INCREMEM: COMPRESSING LONG-TERM AGENT MEMORY IN INCREMENTS OF INFORMATION GAIN

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon LLM agents accumulate interaction histories whose length makes full-context attention prohibitively expensive. Latent memory compression reduces this cost by condensing the history into gist tokens that the model attends to instead of the raw context. Existing methods, however, compress at a uniform rate, as if information were spread evenly across tokens, so a multi-token fact is readily split across gist tokens while fillers and restatements take an equal share of the budget. We argue that capacity should instead follow the information each token contributes. Casting compression as a rate–distortion problem, we show that if a memory unit's distortion is a convex function of the information it holds, regardless of its length, spans of equal information gain minimize the distortion of the LLM's output. We measure this gain through a retrieval head, where a token is informative to the extent that it shifts attention away from its predecessor's focus and onto itself. The resulting method, INCREMEM, closes a span at every fixed increment of cumulative gain, so that each gist token carries an equal share of information rather than of tokens. Across three long-context benchmarks and three backbones, INCREMEM loses virtually no accuracy at half the memory of the uncompressed backbone and outperforms uniform gist compression and KV eviction in nearly every setting up to a compression ratio of 16. Adding only a lightweight gist attention module and gain estimator to a frozen LLM, it runs $3.4\times$ faster with 37.5% less peak memory than the uncompressed backbone at 128K tokens, within 3% of the latency of uniform gist compression. Our code is available at `https://anonymous.4open.science/r/IncreMem-7976`.

1 INTRODUCTION

Large Language Models (LLMs) have exhibited impressive capabilities in handling long input contexts (Brandon et al., 2024; Ainslie et al., 2023; Jiang et al., 2024a; Zhang et al., 2024), giving rise to LLM-based agentic applications that rely on long-term memory (OpenAI, 2026; Anthropic, 2026). However, self-attention in transformer-based LLMs has quadratic time complexity in sequence length, and storing the key-value (KV) cache of a long interaction history imposes a substantial memory footprint. A range of techniques reduce this cost by retaining only a subset or a condensed form of the lengthy memory, including retrieval-augmented generation (RAG) (Zhong et al., 2024; Shi et al., 2024; Gutiérrez et al., 2024), query-guided textual summarization (Jiang et al., 2024b; Xu et al., 2024; Yu et al., 2026; Yuan et al., 2026), or discrete KV eviction methods (Zhang et al., 2023; Li et al., 2024; Feng et al., 2025).

A growing body of work instead turns to the continuous latent space of LLMs, a medium more expressive than discrete text (Zou et al., 2026; Hao et al., 2025), to mitigate the cost of long-context processing. Early approaches condense the context into summary vectors, which are then re-encoded by an LLM that was never trained to interpret them (Chevalier et al., 2023; Ge et al., 2024; Yen et al., 2024); later work compresses the transformer's KV activations directly into gist tokens (Mu et al., 2023; Zhang et al., 2025), the same substrate that holds an agent's working memory (Zou et al., 2026). However, these methods compress text at a uniform rate, implicitly assuming that information is evenly distributed across tokens. As illustrated in Figure 1, a salient fact such as a name or a time is readily split across gist tokens (*Hart*|*ley*, *2:*|*30*), whereas fillers and restatements of content already in memory receive the same budget per token as new facts. This motivates our central

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.

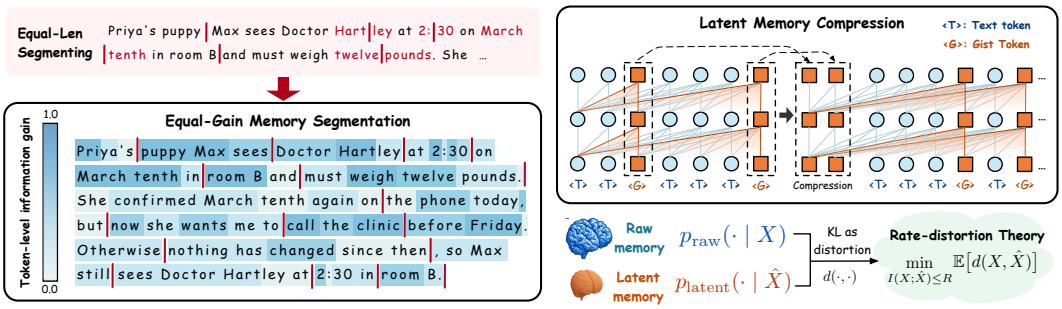


Figure 1: Overview of INCREMEM, which segments the memory at equal increments of information gain on the left and compresses each span into a gist token Mu et al. (2023) on the right.

research question: *how can an agent's long-term memory be compressed such that representational capacity is allocated in proportion to the information each token contributes?*

Addressing this question entails two challenges: (i) estimating, efficiently and online, the contribution of each token conditioned on the content already retained in memory; and (ii) translating these estimates into an allocation of a finite budget of latent tokens. Casting memory compression as a rate–distortion problem (Davisson, 1972; Cover & Thomas, 2001), we show that when the distortion of a memory unit is a convex function of the information it must hold, regardless of its length, the total distortion is minimized by giving every unit an equal share of gain, which reduces challenge (ii) to thresholding a cumulative token-level gain. For challenge (i), we derive the marginal information gain of each token from the attention distribution of a retrieval head of the LLM (Olsson et al., 2022; Wu et al., 2025b), and find that it decomposes into two terms: how far the token's attention shifts from that of the preceding token, and how much attention the token places on itself. Boundaries are placed where the accumulated gain reaches a threshold calibrated to the target compression ratio, with a minimum-gain refinement motivated by our observation that function words and subword continuations tend to carry lower gain. This method, which we call INCREMEM, places a span boundary at every fixed increment of cumulative information gain, so that every latent token carries a comparable share of the context's information.

Instantiating INCREMEM on an existing LLM requires only two lightweight components. The first is a gist attention module, inserted into each self-attention layer, which distills the activations of the text spans into those of gist tokens, one placed after each span (Mu et al., 2023; Zhang et al., 2025; Deng et al., 2025). The second is an auxiliary gain estimator that approximates the attention of the retrieval head from the backbone's input embeddings, since reading the gain from the head itself would require an uncompressed forward pass, precisely the cost that compression is meant to eliminate. The gist attention module is trained to minimize the Kullback–Leibler (KL) divergence (Kullback & Leibler, 1951) between the outputs of the memory-compressed and original models, the very distortion that our analysis minimizes, while the gain estimator and the backbone remain frozen.

Through extensive experiments on long-term memory benchmarks, we demonstrate that allocating latent capacity in proportion to information gain yields a compressed memory that remains faithful to the uncompressed one across a wide range of compression ratios. Halving the memory costs INCREMEM virtually no accuracy relative to the uncompressed backbone, and it stays ahead of uniform gist compression and KV eviction in nearly every setting as the memory shrinks further. The two lightweight modules add little cost, as INCREMEM runs $3.4\times$ faster with 37.5% less peak memory than the uncompressed backbone at 128K tokens, with latency within 3% of uniform gist compression. As LLM agents increasingly reason and recall in the latent space (Zou et al., 2026; Hou et al., 2026), these findings establish information gain as a principled unit for allocating it, pointing toward latent memories that expand in increments of information rather than at a fixed rate.

2 RELATED WORKS

Long-Term Agent Memory. LLM-based agents increasingly operate over interaction histories that far exceed a single context window. Recent benchmarks accordingly assess long-context understanding (Bai et al., 2025) and the ability to recall, update, and reason over information introduced

much earlier in an interaction (Wu et al., 2025a; Tavakoli et al., 2026). The prevailing paradigm maintains memory externally in textual form: past interactions are stored, optionally organized into summaries or structured indexes, and retrieved on demand (Zhong et al., 2024; Shi et al., 2024; Gutiérrez et al., 2024; Packer et al., 2024). Rather than selecting a subset of the history, our method preserves the entire history in a compressed latent form that remains accessible to attention.

Latent Space in LLMs. Beyond discrete tokens, a transformer maintains two continuous representations that recent work has repurposed for computation and storage. Hidden representations, the vectors propagated between transformer blocks at each position, can be trained to encode in a few vectors what would otherwise require many tokens, as exemplified by latent reasoning (Hao et al., 2025; Shen et al., 2025; Li et al., 2026). The KV cache, comprising the key and value vectors retained at every position for subsequent attention, constitutes the transformer's working memory (Zou et al., 2026; Hou et al., 2026). We therefore compress the KV cache directly, operating on the memory itself rather than on the inputs from which it is derived.

Context Compression. Hard prompt compression shortens the context while keeping it as text, either by pruning tokens of low self-information (Li et al., 2023; Jiang et al., 2023) or low query relevance (Jiang et al., 2024b), or by summarizing it (Xu et al., 2024; Yu et al., 2026; Yuan et al., 2026) at the cost of an autoregressive generation pass. KV-cache eviction instead discards entries that receive little attention (Zhang et al., 2023; Li et al., 2024), with later variants allocating the budget non-uniformly across heads and layers (Feng et al., 2025; Cai et al., 2025) or refining how entries are scored or merged (Liu et al., 2025; Yang & Shen, 2026; Tian et al., 2026). Soft-token compression condenses the entire context rather than discarding part of it, either into summary vectors that the LLM must re-encode (Chevalier et al., 2023; Ge et al., 2024; Yen et al., 2024) or into the KV activations of special *gist* tokens trained to absorb the preceding text (Mu et al., 2023; Kim et al., 2024; Zhang et al., 2025; Deng et al., 2025), the line of work on which we build. Across these families, compression is either guided by the user's query, which is unavailable when memory must be formed before future queries are known, or applied at a uniform rate irrespective of content. INCREMEM is query-agnostic yet content-aware, allocating latent capacity in proportion to the information each token contributes.

3 PRELIMINARY

Compressing an agent's long-term memory is a question of where to spend a limited budget: the agent must decide how finely to retain each part of a long context without knowing which details a later query will require. We cast this decision as a rate–distortion problem and derive, under a modeling assumption on how distortion grows with gain, that it is minimized when every unit of latent memory holds the same information gain.

Rate–Distortion Theory. Consider a source X and a lossy reconstruction $\hat{X}$ produced by a channel $f(\hat{x} \mid x)$. The rate $I(X;\hat{X})$ measures how much information the reconstruction retains about the source, and a distortion measure $d(X,\hat{X})$ quantifies the fidelity that is lost. The distortion–rate function (Davisson, 1972; Cover & Thomas, 2001) is the least expected distortion attainable when the rate is capped at R:

$$D(R) \;=\; \min_{f(\hat{x}|x)\,:\,I(X;\hat{X})\le R} \mathbb{E}\big[d(X,\hat{X})\big]. \tag{1}$$

In our setting, $X = (x_1,\dots,x_N)$ is the long context, where each token x_i carries an information gain $g_i \in (0,1)$. Boundaries $0 = \tau_0 < \tau_1 < \cdots < \tau_k = N$ partition X into k contiguous spans $X_j := X_{(\tau_{j-1},\tau_j]}$. A compression module Θ_c maps each span to a memory unit of fixed capacity C, measured in the same units as the gain, and the compressed memory $\hat{X} = \{\Theta_c(X_j)\}_{j=1}^{k}$ plays the role of the reconstruction.

