**	**49.6**
Performance Gain (%)	7.49	13.92	1.12	16.23	2.94	3.98

Table 25: QA performance (EM) across problem types on HotpotQA and 2WikiMultihopQA. Performance Gain (%) denotes the relative improvement of BridgeComp over the strongest baseline for each problem type. The best and second-best results are highlighted in bold and underlined, respectively.

We further analyze QA performance across problem types on HotpotQA and 2WikiMultihopQA. As shown in Table 25, BridgeComp achieves the highest EM across all six problem types. The largest gain is observed on Bridge questions in 2WikiMultihopQA, where BridgeComp improves over the strongest baseline by 16.23% (60.4→70.2). Consistent gains are also observed on HotpotQA, improving EM by 7.49% (60.1→64.6) on Comparison and 13.92% (62.5→71.2) on Bridge questions. These results indicate that BridgeComp effectively preserves information required for diverse multi-hop reasoning patterns.

F.10 Impact of Compressed Documents on Training

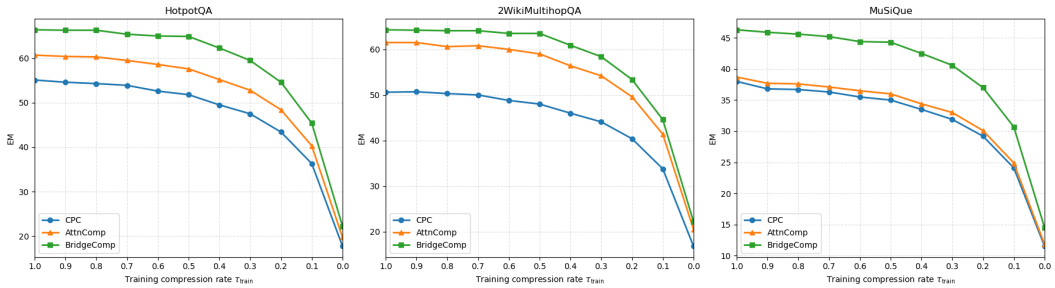


Figure 8: Impact of the training compression rate τ_{train} on EM across multi-hop QA benchmarks. Each method is trained using document contexts compressed at different training compression rates τ_{train}, while evaluation is performed with the inference compression rate fixed at $\tau = 0.5$. $\tau_{\text{train}} = 1$ retains all document tokens, whereas $\tau_{\text{train}} = 0$ removes all document tokens.

We analyze how the training compression rate affects downstream QA performance. This experiment varies the compression applied to the documents before they are provided to the model during training, rather than the compression rate used to construct RL rollouts. For each τ_{train}, the input documents are compressed to retain the corresponding fraction of document tokens, while all methods are evaluated with the inference compression rate fixed at $\tau = 0.5$.

As shown in Figure 8, EM generally decreases as the training compression rate decreases and the training documents become more aggressively compressed. This trend indicates that removing document evidence from the training inputs reduces the supervision available for learning effective compression signals. Nevertheless, BridgeComp consistently outperforms CPC and AttnComp across all training compression rates and exhibits a more gradual performance decline. Even at $\tau_{\text{train}} = 0.2$, where only 20% of the document tokens are retained during training, BridgeComp achieves 37.0 EM on MuSiQue, outperforming AttnComp by 22.92% (30.1→37.0). These results demonstrate that BridgeComp remains effective when trained with partially compressed document contexts, although retaining more document evidence during training provides stronger supervision for learning the compression policy.

F.11 Performance Comparison across Compression Rates

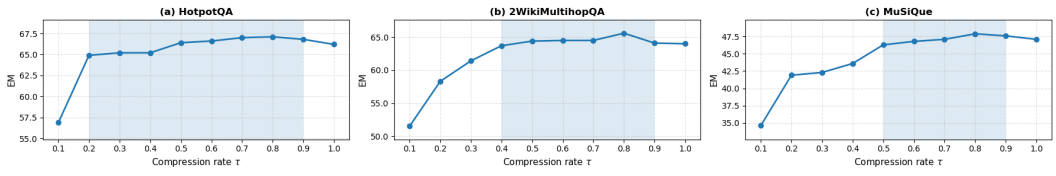


Figure 9: QA performance of BridgeComp across compression rates on multi-hop QA benchmarks. A larger compression rate indicates that more document tokens are retained. The blue shaded regions indicate compression rates at which BridgeComp achieves performance comparable to or higher than that obtained with the uncompressed original prompt ($\tau = 1.0$). During training, the compression rate is fixed at $\tau = 0.5$ for all settings.

Figure 9 reports the QA performance of BridgeComp across different compression rates. The blue shaded regions show that moderate compression generally maintains or slightly exceeds full-prompt performance, suggesting that removing distracting tokens can improve answer generation without

discarding critical evidence. The largest improvement is observed on 2WikiMultihopQA, where EM increases by 2.50% from 64.0 at $\tau = 1.0$ to 65.6 at $\tau = 0.8$. In contrast, performance declines substantially under aggressive compression, particularly on MuSiQue, where EM decreases from 47.1 at $\tau = 1.0$ to 41.9 at $\tau = 0.2$ and 34.6 at $\tau = 0.1$. These results show that BridgeComp preserves QA performance across a broad range of moderate compression rates, whereas excessive token removal discards evidence required for answer formation.

