AdmTree: Compressing Lengthy Context with Adaptive Semantic Trees

Yangning Li[1,2]*, **Shaoshen Chen**[1*], **Yinghui Li**[1‡], **Yankai Chen**[3], **Hai-Tao Zheng**[1,2‡], **Hui Wang**[2], **Wenhao Jiang**[4‡], **Philip S. Yu**[3]
[1]Shenzhen International Graduate School, Tsinghua University
[2]Peng Cheng Laboratory [3]University of Illinois Chicago
[4]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ)

Abstract

The quadratic complexity of self-attention constrains Large Language Models (LLMs) in processing long contexts—a capability essential for many advanced applications. Context compression aims to alleviate this computational bottleneck while retaining critical semantic information. However, existing approaches often fall short: explicit methods may compromise local detail, whereas implicit methods can suffer from positional biases, information degradation, or an inability to capture long-range semantic dependencies. We propose *AdmTree*, a novel framework for adaptive, hierarchical context compression with a central focus on preserving high semantic fidelity while maintaining efficiency. *AdmTree* dynamically segments input based on information density, utilizing gist tokens to summarize variable-length segments as the leaves of a semantic binary tree. This structure, together with a lightweight aggregation mechanism and a frozen backbone LLM (thereby minimizing new trainable parameters), enables efficient hierarchical abstraction of the context. By preserving fine-grained details alongside global semantic coherence, mitigating positional bias, and dynamically adapting to content, *AdmTree* robustly retains the semantic information of long contexts.

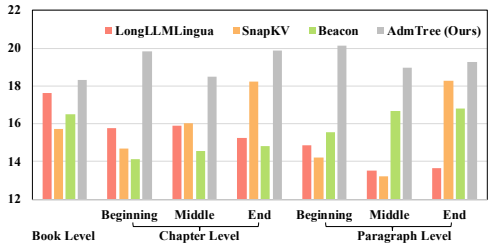
1 Introduction

Large Language Models (LLMs) [2, 3, 52] have demonstrated remarkable proficiency in processing and understanding long contexts, enabling advances in retrieval-ugmented generation [19, 55] and agentic system [54], etc. However, handling long contexts remains computationally intensive due to the quadratic complexity of self-attention with input token length. This leads to high memory consumption and inference latency. Consequently, **context compression** has emerged as a critical technique, aiming to *reduce input token length while preserving maximal semantic integrity*.

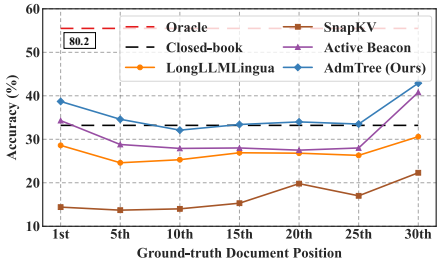
Despite promising results, most existing methods **fail to simultaneously preserve information across multiple semantic dimensions**, such as global versus local semantics, or information across positions. This inability in preserving different information dimensions may leads to poor generalization across real-world tasks, which demand distinct types of semantic information. Specifically, existing methods can be divided into two main categories. ***Explicit methods*** [25, 56, 57] directly shorten text by removing content deemed less essential for overall understanding. Although these methods effectively capture global meaning, they often disrupt local coherence due to excessive omission, leading to the loss of fine-grained details. In contrast, ***implicit methods*** [15, 20, 33, 41] encode long contexts into compact latent vectors (also called "gist tokens") in a flat manner. These methods achieve higher compression ratios, but exhibit different compression efficiency to context at distinct positions. As evidenced by [7, 23, 40], implicit compression methods are prone to positional

*Equal Contribution. ‡: Corresponding Author. This work was primarily conducted at GML under the leadership of Wenhao.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) Multi-Granularity Summarization. Implicit methods tend to prioritize the preservation of global information.



(b) Multi-Document Question Answering. Explicit methods exhibit varying compression effectiveness across content at different positions.

Figure 1: Pre-experiments demonstrate that existing methods struggle to balance semantic information of different dimensions for different types of tasks.

bias, often overlooking information from the earlier or middle parts of the context. In other words, less salient information is easily overshadowed by more prominent content.

To mitigate semantic loss caused by position bias, some implicit methods also explored recursive compression [11, 21, 58]. These methods progressively condense segmented contexts into serialized gist tokens. However, they often rely on fixed-size segments without considering variations in information density, leading to imbalanced compression loads across different input regions. Moreover, such linear manner still causes semantic information to degrade progressively during recursive compression, making it difficult to capture long-range dependencies and maintain global coherence.

To overcome these limitations, we draw inspiration from cognitive science, where hierarchical structures are fundamental to human information processing [6, 22]. Hierarchical representations naturally balance ***semantic breadth*** (coverage across contexts) and ***depth*** (granularity of detail). Moreover, it establishes ***bidirectional connections*** among leaf nodes, mitigating the information degradation by previous unidirectional compression. Motivated by this, we introduce the **Ad**aptive Se**m**antic **Tree** Compressor (**AdmTree**), a novel hierarchical context compression framework designed to preserve information integrity more effectively. Unlike methods relying on fixed chunking or flat representations, AdmTree performs compression in a dynamic and structured manner: *(i)* first, long input context is dynamically segmented into variable-length units based on information density. *(ii)* then, gist tokens are interleaved among segments to compress them, serving as the leaf nodes, and a semantic tree is then constructed bottom-up upon these leaves. *(iii)* finally, AdmTree generates responses conditioned on the semantic tree, which can be dynamically updated as new context arrives.

Extensive experiments are conducted to validate the effectiveness of AdmTree. In main experiments, AdmTree consistently achieves state-of-the-art performance across five task types and more than ten datasets from LongBench [4], surpassing baseline methods by over 10% while maintaining **high inference efficiency**. In some QA tasks, AdmTree even outperforms the strongest baseline Beacon [58], by up to 20 points, demonstrating its remarkable ability to **comprehensively preserve semantic information**. Further analysis also confirms that AdmTree exhibits strong **scalability** and efficiency in dynamic context scenarios. Additionally, appendix results highlight the **interpretability** of AdmTree powered by attention over tree nodes, compared to previous black-box implicit compression methods.

2 Preliminaries

2.1 Task Formulation

Following the instruction tuning setting, we define input context as $\mathbf{X} = (\mathbf{X}^{\text{inst}}, \mathbf{X}^{\text{doc}})$, where $\mathbf{X}^{\text{inst}}$ is task instruction (query) and $\mathbf{X}^{\text{doc}}$ is input long document. The compressed context $\widetilde{\mathbf{X}}$ is either text tokens or semantic vectors, based on which LLMs generate responses. The objective of context compression can be formulated as maximize information retention while minimizing redundancy:

$$\max_{\widetilde{\mathbf{X}}} I(\widetilde{\mathbf{X}}; \mathbf{Y}) - \lambda \mathcal{R}(\widetilde{\mathbf{X}}),$$

in which $I(\widetilde{\mathbf{X}}; \mathbf{Y})$ measures the mutual information between the compressed context $\widetilde{\mathbf{X}}$ and ground-truth answer $\mathbf{Y}$, ensuring that **essential information is preserved**. The second term $\mathcal{R}(\widetilde{\mathbf{X}})$ is a

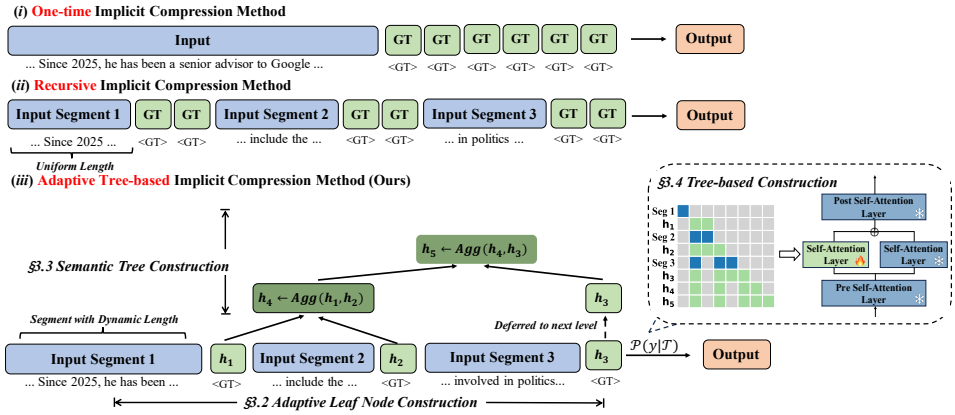


Figure 2: Comparison of compression frameworks. All methods generate responses conditioned on the representation in green. Unlike other methods compress in a linear manner, AdmTree dynamically balances context compression both horizontally and vertically through adaptive gist token allocation and tree hierarchy. Meanwhile, bidirectional aggregation mitigates information degradation.

regularization term that penalizes excessive redundancy, **promoting shorter** $\widetilde{\mathbf{X}}$. The parameter λ controls the trade-off between information retention and compression.

