FAEDKV: Infinite-Window Fourier Transform for Unbiased KV Cache Compression

Runchao Li[1] **Yao Fu**[1] **Mu Sheng**[1] **Xianxuan Long**[1] **Haotian Yu**[1] **Pan Li**[2]*
[1]Case Western Reserve University
[2]Hangzhou Dianzi University, Hangzhou, China
{rxl685,yxf484,mxs2090,xxl1514,hxy692}@case.edu,
lipan@ieee.org

Abstract

The efficacy of Large Language Models (LLMs) in long-context tasks is often hampered by the substantial memory footprint and computational demands of the Key-Value (KV) cache. Current compression strategies, including token eviction and learned projections, frequently lead to biased representations—either by overemphasizing recent/high-attention tokens or by repeatedly degrading information from earlier context—and may require costly model retraining. We present FAEDKV (Frequency-Adaptive Infinite-Window for KV cache), a novel, training-free KV cache compression framework that ensures unbiased information retention. FAEDKV operates by transforming the KV cache into the frequency domain using a proposed Infinite-Window Fourier Transform (IWDFT). This approach allows for the equalized contribution of all tokens to the compressed representation, effectively preserving both early and recent contextual information. A preliminary frequency ablation study identifies critical spectral components for layer-wise, targeted compression. Experiments on LongBench benchmark demonstrate FAEDKV's superiority over existing methods by up to **22%**. In addition, our method shows superior, position-agnostic retrieval accuracy on the Needle-In-A-Haystack task compared to compression based approaches.

1 Introduction

LLM has become the paradigm in language-generating tasks. It can perform variety of important tasks such as text generation, question answering, mathmatical problem-solving. For this types of problems, a long context is often required to provide enough background information, thus they demand the model have long context capability. Recently, chain-of-thought reasoning models have earned popularity due to their ability to breakdown the problems into small steps and solve them in a reasoning process. It also requires sufficiently long generated text to solve complex problems.

However, Transformers(Vaswani et al., 2023) inherently struggle with long sequences. While their quadratic self-attention complexity is a known bottleneck, autoregressive decoding mitigates this for subsequent tokens using a Key-Value (KV) cache. This cache, however, introduces its own challenge: its memory footprint grows linearly with context length, rapidly consuming inference resources.

Recent approaches to mitigate KV cache memory costs primarily involve token pruning. For instance, H2O (Zhang et al., 2023b) evicts tokens with low accumulated attention scores to maintain a target cache size, while PyramidKV (Cai et al., 2024b) extends this by dynamically allocating cache budgets across layers and selecting tokens deemed most important. While such methods can reduce KV cache sizes during inference, their reliance on attention scores as a primary selection criterion introduces a bias. This bias favors tokens with high immediate relevance to the current query, potentially leading to the premature eviction of important tokens, a phenomenon related to the 'lost in the middle' problem (Liu et al., 2023b).

Alternatively, learning-based compression techniques have been applied to the KV cache. For example, ActivationBeacon (Zhang et al., 2024) learns to condense preceding tokens into a compact 'activation beacon,' while LOCOCO (Cai et al., 2024a) employs 1-D convolutional networks to project keys and values into compressed representations. Although these data-driven approaches can effectively reduce cache size, they often rely on a learned compression module activated when the cache exceeds a predefined threshold. This can lead to repeated compression of earlier tokens as the context window expands, progressively de-

*Corresponding author

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 16856–16866
November 4-9, 2025

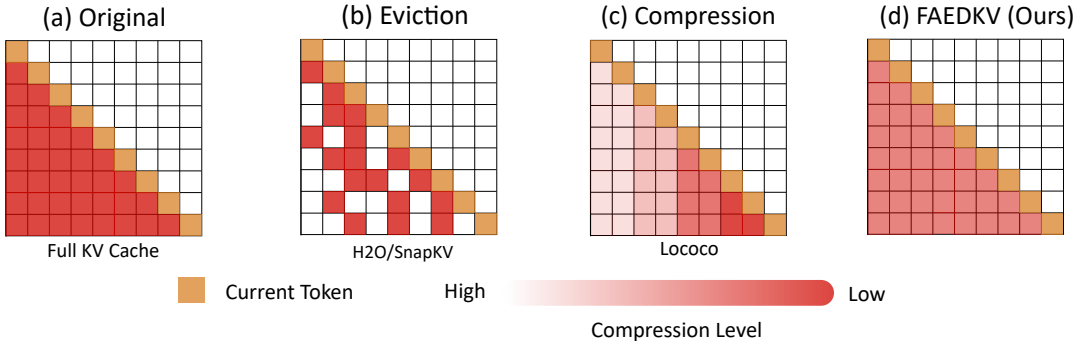

Figure 1: Conceptual illustration of how different KV cache management strategies process past tokens, highlighting their inherent biases. (a) Full KV Cache: Represents unbiased, complete retention. (b) Eviction strategies (e.g., H2O/SnapKV): Clearly show biased token removal. (c) Learned compression (e.g., Lococo): Illustrates bias through targeted, often heavier, compression of older tokens. (d) FAEDKV (Ours): Its visually consistent treatment of all past tokens underscores an algorithmic approach designed to operate without introducing arbitrary biases.

grading their information content. Moreover, these methods frequently necessitate model fine-tuning or the training of auxiliary modules, demanding significant computational resources.