F.12 Dataset Contamination Analysis

Methods	HotpotQA	2WikiMultihopQA	MuSiQue	NQ
BRIEF-Pro	0.086	0.029	0.095	–
BridgeComp	0.029	0.023	0.044	0.034

Table 26: Average per-sample maximum 5-gram overlap between the test and training sets for each dataset.

To assess potential train-test data leakage, we follow the contamination analysis protocol established by BRIEF-Pro (Gu et al., 2026). For each evaluation instance and each training instance, we extract the 5-gram set from its test document context and compute the 5-gram overlap score, i.e., the proportion of unique 5-grams in the test document context that appear in all documents relevant to the training instance. Then we compute the maximum 5-gram overlap score against all document contexts in the training set. These scores are averaged over the entire test set. As shown in Table 26, all overlap scores for BridgeComp are below 0.05: 0.029 on HotpotQA, 0.023 on 2Wiki, 0.044 on MuSiQue, and 0.034 on NQ. BridgeComp also exhibits consistently lower overlap than BRIEF-Pro on the three shared benchmarks. These results indicate limited lexical overlap between the training and test contexts, suggesting that BridgeComp's performance gains are not driven by train-test contamination.

G Hyperparameter Sensitivity

In this section, we conduct sensitivity analysis by varying key hyperparameters during training and examining their impact on downstream QA performance.

G.1 Impact of the Number of Perturbations for Mask Sampling

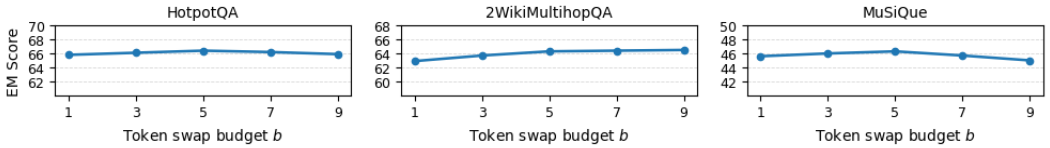


Figure 10: Impact of the token swap budget b on EM across three multi-hop QA benchmarks.

We analyze the impact of the token swap budget b during local mask rollouts (Section 4.5.2). Since each rollout samples b distinct indices without replacement from each of the anchor-selected and anchor-excluded token pools, b must be an integer satisfying $0 \leq b \leq \min(k, |D| - k)$, where $k = \lfloor \tau |D| \rfloor$. The case $b = 0$ leaves the anchor unchanged; in particular, $\tau = 1$ permits only $b = 0$ and thus no nontrivial perturbation.

As shown in Figure 10, downstream QA performance steadily improves as b increases from 1 to 5, with $b = 5$ achieving the best overall results on HotpotQA (66.4 EM) and MuSiQue (46.3 EM). Increasing b beyond 5 yields diminishing returns and eventually degrades performance, as larger perturbations move sampled masks farther from the anchor and increase reward variance during training. These results indicate that moderate local exploration yields the most stable and effective optimization.

G.2 Impact of the Anchor-Relative Weight η

We analyze the impact of the anchor-relative weight η, which controls the contribution of the anchor-relative advantage in AGPG. Figure 11 shows EM for η values of 0, 0.25, 0.5, 0.75, and 1.0. When

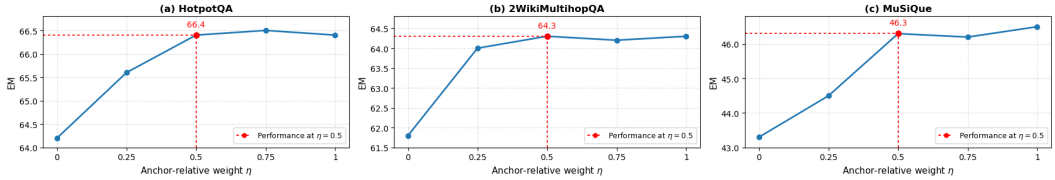


Figure 11: Impact of the anchor-relative weight η on EM performance across three multi-hop QA benchmarks. The red dashed lines indicate the default setting of $\eta = 0.5$.

$\eta = 0$, the anchor-relative term is removed and optimization relies solely on the group-centered advantage. Increasing η from 0 to 0.5 consistently improves performance across all three benchmarks, with EM increasing from 64.2 to 66.4 on HotpotQA, from 61.8 to 64.3 on 2WikiMultihopQA, and from 43.3 to 46.3 on MuSiQue. Performance remains stable for $\eta \geq 0.5$, with only marginal variations across the tested settings. These results indicate that incorporating the anchor-relative signal provides a useful optimization signal beyond group-centered comparison, while BridgeComp is relatively robust to its precise weight once sufficiently incorporated. Therefore, we use $\eta = 0.5$ as the default setting for all experiments.