The distortion caused by compression is measured by the KL divergence between the distributions that the LLM Θ_{LM} assigns to continuations Y of the context with and without compression,

$$d(X,\hat{X}) \;=\; D_{\mathrm{KL}}\Big(p_{\Theta_{LM}}(Y \mid X) \;\Big\|\; p_{\Theta_{LM},\Theta_c}(Y \mid \hat{X})\Big), \tag{2}$$

which vanishes when the compressed model predicts exactly as the full model does. With k and Θ_c fixed, the total capacity kC plays the role of the rate, and since $D(kC)$ is intractable for an LLM, we restrict the channels in Eq. 1 to those that Θ_c induces through a choice of boundaries $\tau_1,\dots,\tau_{k-1}$.

Optimal Memory Allocation. Tokens differ in how much new information they bring, which we quantify by the per-token information gain g_i (detailed in §4.1). The long context thus holds a total gain $V := \sum_{i=1}^{N} g_i$ that is independent of the boundaries, and placing the boundaries amounts to deciding how much of it, $V_j := \sum_{i=\tau_{j-1}+1}^{\tau_j} g_i$, each of the k memory units must hold.

To find the optimal division, we express the distortion $d(X, \hat{X})$ through the span gains V_j, which are what the boundaries τ_j control. Following the order in which spans are compressed, the distortion decomposes exactly, as a telescoping sum, into the distortion that each span adds when it is compressed. We model the expected distortion added by a span as a common function $\phi(v)$ of its gain v alone, regardless of its length, on the premise that a unit of fixed capacity is burdened by the new content it retains, not by the number of its tokens. Given the span gains, the expected distortion then becomes $\mathbb{E}[d(X, \hat{X})] = \sum_{j=1}^{k} \phi(V_j)$, and since additional capacity yields diminishing returns, ϕ is convex, which leads to the following result.

Theorem 3.1 (Optimality of Equal-gain Segmentation)**.** *Under the additive decomposition of distortion (Lemma A.1) and a convex span distortion ϕ of the gain alone (Assumption A.2, Proposition A.3), every segmentation of X into k spans satisfies $\mathbb{E}[d(X, \hat{X})] \geq k\,\phi(V/k)$, with equality at the* equal-gain segmentation $V_1 = \cdots = V_k = V/k$ *whenever the token gains admit one.*

The proofs of the statements in this section are given in §A. In practice, the context arrives as a stream of unknown length, so V is not available in advance. We therefore end a span whenever its accumulated gain reaches a fixed threshold θ, set by the target compression ratio, so that every completed span holds a gain between θ and $\theta + 1$.

4 METHODOLOGY

4.1 TOKEN-LEVEL INFORMATION GAIN

Formulation. Consider a causal attention head of the LLM Θ_{LM} reading the context $x_1, \ldots, x_i$. At position i, the head attends to every position $m \leq i$ with weight $a_{i,m}$ and returns the output $\mathbf{o}_i$:

$$a_{i,m} = \mathrm{softmax}_m\left(\frac{\mathbf{q}_i^\top \mathbf{k}_m}{\sqrt{d_{\text{head}}}}\right), \qquad \mathbf{o}_i = \sum_{m=1}^{i} a_{i,m}\,\mathbf{v}_m, \tag{3}$$

where $\mathbf{q}_i$, $\mathbf{k}_m$ and $\mathbf{v}_m$ are the query, key and value states of the head, d_{head} is its dimension, and $\{(\mathbf{k}_m, \mathbf{v}_m)\}_{m \leq i}$ is the key–value cache carried forward at inference. The weights $\mathbf{a}_i = (a_{i,1}, \ldots, a_{i,i})$ form a probability distribution over the context, so $\mathbf{o}_i$ is a weighted average of the cached values, that is, what the head reads from the context at position i.

Our goal is to quantify the information that each token x_i adds to the context. Attention is the only pathway through which a transformer reads its context, so we characterize the information gain of x_i by the marginal change it induces in the head output, $\mathbf{o}_i - \mathbf{o}_{i-1}$. For $i \geq 2$, the magnitude of this change is bounded as (Proposition B.1)

$$\|\mathbf{o}_i - \mathbf{o}_{i-1}\| \leq \frac{D_i}{2}\Big(\sum_{m<i} |a_{i,m} - a_{i-1,m}| + a_{i,i}\Big), \tag{4}$$

where $D_i := \max_{m,m' \leq i} \|\mathbf{v}_m - \mathbf{v}_{m'}\|$ is the largest distance between two cached values. The head output can thus change only as far as the attention distribution changes, and the bound separates two sources of this change. The first term, the *retrieval shift*, measures how much the attention shifts over the positions visible to both x_{i-1} and x_i. The second term, the *novelty weight*, is the attention that the head places on x_i itself, which has no counterpart in $\mathbf{o}_{i-1}$.

Segmentation accumulates the gains along the stream until they reach a fixed threshold (§3), so a unit of information gain must carry the same meaning at every position. The bound in Eq. 4 fails this requirement on two counts: its scale D_i varies with the cached values, and its absolute differences $|a_{i,m} - a_{i-1,m}|$ add up over the weakly attended positions, whose number grows with the context. We therefore measure both sources through a cosine distance between $\mathbf{a}_{i-1}$ and $\mathbf{a}_i$, which, unlike the bound, is free of the value scale and dominated by the positions that the head actually retrieves.

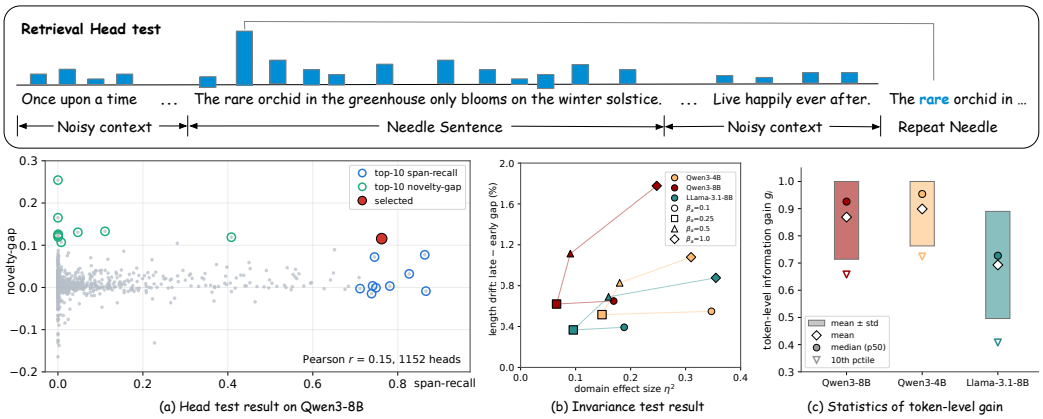


Figure 2: Retrieval head selection and empirical validation of the token-level information gain.

With $a_{i-1,i} := 0$, the novelty weight is absent from the inner product of the two distributions and enters only through the norm of $\mathbf{a}_i$. For $i \geq 2$, the token-level information gain is

$$g_i := 1 - \frac{\sum_{m<i} a_{i,m}\, a_{i-1,m}}{\sqrt{\sum_{m<i} a_{i,m}^2 + a_{i,i}^{2\beta_a}}\ \sqrt{\sum_{m<i} a_{i-1,m}^2}}, \tag{5}$$

where the exponent $\beta_a \in (0, 1]$ rescales the novelty weight, and $\beta_a = 1$ recovers the exact cosine distance between $\mathbf{a}_i$ and $\mathbf{a}_{i-1}$. Since the softmax normalizes over all i positions, $a_{i,i}$ tends to shrink in long contexts, and $\beta_a < 1$ counteracts this drift by lifting small values proportionally more than large ones, which we verify empirically below. The resulting gain lies in $(0, 1)$, and it is large when either source is large and small only when both are small (Proposition B.2).

Head Selection. Attention heads differ widely in function (Olsson et al., 2022; Wu et al., 2025b; Ren et al., 2024), and the information gain in Eq. 5 is informative only when it is computed from a head that exhibits two behaviours: it retrieves the earlier content that the current token relates to, and it responds to the arrival of new content. We test both behaviours in a single pass by extending the needle-in-a-haystack (NIAH) probe of retrieval heads (Wu et al., 2025b) with a second occurrence of the needle: a needle inserted into a long document is repeated in an sentence appended at the end, so that its content appears first as new (positions occ_1) and later as a repeat (positions occ_2). For each head h with attention weights $a_{i,m}^h$, we measure its retrieval ability by span-recall$_h$, the fraction of positions in occ_2 whose largest attention weight falls within occ_1, and sensitivity to novelty by novelty-gap$_h$, the mean self-attention weight $a_{i,i}^h$ over occ_1 minus that over occ_2.

We apply this retrieval-head test to Qwen3-4B, Qwen3-8B (Team, 2025), and Llama-3.1-8B (Grattafiori et al., 2024). As shown in Figure 2(a) and Figure 6, both behaviours are confined to a small minority of heads across all three models, consistent with the sparsity of retrieval heads (Wu et al., 2025b), and the two scores are only weakly correlated across heads. Retrieval and sensitivity to novelty are therefore distinct properties, and a head must be selected for both jointly. Ranking the heads by the two scores jointly singles out one head per backbone with the best combined rank, head 9 of layer 15 on Qwen3-4B and Qwen3-8B and head 31 of layer 8 on Llama-3.1-8B-Instruct, from which we compute g_i.The two Qwen3 backbones select the same head, which suggests that this function sits at a stable location within a model family. More details are in §B.2.

Empirical Validation. We validate the token-level information gain computed from the selected heads on all three backbones. A reliable gain should not depend on where a token appears in the context or on the domain of the context. Using 200 documents sampled from LongAlpaca (Chen et al., 2023), we measure (1) the length drift, the relative difference between the mean gain over the last and the first 30% of each document, and (2) the domain effect, the share of variance in the document-level mean gain explained by the domain, given by the effect size η^2 of a one-way ANOVA (Ross & Willson, 2017). As shown in Figure 2(b), where lower values are better on both axes, $\beta_a = 0.25$ gives the lowest length drift and the lowest domain effect on every backbone, with a drift below 1% throughout, so we adopt it in the rest of the paper.