To quantify the compression degree, the compression ratio is defined as the proportion of original input length to compressed context length, given by: $\tau = \frac{|\mathbf{X}|}{|\widetilde{\mathbf{X}}|} \geq 1$, where $|\cdot|$ is the token number.

2.2 Hard Trade-offs Between Different Information Dimensions in Compression

To preliminarily illustrate the limitations of both explicit and implicit compression methods in balancing semantic information across multiple dimensions, we conduct experiments on two probing tasks: *(i)* **Multi-Granularity Summarization**. Leveraging the BookSum dataset [31], this task requires the model to generate summaries at the book, chapter, and paragraph levels for a given article. It **evaluates the model's capacity to preserve both global and local semantic content**. *(ii)* **Multi-Document Question Answering**. For this task, the NaturalQuestion dataset [32] was employed. The task involves answering questions based on 30 input documents, only one of which contains the ground-truth answer. By placing the ground-truth document at different positions, we verify the model capability to compress information across various locations.

As shown in Figure 1a, we observe that as the summarization granularity becomes finer (from book to paragraph level), the performance of explicit method (LongLLMLingua) consistently declines, while no such trend is observed for implicit methods. This suggests that explicit compression methods tend to prioritize global information while neglecting finer details. In Figure 1b, both Activation Beacon and SnapKV demonstrate better performance when the ground-truth document is placed at the end. This results in a performance curve characterized by an upward trajectory in its latter segment. The "lost in the middle" phenomenon [40] indicates that implicit methods are more prone to positional bias, preferentially retaining information closer to compression tokens. The above preliminary experiments demonstrate the existing compression methods struggle to retain semantic information in multiple dimensions completely, motivating us to propose a more balanced compression framework.

3 AdmTree Framework

3.1 Overview

As shown in Figure 2, existing one-time or recursive compression methods still operate in a linear or flat manner. They all suffer from information degradation due to position bias. In contrast, AdmTree is a novel framework for adaptive hierarchical context compression. Its dynamic tree structure not only balances semantic preservation in terms of both breadth and depth, but also mitigates the information degradation caused by unidirectional compression through hierarchical aggregation. In general, it dynamically segments input context, assigns gist tokens to summarize each segment, and builds a tree structure to capture contextual information at multiple granularities. This hierarchical representation

is then utilized for compression, with only minimal parameters trained. AdmTree comprises three key stages: dynamic leaf gist token construction, semantic tree construction, tree-based compression.

3.2 Adaptive Leaf Gist Token Construction

This stage transforms an input context $\mathbf{X}$ into an alternating sequence of textual sub-segments $\mathbf{s}_k$ and leaf gist tokens $\langle \text{GT} \rangle_k$. The primary objective is to achieve adaptive segmentation based on information density, governed by a compression strategy, to prepare input for hierarchical summarization.

$$\mathbf{X} \xrightarrow{\text{Segmentation \& Budgeting}} \mathbf{X}' = [\mathbf{s}_1, \langle \text{GT} \rangle_1, \mathbf{s}_2, \langle \text{GT} \rangle_2, \ldots, \mathbf{s}_M, \langle \text{GT} \rangle_M]$$

Each leaf gist token $\langle \text{GT} \rangle_k$ is designated to summarize its preceding sub-segment $\mathbf{s}_k$, and the attention scope is restricted to its associated sub-segment and preceding leaf gist tokens. The process involves initial uniform partitioning of $\mathbf{X}$ into segments $\mathbf{X}_i$, followed by adaptive budget reallocation (determining the number of sub-segments b'_i within each $\mathbf{X}_i$) using an information content score (Eq. 2) to define the final sub-segments $\mathbf{s}_k$ and the placement of their corresponding $\langle \text{GT} \rangle_k$. A dedicated token, $\langle \text{GT} \rangle$, is incorporated into the LLM's vocabulary for this purpose.

Initial Segmentation Given an input context $\mathbf{X} = [x_1, \ldots, x_{\mathcal{L}}]$ of length $\mathcal{L}$, AdmTree first partitions it uniformly into initial segments $\mathbf{X_i}$, each of length n:

$$\mathbf{X_i} = \left[x_{(i-1)n+1}, \ldots, x_{\min(i \cdot n, \mathcal{L})}\right], \tag{1}$$

where n is constrained not to exceed the maximum sequence length supported by the LLM.

Adaptive Compression Budget Allocation Recognizing that information density varies across segments, initial uniform segments are adaptively further partitioned: segments with higher information density or complexity are divided into finer sub-segments with more gist tokens (implying a lower local compression ratio).

A simple yet effective strategy is proposed for this budget reallocation. Given the overall compression ratio τ, the initial budget of leaf gist tokens b_i for each segment $\mathbf{X_i}$ (of length n) is $n/2\tau$. We first quantify the informational value of each segment $\mathbf{X_i}$ using an entropy-adjusted perplexity score:

$$\text{Score}(\mathbf{X_i}) = \text{PPL}(\mathbf{X_i}) \cdot \exp\left(-\lambda \cdot \text{Entropy}(\mathbf{X_i})\right), \tag{2}$$

where $\text{Entropy}(\cdot)$ measures its intrinsic information content, and λ is a hyperparameter balancing PPL and entropy. Segments are then ranked based on this score, and the gist token budgets are redistributed as follows:

$$b'_i = \begin{cases} n/\tau, & \text{if } \mathbf{X_i} \text{ ranks in the top 25\% by Score} \\ n/2\tau, & \text{if } \mathbf{X_i} \text{ ranks in the middle 25\% by Score} \\ n/4\tau, & \text{if } \mathbf{X_i} \text{ ranks in the bottom 50\% by Score} \end{cases}. \tag{3}$$

This adaptive allocation strategy still maintains the overall compression ratio τ globally, while adapting locally to information density. Finally, AdmTree partitions each original segment $\mathbf{X_i}$ into b'_i finer sub-segments, appending one gist token $\langle \text{GT} \rangle$ after each sub-segment. Let $\mathbf{s}_{i,j}$ denote the j-th sub-segment of the original segment $\mathbf{X_i}$. The processed segment $\mathbf{X}'_i$ becomes a sequence of sub-segments and their corresponding gist tokens:

$$\mathbf{X}'_i = \left[\mathbf{s}_{i,1}, \langle \text{GT} \rangle_{i,1}, \mathbf{s}_{i,2}, \langle \text{GT} \rangle_{i,2}, \ldots, \mathbf{s}_{i,b'_i}, \langle \text{GT} \rangle_{i,b'_i}\right], \tag{4}$$

where each sub-segment $\mathbf{s}_{i,j}$ has a length $\alpha_{i,j} = n/b'_i$. These $\langle \text{GT} \rangle_{i,j}$ tokens form the leaf nodes of the semantic tree.

3.3 Semantic Tree Construction

Here, we detail the construction of current semantic tree $\mathcal{T}_{\leq k}$ from the set of leaf gist token representations $\{h_{\langle \text{GT} \rangle_1}, \ldots, h_{\langle \text{GT} \rangle_k}\}$. The construction is scalable and incremental: when processing sub-segment $\mathbf{s}_k$, the existing tree $\mathcal{T}_{<k}$ (which summarizes preceding $\langle \text{GT} \rangle_1, \ldots, \langle \text{GT} \rangle_{k-1}$) is partially reused to form $\mathcal{T}_{\leq k}$ by integrating the new gist token's representation $h_{\langle \text{GT} \rangle_k}$ (obtained as described in Sec. 3.4). The general bottom-up aggregation process is:

$$\{h_{\langle \text{GT} \rangle_1}, h_{\langle \text{GT} \rangle_2}, \ldots, h_{\langle \text{GT} \rangle_M}\} \xrightarrow{\text{Hierarchical Aggregation}} \mathcal{T}_M.$$

While the tree structure can be flexible, we employ the binary tree. Meanwhile, to avoid introducing extra consumption, we do not use padding tokens to keep tree balanced. Instead, any remaining unpaired gist token is deferred to the aggregation at next level.