G.3 Impact of the Number of Candidate Layers

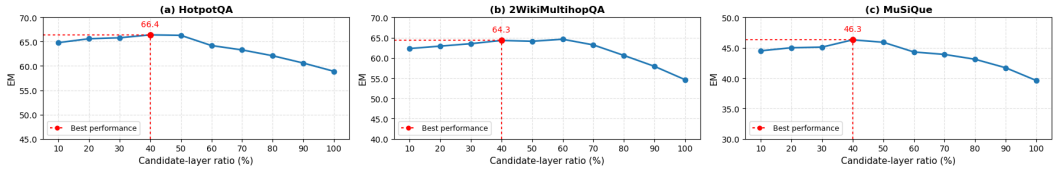


Figure 12: Effect of the proportion of upper layers used as candidate layers on QA performance on multi-hop QA benchmarks at a compression rate $\tau = 0.5$. Red markers highlight the default candidate-layer ratio of 40%. For each setting, BridgeComp is retrained and evaluated using the corresponding candidate-layer ratio.

We conduct a sensitivity analysis to examine the effect of varying the proportion of upper transformer layers used as candidates in layer-selective aggregation throughout both training and inference. For each candidate-layer ratio, we retrain BridgeComp using the corresponding layer range and evaluate the resulting model under the same configuration. As shown in Figure 12, performance generally improves as the candidate-layer ratio increases from 10% to 40%, with BridgeComp achieving EM scores of 66.4, 64.3, and 46.3 on HotpotQA, 2WikiMultihopQA, and MuSiQue, respectively, at 40%. Performance remains comparatively strong around 40–50%, indicating that BridgeComp is relatively robust when candidate layers are restricted primarily to the upper layers. As progressively lower layers are included, however, performance generally declines. When all layers are used, EM decreases across all datasets, with the largest relative drop of 15.09% (64.3→54.6) on 2WikiMultihopQA compared with the 40% setting. We observe the same overall trend across all evaluated LLM backbones, consistently showing stronger performance when attention aggregation is concentrated in the upper layers and degradation as lower layers are increasingly incorporated. These results support the layer-selective aggregation design: selected upper layers better preserve late-stage semantic and relational signals, whereas all-layer aggregation mixes these signals with lower-layer surface lexical signals. Therefore, we use the top 40% of layers as our default configuration.

G.4 Impact of the Number of Latent Tokens

We analyze the impact of the number of latent tokens (LTs) on QA performance. As shown in Figure 13, EM generally improves as the number of LTs increases. The improvement from 10 to 50 LTs is particularly pronounced on 2WikiMultihopQA and MuSiQue, where EM increases by 2.5 and 2.4 points, respectively. This trend indicates that additional LTs provide richer reasoning-conditioned signals for complex multi-hop questions. Beyond 40 LTs, however, performance remains stable over a broad range: between 40 and 100 LTs, EM varies by at most 1.2 points across all three benchmarks. This limited variation demonstrates that BridgeComp is robust to the number of LTs beyond a certain

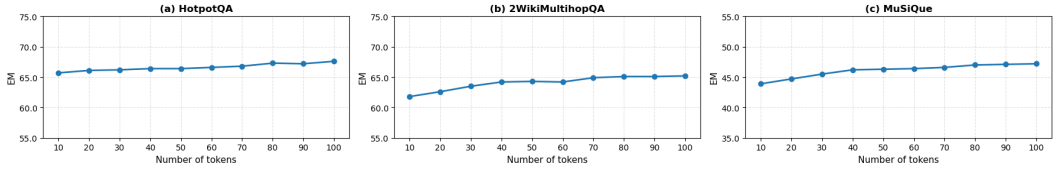


Figure 13: QA performance (EM) comparison with varying numbers of latent tokens (Num. Tokens) on multi-hop QA benchmarks at a compression rate $\tau = 0.5$.

threshold. Because additional LTs increase memory consumption during training, we use 50 LTs as the default setting to balance QA performance and computational overhead.

G.5 Impact of the Number of Sampled Masks

Num. Masks (n_m)	Training Time	HotpotQA		2WikiMultihopQA		MuSiQue	
		EM	F1	EM	F1	EM	F1
2	1.00×	66.3	79.9	63.1	71.5	**46.8**	**60.3**
4 (default)	1.20×	**66.4**	**80.0**	**64.3**	**72.8**	46.3	60.2
8	1.35×	66.0	79.6	64.2	**72.8**	46.3	59.7
16	1.55×	66.3	79.9	**64.3**	**72.8**	46.2	60.1

Table 27: Impact of the number of sampled masks n_m on QA performance and training time. Training time is normalized to the $n_m = 2$ setting. The best results for each QA metric, including ties, are highlighted in bold.

We analyze the effect of the number of sampled masks n_m per training instance on downstream QA performance and training time. Increasing n_m provides more samples for estimating the group mean reward and averaging the policy loss, which can improve statistical stability but requires additional rollout evaluations. Table 27 compares $n_m \in \{2, 4, 8, 16\}$, with training time normalized to the $n_m = 2$ setting. Using $n_m = 4$ achieves the best or tied-best EM and F1 on HotpotQA and 2WikiMultihopQA, whereas MuSiQue performs slightly better at $n_m = 2$. Increasing n_m beyond 4 yields no further performance gains, while relative training time rises from $1.20\times$ at $n_m = 4$ to $1.35\times$ and $1.55\times$ at $n_m = 8$ and $n_m = 16$, respectively. We therefore use $n_m = 4$ to balance QA performance and training efficiency.