We examine the resulting token-level gain on 200K raw tokens per backbone. As shown in Figure 2(c), the mean gain ($\pm$ standard deviation) is 0.87 ± 0.16 on Qwen3-8B, 0.88 ± 0.14 on Qwen3-4B, and 0.69 ± 0.20 on Llama-3.1-8B-Instruct. The distributions are skewed toward the upper bound, so most tokens bring new information, and their lower tails reveal the distinct functions of tokens. Across all three backbones, function words are 1.5 to 2 times as likely as content words to fall in the lowest decile, and the token that begins a word outscores the pieces that complete it in most multi-token words, as in "unreasonably", whose pieces "un", "reason", and "ably" receive 0.959, 0.941, and 0.644. More details can be found in §B.3. Together, these findings highlight the effectiveness of our attention-based formulation of token-level information gain in locating the information increment that each token contributes to its context.

4.2 Memory Compression

Threshold Segmenting. Following Theorem 3.1, span boundaries are placed so that every span holds about the same amount of gain. We set a target compression ratio L, the average number of raw tokens that one memory unit holds. The ratio is realized through a threshold θ: a span is closed once the gain accumulated since the previous boundary reaches θ. Motivated by our observation that word-internal tokens and function words tend to receive a lower gain (§4.1), we apply a *minimum-gain refinement* when thresholding, so that a span is less likely to end in the middle of a unit of content. Formally, the boundary is placed at the position of the least gain within a window of half-width Δ around the position t^* at which the accumulated gain reaches θ:

$$\tau_j := \arg\min_{|t-t^*|\le\Delta} g_t \quad \text{with} \quad t^* = \min\Big\{t : \sum_{i=\tau_{j-1}+1}^{t} g_i \ge \theta\Big\}, \tag{6}$$

where the window is restricted to the positions after τ_{j-1}, and the accumulation restarts after τ_j.

We determine θ and Δ on a subset of training data in §5.1: θ is searched so that the mean span length reaches the target L, and we empirically set $\Delta = 1, 1, 2$, and 2 for $L = 2, 4, 8$, and 16. By looking ahead by only Δ tokens, the minimum-gain refinement, combined with the gain, consistently reduces the boundaries that split a word, by 49% on average across backbones and compression ratios relative to equal-length segmentation, while the segmentation remains streaming. More details can be found in §B.4.

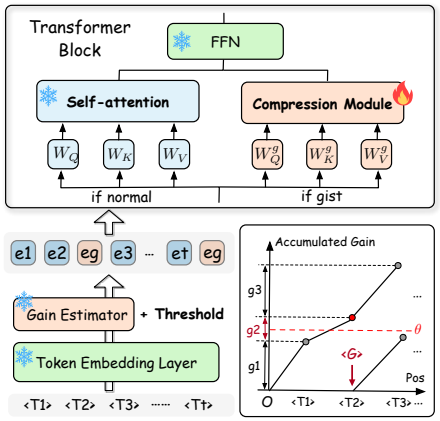


Figure 3: Memory Compression.

Activation Compression. With the boundaries in place, each span is written into memory through a gist token (Mu et al., 2023). Following Ge et al. (2024); Zhang et al. (2025), the context is processed chunk by chunk, with w raw tokens per chunk. Let X^i denote the i-th chunk and $X^i_{(\tau_{j-1},\tau_j]}$ its j-th span. Each span is followed by one gist token $\langle G\rangle^i_j$, so the chunk enters the backbone as

$$\tilde{X}^i = \big[\, X^i_{(\tau_0,\tau_1]}, \langle G\rangle^i_1,\ X^i_{(\tau_1,\tau_2]}, \langle G\rangle^i_2,\ \ldots,\ X^i_{(\tau_{k_i-1},\tau_{k_i}]}, \langle G\rangle^i_{k_i} \,\big], \tag{7}$$

where $k_i \approx w/L$ is the number of spans in the chunk under the target compression ratio L. A gist token carries a single learned embedding, shared by all gist tokens, and is distinguished from the raw tokens only by its projections: in every attention layer, the query, key, and value of a token at position t of $\tilde{X}^i$ with hidden state $\mathbf{h}_t$ are

$$(\mathbf{q}_t, \mathbf{k}_t, \mathbf{v}_t) = \begin{cases} (W^g_Q\mathbf{h}_t,\ W^g_K\mathbf{h}_t,\ W^g_V\mathbf{h}_t) & \text{if } t \text{ is a gist token,} \\ (W_Q\mathbf{h}_t,\ W_K\mathbf{h}_t,\ W_V\mathbf{h}_t) & \text{otherwise,} \end{cases} \tag{8}$$

where W_Q, W_K, W_V are the frozen projections of the backbone and W^g_Q, W^g_K, W^g_V are trained; the gist embedding and these projections constitute the module Θ_c of §3.

The chunk $\tilde{X}^i$ is encoded in one forward pass with plain causal attention over itself and over the memory retained from earlier chunks, namely the KV activations of the gist tokens $\langle G\rangle_1^1, \ldots, \langle G\rangle_{k_{i-1}}^{i-1}$, after which its raw tokens are released, so that w raw tokens leave only k_i memory units. Each gist token thus sees the memory already written and can encode what its span adds to it, as in the per-span decomposition of §3, while the number of memory units grows with the information in the context rather than with its length alone.

4.3 Learning Method

Training Objective. The module Θ_c is trained to minimize the distortion of Eq. 2, that is, to let the backbone predict from the compressed memory what it would predict from the raw context. We realize this as a token-level distillation in which the frozen backbone reading the uncompressed context serves as the teacher and the backbone reading the memory serves as the student:

$$\min_{\Theta_c} \sum_i \sum_{x_t \in X^i} \mathcal{D}_{\mathrm{KL}}\Big(p_{\Theta_{LM}}\big(x_t \mid X_{<t}\big) \,\Big\|\, p_{\Theta_{LM},\Theta_c}\big(x_t \mid \langle G\rangle_1^1, \ldots, \langle G\rangle_{k_{i-1}}^{i-1}, \tilde{X}^i_{<t}\big)\Big), \tag{9}$$

where $\mathcal{D}_{\mathrm{KL}}$ is the Kullback–Leibler (KL) divergence (Kullback & Leibler, 1951), $X_{<t}$ is the raw context preceding x_t, and $\tilde{X}^i_{<t}$ is the prefix of the chunk in Eq. 7 preceding x_t. During training, only the parameters of the compression module Θ_c are updated.

Gain Estimator. The information gain is defined on the attention of the selected head and is thus available only after the backbone has read the uncompressed context, which is the computation that compression is meant to avoid. We therefore approximate it with a lightweight gain estimator (Pagnoni et al., 2025; Owodunni et al., 2026), a three-layer causal decoder that reuses the frozen input embeddings of the backbone, is initialized from backbone layers ending at the layer of the selected head (Hou et al., 2026), and is trained to reproduce the attention distribution of that head. The estimator reads the raw context once ahead of compression, and its estimate $\hat{g}_i$ replaces g_i during segmentation, with θ calibrated on the estimated gain. It approximates the token-level gain closely, with a mean absolute error of at most 4.8% across the three backbones (§B.5), and adds only 3% to the latency (§5.2). It is kept frozen while Θ_c is trained.

5 Experiment

5.1 Settings

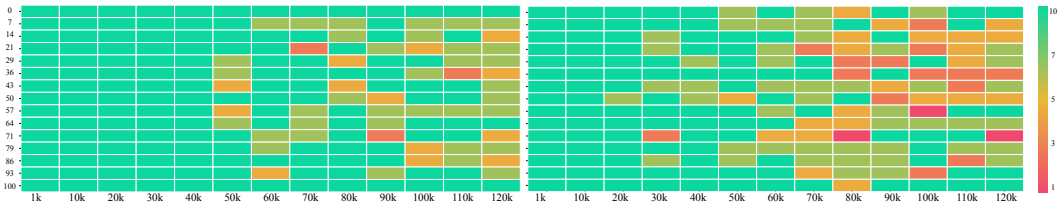
Implementation Details. We apply our method to Qwen3-4B, Qwen3-8B, and Llama-3.1-8B-Instruct. The compression module is pretrained on 1B tokens sampled from RedPajama (Weber et al., 2024) and then finetuned on long-context data from SynthLong (Lazarevich et al., 2025), BookSum (Kryscinski et al., 2022), and LongAlpaca (Chen et al., 2023), with the backbone and the gain estimator kept frozen. The chunk size w is 2048 tokens for the Qwen3 backbones and 1024 for Llama-3.1-8B. During training, the compression ratio of each chunk is drawn at random (Zhang et al., 2025; Deng et al., 2025), from $\{2, 4, 8, 16, 32\}$ in pretraining and from $\{2, 4, 8\}$ in finetuning; at inference, a single ratio is used for all chunks of a context. All models are trained on four NVIDIA RTX Pro 6000 GPUs with 96 GB of memory.

Baselines. We compare against two classes of baselines on the same backbones. The first keeps the context within the model, namely the uncompressed backbone (FullKV); the KV-eviction methods H2O (Zhang et al., 2023), SnapKV (Li et al., 2024), and AdaKV (Feng et al., 2025), which drop cached KV pairs by attention scores at our compression ratios; and Beacon (Zhang et al., 2025), which compresses activations into gist tokens with fixed-length spans and a next-token-prediction objective, and is otherwise trained as ours. The second is RAG, which retrieves the five snippets most relevant to the user's query, either user–assistant turn pairs or 1024-token windows following Tavakoli et al. (2026), using BM25 or a dense retriever built on bge-small-en-v1.5 (Xiao et al., 2024).

Benchmarks and Metrics. We evaluate on three long-context benchmarks, LongBench-v2 (Bai et al., 2025), LongMemEval (Wu et al., 2025a), and BEAM (Tavakoli et al., 2026), which together cover long-term memory over multi-session interaction histories as well as understanding and reasoning over long documents and structured data, with contexts of up to 128K tokens. We further use NIAH (gkamradt, 2023) to examine retrieval accuracy across context lengths and needle depths. We

Table 1: Evaluation on LongBench-v2 (Bai et al., 2025), LongMemEval (Wu et al., 2025a), and BEAM (Tavakoli et al., 2026) with three backbones at compression ratios from 2 to 16. Among the compressed methods, the best result in each column is in **bold** and the second best is underlined.