This paper introduces Frequency-Adaptive Infinite Window for KV cache (FAEDKV), a novel algorithm to address these challenges. FAEDKV transforms the KV cache into the frequency domain, ensuring balanced preservation of information from all tokens. Its methodology involves a layer-wise frequency ablation study to identify critical spectral components and a novel Infinite-Window Fourier Transform (IWDFT) for managing the frequency-domain cache. Unlike common approaches leading to abrupt eviction or repeated compression of older tokens (conceptualized in Figure 1 (b,c)), FAEDKV more consistently retains their information (Figure 1 (d)), enabling targeted frequency filtering for compression. FAEDKV is universally compatible, requires no fine-tuning, and integrates via a one-time ablation study and modification of the attention module.

Contribution:

- We propose **FAEDKV**, a novel method achieving **unbiased Key-Value (KV) cache compression** by transforming entries into the frequency domain, thereby mitigating prevalent contextual biases found in existing techniques.
- We develop a supporting frequency-based framework featuring a novel **Infinite-Window Fourier Transform (IWDFT)** for efficient, recursive cache updates, and a **frequency ablation study** for targeted, layer-wise spectral pruning to optimize compression.
- Experiments demonstrate FAEDKV's superior performance, significantly outperforming established baselines on LongBench by up to 22% with 9% cache size and achieving consistent, position-agnostic retrieval accuracy in Needle-in-a-Haystack tests.

2 Related Works

2.1 KV Cache Compression

Managing the extensive Key-Value (KV) cache in Large Language Models (LLMs) to reduce memory overhead and latency is a significant research focus (Ge et al., 2023; Liu et al., 2024a). Many approaches selectively prune the cache by evicting less important tokens based on attention scores or other heuristics, such as H_2O (Zhang et al., 2023a), Scissorhands (Liu et al., 2023a), and SnapKV (Li et al., 2024c). Other strategies involve learned projections or recursive compression, like LoCoCo (Cai et al., 2024a), which may require fine-tuning as discussed in broader analyses of learned long-context methods (Tan et al., 2024; Fu et al., 2025b).

Another line of work focuses on obtaining more compact cache representations through sparse decomposition. This includes low-rank approximation methods such as PALU (Chang et al., 2024), as well as approaches that achieve sparsity via

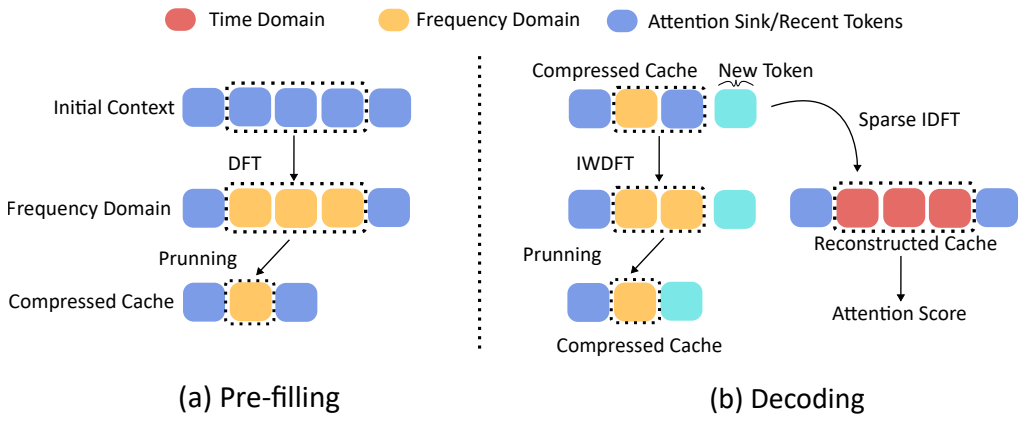

Figure 2: Overview of the FAEDKV workflow. (a) Pre-filling: The middle segment of the initial context is converted to the frequency domain (orange) by DFT, pruned, and stored with sink/recent tokens (blue). (b) Decoding: The compressed frequency-domain segment (orange) is updated via IWDFT (with new token information) and re-pruned. For attention, it's reconstructed to the time domain (red) by Sparse IDFT and combined with sink/recent tokens (blue) before attention score calculation.

aggressive quantization (e.g., KIVI (Liu et al., 2024b)) or structured, layer-wise attention (e.g., MLA (DeepSeek AI, 2024)). However, these methods operate primarily in the *embedding dimension*, reducing the per-token representation size rather than the sequence length, which is the main focus of our work. Therefore, sparse embedding-based approaches like PALU and MLA are complementary but not directly comparable to our method.

Architectural modifications such as Grouped-Query Attention (GQA) (Ainslie et al., 2023) and other attention-aware compression techniques (Ge et al., 2024) further attempt to reduce cache size. While these methods effectively decrease memory usage, they often introduce unequal token influence, with some tokens being discarded or heavily down-weighted. Our approach diverges by ensuring that every token contributes more evenly to the compressed representation.