Once the structure is determined, the aggregation function is defined as: $\text{Agg} : \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}^d$, where $\mathcal{P}$ denotes the power set and $\mathbb{R}^d$ is the hidden state dimension. Starting from the leaf gist tokens, the aggregator iteratively combines hidden states h_u for all children u of node v to form the parent node's hidden state h_v:

$$h_v = \text{Agg}(\{h_u \mid u \in C_v\}), \tag{5}$$

Here, C_v denotes the set of children of node v. For efficiency, the aggregation function $\text{Agg}(\cdot)$ is implemented by a single-layer self-attention mechanism, followed by an averaging operation:

$$\text{Agg}(C_v) = \text{Average}\left(\text{Self-Attn}_{\text{agg}}(\{h_u \mid u \in C_v\}; \theta_{\text{agg}})\right), \tag{6}$$

The trainable parameters size in θ_{agg} is significantly smaller than the entire LLM, and are only built once during inference. Therefore, it does not affect the original inference complexity $\mathcal{O}((\mathcal{L}/\tau)^2)$. Furthermore, the self-attention in aggregation mechanism further allows information from subsequent segments to influence preceding ones, mitigating the unidirectional limitations of causal LLMs.

3.4 Tree-based Compression

This stage processes each sub-segment $\mathbf{s}_k$ and its associated gist token $\langle\text{GT}\rangle_k$ by leveraging the semantic summary $\mathcal{T}_{<k}$ of preceding content (i.e., tree built from gist tokens $\langle\text{GT}\rangle_1, \ldots, \langle\text{GT}\rangle_{k-1}$). The primary goal is to generate contextualized representations for tokens in $\mathbf{s}_k$ and for $\langle\text{GT}\rangle_k$. Once the representation is computed, the key ($K_{\langle\text{GT}\rangle_k}$) and value ($V_{\langle\text{GT}\rangle_k}$) representations of $\langle\text{GT}\rangle_k$ are also stored to accelerate the construction of subsequent semantic tree (as per Sec. 3.3).

$$(\mathbf{s}_k, \langle\text{GT}\rangle_k), \mathcal{T}_{<k} \xrightarrow{\text{LLM Encoding}} (H_{\mathbf{s}_k}, h_{\langle\text{GT}\rangle_k}, \{K_{\langle\text{GT}\rangle_k}, V_{\langle\text{GT}\rangle_k}\})$$

Here, $H_{\mathbf{s}_k}$ and $h_{\langle\text{GT}\rangle_k}$ denote the output hidden states of text tokens in $\mathbf{s}_k$ and gist token $\langle\text{GT}\rangle$, respectively. Following prior work [21, 41], AdmTree leverages the original LLM to encode text token, while the $\langle\text{GT}\rangle_k$ token and the nodes from $\mathcal{T}_{<k}$ are processed independently with another trainable attention heads.

As illustrated in Figure 2, to distinctly encode gist tokens and regular text tokens, we introduce two separate attention branches before merging them for joint self-attention. Assuming $h_j^l \in \mathbb{R}^d$ is the hidden state of the j-th token at layer l. The query, key, and value vectors are computed as:

$$q_j^{l+1} = h_j^l W_q^*, \qquad k_j^{l+1} = h_j^l W_k^*, \qquad v_j^{l+1} = h_j^l W_v^*, \tag{7}$$

where placeholder $* \in \{\text{gt}, \text{tt}\}$ distinguishes gist tokens (gt) from text tokens (tt). Projection matrices for text tokens (e.g., W_q^{tt}) are derived from the original LLM, while those for gist tokens (e.g., W_q^{gt}) are newly introduced and trained. After individually computing, the representations from the preceding tree context ($\mathcal{T}_{<k}$), the current text sub-segment ($\mathbf{s}_k$), and the current gist token ($\langle\text{GT}\rangle_k$) are concatenated back to form a unified sequence. The original order of each token type is maintained. The combined matrices are:

$$Q = [Q_{\text{tt}}, Q_{\text{gt}}], \qquad K = [K_{\mathcal{T}_{<k}}, K_{\text{tt}}, K_{\text{gt}}], \qquad V = [V_{\mathcal{T}_{<k}}, V_{\text{tt}}, V_{\text{gt}}]. \tag{8}$$

Then, the standard self-attention mechanism with a causal mask M is then applied:

$$H^{l+1} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V, \tag{9}$$

where d_k is the dimension of the key vectors. All tokens are encoded with relative positional embeddings for queries and keys. H is actually the concatenation of $H_{\mathbf{s}_k}$ and $h_{\langle\text{GT}\rangle_k}$. Tree nodes from $\mathcal{T}_{<k}$ are flattened in a left-to-right, bottom-to-top order to be integrated into the sequence.

3.5 Compression Learning

AdmTree learns to compress context into the semantic nodes of a tree $\mathcal{T}$ and then generate appropriate responses based on them. Similar to general LLMs, AdmTree is optimized via next-token prediction

task. The prediction for token x_j within the given sub-segment $\mathbf{s}_k$ is conditioned on both the dynamically constructed semantic tree $\mathcal{T}_{<k}$ (summarizing all preceding context) and the local intra-sub-segment context $x_{<j}$. The objective is to minimize the negative log-likelihood:

$$\mathcal{L}_{\text{train}} = \min_{\theta_{\text{gt\_attn}}, \theta_{\text{gt\_emb}}, \theta_{\text{agg}}} \mathbb{E}_{\mathbf{X} \sim \mathcal{D}} \left[- \sum_{\mathbf{s}_k \in \mathbf{X}} \sum_{x_j \in \mathbf{s}_k} \log P\left(x_j \mid \mathcal{T}_{<k}, x_{<j}; \theta_{\text{gt\_attn}}, \theta_{\text{gt\_emb}}, \theta_{\text{agg}}\right) \right], \quad (10)$$

where the outer sum iterates over all sub-segments $\mathbf{s}_k$ derived from a long context input $\mathbf{X}$. The trainable parameters consist of: $\theta_{\text{gt\_attn}}$, for the attention heads processing gist tokens; $\theta_{\text{gt\_emb}}$, for the token embedding of $\langle \text{GT} \rangle$; and θ_{agg}, for the aggregator parameters. The parameters of the backbone LLM, θ_{LLM}, remain frozen throughout the training.

4 Experiment Details

4.1 Experiment Setting

Baselines Following prior work [26, 58] on compression, we introduce two main categories of baseline models for processing long contexts:

(i) Retrieval-based Methods. These methods employ a retriever to obtain the most relevant segments based on their similarities to the given query (instruction). We utilize one sparse retrieval method **BM25** [48], and two dense retrieval methods, **Sentence-BERT** [47] and **OpenAI Embeddings**[2].

(ii) Compression-based Methods. Unlike retrieval-based methods, these methods directly compress the input text into more concise representations. We select **LongLLMLingua** [26] as a representative of explicit compression methods. **AutoCompressor** [11], **ICAE** [21], **Activation Beacon** [58], and **SnapKV** [38] are employed as implicit compression methods. The former three methods recursively compresses long contexts into semantic vectors, enabling preliminary fine-grained compression.

We also fine-tuned the original LLM using the same training data to serve as the strong baseline, denoted as Original LLM-FT.

Evaluation Benchmark We evaluate AdmTree's effectiveness and efficiency on **LongBench** [4]. LongBench is a multitask benchmark assessing LLM comprehension of long contexts. The description for other used dataset can be found in Appendix.

Backbone LLMs `LlaMA-2-7B-Chat` and `Qwen-2-7B-Instruct` are employed as the backbone LLMs. We use LlaMA-2-7B-Chat because three out of the four compression baselines utilize it.

Implementation Details The training details for AdmTree can be found in Appendix. For the compression ratio in main experiments, given LLaMA-2's maximum sequence length of 4K, we apply $\times 2$ compression for 4K–8K contexts, $\times 4$ for 8K–16K, and $\times 8$ for 16K–32K. For Qwen-2, we uniformly set the compression ratio to $\times 4$. The results are averaged over three inference runs.

4.2 Main Experiments

We evaluate the AdmTree and other baseline on LongBench. The performance and latency are reported in Table 1, from which we can observe that:

1. **AdmTree significantly surpasses all other baselines.** Specifically, on the Llama-2-7B and Qwen-2-7B models, AdmTree outperforms the current state-of-the-art compression method Activation Beacon, by 10% and 5% in average scores, respectively. The performance gains are particularly striking in QA tasks, where AdmTree achieves improvements exceeding 327.5% in some tasks. Notably, AdmTree achieves these gains without incurring additional latency compared to other recursive compression methods. This performance gain is primarily attributed to AdmTree's adaptive leaf node allocation and hierarchical architecture, which effectively balances semantic information across different dimensions. Meanwhile, the utilization of a single-layer Transformer for the leaf node representation in AdmTree's semantic tree contributes to its computational efficiency. In contrast, AutoCompressor and ICAE exhibit poor performance, primarily due to their fixed number of gist tokens, which hinders their ability to adeptly handle the diverse long-context scenarios in LongBench.