	Method	LongBench-v2				LongMemEval				BEAM			
		×2	×4	×8	×16	×2	×4	×8	×16	×2	×4	×8	×16
Llama-3.1-8B	RAG (*BM25* / *Dense*)	25.4 / 26.2				53.6 / 55.6				34.1 / 34.9			
	FullKV	29.2				44.0				27.9			
	H2O	26.7	26.2	25.9	23.5	29.4	21.4	17.6	15.4	20.3	17.3	16.8	12.3
	SnapKV	27.1	25.5	25.2	24.1	30.1	23.2	19.2	17.0	22.8	19.9	17.5	15.1
	AdaKV	27.7	26.7	26.5	24.8	33.3	24.6	21.8	20.6	24.8	21.6	16.8	14.9
	Beacon	27.4	25.8	25.9	25.8	40.5	38.8	35.6	29.2	23.9	23.0	19.3	17.7
	INCREMEM (Ours)	**29.2**	**27.8**	**27.0**	**26.4**	**44.8**	**43.2**	**38.6**	**33.4**	**26.3**	**26.1**	**24.6**	**20.5**
Qwen3-8B	RAG (*BM25* / *Dense*)	29.0 / 31.8				52.4 / 56.4				35.1 / 35.6			
	FullKV	34.8				47.2				28.6			
	H2O	27.8	26.8	26.0	25.2	33.6	23.2	19.8	16.2	22.8	19.4	17.9	17.4
	SnapKV	28.2	28.0	27.2	27.0	35.6	26.0	22.0	18.3	26.3	20.2	18.4	17.0
	AdaKV	28.4	27.0	26.8	23.3	35.2	27.2	22.2	20.6	27.1	22.3	22.2	21.0
	Beacon	31.2	30.4	28.9	28.0	41.0	36.4	31.4	24.0	28.2	26.6	24.6	24.0
	INCREMEM (Ours)	**35.4**	**31.0**	**30.2**	**29.1**	**48.8**	**42.3**	**38.7**	**31.2**	**32.2**	**28.3**	**26.3**	**25.7**
Qwen3-4B	RAG (*BM25* / *Dense*)	29.4 / 30.6				53.8 / 54.4				33.8 / 34.5			
	FullKV	30.2				42.2				26.9			
	H2O	27.4	26.6	26.0	25.5	23.2	19.6	16.4	12.8	20.3	18.5	16.9	16.2
	SnapKV	28.6	28.4	**27.4**	26.0	26.0	21.6	17.4	15.2	21.6	19.8	17.6	17.3
	AdaKV	29.4	28.8	27.0	22.7	36.0	23.0	18.4	16.6	25.5	21.8	20.7	19.8
	Beacon	28.9	28.0	25.2	26.5	36.8	35.1	31.7	27.4	25.3	23.2	21.8	**21.3**
	INCREMEM (Ours)	**33.4**	**28.9**	**27.4**	**27.1**	**42.3**	**39.0**	**35.2**	**28.3**	**27.8**	**23.8**	**22.3**	20.8



(a) Qwen3-8B w. IncreMem (b) Qwen3-8B w. Beacon

Figure 4: Needle-in-a-Haystack test (gkamradt, 2023) results on Qwen3-8B with x8 compression ratio. Results on LLama-3.1-8B and Qwen3-4B can be found in Figure 7.

follow the official protocol of each benchmark, reporting multiple-choice accuracy on LongBench-v2 and LLM-as-judge scores on the other three, with the prompts provided by the benchmarks and Gemini-3.7-Flash as the judge. More details about experimental setup are provided in §C.1.

5.2 MAIN RESULTS

As shown in Table 1, INCREMEM achieves the highest accuracy among the compressed memories across nearly all backbones, benchmarks, and compression ratios, and three findings stand out. First, INCREMEM *preserves the quality of the memory at a compression ratio of* 2, performing on par with FullKV with half of its memory and narrowing the gap to RAG frameworks, which retrieves from the full history with the user's query in hand. Second, *compressing the memory retains what evicting it discards*. As the compression ratio rises to 16, the KV-eviction baselines lose nearly half of their accuracy on LongMemEval, whereas INCREMEM keeps every span in compressed form and still leads them by more than 10 points. Third, *the advantage over uniform compression comes from where the boundaries are placed*. Beacon, which uses equal-length spans and a next-token-prediction objective, trails INCREMEM in almost all settings, and our ablation that reverts only the spans to equal length trails by a similar margin (Table 2), identifying equal-gain segmentation as the main source of the advantage.

We further conduct the Needle-in-a-Haystack test to assess the long-context retrieval ability of our memory compression method, with the needle buried at 15 depths across contexts of up to 120K tokens. As shown in Figure 4, INCREMEM preserves fine-grained retrieval in a $\times 8$ compressed memory on Qwen3-8B. It recovers the needle at every depth for contexts up to 40K tokens, a range in which Beacon already misses several needles, and its accuracy declines only gradually up to 120K. Beacon, by contrast, fails at many depths beyond 70K, with its errors concentrated in the middle of the context. Llama-3.1-8B-Instruct and Qwen3-4B show the same pattern in Figure 7. The advantage of INCREMEM thus extends beyond aggregate benchmark scores to fine-grained fact retrieval, even though its memory is written before the query is known.

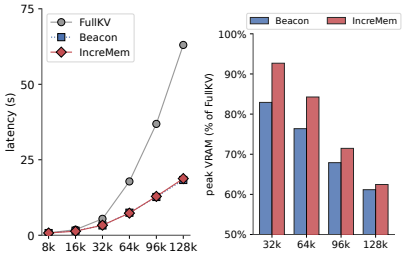


Figure 5: Efficiency analysis on Qwen3-8B with more in §C.2.

We conduct the efficiency analysis on a single NVIDIA Pro6000-96GB GPU. We reuse the NIAH documents and average over 30 per context length (8K–128K) and over compression ratios of 2, 4, 8, and 16. As shown in Figure 5, at 128K context, INCREMEM runs $3.4\times$ faster than the full backbone with 37.5% less peak memory on Qwen3-8B. Compared with Beacon, which lacks the gain estimator, INCREMEM's latency stays within 3% of Beacon's at every length, and its peak-memory gap to Beacon shrinks sharply with longer context, from 9.8 points of FullKV's at 32K to 1.3 points at 128K. Llama-3.1-8B-Instruct and Qwen3-4B show the same trend in Figure 8.

5.3 ABLATION STUDIES

We ablate each component on Qwen3-8B with LongMemEval in Table 2. First, accuracy falls with each shallower gain estimator, in the order of its estimation error (§B.5), and shuffling the estimated gains within each chunk, which keeps a similar distribution of span lengths but detaches the gains from their tokens, falls below even equal-length segmentation, so variable span lengths pay off only when the gain decides where they fall. Second, equal-length segmentation under our objective, the controlled counterpart of Beacon, trails INCREMEM at every ratio, and equalizing span gains accounts for most of this gap, with the minimum-gain refinement adding a further, smaller improvement by ending spans at low-gain tokens. A rule-based refinement that snaps equal-length boundaries to the nearest function word, punctuation mark, or word-final token recovers little of this gap, showing that the gain matters for deciding which tokens deserve a boundary, which a tokenization rule does not capture. Third, the training recipe amplifies equal-gain segmentation, where KL distillation and chunk-wise random ratios both contribute consistently.

Table 2: Ablation study on Qwen3-8B.

Method	**LongMemEval**			
	$\times 2$	$\times 4$	$\times 8$	$\times 16$
INCREMEM	48.8	42.3	38.7	31.2
Gain signal				
w. 2-layer estimator	45.6	40.1	36.6	27.3
w. 1-layer estimator	43.1	38.2	35.1	24.7
w. shuffled gain	37.4	32.5	22.1	19.8
Segmentation				
w/o min-gain refine	46.3	40.7	37.1	28.6
w. equal-len seg	40.3	36.9	32.5	23.9
w. rule-based refine	41.8	37.2	33.4	25.7
Training				
w/o KL distillation	46.2	39.7	35.9	28.5
w/o chunk-wise ratio	46.7	38.4	36.0	27.7

6 CONCLUSION

This work rests on the premise that the capacity of a latent memory should follow the information in the context rather than its length. Casting memory compression as a rate–distortion problem, we showed that equal-gain spans minimize distortion when it is a convex function of a unit's gain alone, and grounded this gain in a retrieval head. INCREMEM closes a span at every fixed increment of cumulative gain and trains its gist tokens to minimize this same distortion. Halving the memory costs it virtually no accuracy, and it stays ahead of KV eviction and uniform compression as the ratio grows, with ablations tracing its edge over uniform compression to where the gain places span boundaries. Information gain thus emerges as a principled unit for organizing latent memory, pointing toward memories that grow and consolidate over an agent's lifetime in increments of the information it acquires.

AI USE STATEMENT

In this work, we used generative AI tools to polish writing and refine mathematical derivations. We have not used generative AI tools for literature retrieval or discovery, research ideation, experimental execution, drafting paper sections, generating synthetic datasets, or proving mathematical claims. We additionally used generative AI models as an LLM-as-Judge metric where required by the official evaluation protocols of the benchmarks we use. All AI-generated content was manually reviewed and verified by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Anthropic. Claude code. https://claude.com/product/claude-code, 2026. Accessed: 2026-09-19.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. 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. 3639–3664, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.183. URL https://aclanthology.org/2025.acl-long.183/.

Rajendra Bhatia and Chandler Davis. A better bound on the variance. *The American Mathematical Monthly*, 107(4):353–357, 2000. ISSN 00029890, 19300972. URL http://www.jstor.org/stable/2589180.

Steven Bird and Edward Loper. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pp. 214–217, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/P04-3031/.

William Brandon, Mayank Mishra, Aniruddha Nrusimha, Rameswar Panda, and Jonathan Ragan-Kelley. Reducing transformer key-value cache size with cross-layer attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=M2UzLRoqic.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=ayi7qezU87.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. *ArXiv*, abs/2309.12307, 2023. URL https://api.semanticscholar.org/CorpusID:262084134.

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/.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, 2001. ISBN 9780471062592. doi: 10.1002/0471200611. URL https://doi.org/10.1002/0471200611.

L. Davisson. Rate distortion theory: A mathematical basis for data compression. *IEEE Transactions on Communications*, 20(6):1202–1202, 1972. doi: 10.1109/TCOM.1972.1091311.

Chenlong Deng, Zhisong Zhang, Kelong Mao, Shuaiyi Li, Xinting Huang, Dong Yu, and Zhicheng Dou. A silver bullet or a compromise for full attention? a comprehensive study of gist token-based context 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. 4861–4879, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.241. URL https://aclanthology.org/2025.acl-long.241/.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Ada-KV: Optimizing KV cache eviction by adaptive budget allocation for efficient LLM inference. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=tcisuhGsQZ.

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 B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 2591–2607, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/0b276510ec2d3f6613a8b60c41ff0438-Paper-Conference.pdf.

gkamradt. Llmtest_needleinahaystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 59532–59569. Curran Associates, Inc., 2024. doi: 10.52202/079017-1902. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/6ddc001d07ca4f319af96a3024f6dbd1-Paper-Conference.pdf.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Yubo Hou, Zhisheng Chen, Tao Wan, and Zengchang Qin. FlashMem: Distilling intrinsic latent memory via computation reuse. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 4687–4705, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.230. URL https://aclanthology.org/2026.findings-acl.230/.

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, YUCHENG LI, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a. URL https://openreview.net/forum?id=fPBACAbqSN.

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 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.91. URL https://aclanthology.org/2024.acl-long.91/.