2.2 Fourier Transform in LLM

Concurrently, Fourier transforms are proving instrumental in analyzing and enhancing LLMs. Studies reveal LLMs utilize Fourier features internally for tasks like arithmetic, encoding information across different frequency components (Nanda et al., 2023; Murty et al., 2024). Beyond analysis, frequency-domain techniques are actively improving model efficiency and capabilities. For example, Transformer FFNs have been reinterpreted as frequency transformers (Lee et al., 2024), and methods like AFFormer (Li et al., 2024b) incorporate Fourier-inspired adaptive filters. Fourier features are also applied in specialized areas such as LLM-based time-series forecasting (Zhou et al., 2024), creating robust positional embeddings like RoFormer (Su et al., 2024), and developing novel frequency-domain attention mechanisms like FNet (Lee-Thorp et al., 2021). While these studies highlight the versatility of Fourier methods in LLMs, their specific application to KV cache compression with an emphasis on uniform token contribution—as explored in our work—remains a novel direction(Fu et al., 2025a; Long et al., 2025).

3 Preliminaries

Existing KV cache compression largely relies on eviction or learning-based techniques, which can introduce recency bias. Our approach differs by applying a novel frequency-domain transformation to the KV cache, aiming for equalize token contribution to the compressed state. This requires empirically analyzing frequency component importance via ablation experiments that measure perplexity changes upon pruning. This section outlines preliminaries and the frequency ablation study.

3.1 Background

3.1.1 KV Cache in Autoregressive Decoding

During generation, we omit batch and head dimensions for clarity. Let $\mathbf{x}^t \in \mathbf{R}^{1\times d}$ be the input embedding at step t, and let $W^Q, W^K, W^V \in \mathbf{R}^{d\times d}$

be the projection matrices. The query, key, and value vectors are computed as

$$\mathbf{q}^t = \mathbf{x}^t W^Q, \mathbf{k}^t = \mathbf{x}^t W^K, \mathbf{v}^t = \mathbf{x}^t W^V. \quad (1)$$

The key and value caches grow by appending the new vectors:

$$\mathbf{K}^{t+1} = \begin{bmatrix} \mathbf{K}^t \\ \mathbf{k}^t \end{bmatrix}, \mathbf{V}^{t+1} = \begin{bmatrix} \mathbf{V}^t \\ \mathbf{v}^t \end{bmatrix}, \quad (2)$$

so that $K^t, V^t \in \mathbf{R}^{t \times d}$.

The attention output for the next token is then

$$o^t = \text{Softmax}\Big(\frac{\mathbf{q}^t \left[\mathbf{K}^t\right]^\top}{\sqrt{d}}\Big) \mathbf{V}^t. \quad (3)$$

While caching prevents the redundant recomputation of past keys and values, allowing the attention calculation for each new token to be performed in $O(t)$ time with respect to the current context length t, the KV cache itself incurs a memory cost of $O(t)$. For very long sequences, both this linear growth in memory and the per-step computational cost become substantial. This challenge motivates our frequency-domain compression strategy, aimed at reducing the effective cache size—and thereby both memory and computational overheads for attention—without sacrificing the model's ability to attend to both recent and distant tokens.

3.1.2 Discrete Fourier Transform

The Discrete Fourier Transform (DFT) is a fundamental tool for analyzing signals in the frequency domain, widely used in fields like voice and image processing. In the context of Large Language Models (LLMs), a sequence of vectors (such as those in the KV cache along the token dimension) can be treated as a 1-D time-domain signal. Applying a 1-D DFT to such sequences transforms them into the frequency domain, which can reveal structural properties and allow for targeted manipulation, forming the basis for our compression approach.

The DFT converts a finite discrete-time sequence $x[0], x[1], \ldots, x[N-1]$ into its frequency-domain representation $X^f[0], X^f[1], \ldots, X^f[N-1]$. This transformation is defined as:

$$X^f[k] = \sum_{n=0}^{N-1} x[n](W_k)^n,$$
$$\text{where } W_k = e^{-j\frac{2\pi k}{N}}, \quad k = 0, 1, \ldots, N-1. \quad (4)$$

In this formulation, $x[n]$ is the input signal at time n, $X^f[k]$ is the k-th frequency component, N is the sequence length, j is the imaginary unit, and W_k is the complex exponential term $e^{-j\frac{2\pi k}{N}}$ (often referred to as a twiddle factor). This transformation can be computed efficiently using the Fast Fourier Transform (FFT) algorithm, which has an $O(N \log N)$ time complexity and $O(N)$ memory cost.

To utilize the frequency-domain representation for attention, the time-domain KV cache vectors must be reconstructed. This is achieved using the *inverse DFT* (IDFT). Given the frequency components $X^f[k]$ and our previously defined $W_k = e^{-j\frac{2\pi k}{N}}$, the IDFT reconstructs the time-domain sequence $\widetilde{x}[n]$ as:

$$\widetilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} X^f[k](W_k^*)^n,$$
$$\text{where } W_k^* = e^{j\frac{2\pi k}{N}}, n = 0, 1, \ldots, N-1. \quad (5)$$

This reconstruction can be performed efficiently using an Inverse Fast Fourier Transform (IFFT) with $O(N \log N)$ time complexity. We adopt this method for reconstructing our KV cache from its compressed frequency-domain representation. While the initial pre-filling stage for attention calculation over N tokens is typically $O(N^2)$, our DFT-based enables us to reduce the initial latency. We verify it at Section 5.4.