[2]We use `text-embedding-ada-002` at OpenAI.

2. **AdmTree's advantages are more pronounced when compared with retrieval-based methods.** While retrieval methods enable processing lengthy context without any modification on original model, this comes at the expense of performance. These methods particularly struggle with complex, multi-hop question answering, since they may fail to retrieve all necessary relevant context. Consequently, there is a marked performance decline on single and multi-document QA compared to AdmTree.

3. **AdmTree is the only compression-based method that consistently outperforms the strong baseline of full fine-tuning on the original LLM (Original LLM-FT) across both backbone models.** We posit that AdmTree's ability to surpass Original LLM-FT stems from its capacity to retain more comprehensive information within its hierarchical structure. When presented with a task query, AdmTree can efficiently "retrieval" and utilize the most relevant information while effectively filtering out contextual noise. Other methods, such as Beacon, do not consistently achieve this, occasionally underperforming Original LLM-FT on specific tasks.

Table 1: Out of domain evaluation on LongBench [4]. Each task type includes at least two datasets. We report the micro-averaged performance across all datasets. †: the results cited from Bai et al. [4], Zhang et al. [58]

Methods	LongBench						Latency	
	SingleDoc	MultiDoc	Summ.	FewShot	Code	**AVG**	Latency	Speedup
LLama-2-7B								
Original LLM†	24.7	22.4	24.6	63.2	57.7	37.2	6.4	4.0×
Original LLM-FT†	34.8	27.5	23.2	61.8	57.8	39.8	25.6	-
Retrieval-based Methods								
BM25	25.1	23.9	24.4	56.4	33.1	32.5	6.4	4.0×
SBERT	17.1	15.8	23.6	53.2	36.8	28.8	6.4	4.0×
OpenAI	28.3	16.4	16.9	23.7	50.3	25.5	6.4	4.0×
Compression-based Methods								
AutoCompressor† [11]	12.9	16.4	16.3	23.8	39.4	20.5	12.8	2.0×
ICAE† [21]	19.5	19.2	19.5	24.8	27.8	21.8	10.2	2.5×
LongLLMLingua† [26]	21.5	18.8	21.7	49.5	53.2	31.5	8.5	3.0×
SnapKV† [38]	24.2	22.6	16.3	60.1	57.7	34.6	8.5	3.0×
Beacon† [58]	34.9	27.5	25.0	61.4	57.8	40.1	8.0	3.2×
AdmTree (Ours)	**36.5**	**36.3**	**26.9**	**65.5**	**60.9**	**44.1**	7.8	3.3×
Qwen-2-7B								
Original LLM†	38.8	37.5	26.7	70.1	60.3	45.7	23.5	1.0×
Original LLM-FT†	41.0	40.6	26.8	68.5	66.1	47.4	23.5	-
Retrieval-based Methods								
BM25	28.8	31.1	23.8	54.0	32.0	34.1	5.9	4.0×
SBERT	18.4	22.9	21.6	50.4	34.6	28.9	5.9	4.0×
OpenAI	30.0	19.1	23.8	22.9	51.6	27.9	5.9	4.0×
Compression-based Methods								
LongLLMLingua†	24.7	20.3	26.3	55.9	50.1	34.4	7.8	3.0×
SnapKV†	38.7	37.6	26.2	67.1	60.3	45.0	7.8	3.0×
Beacon†	40.5	40.3	26.8	68.4	66.4	47.2	7.3	3.2×
AdmTree (Ours)	**41.6**	**45.9**	**30.2**	**69.9**	**66.8**	**49.7**	7.0	3.4×

4. Among compression-based methods, **explicit method (LongLLMLingua) and implicit methods (represented by Activation Beacon) demonstrate varied efficacy across different task types.** For instance, LongLLMLingua achieves competitive performance with AdmTree on summarization tasks but significantly lags behind on question answering tasks. This observation aligns with our assertion in the introduction that explicit compression methods may struggle with tasks demanding fine-grained details. Conversely, AdmTree exhibits consistently strong and balanced performance across all task types. As further evidenced by our analyses in Figure 1, AdmTree maintains stable performance across summarization tasks of different granularities and QA tasks with different ground-truth information location. This consistent performance, with lower variance, demonstrating AdmTree's superior capability in preserving semantic integrity across multiple information dimensions, and also suggests the potential of integrating Mixture-of-Experts (MoE) architectures [8] to further specialize compression across diverse task requirements.

4.3 Ablation Experiments

To quantify the contributions of each modules to overall performance, we conducted ablation experiments covering different training stages, model structures, and tree construction strategies. As shown in Table 2, both pre-training and instruction fine-tuning enhance the compression learning. Pre-training exerts the larger impact, probably since it learned from the full-sentence contexts, whereas instruction tuning optimized LLMs only on task outputs. Removing the tree structure (while maintaining the compression ratio) resulted in a severe performance drop to 28.5, demonstrating the effectiveness of our proposed tree architecture. When the adaptive leaf gist token allocation was replaced with uniform allocation, indicating that adaptive compression better balances semantic information across segments. Furthermore, removing the parameters from the self-attention layer in the tree aggregation process also led to a notable performance decrease. In general, single-layer self-attention strikes an effective trade-off between performance and efficiency.

Tree-node retrieval To further enhance inference efficiency, we experimented with directly retrieving only the top 75% of tree nodes during inference, based on their attention scores relative to

Table 3: Experiments in dialogue scenarios, where context arrives dynamically rather than requiring one-time compression. ShareGPT dataset [12] is used. AdmTree demonstrates notable flexibility.

Method	1 Turn (765 tokens)			2 Turn (3006 tokens)			3 Turn (6491 tokens)		
	PPL↓	Latency	TFLOPs	PPL↓	Latency	TFLOPs	PPL↓	Latency	TFLOPs
AutoCompressors	8.32	0.76	8.41	6.09	1.38	38.18	7.51	1.82	96.63
ICAE	7.72	0.45	8.78	6.55	0.76	40.62	8.13	1.20	92.37
LongLLMLingua	5.91	0.78	8.78	4.96	1.41	48.32	4.77	1.98	145.71
SnapKV	4.91	0.32	8.30	4.20	0.69	33.97	4.63	0.83	75.45
Beacon	4.27	0.37	8.63	3.08	0.72	34.54	2.98	0.94	75.41
AdmTree (Ours)	**4.01**	0.37	8.37	**2.91**	0.70	34.09	**2.79**	0.92	75.38

the last token. The resulting performance loss is marginal: the model still surpasses competitive baselines such as SnapKV on single-document QA. The scalability of AdmTree thus enables further inference-time optimizations under limited computational budgets.

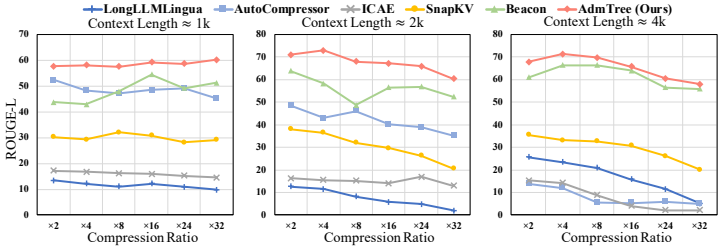


Figure 3: Model performance under varying context lengths and compression ratios.

Table 2: Ablation experiments on LLaMA-based AdmTree.

Ablation	Single-Doc
Full	36.5
w/o Pre-training	26.6
w/o Fine-tuning	29.3
w/o Tree Structure	28.5
w/o Self-attention	29.6
w/o Adaptive Leaf Constr.	34.1
+ Tree Nodes Retrieval	32.8

5 Analysis Experiments

In this section, several characteristics of AdmTree are investigated, including flexibility, effects across varying context lengths and compression ratios, and compression effects on fine-grained information.