Jang-Hyun Kim, Junyoung Yeom, Sangdoo Yun, and Hyun Oh Song. Compressed context memory for online language model interaction. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=64kSvC4iPg.

Wojciech Kryscinski, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. BOOKSUM: A collection of datasets for long-form narrative summarization. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 6536–6558, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.488. URL https://aclanthology.org/2022.findings-emnlp.488/.

Solomon Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951. URL https://api.semanticscholar.org/CorpusID:120349231.

Ivan Lazarevich, David Bick, Harsh Gupta, Srinjoy Mukherjee, Nishit Neema, Gokul Ramakrishnan, and Ganesh Venkatesh. Extending llm context with 99% less training tokens. Cerebras Blog, 2025. URL https://www.cerebras.ai/blog/extending-llm-context-with-99-less-training-tokens. Accessed: 2026-09-18.

Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=j84WR5ORsC.

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/.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 22947–22970. Curran Associates, Inc., 2024. doi: 10.52202/079017-0722. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/28ab418242603e0f7323e54185d19bde-Paper-Conference.pdf.

Xiang Liu, Zhenheng Tang, Peijie Dong, Zeyu Li, Liuyue, Bo Li, Xuming Hu, and Xiaowen Chu. ChunkKV: Semantic-preserving KV cache compression for efficient long-context LLM inference. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=20JDhbJqn3.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=2DtxPCL3T5.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish,

and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

OpenAI. Codex. https://openai.com/codex/, 2026. Accessed: 2026-09-19.

Abraham Toluwase Owodunni, Orevaoghene Ahia, and Sachin Kumar. FLEXITOKENS: Flexible tokenization for evolving language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17170–17190, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.848. URL https://aclanthology.org/2026.findings-acl.848/.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024. URL https://arxiv.org/abs/2310.08560.

Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason E Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srini Iyer. Byte latent transformer: Patches scale better than tokens. 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. 9238–9258, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.453. URL https://aclanthology.org/2025.acl-long.453/.

Jie Ren, Qipeng Guo, Hang Yan, Dongrui Liu, Quanshi Zhang, Xipeng Qiu, and Dahua Lin. Identifying semantic induction heads to understand in-context learning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6916–6932, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.412. URL https://aclanthology.org/2024.findings-acl.412/.

Amanda Ross and Victor L. Willson. *One-Way Anova*, pp. 21–24. SensePublishers, Rotterdam, 2017. ISBN 978-94-6351-086-8. doi: 10.1007/978-94-6351-086-8_5. URL https://doi.org/10.1007/978-94-6351-086-8_5.

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/.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. REPLUG: Retrieval-augmented black-box language models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8371–8384, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.463. URL https://aclanthology.org/2024.naacl-long.463/.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2158–2170, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.195. URL https://aclanthology.org/2020.acl-main.195/.

Mohammad Tavakoli, Alireza Salemi, Carrie Ye, Mohamed Abdalla, Hamed Zamani, and J Ross Mitchell. Beyond a million tokens: Benchmarking and enhancing long-term memory in LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=y59hf5lrMn.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Haolin Tian, Yuzhe Liu, and Tonghan Wang. Every cache entry earns its place: Global allocation of resolution and coverage for kv cache compression, 2026. URL https://arxiv.org/abs/2608.07001.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. Redpajama: an open dataset for training large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Frank Wilcoxon. Individual comparisons by ranking methods. In Samuel Kotz and Norman L. Johnson (eds.), *Breakthroughs in Statistics: Methodology and Distribution*, pp. 196–202. Springer New York, New York, NY, 1992. ISBN 978-1-4612-4380-9. doi: 10.1007/978-1-4612-4380-9_16. URL https://doi.org/10.1007/978-1-4612-4380-9_16.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=pZiyCaVuti.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 62143–62156, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/9b77f07301b1ef1fe810aae96c12cb7b-Paper-Conference.pdf.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings, 2024. URL https://arxiv.org/abs/2309.07597.

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.

Junzhe Yang and Xiaoyu Shen. Adaptive mass-segmented kv compression for long-context reasoning, 2026. URL https://arxiv.org/abs/2605.23200.

Howard Yen, Tianyu Gao, and Danqi Chen. Long-context language modeling with parallel context encoding. 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. 2588–2610, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.142. URL https://aclanthology.org/2024.acl-long.142/.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. Memagent: Reshaping long-context LLM with multi-conv RL-based memory agent. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=k5nIOvYGCL.

Qianhao Yuan, Jie Lou, Zichao Li, Jiawei Chen, Yaojie Lu, Hongyu Lin, Le Sun, Debing Zhang, and Xianpei Han. MemSearcher: Iterative memory integration for search agent via end-to-end reinforcement learning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 14965–14977, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.736. URL https://aclanthology.org/2026.findings-acl.736/.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Long context compression with activation beacon. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=1eQT9OzfNQ.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang "Atlas" Wang, and Beidi Chen. H2o: Heavy-hitter oracle for efficient generative inference of large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 34661–34710. Curran Associates, Inc., 2023. doi: 10.52202/075280-1506. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6ceefa7b15572587b78ecfcebb2827f8-Paper-Conference.pdf.

Zhihan Zhang, Yixin Cao, Chenchen Ye, Yunshan Ma, Lizi Liao, and Tat-Seng Chua. Analyzing temporal complex events with large language models? a benchmark towards temporal, 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. 1588–1606, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.87. URL https://aclanthology.org/2024.acl-long.87/.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: enhancing large language models with long-term memory. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i17.29946. URL https://doi.org/10.1609/aaai.v38i17.29946.

Jiaru Zou, Ruizhong Qiu, Gaotang Li, Xiyuan Yang, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, James Zou, Mengdi Wang, and Ling Yang. Latent collaboration in multi-agent systems. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=syG9I9ofd8.

A Theoretical Analysis

We follow the notation of Section 3. The context $X = (x_1, \ldots, x_N)$ has token gains $g_i \in (0, g_{\max}]$, with $g_{\max} = 1$ for the gain of §4.1, and total gain $V = \sum_i g_i$. Boundaries $0 = \tau_0 < \cdots < \tau_k = N$ define the spans X_j with gains V_j, the module Θ_c maps each span to a memory unit $\hat{X}_j$ of capacity C, and $d(X, \hat{X})$ is the distortion in Eq. 2. The gains are computed on the raw context and do not depend on the boundaries. A continuation Y is a token sequence of a fixed length, and since the output distribution of the model is a softmax, every continuation has positive probability under both the full and the memory-compressed model, so all log-probabilities below are finite. $\mathbb{E}[\cdot]$ denotes the expectation over contexts conditionally on the span gains $(V_1, \ldots, V_k)$, so the results below concern the expected distortion as a function of the span gains rather than the distortion of an individual context. Since the unconditional expectation in Eq. 1 averages this conditional one, choosing for each context the segmentation that minimizes the conditional expected distortion also minimizes the unconditional one.

A.1 Proof of Theorem 3.1

The proof shows in turn that the distortion of the memory is the sum of the distortions of its spans, that the expected distortion of a span is a convex function of its gain, and that a convex cost with a fixed total is minimized by an equal split.

Step 1. The distortion decomposes over spans. Let $p(Y \mid X) := p_{\Theta_{LM}}(Y \mid X)$ be the continuation distribution of the full model and $q(Y \mid M) := p_{\Theta_{LM},\Theta_c}(Y \mid M)$ that of the memory-compressed model reading a memory state M. As spans are compressed one after another, the memory passes through states $M_0, M_1, \ldots, M_k$, where the spans $X_1, \ldots, X_j$ are compressed in M_j and the remaining spans are still raw. Thus $M_k = \hat{X}$ and $M_0 = X$, for which $q(Y \mid M_0) = p(Y \mid X)$, since the backbone is frozen and Θ_c acts only on gist tokens, which M_0 does not contain. The distortion of span X_j is the additional loss incurred when it is compressed after the spans before it,

$$d_j \;:=\; \mathbb{E}_{Y\sim p(\cdot|X)}\big[\log q(Y \mid M_{j-1}) - \log q(Y \mid M_j)\big]. \tag{10}$$

Lemma A.1 (Additivity)**.** *The distortion of the memory is exactly the sum of the distortions of its spans,* $d(X, \hat{X}) = \sum_{j=1}^{k} d_j$.

Proof. The sum telescopes,

$$\begin{aligned}\sum_{j=1}^{k} d_j &= \mathbb{E}_{Y\sim p(\cdot|X)}\big[\log q(Y \mid M_0) - \log q(Y \mid M_k)\big] \\ &= \mathbb{E}_{Y\sim p(\cdot|X)}\big[\log p(Y \mid X) - \log q(Y \mid \hat{X})\big] \\ &= d(X, \hat{X}),\end{aligned}$$

where the last equality is the definition of the KL divergence in Eq. 2. □

The identity requires neither small nor independent compression errors and holds when a memory unit attends to the units written before it. An individual d_j is a difference of log-losses rather than a divergence and may be negative, so only its expectation enters the analysis.

Step 2. The expected distortion of a span is a convex function of its gain.

Assumption A.2 (Homogeneous spans)**.** *The expected distortion that a span adds depends on the span only through its gain, and in particular not on its length, its position, or how the spans before it were compressed, so that* $\mathbb{E}[d_j] = \phi(V_j)$ *with*

$$\phi(v) \;:=\; v\,\delta\!\left(\frac{C}{v}\right) \;\; \textit{for } v > 0, \qquad \phi(0) := 0, \tag{11}$$

where $\delta : (0, \infty) \to [0, \infty)$*, the expected distortion of one unit of gain stored with capacity* r*, is non-increasing and convex.*

The assumption is a modeling premise rather than a consequence of the architecture. Since every span is compressed by the same module into a unit of the same capacity, it posits that the difficulty of compressing a span is governed by the new content it carries rather than by the number of its tokens. If distortion followed a span's length instead, the same argument would favor equal-length spans, so the choice between the two segmentations rests on this premise. Table 2 tests its implications, as equal-length spans trail equal-gain spans at every ratio, and shuffling the gains within each chunk, which keeps variable span lengths but detaches them from the gain, performs worst. Because a memory unit may attend to the content before its span, spans are not compressed in isolation, and the assumption describes the expected effect of a span given the spans before it, in the streaming order of the context.

The proof uses only the convexity of ϕ, and Eq. 11 is one natural model of how it arises. A span with gain v holds v units of gain that share the capacity C, and for an integer v an equal share C/v per unit is the best split, since $\sum_{i=1}^{v} \delta(r_i) \geq v\,\delta(C/v)$ for any $r_1 + \cdots + r_v = C$ by Jensen's inequality, which Eq. 11 extends to real-valued gain. The conditions on δ state that additional capacity reduces the distortion with diminishing returns, as the distortion–rate function of a fixed source does (Cover & Thomas, 2001), which for the learned module is part of the assumption rather than a consequence of rate–distortion theory.