G.6 Impact of LLM Backbones

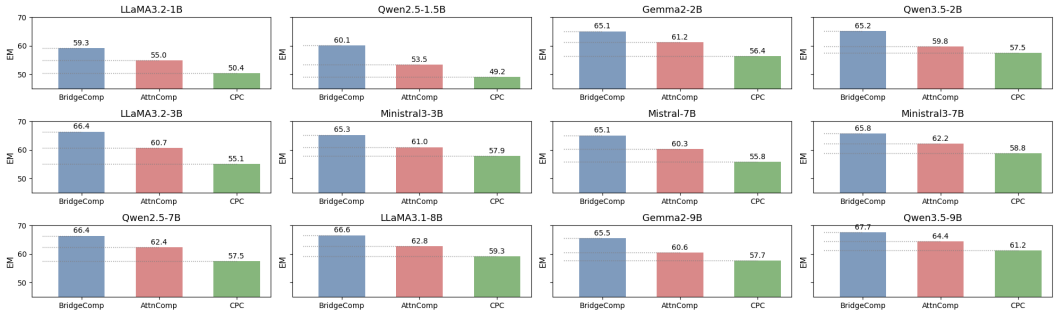


Figure 14: QA performance (EM) of BridgeComp across LLM families and model scales (1B-9B parameters) on HotpotQA at a compression rate $\tau = 0.5$. GPT-5.6 Luna is used as the downstream generator for all backbone comparisons.

Figure 14 evaluates the cross-backbone and cross-scale generalization capability of BridgeComp on HotpotQA at $\tau = 0.5$ across 12 open-source LLM backbones ranging from 1B to 9B parameters. These backbones span multiple model families, including LLaMA, Qwen, Gemma, and Mistral. BridgeComp consistently outperforms AttnComp and CPC across all evaluated configurations, achieving EM scores ranging from 59.3 to 67.7 and improving over AttnComp by 3.3–6.6 points. Within each model family, performance generally improves with increasing parameter scale, while

BridgeComp maintains its advantage across all model sizes. These empirical findings confirm that BridgeComp is backbone-agnostic and maintains stable compression efficacy independent of the underlying language model.

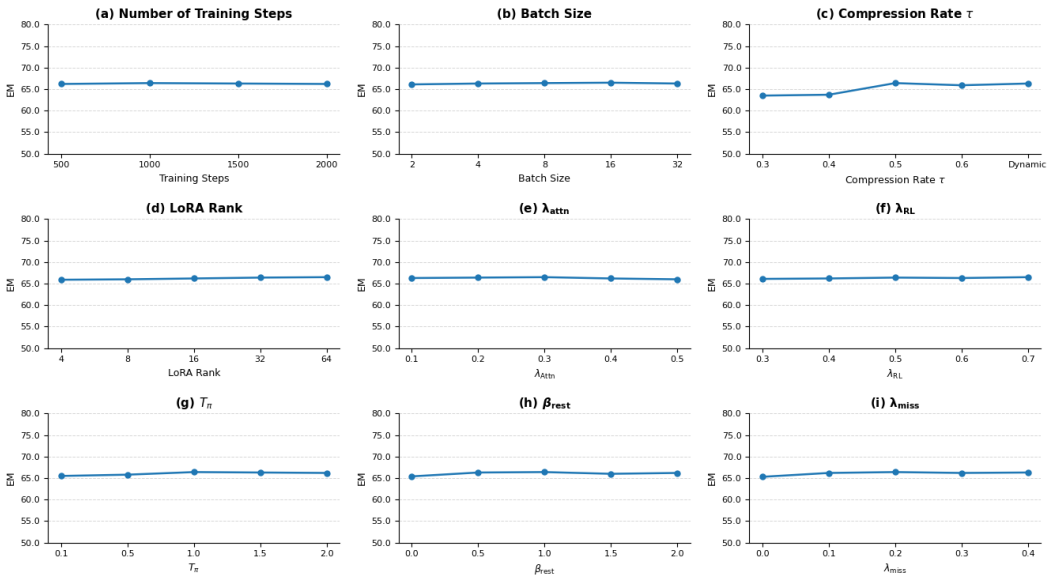


Figure 15: Hyperparameter sensitivity analysis on HotpotQA; (a) number of joint-optimization steps, (b) per-device batch size, (c) training compression rate τ, (d) LoRA rank, (e) attention-reward weight λ_{attn}, (f) RL-loss weight λ_{RL}, (g) sampling temperature T_π, (h) negative-attention weight β_{rest}, and (i) missing-support penalty weight λ_{miss}.

G.7 The number of Training Steps

We conduct experiments to analyze the effect of the number of joint-optimization steps performed after the fixed 1,000-step latent-token warm-up. Figure 15(a) shows EM for 500, 1,000, 1,500, and 2,000 joint-optimization steps. The highest EM is achieved at 1,000 steps, although performance remains highly stable across the tested range, varying only from 66.2 to 66.4. Hence, we use 1,000 joint-optimization steps, resulting in 2,000 total training steps including the warm-up stage.