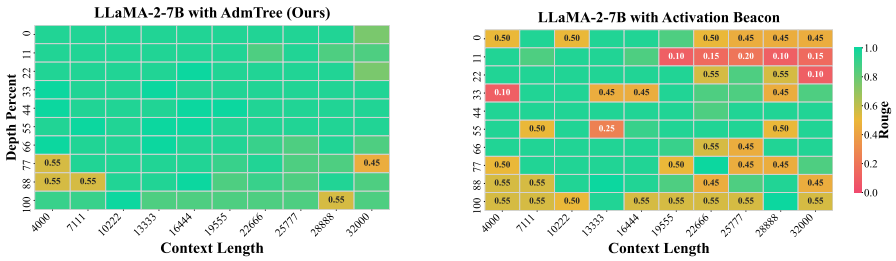
5.1 Compression Flexibility

We evaluate the flexibility of compression methods in multi-turn dialogue scenarios, where demands efficient compression of dynamically updated conversational context (utterances). The ShareGPT dataset [12], comprising lengthy dialogues between humans and ChatGPT, was utilized for evaluation. We assessed methods over three consecutive dialogue turns, with the average context length progressively increasing from 765 to 6,491 tokens. Perplexity (PPL) was employed to assess responses quality, while latency and TeraFLOPs (TFLOPs) were reported to measure compression efficiency.

As shown in Table 3, AdmTree consistently achieves the lowest PPL scores across all three dialogue turns. Notably, its PPL decreases as the dialogue progresses and context length increases, which indicates that AdmTree effectively compresses the dialogue history, preserving semantic integrity and conversational structure. Meanwhile, one-time compression methods (e.g., LongLLMLingua and ICAE) incur a dramatic rise in computational cost as the dialogue extends, since they need to the recompress entire context at each turn. While AdmTree and Activation Beacon mitigate this issue as recursive compression methods, which allow for the reuse of compressed representations from previous turns. This scalability is highly desirable for dynamic, real-world applications like online dialogue systems, where context is continuously updated, not merely processed in a single pass.

5.2 Compression with Different Context Length and Compression Ratio

Figure 3 presents the performance of various compression methods under different context lengths and compression ratios. Specifically, we use the Multi-Session Conversation (MSC) dataset, constructed by MemGPT [44]. We further process it to obtain dialogue samples with average lengths of 1K, 2K, and 4K tokens. Following MemGPT, ROUGE-L is employed to evaluate response quality. We can found that AdmTree consistently outperforms all baseline models across all context lengths and compression ratios by a substantial margin. At a 1K context length, AdmTree maintains stable performance, even showing slight improvement as the compression ratio increases. For longer contexts, AdmTree exhibits a much smaller degradation in performance compared to other methods.



(a) ROUGE scores of responses generated by AdmTree. We show the scores below 0.6.

(b) ROUGE scores of responses generated by Activation Beacon.

Figure 4: Comparison of Needle-in-The-Haystack [27] results with LLaMA-2-7B as backbone LLM. Notably, several baselines collapse under the combination of long context and high compression ratio, with ROUGE-L scores dropping below 10.

5.3 Fine-grained Information Compression

We evaluated the fine-grained information retention capabilities of AdmTree and the state-of-the-art Activation Beacon with the challenging Needle-in-the-Haystack dataset. In this task, a specific piece of information ("needle") is embedded at a specific position within a lengthy document ("haystack"), and the model is prompted to retrieve this detail. As illustrated in Figure 4, AdmTree consistently and precisely extracts the target information across documents of varying lengths and needle positions, demonstrating strong semantic preservation. Notably, the length of input contexts in this task substantially exceeds those seen during AdmTree's training, yet the model maintains stable performance.

6 Related Work

6.1 Compression Methods

As input contexts grow increasingly long, context compression has attracted significant attention. The goal is to reduce input token length while maintaining semantic integrity. The existing methods can be broadly categorized into two categories: (1) Explicit methods that directly shorten text by removing less crucial content based on information theory [25, 26, 37, 45, 51]. These methods range from token-level pruning [25, 45], to sentence-level methods such as CPC [39]. (2) Implicit methods include encoding inputs into soft tokens using methods such as Gisting [41] and Gist-COCO [35], or applying recursive-based compression as in AutoCompressor [11], ICAE [20] and Activation Beacon [58]. Specialized compression methods have also been proposed, including xRAG [10] for retrieval-augmented generation and CCM [29] for online scenarios.

6.2 Retrieval Methods

Retrieval methods enable LLMs to process long context by selecting the most relevant segments based on their similarity to a given query. These methods are generally categorized into two types: (1) Sparse Retrieval: use term-frequency-based models, such as TF-IDF [46] and BM25 [48], which represent documents as sparse vectors based on keyword occurrences. (2) Dense Retrieval: employ neural networks to encode queries and documents into continuous vector spaces, enabling semantic similarity matching. Common models include Sentence-BERT [47], Contriever [24], and DPR [28].

7 Conclusion

In this paper, to address the limitations of existing context compression methods in preserving semantic fidelity, we propose AdmTree, a novel hierarchical compression framework that dynamically allocates gist tokens and constructs adaptive semantic trees. By integrating information-aware segmentation with lightweight tree-based aggregation, AdmTree comprehensively preserves semantic information across multiple dimensions. Experiments on LongBench show that AdmTree consistently surpasses state-of-the-art baselines in both performance and inference efficiency across diverse tasks. Further analysis also confirms that AdmTree exhibits strong scalability in dynamic context scenarios.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant No.62276154), the Research Center for Computer Network (Shenzhen) Ministry of Education, the Natural Science Foundation of Guangdong Province (Grant No. 2024TQ08X729 and 2023A1515012914), the Basic Research Fund of Shenzhen City (Grant No.JCYJ20210324120012033, JCYJ20240813112009013, and GJHZ20240218113603006), and the Major Key Project of PCL for Experiments and Applications (PCL2023A09).

References

[1] Sharegpt. 2023. URL `https://sharegpt.com/`.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

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

[5] Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, et al. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. *arXiv preprint arXiv:2412.15204*, 2024.

[6] Matthew M Botvinick. Hierarchical models of behavior and prefrontal function. *Trends in cognitive sciences*, 12(5):201–208, 2008.

[7] Ruisi Cai, Yuandong Tian, Zhangyang Wang, and Beidi Chen. Lococo: Dropping in convolutions for long context compression. In *Forty-first International Conference on Machine Learning*, 2024.

[8] Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[9] 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. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. xrag: Extreme context compression for retrieval-augmented generation with one token. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL `http://papers.nips.cc/paper_files/paper/2024/hash/c5cf13bfd3762821ef7607e63ee90075-Abstract-Conference.html`.

[11] Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, 2023.

[12] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[13] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

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

[15] 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. *arXiv e-prints*, pages arXiv–2412, 2024.

[16] Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=yf1icZHC-l9.

[17] Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=TaAqeo7lUh.

[18] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[19] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[20] Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*, 2023.

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

[22] Stephen C Hirtle and John Jonides. Evidence of hierarchies in cognitive maps. *Memory & cognition*, 13(3):208–217, 1985.

[23] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2024.

[24] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

[25] Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Llmlingua: Compressing prompts for accelerated inference of large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[26] 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, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[27] Greg Kamradt. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2024.

[28] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen Tau Yih. Dense passage retrieval for open-domain question answering. In *2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 6769–6781. Association for Computational Linguistics (ACL), 2020.

[29] 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, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=64kSvC4iPg.

[30] Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. Learned token pruning for transformers. In Aidong Zhang and Huzefa Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 784–794. ACM, 2022. doi: 10.1145/3534678.3539260. URL https://doi.org/10.1145/3534678.3539260.

[31] Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. Booksum: A collection of datasets for long-form narrative summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6536–6558, 2022.

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

[33] Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.

[34] Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. How long can open-source llms truly promise on context length? 2023. URL https://lmsys.org/blog/2023-06-29-longchat.

[35] Xinze Li, Zhenghao Liu, Chenyan Xiong, Shi Yu, Yukun Yan, Shuo Wang, and Ge Yu. Say more with less: Understanding prompt learning behaviors through gist compression. *arXiv preprint arXiv:2402.16058*, 2024.

[36] Yucheng Li. Unlocking context constraints of llms: Enhancing context efficiency of llms with self-information-based content filtering. *CoRR*, abs/2304.12102, 2023. doi: 10.48550/ARXIV.2304.12102. URL https://doi.org/10.48550/arXiv.2304.12102.

[37] 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, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 6342–6353. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.391. URL https://doi.org/10.18653/v1/2023.emnlp-main.391.

[38] 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. *arXiv preprint arXiv:2404.14469*, 2024.

[39] Barys Liskavets, Maxim Ushakov, Shuvendu Roy, Mark Klibanov, Ali Etemad, and Shane K Luke. Prompt compression with context-aware sentence encoding for fast and improved llm inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24595–24604, 2025.