Proposition A.3 (ϕ is convex)**.** *Under Assumption A.2,* ϕ *is non-negative, non-decreasing and convex on* $[0, \infty)$*, and strictly convex on* $[0, \infty)$ *if* δ *is strictly convex.*

Proof. For $v_1, v_2 > 0$ and $\lambda \in [0,1]$, let $v = \lambda v_1 + (1-\lambda)v_2$ and $\alpha = \lambda v_1 / v \in [0,1]$, so that $\alpha \frac{C}{v_1} + (1-\alpha)\frac{C}{v_2} = \frac{C}{v}$. The convexity of δ gives

$$\phi(v) = v\,\delta\Big(\alpha\tfrac{C}{v_1} + (1-\alpha)\tfrac{C}{v_2}\Big) \;\le\; v\,\alpha\,\delta\big(\tfrac{C}{v_1}\big) + v\,(1-\alpha)\,\delta\big(\tfrac{C}{v_2}\big) \;=\; \lambda\,\phi(v_1) + (1-\lambda)\,\phi(v_2),$$

strictly if δ is strictly convex, $v_1 \neq v_2$ and $\lambda \in (0,1)$. For $v > 0$ and $t \in (0,1]$, since δ is non-increasing and non-negative,

$$\phi(tv) = tv\,\delta\big(\tfrac{C}{tv}\big) \;\le\; tv\,\delta\big(\tfrac{C}{v}\big) = t\,\phi(v) \;\le\; \phi(v).$$

With $t = 1-\lambda$, the first inequality reads $\phi\big(\lambda \cdot 0 + (1-\lambda)v\big) \le \lambda\,\phi(0) + (1-\lambda)\,\phi(v)$, which extends convexity to the endpoint $v = 0$, and the second, together with $\phi \ge 0 = \phi(0)$, shows that ϕ is non-decreasing. A strictly convex, non-increasing δ is strictly decreasing, since $\delta(a) = \delta(b)$ for some $a < b$ would give $\delta(c) < \delta(a) = \delta(b)$ for $a < c < b$, contradicting monotonicity, so the first inequality is then strict for $t < 1$ and strict convexity extends to the endpoint as well. □

Step 3. An equal split minimizes a convex cost. By Lemma A.1 and Assumption A.2, $\mathbb{E}[d(X, \hat{X})] = \sum_j \mathbb{E}[d_j] = \sum_j \phi(V_j)$, where the span gains always satisfy $\sum_j V_j = V$. Since ϕ is convex, Jensen's inequality gives

$$\frac{1}{k}\sum_{j=1}^{k} \phi(V_j) \;\ge\; \phi\Big(\frac{1}{k}\sum_{j=1}^{k} V_j\Big) = \phi\big(V/k\big),$$

with equality when $V_1 = \cdots = V_k = V/k$, and only then if ϕ is strictly convex. Since every $g_i > 0$, the cumulative gain is strictly increasing in the position, so at most one choice of boundaries attains $V_j = V/k$ for all j, and for strictly convex ϕ the minimizer is unique as a segmentation. The bound thus holds for every segmentation and is attained whenever the token gains admit an exactly equal split, and Remark A.4 bounds how far our threshold segmentation falls from it otherwise. □

Remark A.4 (Threshold segmentation)**.** *Our method does not know V or k in advance and instead ends a span at the first token at which the gain accumulated since the previous boundary reaches a threshold θ. The accumulated gain is below θ before this token and grows by at most $g_{\max}$ at it, so every completed span satisfies $\theta \le V_j < \theta + g_{\max}$, whereas equal-length spans of L tokens may have any gain in $(0, Lg_{\max})$. This bounded spread in turn bounds the excess distortion. Let $\bar{V}$ be the mean gain of the n completed spans, and suppose that ϕ is twice differentiable with $\phi'' \le M$ on $[\theta, \theta + g_{\max}]$. A second-order Taylor expansion of ϕ around $\bar{V}$, in which the first-order terms cancel because $\sum_j (V_j - \bar{V}) = 0$, gives*

$$\sum_{j=1}^{n} \phi(V_j) - n\,\phi(\bar{V}) \;\le\; \frac{M}{2}\sum_{j=1}^{n}(V_j - \bar{V})^2 \;\le\; \frac{n}{8}\,M g_{\max}^2,$$

since values in an interval of width $g_{\max}$ deviate from their mean by at most $g_{\max}^2/4$ in mean square (Bhatia & Davis, 2000), where $n\,\phi(\bar{V})$ is the lower bound of Theorem 3.1 for n spans of the same total gain. Relative to this optimum, the excess is at most $Mg_{\max}^2/(8\,\phi(\theta))$ whenever $\phi(\theta) > 0$, since $\bar{V} \ge \theta$ and ϕ is non-decreasing, so it shrinks as θ grows with L. The minimum-gain refinement moves each boundary by at most Δ tokens, so completed spans satisfy $\theta - \Delta g_{\max} \le V_j < \theta + (\Delta+1)g_{\max}$, and the bound holds with $g_{\max}$ replaced by $(2\Delta+1)g_{\max}$, provided $\phi'' \le M$ on this interval. Both guarantees are loose at $L = 2$, where $\theta \approx g_{\max}$ (Table 4). Span gains may then differ by about a factor of two, and with $\Delta = 1$ the refined interval spans nearly $[0, 3)$, wider than that of equal-length spans, so at this ratio the refinement rests on the boundaries it selects (§B.4) rather than on this bound.

B Token-Level Information Gain

B.1 Derivation

We use the notation of §4.1 and fix a position $i \ge 2$. The attention weights $\mathbf{a}_i = (a_{i,1}, \ldots, a_{i,i})$ are positive and sum to one, and $\mathbf{a}_{i-1}$ is extended to position i with $a_{i-1,i} := 0$, which changes neither its norm nor its inner products over the first $i-1$ positions. We write $\langle \cdot, \cdot \rangle$ for the inner product and $\|\cdot\|$ for the Euclidean norm.

Proposition B.1 (Bound on the change of the head output)**.** *For any value states* $\mathbf{v}_1, \ldots, \mathbf{v}_i$*, let* $D_i := \max_{m,m' \le i} \|\mathbf{v}_m - \mathbf{v}_{m'}\|$*. Then*

$$\|\mathbf{o}_i - \mathbf{o}_{i-1}\| \le \frac{D_i}{2} \Big(\sum_{m<i} |a_{i,m} - a_{i-1,m}| + a_{i,i} \Big).$$

Proof. Let $\delta_m := a_{i,m} - a_{i-1,m}$ for $m \le i$, so that $\mathbf{o}_i - \mathbf{o}_{i-1} = \sum_m \delta_m \mathbf{v}_m$ by Eq. equation 3, where both outputs read the same value states because the key–value cache is causal, and $\sum_m \delta_m = 0$ because both weight vectors sum to one. With $\delta_m^+ := \max(\delta_m, 0)$ and $\delta_m^- := \max(-\delta_m, 0)$, this gives $\sum_m \delta_m^+ = \sum_m \delta_m^- =: t$, where

$$2t = \sum_m |\delta_m| = \sum_{m<i} |a_{i,m} - a_{i-1,m}| + a_{i,i} > 0.$$

Hence

$$\begin{aligned} \mathbf{o}_i - \mathbf{o}_{i-1} &= t \Big(\sum_m \frac{\delta_m^+}{t} \mathbf{v}_m - \sum_{m'} \frac{\delta_{m'}^-}{t} \mathbf{v}_{m'} \Big) \\ &= t \sum_{m,m'} \frac{\delta_m^+ \delta_{m'}^-}{t^2} (\mathbf{v}_m - \mathbf{v}_{m'}), \end{aligned}$$

where the second equality uses $\sum_m \delta_m^+/t = \sum_{m'} \delta_{m'}^-/t = 1$. The coefficients $\delta_m^+ \delta_{m'}^-/t^2$ are non-negative and sum to one, so the triangle inequality gives $\|\mathbf{o}_i - \mathbf{o}_{i-1}\| \le t\, D_i$. □

Proposition B.2 (Decomposition of the information gain)**.** *Let* $\mathbf{a}_{i,<i} := (a_{i,1}, \ldots, a_{i,i-1})$ *and* $S_i := \|\mathbf{a}_{i,<i}\|^2$*. Define the contribution of the retrieval shift as the cosine distance between the two distributions over the shared positions, and the contribution of the novelty weight through its size relative to the attention placed on the past:*

$$g_i^{ret} := 1 - \frac{\langle \mathbf{a}_{i,<i},\, \mathbf{a}_{i-1} \rangle}{\|\mathbf{a}_{i,<i}\|\, \|\mathbf{a}_{i-1}\|},$$

$$g_i^{nov} := 1 - \Big(1 + \frac{a_{i,i}^{2\beta_a}}{S_i} \Big)^{-1/2}.$$

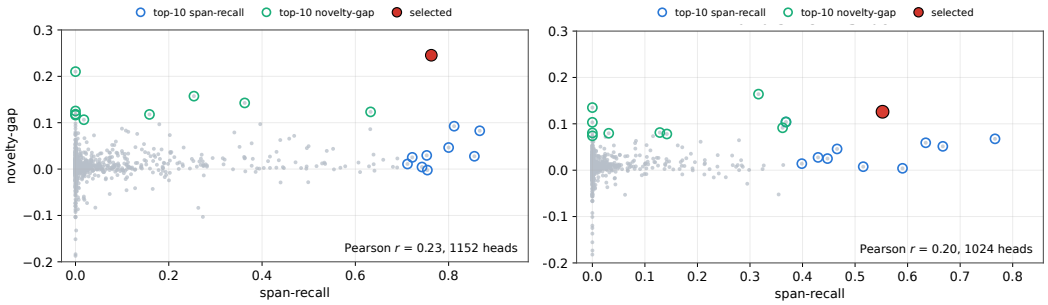
Then the gain g_i *of Eq. equation 5 satisfies* $1 - g_i = (1 - g_i^{ret})(1 - g_i^{nov})$*. Consequently,* $\max(g_i^{ret}, g_i^{nov}) \le g_i \le g_i^{ret} + g_i^{nov}$ *and* $0 < g_i < 1$.