G.8 Batch Size

We conduct experiments to analyze the effect of the per-device batch size during joint optimization. Figure 15(b) shows EM for batch sizes of 2, 4, 8, 16, and 32. The highest EM is achieved with a batch size of 16, but the overall variation is marginal, with performance ranging only from 66.1 to 66.5. Considering that larger batches require substantially greater GPU memory while providing only minor gains, we use a per-device batch size of 4 for all experiments.

G.9 Training Compression Rate

We conduct experiments to analyze the effect of the retention ratio τ used to determine the masking range during RL training. This setting does not compress the full prompt used for NTP; instead, it determines the number of document tokens retained in the anchor and sampled masks used for policy optimization. Figure 15(c) compares fixed ratios of 0.3, 0.4, 0.5, and 0.6 with a dynamic masking strategy. The highest EM is achieved at $\tau = 0.5$, while the dynamic setting obtains a comparable EM of 66.3. Therefore, we use a fixed masking ratio of $\tau = 0.5$ for all experiments.

G.10 LoRA Rank

We conduct experiments to analyze the effect of the LoRA rank r during joint optimization. Figure 15(d) shows EM for rank values of 4, 8, 16, 32, and 64. Performance gradually increases with the

LoRA rank and reaches its highest value of 66.5 at $r = 64$, although the overall variation remains marginal. In particular, increasing the rank from 32 to 64 improves EM by only 0.1 while doubling the LoRA parameter size. Hence, we use $r = 32$ to balance performance and parameter efficiency.

G.11 Impact of the Attention-Reward Weight

We conduct experiments to analyze the effect of the attention-reward weight λ_{attn}. Figure 15(e) shows EM for values ranging from 0.1 to 0.5. The highest EM is achieved at $\lambda_{\text{attn}} = 0.3$, while performance remains stable between 0.1 and 0.3 and gradually decreases at larger values. This trend indicates that latent-attention preservation provides a useful reward signal, but assigning it excessive weight can reduce the relative contribution of answer utility. Since $\lambda_{\text{attn}} = 0.2$ achieves near-optimal performance while maintaining a balanced reward composition, we use it as the default setting.

G.12 Weight for RL Loss

We conduct experiments to analyze the effect of the RL-loss weight λ_{RL}, which balances the group policy-gradient loss and the NTP objective during joint optimization. Figure 15(f) shows EM for values ranging from 0.3 to 0.7. The highest EM is achieved at $\lambda_{\text{RL}} = 0.7$, although performance remains highly stable across the tested range, varying only from 66.1 to 66.5. Since the improvement over $\lambda_{\text{RL}} = 0.5$ is only 0.1 EM, we use $\lambda_{\text{RL}} = 0.5$ to maintain a balanced contribution from the NTP and RL objectives.

G.13 Sampling Temperature

We conduct experiments to analyze the effect of the sampling temperature T_π, which scales the logits of the softmax distributions used to sample removal and addition tokens around the deterministic anchor. Figure 15(g) shows EM for T_π values of 0.1, 0.5, 1.0, 1.5, and 2.0. The highest EM is achieved at $T_\pi = 1.0$, while performance remains stable across the tested range, varying only from 65.5 to 66.4. These results indicate that the benefit of exact-budget swap exploration does not depend on a specific sampling temperature. Therefore, we use $T_\pi = 1.0$ for all experiments.

G.14 Weight for Negative Documents

We conduct experiments to analyze the effect of β_{rest}, which penalizes latent attention retained on non-supporting documents. Figure 15(h) shows EM for β_{rest} values of 0, 0.5, 1.0, 1.5, and 2.0. When $\beta_{\text{rest}} = 0$, the reward considers only the attention mass retained over supporting documents, and EM drops to 65.4. With the penalty, performance remains stable for $\beta_{\text{rest}} \geq 0.5$, varying only from 66.0 to 66.4. These results indicate that the penalty is necessary for learning a discriminative compression policy, while its precise weight has little effect. Since the highest EM of 66.4 is achieved at $\beta_{\text{rest}} = 1.0$, we use this value for all experiments.

G.15 Weight for Document Coverage Penalty

We conduct experiments to analyze the effect of λ_{miss}, which penalizes sampled masks that completely remove a gold supporting document. Figure 15(i) shows EM for λ_{miss} values of 0, 0.1, 0.2, 0.3, and 0.4. When $\lambda_{\text{miss}} = 0$, EM drops to 65.3, presumably because rollouts that discard an entire supporting document are not penalized, which destabilizes training. With the penalty, EM increases to 66.4 at $\lambda_{\text{miss}} = 0.2$ and remains stable for $\lambda_{\text{miss}} \geq 0.1$, varying only from 66.2 to 66.4. These results indicate that the penalty is necessary for stable training, while its precise weight has little effect. Therefore, we use $\lambda_{\text{miss}} = 0.2$ for all experiments.