[40] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[41] Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36, 2023.

[42] Erik Nijkamp, Tian Xie, Hiroaki Hayashi, Bo Pang, Congying Xia, Chen Xing, Jesse Vig, Semih Yavuz, Philippe Laban, Ben Krause, Senthil Purushwalkam, Tong Niu, Wojciech Kryscinski, Lidiya Murakhovs'ka, Prafulla Kumar Choubey, Alexander R. Fabbri, Ye Liu, Rui Meng, Lifu Tu, Meghana Bhat, Chien-Sheng Wu, Silvio Savarese, Yingbo Zhou, Shafiq Joty, and Caiming Xiong. Xgen-7b technical report. *CoRR*, abs/2309.03450, 2023. doi: 10.48550/ARXIV.2309.03450. URL `https://doi.org/10.48550/arXiv.2309.03450`.

[43] Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems. *CoRR*, abs/2310.08560, 2023. doi: 10.48550/ARXIV.2310.08560. URL `https://doi.org/10.48550/arXiv.2310.08560`.

[44] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

[45] Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 963–981. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.57. URL `https://doi.org/10.18653/v1/2024.findings-acl.57`.

[46] Shahzad Qaiser and Ramsha Ali. Text mining: use of tf-idf to examine the relevance of words to documents. *International journal of computer applications*, 181(1):25–29, 2018.

[47] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics.

[48] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[49] ByteDance Seed, Yufeng Yuan, Yu Yue, Mingxuan Wang, Xiaochen Zuo, Jiaze Chen, Lin Yan, Wenyuan Xu, Chi Zhang, Xin Liu, et al. Seed-thinking-v1. 5: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*, 2025.

[50] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics.

[51] Jiwei Tang, Jin Xu, Tingwei Lu, Hai Lin, Yiming Zhao, and Hai-Tao Zheng. Perception compressor:a training-free prompt compression method in long context scenarios. *CoRR*, abs/2409.19272, 2024. doi: 10.48550/ARXIV.2409.19272. URL `https://doi.org/10.48550/arXiv.2409.19272`.

[52] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[53] Maurice Weber, Daniel Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, et al. Redpajama: an open dataset for training large language models. *arXiv preprint arXiv:2411.12372*, 2024.

[54] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023.

[55] Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

[56] Yang Xu, Yunlong Feng, Honglin Mu, Yutai Hou, Yitong Li, Xinghao Wang, Wanjun Zhong, Zhongyang Li, Dandan Tu, Qingfu Zhu, et al. Concise and precise context compression for tool-using language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 16430–16441, 2024.

[57] Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. Compact: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439, 2024.

[58] 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`.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We demonstrate the advantages of our AdmTree in both performance and efficiency in the experiment section.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: In conclusion section, we point out the limitations.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide implemention details in experiment section and appendix.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We use public dataset.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide implemention details in experiment section and appendix.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report the average results over three inference runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide it in the Appendix

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Our research fully adheres to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discussed it in introduction.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: the paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: all creators and original owners of assets used in our paper have been properly credited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

Answer: [Yes]

Justification: we will release our code and model in Github with well-documented readme.

Guidelines:

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

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

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

Answer: [NA]

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

Guidelines:

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

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

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

Answer: [NA]

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

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [NA]

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

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Additional Related Work

A.1 Extending LLM Window Size

Unlike the above methods that improve LLM efficiency for long-text processing, some studies focus on expanding the context window of LLMs. A mainstream approach involves progressive pre-training strategies [17, 42, 49], which incrementally extend the maximum context length through phased training. Although these methods achieve strong performance, they incur high training costs and do not reduce inference overhead. In our experiments, we include fine-tuned LLMs trained on the same long-context datasets as strong baselines (Table 1).

B Experiment Setting

B.1 Method Details

Baselines Following prior work [26, 58] on compression, we introduce two main categories of baseline models for processing long contexts:

(i) Retrieval-based Methods. These methods employ a retriever to rank segmented long contexts based on their relevance to the given query (instruction). We utilize one sparse retrieval method **BM25** [48], and two dense retrieval methods, **Sentence-BERT** [47] and **OpenAI Embeddings**[3]. After assessing the relevance of the input context to the query using these models, segments with lower relevance are discarded iteratively until the required compression ratio is achieved.

(ii) Compression-based Methods. We select **LongLLMLingua** [26] as a representative of explicit compression methods. **AutoCompressor** [11], **ICAE** [21], **Activation Beacon** [58], and **SnapKV** [38] are employed as implicit compression methods. The former three methods recursively compresses long contexts into semantic vectors, enabling preliminary fine-grained compression.

We also fine-tuned the original LLM using the same training data to serve as the baseline, denoted as Original LLM-FT.

Backbone LLMs `LlaMA-2-7B-Chat` and `Qwen-2-7B-Instruct` are employed as the backbone LLMs. We use LlaMA-2-7B-Chat because three out of the four compression baselines utilize it. Throughout the experiments, we adopt LlaMA-2 as the backbone model unless explicitly stated otherwise.

Implementation Details In training AdmTree, we use the same dataset as Zhang et al. [58] to ensure fairness. Specifically, pre-training is conducted on 1B tokens sampled from RedPajama [53]. During fine-tuning, we utilize LongAlpaca [9], BookSum [31], and 16K synthetic samples generated by GPT-3.5 [18]. AdmTree is trained with a batch size of 8 on 8 NVIDIA A800 GPUs, with learning rates set to 5e-5 for pre-training and 1e-5 for fine-tuning, respectively.

For the compression ratio in main experiments, given LLaMA-2's maximum sequence length of 4K, we apply $\times 2$ compression for 4K–8K contexts, $\times 4$ for 8K–16K, and $\times 8$ for 16K–32K. For Qwen-2, we uniformly set the compression ratio to $\times 4$.

B.2 Dataset Details

Evaluation Benchmark In main experiments, we evaluate AdmTree's effectiveness and efficiency on **LongBench** [4] and **Needle-In-The-Haystack** [27]. LongBench is a multitask benchmark assessing LLMs' comprehension of long contexts across various tasks such as single- and multi-document QA and summarization, with 32K maximum length. Needle-In-The-Haystack assesses the model's ability to extract pertinent fact ("needle") from lengthy context ("haystack").

In other section, we further experiment on diverse benchmarks, including **GSM8K** [14] and **BBH** [50] for demonstration compression, **MSC** [44] and **ShareGPT** [12] for dialogue compression, Arxiv-March23 for summarization scenario, Topic Retrieval for retrieval scenario:

[3]We use `text-embedding-ada-002` at OpenAI.

longbenchv2 This dataset, introduced by Bai et al. [5], spans six major task categories and features documents ranging from 8,000 words to 2 million words in length. The benchmark presents a challenging four-choice question format, with performance evaluated using accuracy as the primary metric.

GSM8K This dataset [13] is a widely recognized benchmark for evaluating mathematical reasoning capabilities. It specifically assesses a model's proficiency in performing arithmetic operations and constructing coherent, step-by-step solutions using natural language. In our implementation, we employ a 2-shot learning approach with the original prompt being a complex, multi-step chain-of-thought (CoT) [16] formulation.

BBH This dataset [30] is designed for language and symbolic reasoning tasks, with a specific focus on evaluating chain-of-thought (CoT) reasoning capabilities. In our experiments, we employ two distinct CoT prompts as the baseline for comparison.

ShareGPT This dialogue dataset [1] is sourced from ShareGPT.com, a platform where users publicly share their ChatGPT interactions across multiple languages and diverse conversational scenarios (e.g., casual chats, writing assistance, and more). For our evaluation, we adopt the curated subset provided by [58] as our test set.

Arxiv-March23 This dataset is constructed from the most recent academic papers published in the arXiv preprint repository during March 2023. For evaluation purposes, we utilize a curated subset of 500 samples collected as our test set [36]. Input text length is limited to 10,000 characters and we employ DeepSeek to automatically generate summaries for each test instance, which serve as the reference answers [25].

Topic Retrieval This evaluation dataset [34] systematically examines model adaptability to varying context lengths and compression requirements. The framework employs structured multi-turn dialog sequences between human users and chatbots.

MSC Derived from MemGPT [43], this multi-session benchmark comprises the MSC dataset containing extended human-human dialogues (avg. 2,000 tokens). We systematically transformed the original dataset through prompt engineering to create length variants (1K and 4K tokens) for comprehensive evaluation of context-length robustness.