Proof. The weights are positive, so $S_i > 0$. The numerator of the fraction in Eq. equation 5 is $\langle \mathbf{a}_{i,<i}, \mathbf{a}_{i-1} \rangle$ and its denominator is $\sqrt{S_i + a_{i,i}^{2\beta_a}}\, \|\mathbf{a}_{i-1}\| = \|\mathbf{a}_{i,<i}\| \left(1 + a_{i,i}^{2\beta_a}/S_i\right)^{1/2} \|\mathbf{a}_{i-1}\|$, which gives the identity. Let $c := 1 - g_i^{\text{ret}}$ and $\nu := 1 - g_i^{\text{nov}}$. The inner product of two positive vectors is positive and, by the Cauchy–Schwarz inequality, their cosine is at most one, so $c \in (0, 1]$, while $a_{i,i} > 0$ gives $\nu \in (0, 1)$. Hence $1 - g_i = c\,\nu \in (0, 1)$, the lower bound follows from $c\,\nu \le \min(c, \nu)$, and the upper bound from $1 - c\,\nu = (1 - c) + c\,(1 - \nu) \le (1 - c) + (1 - \nu)$. □

The gain is therefore large when either source is large and small only when both are small, and it is bounded by $g_{\max} = 1$, the constant used in Remark A.4. The exponent acts on g_i^{nov} alone: since $a_{i,i} < 1$ for $i \ge 2$, choosing $\beta_a < 1$ enlarges the ratio $a_{i,i}^{2\beta_a}/S_i$ by the factor $a_{i,i}^{-2(1-\beta_a)}$ relative to $\beta_a = 1$, which is larger for smaller $a_{i,i}$, as happens when a long context dilutes the novelty weight. For $\beta_a < 1$, g_i is no longer a cosine distance, but it retains the properties above, which are all that segmentation requires.

B.2 Head Selection

Setup. Our test builds on the needle-in-a-haystack (NIAH) protocol with which Wu et al. (2025b) detect retrieval heads: a needle, a short statement unrelated to its surroundings, is inserted at a controlled depth of a long document and queried at the end. We extend the protocol by appending the needle content as a teacher-forced answer after the query, so that the same content occurs twice,



(a) Head test result on Qwen3-4B (b) Head test result on Llama-3.1-8B

Figure 6: Retrieval head selection on (a) Qwen3-4B and (b) Llama-3.1-8B.

first as new content inside the document, at positions occ_1, and then as a repeat in the answer, at positions occ_2. Following Wu et al. (2025b), the evaluation grid spans 20 context lengths spaced logarithmically from 1K to 50K tokens and 10 needle depths spaced uniformly from the beginning to the end of the document. For each document, we record the attention weights $a^h_{i,m}$ of every head h in a single forward pass and compute

$$\text{span-recall}_h := \frac{1}{|\text{occ}_2|}\Big|\Big\{\, i \in \text{occ}_2 : \arg\max_{m \le i} a^h_{i,m} \in \text{occ}_1 \Big\}\Big|, \tag{12}$$

$$\text{novelty-gap}_h := \frac{1}{|\text{occ}_1|}\sum_{i \in \text{occ}_1} a^h_{i,i} - \frac{1}{|\text{occ}_2|}\sum_{i \in \text{occ}_2} a^h_{i,i}, \tag{13}$$

each averaged over the documents. Span-recall adapts the retrieval score of Wu et al. (2025b), measuring how often the repeat attends back to the needle, and novelty-gap measures how much more self-attention the head places on the needle when it first appears than when it recurs, which is the quantity underlying the novelty weight.

Results. We rank the heads of each backbone, 1152 on the Qwen3 backbones and 1024 on Llama-3.1-8B-Instruct, separately under the four measurements given by the two scores on the two needle types, and select the head with the lowest rank sum. Figure 2(a) and Figure 6 place every head by its two scores, averaged over the needle types. On each backbone, the selected head lies within the top ten of both scores, whereas the other heads in either top ten score low on the other; the two Qwen3 backbones select the same head and share the same five heads with the lowest rank sum.

Both behaviours are sparse. The median head scores below 0.01 on both, only about 6% of the heads on the Qwen3 backbones and 1.6% on Llama-3.1-8B-Instruct exceed a span-recall of 0.3, and 3.3–4.6% exceed a novelty-gap of 0.05. The two scores are also weakly correlated across heads, with a Pearson correlation of 0.234 on Qwen3-4B, 0.146 on Qwen3-8B, and 0.205 on Llama-3.1-8B-Instruct, so the heads that retrieve and the heads that respond to new content largely differ and must be selected jointly. We further check that the selected heads are robust in two respects. First, their behaviour does not depend on the context length. Across short (1K–16K tokens), medium (16K–32K), and long (32K–50K) contexts, their span-recall changes by at most 0.09, and their novelty-gap stays positive in every range and in 95–100% of the individual documents.

B.3 Empirical Validation

Length-invariance Test. A token-level gain that is accumulated up to a fixed threshold should not depend on where the token appears in the context. For each document, we compare the mean gain over its last 30%, $\bar{g}_{\text{late}}$, with that over its first 30%, $\bar{g}_{\text{early}}$, relative to the mean gain of the whole document:

$$\gamma := \frac{\bar{g}_{\text{late}} - \bar{g}_{\text{early}}}{\bar{g}_{\text{doc}}} \times 100\%, \tag{14}$$

and report the median of γ over the documents as the length drift in Figure 2(b). With $\beta_a = 0.25$, the drift is the lowest among the tested values of β_a and stays below 1% on every backbone, so a fixed threshold on the cumulative gain yields spans of nearly the same expected content at every position of the context. A two-one-sided equivalence test using the Wilcoxon signed-rank test (Wilcoxon, 1992), confirms that the drift lies within a margin of $\pm 1\%$ on every backbone ($p < 0.001$).

Table 3: Token-level information gain by word class and position within the word. Ratio compares the fractions of function and content words in the lowest decile of the gain, and Share is the percentage of multi-token words whose word-initial token outscores its word-internal tokens.

	Content vs. function			Initial vs. internal		
Backbone	Content	Function	Ratio	Initial	Internal	Share
Qwen3-4B	0.924	0.876	2.0	0.951	0.811	81.0
Qwen3-8B	0.891	0.845	1.5	0.923	0.779	78.9
Llama-3.1-8B	0.692	0.651	1.7	0.729	0.639	75.4

Domain-invariance Test. For domain invariance, let $\bar{g}_{c,j}$ be the mean gain of document j in domain c, $\bar{g}_c$ the mean over the n_c documents of domain c, and $\bar{g}$ the mean over all documents. A one-way ANOVA across the domains (Ross & Willson, 2017) gives the effect size

$$\eta^2 := \frac{\sum_c n_c \, (\bar{g}_c - \bar{g})^2}{\sum_c \sum_j (\bar{g}_{c,j} - \bar{g})^2}, \tag{15}$$

the fraction of the variance of the document means explained by the domain, which we report in Figure 2(b). The documents are drawn from four LongAlpaca domains, covering Book Question, Summarization, Paper Review, and Document Analysis. With $\beta_a = 0.25$, the effect size is the lowest among the tested values of β_a on every backbone and at most 0.15, so most of the variation in the gain lies between documents of the same domain rather than between domains.

Token Function Analysis. We relate the token-level information gain to two properties of a token, its word class and its position within the word, on 100 documents randomly sampled from LongAlpaca and LongMemEval. Tokens are grouped into words, each consisting of a *word-initial* token, which carries the word-boundary marker of the tokenizer, and the *word-internal* tokens that complete it.

Word class. Each word is tagged in context with the NLTK part-of-speech tagger (Bird & Loper, 2004); nouns, adjectives, adverbs, and main verbs are content words (58% of the words), and determiners, prepositions, conjunctions, pronouns, particles, and modal and auxiliary verbs are function words (41%). As shown in Table 3, content words receive a higher gain than function words at their word-initial token on every backbone (Mann–Whitney U test, $p < 10^{-25}$). By measuring how often each class falls among the tokens with the lowest 10% of the gain, we further find that function words are 1.5 to 2.0 times as likely as content words to fall there, so the difference is most pronounced where the gain is lowest.

Position within the word. For each word that the tokenizer splits into several tokens (6% of the words), the word-initial token receives a higher gain than the mean of its word-internal tokens in 75–81% of the words on every backbone (Wilcoxon signed-rank test, $p < 10^{-200}$). Once the first piece of a word is read, the remaining pieces are largely implied, and the gain reflects this even though subword boundaries play no role in its definition.

Together, these patterns show that the gain, defined purely by the attention dynamics of a retrieval head, is lowest where the context already determines the token, whether through its grammatical role or its place within a word, and that this correspondence recurs across model families and tokenizers.

Comparison with Output Entropy. We further investigate how the information gain relates to the output entropy, a widely used token-level indicator given by the entropy of the distribution from which the backbone predicts each token. We compute both on the random dataset collected above, and find that they are nearly uncorrelated, with a Spearman correlation of 0.12 on Qwen3-4B, 0.07 on Qwen3-8B, and 0.02 on Llama-3.1-8B-Instruct. This is expected, since the entropy measures how uncertain the model is about the next token, while the gain measures how much a token changes what the model retrieves from the context after it is read.

The two signals also spread differently over tokens. On Qwen3-8B, the 10% of tokens with the highest entropy account for 29% of its total, compared with 11% for the gain, so the entropy is dominated by a few highly uncertain positions. For instance, sentence-initial function words such as "The" rank at the 81st percentile of the entropy, as many continuations are open, yet add little content. Digits show the reverse, ranking at the 31st percentile of the entropy but the 65th of the

Table 4: Half-width Δ of the refinement window and threshold θ calibrated on the gain of the selected head and on the estimated gain, for each backbone and target ratio L.

		Qwen3-4B		Qwen3-8B		Llama-3.1-8B	
L	Δ	Head	Estimator	Head	Estimator	Head	Estimator
2	1	0.997	0.991	0.995	0.990	0.961	0.930
4	1	2.942	2.928	2.923	2.932	2.666	2.680
8	2	6.764	6.807	6.701	6.823	5.922	6.121
16	2	14.142	14.440	13.962	14.485	12.255	12.730

gain, since a number carries precise information the context cannot supply. The other two backbones show the same behaviour, suggesting that the information gain and the output entropy capture complementary aspects of a token.

B.4 Threshold Segmenting

Role of the Refinement. In practice, the equal-gain segmentation (Theorem 3.1) can only be approximated, since the gain arrives in discrete tokens. The refinement uses the Δ tokens around t^* to end each span at the lowest-gain token nearby, typically the end of a word or phrase. This trades part of the equality of span gains, most at small L where a span holds only a few tokens, for boundaries that keep units of content intact, an aspect of segmentation that Assumption A.2 does not model. The ablation in Table 2 shows that the trade improves the memory at every ratio.

Threshold Calibration. The segmentation procedure in §4.2 involves two parameters for each target ratio L, namely the half-width Δ of the refinement window and the threshold θ. We set $\Delta = 1, 1, 2$, and 2 for $L = 2, 4, 8$, and 16, respectively. Given Δ, we determine θ by binary search such that the full procedure, including the refinement, yields a mean span length of L on 1,000 documents randomly sampled from the training data (§5.1). The calibration is performed on both the gain of the selected head and the gain predicted by the estimator (§B.5), the latter being the one deployed at inference. As shown in Table 4, the two thresholds remain close at every ratio.