3.2 Frequency Ablation Study

Previous studies (He et al., 2023; Kai et al., 2025) have employed the Discrete Cosine Transform (DCT) for analyzing model components, often finding energy concentrated in lower frequencies and thus retaining only these for compression. We choose a DFT approach over DCT because DCT's implicit symmetric signal extension (mirroring) contributes to its strong emphasis on lower-frequency bins. It could leads to the loss of critical higher-frequency details.

To assess the relative importance of different spectral bands in the KV cache, we perform a layer-wise frequency ablation study. We randomly sampled 100 texts from WikiText-103-v1 (Merity et al., 2016), processing each up to the model's maximum token sequence length, denoted as N. For each Transformer layer $\ell \in \{1, \ldots, L_{layers}\}$ (where L_{layers} is the total number of layers), we compute the DFT of the keys and values along

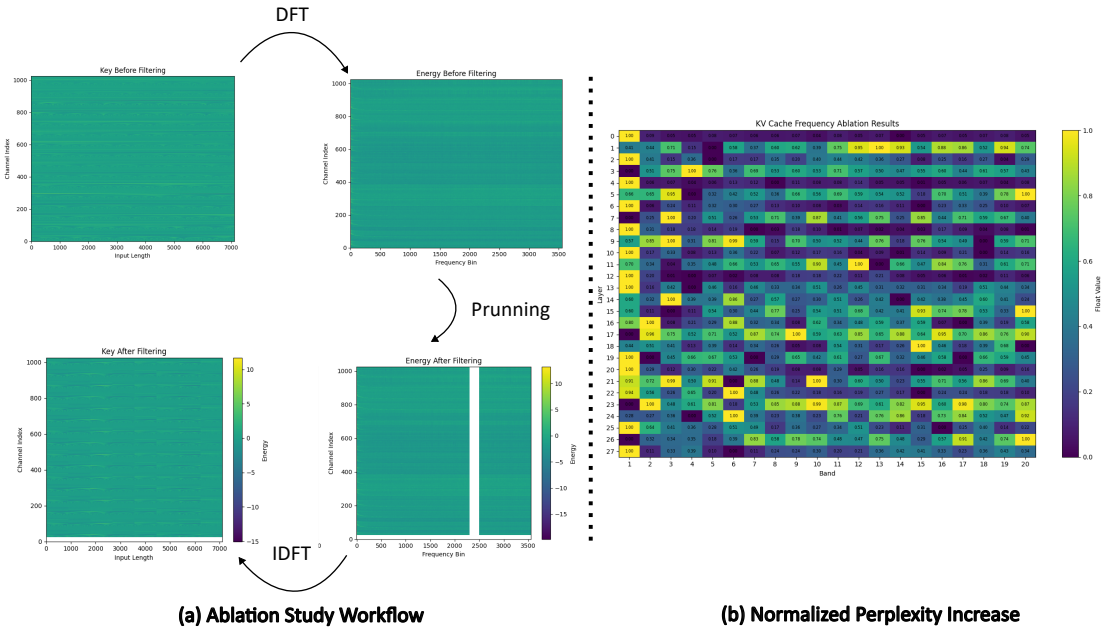

Figure 3: Overview of the Frequency Ablation Study. (a) The workflow illustrates the process: time-domain Key/Value cache data ("Key Before Filtering") is transformed via DFT into its frequency-domain energy representation ("Energy Before Filtering"). Specific frequency bands are then pruned ("Energy After Filtering"), and the corresponding time-domain data is reconstructed via IDFT ("Key After Filtering") to evaluate performance impact. (b) A heatmap displays the Normalized Perplexity Increase ($\Delta_{\ell,c}$) resulting from ablating different frequency bands (x-axis, 1-20) across various model layers (y-axis, 0-27). Higher float values (brighter colors, from 0.0 to 1.0) indicate greater importance of the ablated band to model performance.

the token sequence length dimension. This DFT yields N frequency bins. These N bins are partitioned into C contiguous *chunks*. Each chunk $c \in \{1, \ldots, C\}$ consists of N/C frequency bins. Let $B_c \subset \{0, \ldots, N-1\}$ denote the set of frequency indices belonging to chunk c.

During the ablation study, for each layer ℓ and each chunk c, we zero out (prune) all DFT coefficients $X_\ell^f[k]$ where $k \in B_c$. The modified coefficients $\hat{X}_\ell^f[k]$ are thus defined as:

$$\hat{X}_\ell^f[k] = \begin{cases} 0, & \text{if } k \in B_c, \\ X_\ell^f[k], & \text{if } k \notin B_c. \end{cases} \tag{6}$$

Using these pruned coefficients $\hat{X}_\ell^f[k]$, we reconstruct the key and value tensors for layer ℓ via the IDFT. This reconstructed KV cache temporarily replaces the original one for that layer to evaluate the impact of pruning chunk B_c. We record the resulting model perplexity as $\mathrm{PPL}_{\ell,c}$. For baseline comparison, let $\mathrm{PPL}_{\mathrm{orig}}$ be the perplexity of the model with an unmodified KV cache. The normalized perplexity increase, $\Delta_{\ell,c}$, is then defined as:

$$\Delta_{\ell,c} = \frac{\mathrm{PPL}_{\ell,c} - \mathrm{PPL}_{\mathrm{orig}}}{\mathrm{PPL}_{\mathrm{orig}}}, \tag{7}$$

This metric, $\Delta_{\ell,c}$, quantifies the importance of frequency chunk c at layer ℓ; a larger value indicates a more significant contribution of that chunk to the model's performance.