C Analysis Experiments on More Benchmarks and Scenarios

C.1 Compression on LongBench v2

To ensure consistency with settings in [58], the experiments in main body are conducted on the LongBench benchmark. In Table 4, we further report results on the more challenging LongBench v2 [5], which requiring deeper understanding and reasoning over long contexts. On LongBench v2, AdmTree demonstrates consistent absolute improvements and even more pronounced relative gains. Specifically, it outperforms the state-of-the-art Activation Beacon by +18.5% on LLaMA and +12.7% on Qwen. Moreover, on the newly introduced Long-dialogue History Understanding task, AdmTree brings a notable performance boost, demonstrating its superior ability to preserve dialogue structure and semantics during compression. Besides, compared to the performance on the original LongBench, Activation Beacon no longer consistently surpasses other baselines on some tasks (e.g. Multi-Document QA) in LongBench v2. This trend underscores the potential advantages of tree-structured compression in facilitating structured reasoning over long-context scenarios.

C.2 Compression on Reasoning Data

Following the experimental setup of Jiang et al. [25], GSM8K and BBH are employed to assess model performance in reasoning and in-context learning scenarios. We introduce a strong baseline, Full-shot, whose prompt lengths on both datasets remain within the maximum context window of LLaMA-2-7B. As shown in Table **??**, AdmTree consistently surpasses all baseline methods. Notably, even under the quarter-shot compression constraint, AdmTree outperforms the Full-shot baseline, achieving

Table 4: Experiment results (%) on LongBench v2 [5]. We also report the micro-averaged performance across all datasets.

Methods	LongBenchv2						
	SingleDoc	MultiDoc	ICL	History	Code	Data	**AVG**
LLama-2-7B							
AutoCompressor[11]	3.1	1.5	2.6	0.8	8.9	7.2	3.1
ICAE [21]	4.2	2.7	2.1	3.5	9.6	8.5	4.1
LongLLMLingua [26]	20.0	17.6	8.6	2.6	16.0	11.8	14.9
SnapKV [38]	18.1	15.2	23.1	16.2	18.3	19.2	18.1
Beacon [58]	20.0	14.4	25.9	18.0	22.0	21.2	19.5
AdmTree (Ours)	**22.9**	**19.2**	**27.2**	**25.6**	**24.0**	**24.2**	**23.1**
Qwen-2-7B							
LongLLMLingua [26]	22.3	19.6	12.3	8.5	13.2	20.1	18.1
SnapKV [38]	29.5	27.2	28.6	29.1	15.3	27.9	27.9
Beacon [58]	34.3	18.4	29.6	30.8	16.0	33.3	28.3
AdmTree (Ours)	**34.9**	**25.6**	**33.3**	**38.5**	**22.0**	**33.3**	**31.9**

Table 5: Experiment results (Exact Match, EM) under different compression ratios on the GSM8K mathematical reasoning [14] and Big-bench Hard (BBH) datasets [50].

Method	GSM8K			BBH		
	1-st. const. (∼5 x)	half-st. const. (∼14 x)	quarter-st. const. (∼20 x)	1-st. const. (∼3 x)	half-st. const. (∼5 x)	quarter-st. const. (∼7 x)
AutoCompressor	22.7	21.5	21.2	13.71	11.56	11.23
ICAE	37.76	35.21	33.12	21.67	20.12	19.83
Snapkv	44.56	42.95	43.81	38.61	36.83	36.01
LongLLMLingua	42.55	43.46	43.23	33.37	39.42	38.81
Beacon	45.74	44.36	44.13	43.95	44.25	42.44
AdmTree (Ours)	**48.12**	**47.93**	**46.85**	**45.06**	**49.7**	**53.53**
Zero-shot		41.87			36.49	
Full-shot		46.88			43.34	

compression ratios exceeding 20× and 7× on GSM8K and BBH, respectively. These results indicate that AdmTree effectively retains the reasoning process encoded in the original prompts. Interestingly, on the BBH dataset, higher compression ratios yield performance improvements for AdmTree. We attribute this to the relatively short full-shot prompts in BBH (fewer than 500 tokens), where fewer but more compact compressed tokens may enhance compression quality and thereby improve overall performance.

C.3 Compression on Summarization Scenario

We evaluate the performance of different compression models on the summarization task using an additional dataset, ArXiv-March23. As shown in Table 6, we compare the models under 2× and 4× compression ratios. Across all metrics, AdmTree consistently achieves strong performance. Besides, its performance under the 4× compression ratio does not show significant degradation compared to the 2× setting. In contrast, Activation Beacon performs noticeably worse on the BLEU metric. This may be attributed to the nature of BLEU, which emphasizes n-gram continuity, unlike ROUGE, which focuses on longest common subsequences. The lower BLEU score potentially suggests that Activation Beacon may generate less coherent summaries, potentially introducing unwanted or disjointed tokens.

C.4 Compression on Retrieval Scenario

We conduct additional experiments in a retrieval scenario using the Topic Retrieval dataset [] as a supplement to the Needle-in-a-Haystack experiment. Different from Needle-in-a-Haystack, which varies the position of the "needle" within a fixed-length context, Topic Retrieval varies the context length by modifying the number of total topics and then querying information about the first topic.

Table 6: Compression performance in summarization scenarios. The ArXiv-March23 dataset is used for evaluation. Metrics include ROUGE-1, ROUGE-2, ROUGE-L, BERTScore, and BLEU.

Methods	ROUGE-1	ROUGE-2	ROUGE-L	BS F1	BLEU
Original LLM	40.35	16.22	27.22	81.57	34.88
2× Compression Constraint					
AutoCompressor	9.05	0.24	6.31	12.43	5.05
ICAE	12.87	1.03	8.42	15.25	15.77
LongLLMLingua	31.78	9.29	20.01	74.24	24.81
SnapKV	33.23	12.56	25.1	73.09	30.23
Beacon	35.42	15.47	25.91	77.69	16.05
AdmTree (Ours)	**36.10**	**15.69**	**26.60**	**79.83**	**35.99**
4× Compression Constraint					
AutoCompressor	7.69	0.21	5.21	11.36	4.21
ICAE	10.56	0.89	7.65	13.21	13.52
LongLLMLingua	29.3	7.43	18.2	69.04	26.36
SnapKV	31.25	8.92	20.53	70.21	23.21
Beacon	32.18	14.12	23.96	74.25	17.22
AdmTree (Ours)	**35.51**	**15.06**	**25.84**	**79.97**	**34.82**

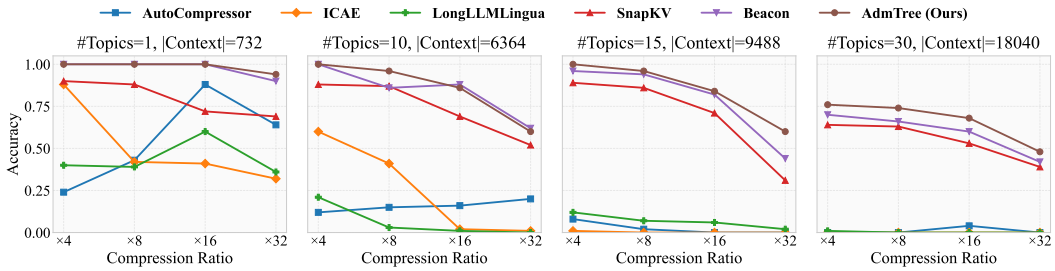


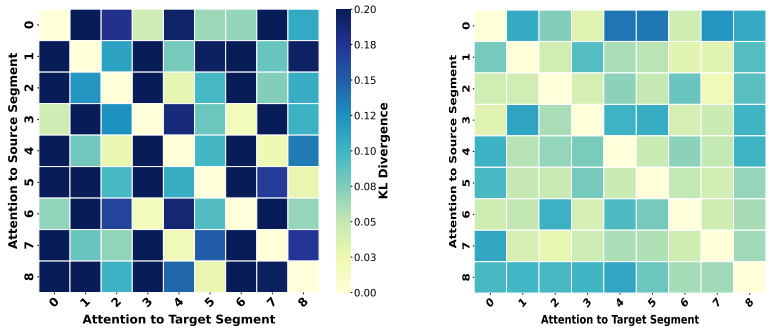
Figure 5: Retrieval accuracy comparison on the Topic Retrieval dataset under different context lengths and compression ratios.