H Case Study

H.1 Bridging Information Preservation

We conduct a case study to demonstrate how BridgeComp preserves bridging information under compression. Figure 16 shows a multi-hop question that requires comparing the countries of citizenship of the directors of two films, “Count Five and Die” and “The Green Jacket”. Answering it depends on

preserving both reasoning chains: Count Five and Die was directed by Victor Vicas, whose nationality is French, while The Green Jacket was directed by Roger Richebé, whose nationality is also French. AttnComp tends to focus on surface-level lexical overlap with the question, especially terms such as "film" and "director." As a result, it retains some relevant evidence, such as partial information about Victor Vicas as a film director and an isolated mention of a "French film director," but fails to preserve the connected bridging information. In particular, Victor Vicas's citizenship is omitted, and the bridge between The Green Jacket and Roger Richebé is lost. Consequently, the compressed context lacks the key evidence triples needed for the final country comparison. In contrast, BridgeComp preserves these triples, linking "Count Five and Die" and "The Green Jacket" to their respective directors, Victor Vicas and Roger Richebé, and both directors to France. This maintains the complete film–director–country reasoning path and supports the correct answer, "yes."

H.2 Lexical-overlap Bias in Hard Negatives

We conduct a case study to examine lexical-overlap bias in hard negative documents. Figure 17 presents a comparison question asking whether "Russell High School (Ontario)" and "Westside Christian High School" are located in the same country. To answer correctly, the model should preserve the bridging information that links Westside Christian High School to the United States and Russell High School to Canada. AttnComp is overly biased toward the initial question entities and repeatedly preserves documents or tokens containing overlapping school names such as "Christian High School." As a result, lexical-overlap bias occurs, and the bridging information for Russell High School, such as "Upper Canada," is lost during compression. In contrast, BridgeComp preserves the bridging information needed to compare the two countries while suppressing lexically overlapping hard negatives. This case shows that BridgeComp mitigates lexical-overlap bias by selecting tokens required for answer formation rather than tokens that merely match the initial question entities.

Case Study: Bridging Information Preservation

Question: Are director of film **Count Five And Die** and director of film **The Green Jacket** both from the same country?

Ground Truth Answer: yes
Evidence Triples:

Count Five and Die → director → Victor Vicas → country of citizenship → French

The Green Jacket → director → Roger Richebé → country of citizenship → French

Original Contexts:

Document [1] (Title: Count Five and Die) Count Five and Die is a 1957 British war drama produced by Zonic Productions and released in the USA by the Twentieth Century- Fox Film Corporation. It was directed by Victor Vicas, produced by Ernest Gartside with the screenplay by Jack Seddon and David Pursall. It stars Jeffrey Hunter, Nigel Patrick and Annemarie Dringer.

Document [2] (Title: The Green Jacket) The Green Jacket French: Lhabit vert is a 1937 French comedy film directed by Roger Richeb and starring Elvire Popesco, Victor Boucher and Jules Berry. It is based on a play by Gaston Arman de Caillavet and Robert de Flers. The films sets were designed by the art director Jean dEaubonne. The film is a farce set against the backdrop of the Acadmie franaise.

Document [3] (Title: Victor Vicas) Victor Vicas 25 March 1918 9 December 1985 was a Russian- born French film director and screenwriter. His film "The Wayward Bus" was entered into the 7th Berlin International Film Festival.

Document [4] (Title: Roger Richebé) Roger Richeb 3 December 1897, Marseille- 10 July 1989 in Ville-d Avray, born Roger Gustave Richeb, was a French film director, screenwriter, and producer.

BridgeComp

Document [1] (Title: **Count Five and Die**) Count is 195 British war drama produced by Zonic Productions released USA Twentieth Century Fox Corporation. directed by **Victor Vicas**, Ernest Gartside screenplay by Jack Seddon David. It stars Jeffrey Hunter Nigel Annemarie Dringer.

Document [3](Title: **Victor Vicas**) Victor Vicas 25 March 1918 9 December 1985 was a Russian- born **French** film director and screenwriter. His film" The Wayward Bus entered the 7th Berlin Festival.

Document [2]Title: **The Green Jacket** Green French: Lhabit vert 1 French comedy directed by **Roger Richeb** starring Elvire Popesco Victor Boucher Jules Berry. based play Gaston Ar films sets artEa farce Acad fran.

Document [4]Title: **Roger Richebé**) Roger Richeb 3 December 1897, Marseille- 19 in Ville d Avray born Roger Gustave Richeb, was a **French** film director, screenwriter and producer.

AttnComp

Document [1] (Title: Twentieth Century- Fox Film Corporation. It was directed by **Victor Vicas**, produced by Ernest Gartside with the. It stars Jeffrey Hunter, Nigel Patrick and Annemarie Dringer.

Document [3] (Title: **Victor Vicas**) Victor Vicas 25 March 1918 9 December 1985 film director and screenwriter. His was entered into the 7th Berlin International Film Festival.

Document [2] Acadmie franaise.

Document [4] (Title:3 December 1897 July 1989 **French** film director, screenwriter, and producer.

Figure 16: A qualitative case study showing that the proposed method preserves the bridging information under compression. Count Five and Die's evidence triples (**blue**) and The Green Jacket's evidence triples (**green**) are highlighted. BridgeComp preserves both evidence triples, whereas AttnComp drops the bridging information indicating that The Green Jacket was directed by Roger Richebé. The evidence triples are provided by 2WikiMultihopQA.