Our analysis of these $\Delta_{\ell,c'}$ values reveals that while low-frequency components often demonstrate greater importance, many high-frequency components (or chunks containing them) also yield significant Δ values, indicating their non-negligible role. This finding suggests that a simple low-pass filtering approach might be suboptimal. The whole process is visualized in Figure 3.

Thus, to retain as much critical information as possible across the spectrum, we employ a greedy compression strategy. For each layer, given a desired retention ratio r, we select the top $r \cdot C$ most important frequency chunks and discard the remainder. C is a hyperparameter that controls the granularity of ablation study. We evaluate its effect in our experiments at Section 5.5.

4 FAEDKV

4.1 Infinite Window DFT

The standard Discrete Fourier Transform (DFT), as defined in Equation (4), provides a static analysis of an entire input sequence. However, in autoregressive Transformer models, new tokens are generated sequentially. If we were to recompute the DFT over the entire growing KV cache (of current length N) at each decoding step, this would incur a computational cost of $O(N \log N)$ using FFT (or $O(N^2)$ naively) for that single step. This is problematic when compared to the typical complexities of decoder-only Transformers. We notice that *sliding-window DFT* is better being recursively updated. With window size M, it updates each frequency bin as:

$$S_{t+1}[k] = W_k\big(S_t[k] - x_{t-M+1} + x_{t+1}\big),$$
$$W_k = e^{-j\frac{2\pi k}{M}}, \quad \text{for } k = 0, 1, \ldots, M-1. \tag{8}$$

Here, $x[t+1]$ is the new input sample entering the window, and $x[t-M+1]$ is the oldest sample leaving the window.The *sliding-window DFT* efficiently updates the windowed frequency representation at $O(M)$ per time step to update all M frequency bins.

Despite this, it presents two major drawbacks for our goal of KV cache compression. Firstly, it necessitates storing all M time-domain samples of the current window. Secondly, this subtraction of $x[t-M+1]$ completely removes information about tokens older than the M-sample window.

To address the fixed memory window and associated storage overhead of the standard sliding-window DFT, an intuitive first step is to remove the subtraction of the oldest sample ($x[t-M+1]$). This creates a conceptually infinite, recursive window. However, this simpler recursion would lead to unbounded accumulation of values in the frequency-domain state $S_t[k]$, risking floating-point overflow for very long sequences common in LLMs(Lee et al., 2025).

Our Infinite Window DFT (IWDFT), defined in Equation 9, prevents such overflow by incorporating a normalization factor based on the current sequence length, N, into each recursive update:

$$S_{t+1}[k] = W_k \left(\frac{N-1}{N} S_t[k] + \frac{1}{N} x[t+1] \right). \tag{9}$$

Here, $S_t[k]$ is the previous state for the k-th frequency bin. We approximate term $\frac{N-1}{N}$ to 1 since most context is larger than 1000.

This IWDFT approach is applied during decoding to update the K and V caches. It resolves the primary issues of the standard sliding window by avoiding extra time-domain storage and the hard cut-off of past information. Unlike methods that can introduce bias (Cai et al., 2024a; He et al., 2023), IWDFT processes each token's contribution consistently. The update is efficient ($O(N_{DFT})$ per step), training-free, and compatible with autoregressive LLMs.

4.2 Workflow

In this section, we detail our workflow, shown in Figure 2. Our overall idea is to transform Key and Value (KV) caches into the frequency domain using the Infinite Window DFT (IWDFT), and then compress this representation by selectively retaining frequency regions based on importance scores (Δ) derived from an ablation study. Initially, for each model, this ablation study is performed.By measuring the normalized perplexity increase (Δ) that occurs when each chunk is temporarily removed, we obtain layer-specific importance scores for these C distinct spectral chunks. Given a desired retention ratio r, we then select the top $r \cdot C$ most important frequency chunks per layer based on these Δ scores. We refer to this process "pruning".