Since the gain of an unseen context may be distributed differently from that of the calibration data, we further measure the mean span length realized on the three evaluation benchmarks. The realized length deviates from the target L by 1.8% on average across benchmarks, backbones, and ratios, with the largest deviation of 2.5% observed on BEAM. Therefore, IncreMem and the fixed-ratio baselines in §5.1 are compared under matched memory budgets.

We further report the rate of boundaries that split a word in Table 5, that is, boundaries followed by a word-internal token, under equal-length segmentation, equal-gain segmentation, and equal-gain segmentation with the minimum-gain refinement, all at the same mean span length. Equal-gain segmentation alone lowers the rate at every ratio, since word-internal tokens receive a lower gain (§B.3) and rarely complete the accumulation, and the refinement lowers it further, most at larger L, where a longer span leaves more room to move the boundary. Relative to equal-length segmentation, the two together reduce the rate by 26%, 50%, 55%, and 63% at $L = 2, 4, 8$, and 16, which average to the 49% reported in §4.2.

Table 5: Word-splitting boundaries (%).

	L			
Segmentation	2	4	8	16
Equal-length	10.9	10.9	10.7	10.6
Equal-gain	8.8	7.2	6.8	6.5
+ refinement	**8.1**	**5.4**	**4.8**	**3.9**

Case Study. The span below is segmented at $L = 8$ with Qwen3-8B, where | marks a boundary. Fixed-length segmentation splits four words: *the antigen, stabil|izers increase the storage life, and pres|ervatives allow the use of multidose v|ials. Combination vaccines are harder to develop| and produce, because of potential incompat|ibilities and interactions among*. Gain-guided segmentation keeps every word intact and closes its spans at the ends of words and phrases: *the antigen, stabilizers| increase the storage life, and preservatives| allow the use of multidose| vials. Combination vaccines are harder to develop and| produce, because of potential incompatibilities| and interactions among*.

Table 6: Mean absolute error (%) of the estimated gain per token, in percent of the mean true gain, for each selection of transplanted backbone layers. The deployed selections are in bold.

	Qwen3-4B		Qwen3-8B		Llama-3.1-8B	
Transplanted layers	Layers	MAE	Layers	MAE	Layers	MAE
One	15	7.8	15	6.2	8	9.7
Two	10, 15	6.1	7, 15	5.0	5, 8	8.2
Three	**5, 10, 15**	**3.9**	**9, 12, 15**	**3.8**	**0, 4, 8**	**4.8**

B.5 Gain Estimator

Architecture. The estimator is a decoder with the hidden size, number of heads, and head dimension of its backbone, preceded by the frozen input embeddings of the backbone and a trainable projection initialized to the identity; with three layers, it amounts to 6–8% of the parameters of the backbone. Its layers are initialized with the attention weights of backbone layers evenly spaced up to the layer of the selected head, so that the last layer starts from the computation that produces the target attention and the earlier ones bridge the gap from the input embeddings.

Training Strategy. At each position i, the estimator outputs a learned convex combination $\hat{\mathbf{a}}_i$ of the attention heads in its last layer, and it is trained to match the attention $\mathbf{a}_i$ of the selected head by minimizing the Kullback–Leibler divergence between the two, the standard objective for distilling attention distributions (Wang et al., 2020; Sun et al., 2020). Because $\mathbf{a}_i$ is stored only through its $K = 64$ largest entries $\mathcal{T}_i$, which always include the attention of the token to itself, the divergence is computed over these entries, with the remaining mass on either side pooled into a single bucket:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\Big[\sum_{m\in\mathcal{T}_i} a_{i,m}\log\frac{a_{i,m}}{\hat{a}_{i,m}} + r_i\log\frac{r_i}{\hat{r}_i}\Big], \qquad r_i = 1-\sum_{m\in\mathcal{T}_i} a_{i,m}, \quad \hat{r}_i = 1-\sum_{m\in\mathcal{T}_i}\hat{a}_{i,m}. \tag{16}$$

The estimator is trained on 2,000 documents sampled from the training data (§5.1). At inference, it reads the raw context ahead of compression in windows of 24K tokens with a stride of 12K.

Results. We measure the mean absolute error (MAE) between the estimated gain and the gain of the selected head on 25 documents from the training data that are held out from the estimator's training, expressed as a percentage of the mean gain. Table 6 reports it for estimators initialized from one, two, and three backbone layers. Each added layer lowers the error on every backbone, and the deployed three-layer estimators reach 3.8–4.8%.

B.6 Complexity Analysis

We compare the cost of IncreMem with that of the uncompressed backbone as a function of the context length N, for a backbone with n_ℓ layers and a fixed hidden size. Since IncreMem reads the context in chunks of w tokens and each token attends only to its own chunk and to the at most $k \approx N/L$ memory units written before it, the attention cost of encoding drops from $O(N^2 n_\ell)$ to $O\big((N^2/L + Nw)\, n_\ell\big)$, and both the retained KV cache and the attention cost of each generated token drop from $O(N n_\ell)$ to $O\big((N/L + w)\, n_\ell\big)$. Both savings thus grow with the context toward a factor on the order of L. The bounds depend on the segmentation only through the number of memory units k, so allocating memory by gain rather than by length has the same asymptotic cost as uniform gist compression at the same ratio, apart from a linear-time segmentation pass and the gain estimator. The gain estimator adds only a lower-order term to this cost. With $n_e \ll n_\ell$ layers and attention over a bounded context of M tokens, its attention cost is $O(NMn_e)$ and its activation memory $O(Mn_e)$. The ratio of its attention cost to that of the backbone therefore scales as $O\big(\frac{n_e}{n_\ell}\cdot\frac{M}{N/L+w}\big)$, which carries the small factor n_e/n_ℓ and decreases as the context grows, while its memory does not grow with N at all. This agrees with the efficiency analysis in §5.2, where the savings of IncreMem grow with the context and its overhead over Beacon remains small.

C Detailed Experiments

C.1 Experimental Setups

Training Details. The pretraining stage uses the next-token-prediction objective (Zhang et al., 2025; Deng et al., 2025), in which the backbone predicts the raw tokens of each chunk conditioned on the gist tokens of the preceding chunks. The finetuning stage uses the distillation objective of Eq. equation 9, in which the teacher distribution is truncated to its top-64 entries at every position. The learning rate is 5×10^{-5} in the pretraining stage and 1×10^{-5} in the finetuning stage. Both stages train for one epoch with AdamW, a batch size of one sequence per device without gradient accumulation, bf16 precision, and gradient checkpointing, with a linear warmup over the first 5% of the finetuning steps, on four NVIDIA RTX Pro 6000 GPUs with 96 GB of memory; pretraining and finetuning take about 13.5 and 19.0 hours per backbone, respectively.

Evaluation Details. Following existing works Wu et al. (2025a); Tavakoli et al. (2026), we employ greedy search during evaluation and disable the thinking mode for the Qwen3 backbones. For the KV-eviction baselines, the memory budget is computed per example as $B = \lceil L/r \rceil$, where L is the length of the current example and r is the target compression ratio; AdaKV is built upon SnapKV following its official protocol Feng et al. (2025). Beacon is trained with the same two-stage recipe and the same gist attention module as ours, but with fixed-length spans and the next-token-prediction objective in both stages, so that the two differ only in the placement of span boundaries and in the finetuning objective. The maximum raw (pre-compression) context length is capped at 130K tokens for IncreMem and all baselines, including , with longer contexts truncated from the middle following prior work (Bai et al., 2025; Wu et al., 2025a; Tavakoli et al., 2026; Zhang et al., 2025). As the cap applies before compression, every method sees identical document content regardless of compression ratio L.

LongBench-v2 Bai et al. (2025) is a long-context understanding and reasoning benchmark spanning six task categories: single-document QA, multi-document QA, long in-context learning, long-dialogue history understanding, code repository understanding, and long structured-data understanding. Across its 503 documents, context length is highly skewed (mean 242.7K tokens, median 98.5K, range 9.9K–4.14M), reflecting a long tail of very large code repositories; we report the median as the more representative statistic. Following the official protocol, we use the direct-answer setting with at most 32 new tokens and report multiple-choice accuracy.

LongMemEval-S Wu et al. (2025a) evaluates the long-term memory ability of conversational assistants by testing recall and reasoning over a synthetic multi-session interaction history. It covers five core memory abilities: information extraction, multi-session reasoning, temporal reasoning, knowledge updates, and abstention. Each of the 500 questions is paired with a haystack of prior sessions averaging 103K tokens (median 103K, range 96.5K–105.1K), giving a tightly controlled context length by construction. Following the official protocol, the answer is generated with at most 64 new tokens and judged for correctness against the reference by an LLM with the task-specific prompts of the benchmark, including the abstention variant.

BEAM Tavakoli et al. (2026) similarly evaluates long-term conversational memory, using synthetic user profiles and conversation plans anchored to explicit dates across sessions, and probing questions that require consistency and temporal reasoning over the accumulated history. We use the 100K split (20 conversations, up to 20 probing questions each), with a mean context length of 130K tokens (median 120K, range 92.9K–180.8K). Following the official protocol, the answer is generated with at most 256 new tokens and scored against a per-question rubric, with one judge call per rubric item averaged over items.

NIAH gkamradt (2023) inserts a single needle sentence into a haystack of unrelated essays and asks the model to retrieve it, with the context length and the depth of the needle varied on a grid. Following the official protocol, each response is scored from 1 to 10 against the needle by an LLM with the official rubric.

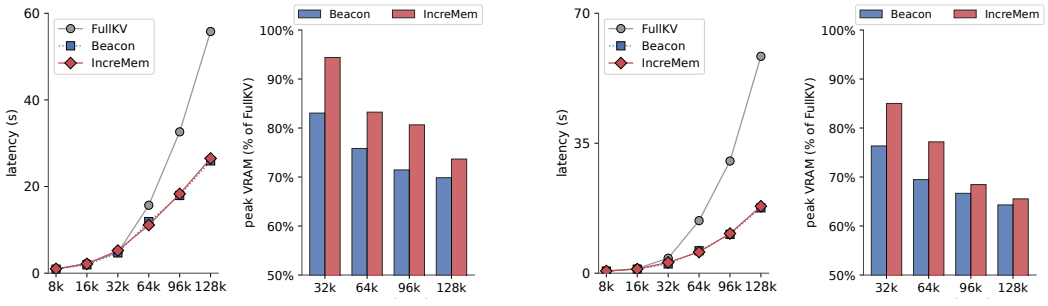
C.2 Additional Results



(a) Lllama-3.1-8B w. IncreMem (b) Lllama-3.1-8B w. Beacon

(c) Qwen3-4B w. IncreMem (d) Qwen3-4B w. Beacon

Figure 7: NIAH test results on Llama-3.1-8B and Qwen3-4B with x8 compression ratio.



(a) Llama-3.1-8B (b) Qwen3-4B

Figure 8: Efficiency Analysis of (a) Llama-3.1-8B and (b) Qwen3-4B.