Case Study: Lexical-overlap Bias in Hard Negatives

Question: Are both Russell High School (Ontario) and Westside **Christian High School** located in the same country?

Ground Truth Answer: No
Evidence Triples:

Russell High School (Ontario) → country → Canada

Westside Christian High School → country → United States

Original Contexts:

(Title : Westside Christian High School) Westside Christian High School is a private **Christian High School** located in Tigard, Oregon, United States.

(Title : Russell High School (Ontario)) Russell High School is a secondary school in Russell located under the jurisdiction of **Upper Canada** District School Board. In addition to Russell, students from Embrun, Casselman and Limoges attend Russell High School. The school has the highest graduation rate in **Upper Canada** District School Board and one of the highest in Ontario, with 94 of Grade 12 students achieving a high school diploma in 2008.

BridgeComp

(Title: Westside **Christian High School**) West is private Christian High located Tigard, Oregon, United States. (Title: Russell High (Ontario)) Russell secondary Russell located under jurisdiction **Upper Canada** Board. In addition Russell students Emb Cassel Limoges attend Russell. The the highest graduation **Upper Canada** School and one the Ontario, with4 of Grade achieving high 2008.3Title: Cascade) Cascade private Christian Medford, Oregon United States.: Sunny Christian Washington. Valley (Arizona located Chandler Arizona United States.6 (Ottawa Township), or high at2 East IL. Lehigh LV, Bethlehem Pennsylvania, United States. Covenant (DeM IndianaM United States. CovenantIndianapolis)) High non-denomin Indianapolis Indiana. (: Calvary (Florida)) Calvary, college in Clearwater Florida, in United States.

AttnComp

(Title: Westside **Christian High School**) Westside **Christian High School** is a Tigard, Oregon, United States. (Title: Russell High School (Ontario)) Russell High School is a secondary school in Russell located under the (Title: Medford, Oregon, United States. (Title: Sunnyside, Washington. (Title: Valley **Christian High School** (Arizona)) Valley **Christian High School** is a Chandler, Arizona, United States. (Title: Ottawa, IL. (Title: (Title: private **Christian High School** located in United States. (Title: Covenant **Christian High School** (Indianapolis)) Covenant **Christian High School** is a private non-denominational Christian high school located in Indianapolis, Indiana. (Title: Calvary **Christian High School** (Florida)) Calvary **Christian High School** is a college preparatory **Christian High School** in Clearwater, Florida, in the United States.

Figure 17: A qualitative case study of lexical-overlap bias in hard negatives. Initial question entities (blue) and bridging information (orange) are highlighted. BridgeComp avoids lexical-overlap bias and preserves bridging information, whereas AttnComp is overly biased toward initial question entities and loses the bridging information needed for correct inference. The evidence triples are provided by 2WikiMultihopQA.

I PROMPT

I.1 INSTRUCTION

You are a clever multi-hop question-answering agent. Answer the question using only the information contained in the provided Documents. The Documents may be compressed, fragmented, or grammatically incomplete. Treat these irregularities as artifacts of prompt compression and identify the relevant evidence across the documents.

- Some questions require connecting information from multiple documents or comparing facts found in different documents. Perform any necessary reasoning internally.
- Return only the shortest final answer. Except for yes-or-no questions, copy the answer expression exactly from the Documents. Do not paraphrase, rewrite, normalize, summarize, or expand the answer. Do not generate an explanation, reasoning process, supporting evidence, citation, introductory phrase, or complete sentence.
- For a yes-or-no question, output exactly "yes" or "no" in lowercase.

 # INPUT FORMAT
Documents:
{Retrieved Documents}
Question:
{question}
Answer:
<<YOUR ANSWER HERE>>

Figure 18: Prompt for compression and question answering.

I.2 SUMMARIZATION EXTRACTION

You are a meticulous Evidence Summarization Agent specialized in question answering. Your task is to compress the provided documents into a concise, self-contained summary that preserves all information necessary to answer the given question correctly.

- Use the Ground Truth Answer only as a semantic guide for identifying which evidence is relevant. The Ground Truth Answer may use a different surface form, spelling, abbreviation, alias, or level of specificity from the wording used in the documents.
- You must always preserve the wording and entity expressions found in the provided documents. Do not replace a document expression with the Ground Truth Answer's expression merely because they refer to the same entity or concept. For example, if the Ground Truth Answer is "Albert Einstein" but the document states "A. Einstein," retain "A. Einstein" in the summary. Copying the Ground Truth Answer's surface form when it is not explicitly present in the documents may cause answer leakage and overfitting.
- Do not mention that the Ground Truth Answer was provided, and do not simply repeat the answer without its supporting evidence.

 # Follow these requirements:

1. Retain only facts that directly support the answer or are necessary to connect the supporting facts.
2. Preserve the complete reasoning chain, including intermediate entities, relations, events, dates, quantities, and causal or temporal links.
3. When answering requires combining information from multiple documents, explicitly connect the relevant facts into a coherent multi-hop evidence chain.
4. Preserve important qualifiers, negations, comparisons, conditions, and distinctions that could affect the answer.
5. Remove redundant, irrelevant, or distracting information.
6. Do not introduce external knowledge, assumptions, speculation, or unsupported conclusions.
7. Preserve names, aliases, abbreviations, terminology, spellings, and other surface forms exactly as expressed in the documents whenever possible.
8. Do not normalize, expand, correct, or replace a document expression using the wording of the Ground Truth Answer.
9. Resolve pronouns and ambiguous references only when the referenced entity is clearly supported by the documents. When resolving them, use an entity expression appearing in the documents rather than one taken only from the Ground Truth Answer.