4.2.1 Pre-filling Stage

In the pre-filling stage, an initial input context of length N (e.g., up to $100k$ tokens) is processed with an approximate $O(N^2)$ attention complexity to generate the Key-Value (KV) cache. To prepare for compression, we transform a segment of this cache to the frequency domain using the Fast Fourier Transform (FFT) with Equation 4). Guided by studies highlighting the importance of initial and recent tokens for "attention sinks"(Han et al., 2024), we exclude the first S and last R tokens from this transformation. Thus, only the middle segment of $M = N - S - R$ tokens from each layer's KV cache undergoes DFT:

$$\mathbf{K}^f_{0:M-1} = \mathrm{DFT}\big(\mathbf{K}[S : S+M-1]\big),$$
$$\mathbf{V}^f_{0:M-1} = \mathrm{DFT}\big(\mathbf{V}[S : S+M-1]\big). \tag{10}$$

Subsequently, these M-length frequency-domain representations, $\mathbf{K}^f_{0:M-1}$ and $\mathbf{V}^f_{0:M-1}$, are pruned layer-wise. Using the set of important frequency components B^*_ℓ identified in Section 3.2 and the rule from Equation 6, we obtain the compressed

versions $\hat{\mathbf{K}}^f_{0:M-1}$ and $\hat{\mathbf{V}}^f_{0:M-1}$. Storing only these selected components significantly reduces memory for the full N-token KV cache from $O(N)$ to $O(N \cdot r)$, where r denotes the cache compression ratio.

4.2.2 Decoding Stage

At each decoding step t, we have three operations:

Cache Reconstruction. At each decoding step t, the compressed frequency-domain KV caches, $\hat{\mathbf{K}}^f_t$ and $\hat{\mathbf{V}}^f_t$ (inherited and updated from the previous step), are transformed back to the time domain using the IDFT operation from Equation 5. This yields the reconstructed caches $\widetilde{\mathbf{K}}_t$ and $\widetilde{\mathbf{V}}_t$:

$$\begin{aligned} \widetilde{\mathbf{K}}_t &= \mathrm{IDFT}\big(\hat{\mathbf{K}}^f_t\big), \\ \widetilde{\mathbf{V}}_t &= \mathrm{IDFT}\big(\hat{\mathbf{V}}^f_t\big). \end{aligned} \tag{11}$$

To optimize this reconstruction, we leverage the sparsity inherent in the compressed $\hat{\mathbf{K}}^f_t$ and $\hat{\mathbf{V}}^f_t$ by employing a sparse IDFT implementation. It speed up the process by only working on the non-zero frequency components. We show this in our experiment at Section 5.4

Updating Cache As new tokens (k_t, v_t) are generated and added to a recent time-domain window of size R, the tokens that age out of this window (e.g., k_{t-R}, v_{t-R}) are incorporated into the historical compressed KV cache. This update leverages our IWDFT mechanism, as defined in Equation 9. The update process is:

$$\begin{aligned} \mathbf{K}^f_{t+1} &= \mathrm{IWDFT}\big(\hat{\mathbf{K}}^f_t, k_{t-R}\big), \\ \mathbf{V}^f_{t+1} &= \mathrm{IWDFT}\big(\hat{\mathbf{V}}^f_t, v_{t-R}\big). \end{aligned} \tag{12}$$

Here, $\hat{\mathbf{K}}^f_t$ and $\hat{\mathbf{V}}^f_t$ are the previous compressed frequency-domain states, and $\mathbf{K}^f_{t+1}, \mathbf{V}^f_{t+1}$ are the updated states. This IWDFT process ensures that each token aging into the historical cache is incorporated consistently, allowing early context to maintain a sustained influence. Since the IWDFT update operates on all its maintained frequency bins, it can repopulate components that were previously zero due to pruning. After each IWDFT update, the pruning rule (Equation 6, using the selected components B^*_ℓ) must be reapplied to maintain the compression level. In practice, we only compute and store the coefficients for the selected frequency components B^*_ℓ.

Calculating Attention We assemble our $\mathbf{K}$ and $\mathbf{V}$ as follow:

$$\begin{aligned} \mathbf{K}_t &= \big[\mathbf{K}_{[0:S]},\ \widetilde{\mathbf{K}}_t,\ \mathbf{K}_{[N-R:N-1]}\big], \\ \mathbf{V}_t &= \big[\mathbf{V}_{[0:S]},\ \widetilde{\mathbf{K}}_t,\ \mathbf{V}_{[N-R:N-1]}\big], \end{aligned} \tag{13}$$

As shown in the equation, the current KV pair consist of "attention-sink" tokens, reconstructed tokens and the last R recent tokens.

Finally, the attention output $\mathbf{o}^t$ is computed using the assembled $\mathbf{K}_t$ and $\mathbf{V}_t$ caches (Equation 3). Critically, FAEDKV requires no model fine-tuning for its integration. By reconstructing compressed KV cache segments to their original length, our method ensures the model operates within its existing architectural and positional embedding limits. This approach differs from other works(Tan et al., 2024; Cai et al., 2024a) that designed to extend the model's inherent context.

5 Experiments

5.1 Setup

We use Llama3-8B for long-context Question Answering (QA) and Qwen2-7B-Instruct for Needle-in-the-Haystack evaluations. Initial frequency ablation studies on both models informed our approach, leading us to segment the frequency spectrum into $C_{chunks} = 22$ chunks (further details in Section 5.5). Our method explicitly retains the first $S = 10$ tokens as attention sinks and the most recent $R = 50$ tokens, incurring negligible KV cache overhead from these. All experiments employed greedy decoding, and baseline methods were implemented using their officially provided code for fair comparison.