Accuracy is used to measured the retrieval effectiveness after compression. Figure 5 presents the retrieval accuracy of different methods under varying compression ratios and context lengths. AdmTree consistently achieves the best performance across all conditions. Notably, at a shorter context length (Context Length = 732), AdmTree maintains near-lossless compression even at a 16 × compression ratio. For longer contexts, AdmTree also exhibits lower performance degradation compared to other methods. In contrast, AutoCompressor and LongLLMLingua fail to handle context lengths beyond 6000 tokens effectively.

D Visualization Experiments

D.1 Attention Pattern Analysis

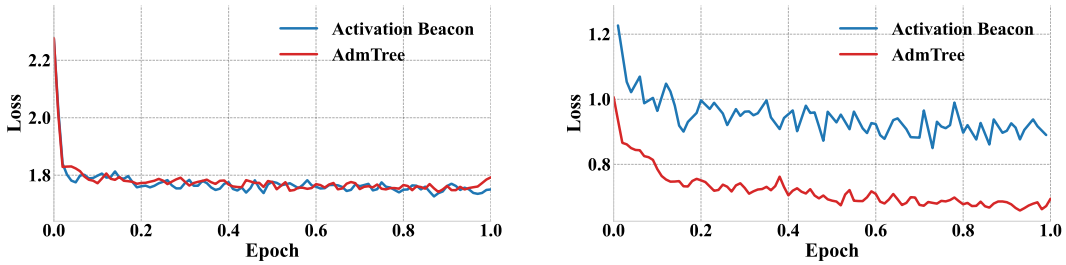
Figure 6 presents a KL divergence analysis of the attention distributions for tasks requiring varied semantic information, comparing AdmTree and Activation Beacon. Specifically, we measure the attention scores from the sentence's final token to the preceding compressed (gist) tokens at the onset of generation. Tasks 0–9 correspond to repetition tasks involving different semantic requirements, including the repetition of content from different positions (beginning, middle, end) and at varying granularities (word, sentence, paragraph). AdmTree demonstrates substantially greater diversity in its attention distributions across tasks, indicating that its compressed tokens encode distinct and semantically differentiated information. This diversity likely contributes to its improved performance by enabling more task-specific representation retrieval. Conversely, recursive compression methods such as Activation Beacon not only suffer from semantic degradation through iterative compression,



(a) AdmTree: Higher KL divergence across tasks, indicating greater variation in attention to compressed tokens.

(b) Activation Beacon: Lower KL divergence across tasks, indicating more uniform attention to compressed tokens.

Figure 6: The Kullback–Leibler (KL) divergence of the attention distribution from the final token to the compressed tokens across tasks of varying semantic types.



(a) Pre-training stage.

(b) Fine-tuning stage.

Figure 7: Loss curve during the different training stage of AdmTree and Activation Beacon.

but also tend to homogenize the semantic content across compressed tokens, thereby limiting their generalizability across diverse tasks.

D.2 Training Trends Comparison

AdmTree was trained on the same dataset as Activation Beacon. Figure 7 illustrates the loss convergence trends during both the pre-training and fine-tuning stages for the two methods. As shown, AdmTree achieves better convergence in the multi-task fine-tuning stage.

D.3 Case Study

We introduce the repetition task, which requires models to faithfully reproduce the original input text without omission or alteration. As shown in Table 7, this task provides an intuitive perspective through which we can observe how different methods preserve semantic information. Compared to the baseline model, our AdmTree achieves near-perfect reconstruction of the original text in both cases, merely leveraging the gist tokens within its semantic tree structure. In contrast, Activation Beacon exhibits severe hallucinations. In the first case, it fabricates content that does not exist in the original text based on its internal knowledge; in the second case, it falls into uncontrolled repetition. These qualitative comparisons clearly demonstrate that AdmTree is more capable of preserving the original semantics in a faithful and complete manner.

Table 7: Case studies on the repetition task. We sample several contexts from the GovReport subset of LongBench and evaluate whether models can accurately faithfully repeat the original content. We compare the outputs of our AdmTree and Activation Beacon. Content in red indicates hallucinated or redundant content that deviates from the original text.

Original Context	**Activation Beacon**	**AdmTree (Ours)**
Colombia, one of the oldest democracies in the Western Hemisphere and the third most populous Latin American country, has endured a multisided civil conflict for more than five decades until President Juan Santos declared the conflict over in August 2017 at the end of a U.N.-monitored disarmament. According to the National Center for Historical Memory 2013 report, presented to the Colombian government as part of the peace process to end the fighting, some 220,000 Colombians died in the armed conflict through 2012, 81% of them civilians. The report also provided statistics quantifying the scale of the conflict, which has taken a huge toll on Colombian society: more than 23,000 selective assassinations between 1981 and 2012; internal displacement of more than 5 million Colombians due to land seizure and violence; 27,000 kidnappings between 1970 and 2010; and 11,000 deaths or amputees from anti-personnel land mines laid primarily by Colombia's main insurgent guerrilla group, the Revolutionary Armed Forces of Colombia (FARC).	Colombia is a founding member of the Pacific Alliance, a trade bloc that seeks to deepen trade integration and investment with its partner countries and reduce barriers to bilateral trade between the four countries. The Pacific Alliance aims to create a common market, allowing for the free movement of businesses and people, and to serve as an export platform to the Asia-Pacific region. Colombia's leadership role in the Pacific Alliance and its accession to the Organization for Economic Cooperation and Development (OECD) in May 2018, following a review of its macroeconomic policies, are new developments. The record expansion of Colombia's coca crop and increasing cocaine exports to the United States, however, may significantly hinder the effort to consolidate peace in Colombia and could potentially increase corruption and extortion. A significant portion of the Colombian public remains skeptical of the peace process and the FARC 's role in Colombia's democracy. Other	Colombia, one of the oldest democracies in the Western Hemisphere and the third most populous Latin American country, has endured a multisided civil conflict for more than five decades until President Juan Ma nuel Santos declared the conflict over in August 2017 at the end of a U.N.-monitored disarmament. According to the National Center for Historical Memory 2013 report, presented to the Colombian government as part of the peace process to end the fighting, some 220,000 Colombians died in the armed conflict through 2012, 81% of them civilians. The report also provided statistics quantifying the scale of the conflict, which has taken a huge toll on Colombian society: more than 23,000 selective assassinations between 1981 and 2012; internal displacement of more than 5 million Colombians due to land seizure and violence; 27,000 kidnappings between 1970 and 2010; and 11,000 deaths or amputees from anti-personnel land mines laid primarily by Colombia's main insurgent guerrilla group, the Revolutionary Armed Forces of Colombia (FARC).
The Navy's proposed FY2019 budget requests a total of $2,347 million (i.e., about $2.3 billion) in procurement funding for the CVN-78 program. Congress's decisions on the CVN-78 program could substantially affect Navy capabilities and funding requirements and the shipbuilding industrial base. The Navy's FY2020 budget submission also proposed to not fund the mid-life nuclear refueling overhaul (called a Refueling Complex Overhaul, or RCOH) for the aircraft carrier CVN-75 (Harry S. Truman), and to instead retire the ship around FY2024 and also deactivate one of the Navy's carrier air wings at about the same time. On April 30, 2019, however, the Administration announced that it was effectively withdrawing this proposal from the Navy's FY2020 budget submission. The Administration now supports funding the CVN-75 RCOH and keeping CVN-75 (and by implication its associated air wing) in service past FY2024. For additional discussion of this withdrawn budget proposal, see Appendix A .	The Navy's to perform the test on the Navy's to perform the test on the Navy's to perform the test on the Navy's to perform the Navy's to the Navy's new aircraft carrier, the USS Gerald R. Ford, is scheduled to be delivered to the Navy in 2017. The Navy has been working to complete the ship's initial sea trials, which are scheduled to begin in May 2017.	The Navy's proposed FY2019 budget requests a total $2,347 million (i.e., about $2.3 billion) in procurement funding for the CVN-78 class shipbuilding program. Congress's decisions on the CVN-78 class program and other Navy shipbuilding programs may have significant substantially effects on the Navy's capabilities and funding requirements and the shipbuilding industrial base. The Navy's FY2020 budget submission also proposed not to fund the mid-life nuclear refueling overhaul (called a Refueling Complex Overhaul, or RCOH), for the aircraft carrier CVN-75 (Harry S. Truman), and to instead retire the ship around FY2024 and also deactivate one of the Navy's carrier air wings at about the same time. On April 30, 2019, however, the Administration announced that it was effectively withdrawing this proposal from the Navy's FY2020 budget submission. The Administration now supports funding the CVN-75 RCOH and keeping CVN-75 (and by implication its associated air wing) in service past FY2024. For additional discussion of this withdrawn budget proposal, see Appendix A.