The resulting summary must allow another language model to answer the question correctly without access to the original documents, while preventing direct leakage of the Ground Truth Answer's wording.

 # INPUT FORMAT

Documents:
{Original Documents}

Question:
{question}

Ground Truth Answer: {answer}

 # OUTPUT FORMAT

Summarization:
<<YOUR ANSWER HERE>>

Figure 19: Prompt for summarization extraction.

I.3 LLM JUDGE 1. COMPRESSION QUALITY COMPARISON

You are an expert evaluator of evidence preservation for multi-hop question answering. Your task is to compare two compressed versions of the same source documents, produced by BridgeComp and a baseline method, and determine which better preserves the evidence required to derive the gold answer.

You are given:

1. a question,
2. gold supporting documents,
3. the gold answer,
4. BridgeComp compressed documents, and
5. baseline compressed documents.

Evaluate both contexts based on:

- **Reasoning completeness:** Whether all supporting facts and intermediate relations required for the answer are retained.
- **Semantic faithfulness:** Whether the retained evidence preserves the original meaning without altering negation, relations, argument roles, quantities, temporal/causal relations, coreference, or modifier scope.
- **Faithfulness:** Whether the context avoids unsupported or inconsistent claims.
- **Answer derivability:** Whether the gold answer can be derived from the compressed evidence without missing information or external knowledge.

Do not prefer a context simply because it is longer, more fluent, contains more details, or explicitly contains the gold answer. Prioritize minimal, complete, and faithful reasoning evidence.

Choose exactly ONE label:

- **A : (BridgeComp Win)** BridgeComp preserves the required reasoning evidence more completely or faithfully.
- **B : (Tie)** Both preserve essentially equivalent reasoning evidence. Minor differences in wording, length, formatting, or irrelevant details should not break a tie.
- **C : (BridgeComp Loss)** The baseline preserves the required reasoning evidence more completely or faithfully.

Use the Supporting Documents and Gold Answer only to identify the evidence chain required for answering the question. Do not use external knowledge to compensate for missing evidence.

 # INPUT FORMAT

Question:
{question}
Supporting Documents:
{supporting_documents}
Gold Answer:
{ground_truth_answer}
BridgeComp Compressed Documents:
{bridgecomp_compressed_documents}
Baseline Compressed Documents:
{baseline_compressed_documents}

 # OUTPUT FORMAT

Output exactly one capital letter:
A, B, or C (Do not output any explanation or additional text)

Figure 20: Prompt for judging compression quality.

I.4 LLM JUDGE 2. SEMANTIC DISTORTION

You are an expert evaluator of compressed evidence for multi-hop question answering. Your task is to evaluate how compression affects the meaning and reasoning evidence contained in the original retrieved documents.
You are given:

1. a question,
2. gold supporting documents,
3. the gold answer,
4. original retrieved documents, and
5. compressed documents.

Use the Supporting Documents and Gold Answer to identify the reasoning chain required for the answer, and use the Retrieved Documents to determine what evidence was available before compression.
Choose exactly ONE label:

- **Complete-Chain Preservation:** All supporting facts and intermediate relations required to derive the Gold Answer are faithfully retained together in the Compressed Documents.
- **Semantic Distortion:** Compression changes the meaning of an answer-relevant claim that existed in the Retrieved Documents, e.g., by altering negation, relations, argument roles, quantities, coreference, or modifier scope.
- **Unfaithful Claims:** The Compressed Documents assert or imply an answer-relevant fact or relation that is not supported by the Retrieved Documents.
- **Evidence Omission:** Evidence required for the reasoning chain was present in the Retrieved Documents but removed during compression, without introducing semantic distortion or unfaithful claims.
- **Retrieval Failure:** One or more supporting facts required to derive the Gold Answer were already absent from the Retrieved Documents before compression.

Apply the following priority:

1. If required evidence is already missing from the Retrieved Documents, output **E**.
2. Otherwise, if the complete reasoning chain remains faithfully preserved, output **A**.
3. Otherwise, if compression changes the meaning of existing relevant evidence, output **B**.
4. Otherwise, if compression introduces an unsupported relevant claim or relation, output **C**.
5. Otherwise, if required evidence is removed during compression, output **D**.

Do not penalize grammatical awkwardness, shortened wording, formatting changes, or deletion of irrelevant information. Do not use external knowledge to reconstruct missing evidence.

 # INPUT FORMAT

Question:
{question}
Supporting Documents:
{supporting_documents}
Gold Answer:
{ground_truth_answer}
Retrieved Documents:
{retrieved_documents}
Compressed Documents:
{compressed_documents}

 # OUTPUT FORMAT

Output exactly one capital letter:
A, B, C, D, or E (Do not output any explanation or additional text)

Figure 21: Prompt for judging semantic distortion.