5.2 QA Datasets

We evaluate our approach on Llama3-8B-Instruct model on the LongBench benchmark(Bai et al., 2024b), which comprises 16 tasks across six categories—single-document QA, multi-document QA, summarization, few-shot learning, synthetic tasks, and code completion—with an average context length of roughly 11 000 tokens. We compare against three baselines: **H2O**(Zhang et al., 2023b), an eviction based compression method; SnapKV(Li et al., 2024c), the state of art for long context tasks; and **full KV cache** that stores all KV pairs without compression. We set the baseline compression method's cache size to 512, 1024,2048 respectively. Correspondingly, we set our method's compression ratio r to 0.094, 0.125 and 0.25 to facilitate fair comparison.

Table 1 presents our model's performance across all LongBench tasks and cache sizes. On average, FAEDKV improves accuracy by 2.91 points compared to H2O and by 2.12 points compared to

Method	Single-Doc QA			Multi-Doc QA			Summarization			Few-shot			Synthetic		Code		Avg.
	NtrvQA	QAsper	MF-en	HotpotQA	2WikiMQA	Musique	GovReport	QMSum	MultiNews	TREC	TriviaQA	SAMSum	PCount	PRe	LCC	RB-P	
FullKV	22.53	26.29	41.52	37.46	29.24	21.56	29.47	22.23	25.85	65.00	81.00	40.12	4.65	4.75	39.42	43.63	33.42
Cache Size = 768 (Compression Ratio = 0.094)																	
H2O	7.67	11.81	25.92	20.40	17.37	7.32	12.86	9.18	12.56	24.50	49.18	26.72	0.0	0.0	27.61	32.29	17.83
SnapKV	11.84	12.35	30.86	26.54	17.01	13.22	15.71	9.86	16.66	29.50	55.89	28.54	0	3.00	20.97	26.15	19.91
FAEDKV	15.23	18.08	39.12	31.62	21.09	15.48	19.02	14.26	19.70	43.5	66.54	30.70	2.45	3.20	28.47	34.86	**25.21**
Cache Size = 1024 (Compression Ratio = 0.125)																	
H2O	16.97	19.94	41.54	34.26	24.03	17.53	22.83	19.40	23.26	55.50	78.65	33.68	3.24	4.25	33.89	38.52	28.69
SnapKV	17.98	19.35	40.67	35.02	24.46	16.24	24.72	18.00	23.27	60.00	77.73	35.03	3.18	4.16	33.07	38.62	28.86
FAEDKV	17.49	20.24	42.40	34.26	25.84	17.39	23.13	18.55	21.61	61.50	77.51	34.27	3.26	4.50	35.30	39.68	**29.45**
Cache Size = 2048 (Compression Ratio = 0.25)																	
H2O	21.63	23.89	45.32	37.53	29.21	20.72	28.11	22.54	25.78	63.50	81.10	38.47	3.50	4.50	38.13	43.22	32.95
SnapKV	22.12	25.26	44.03	38.13	30.06	22.99	28.53	22.53	26.35	64.50	80.10	38.57	3.50	4.50	32.19	38.22	32.60
FAEDKV	21.47	24.61	45.61	38.50	30.04	21.29	27.98	22.67	26.29	64.50	81.05	39.21	3.50	4.25	37.84	43.16	**33.24**

Table 1: LongBench performance (perplexity) across 16 tasks and varying cache sizes. Bold indicates the best compressed method at each cache size.

SnapKV, while it remains slightly below the full-attention baseline.

Importantly, under extremely tight cache budgets, FAEDKV outperforms eviction-based methods by up to **22 %**. We attribute this advantage to the fact that eviction strategies tend to bias toward current tokens and discard valuable information that has low attention scores toward the current tokens. In contrast, our frequency-domain compression precisely identifies and preserves the most informative spectral components across the entire context, yielding a more balanced retention of information in the most constrained settings.

5.3 Fact Retrieval

To assess FAEDKV's ability to preserve information integrity across varying context lengths and token positions, we employed the "Needle in a Haystack" benchmark(Li et al., 2024a). This test evaluates an LLM's capacity to retrieve specific information embedded within a larger text corpus. We used excerpts from THUDM's implementation(Bai et al., 2024a), creating contexts of 8K-30K tokens. A unique factual statement was inserted as the needle at 9 relative positions within each from 0% to 100% document depth.

Our experimental procedure involved presenting a Qwen2.5-7B-Instruct model with these augmented documents. We compared with LoCoCo(Cai et al., 2024a), a convulution based compression approach. Both LoCoCo and FAEDKV were evaluated on 1024 ($r = 0.05$ of 24K) Cache size.

The results demonstrate two key advantages of FAEDKV. Firstly, FAEDKV generally achieved higher retrieval accuracy compared to the LoCoCo baseline across various context lengths and compression ratios. Secondly, and critically for our design, FAEDKV exhibited markedly more consistent accuracy irrespective of the needle's position within the haystack. This is attributed to its core mechanism employing the DFT, which inherently processes all token information with equal weight, ensuring a more uniform preservation of contextual details throughout the compressed KV cache.

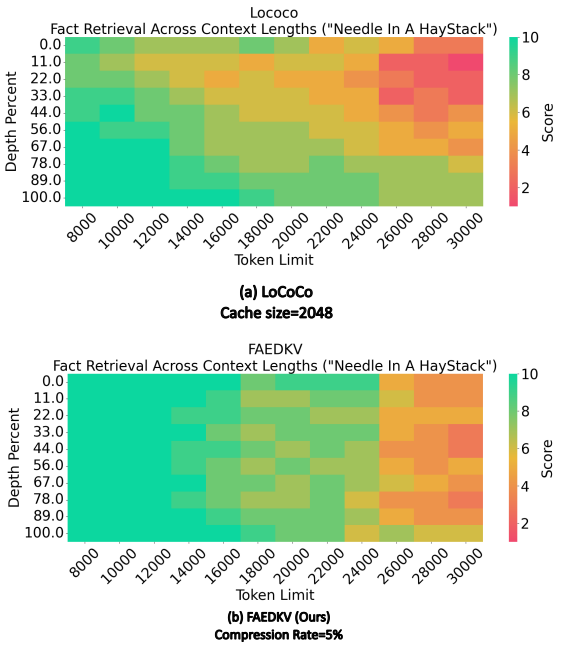

Figure 4: Results of Fact Retrieval Across Context Lengths ("Needle In A Haystack"). The x-axis denotes the length of the document (the "haystack") from 8K to 300K tokens; the axis indicates the position that the "needle" (a short sentence) is located within the document.

5.4 Pre-filling and Decoding Latency

To evaluate FAEDKV's computational efficiency, we benchmarked its inference latency against LoCoCo and a full KV cache baseline using a Llama-

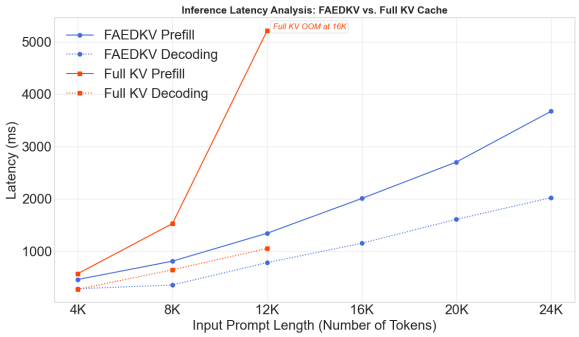

Figure 5: Results of Pre-filling and Decoding Latency

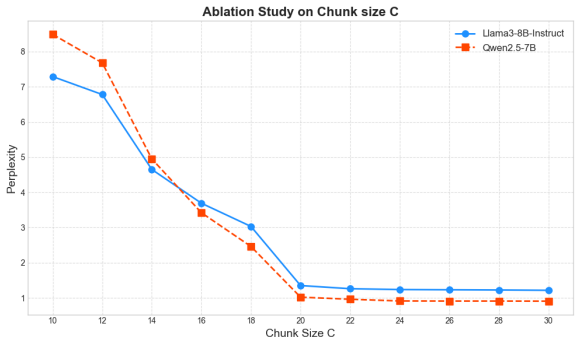

Figure 6: Results of Ablation Study on Chunk Size.

3 8B model (Hugging Face Transformers) on an NVIDIA A6000 GPU. We measured prefill latency and decoding latency for generating 10 tokens. We sample input prompts from the PG19(Rae et al., 2019) dataset, pruned to lengths of 4K, 8K, 12K, and 16K tokens. All tests used a batch size of 1 and compression ratios of 0.1. As shown in Figure 5, While the baseline exceeds GPU VRAM limits for sequences longer than 12k tokens, FAEDKV maintains efficient generation for sequences up to 24k tokens. Our result shows consistant improvement over the baseline, demonstrating the effectiveness of our optimizations in the workflow.

5.5 Ablation Study Of Chunk Size

We evaluated the impact of different chunk sizes (C)—a key hyperparameter for our frequency ablation study and subsequent workflow—on model performance. The ablation study was conducted with varying C values, measuring perplexity increase on the PG-19 dataset for both Llama3-8B and Qwen2.5-7B models. As illustrated in Figure 6, perplexity drops sharply around $C = 12$ and stabilizes near its minimum at $C = 22$. This finding supports $C = 22$ as an optimal choice, balancing model performance retention with the computational cost of the ablation study.

6 Conclusion

In this paper, we introduced FAEDKV, a novel training-free KV cache compression algorithm designed to mitigate memory overhead in LLMs while promoting unbiased information retention. Our approach uniquely leverages frequency-domain transformations, guided by an empirical frequency ablation study to identify critical spectral components for preservation. The core of our method, the IWDFT, enables efficient and normalized updates to the compressed cache during autoregressive decoding, ensuring a more consistent treatment of token contributions over time. Experimental results on multiple benchmarks demonstrate FAEDKV's ability to achieve significant cache compression with competitive, and often superior, performance compared to existing methods, particularly in preserving information uniformly across long contexts.

7 Limitations

Our study has several limitations. Firstly, experiments were conducted on models deployable on a single A6000 GPU, simulating resource-constrained scenarios. While this provides practical insights, the behavior of significantly larger models in the frequency domain and the scalability of our approach warrant further investigation. Secondly, although FAEDKV improves efficiency, opportunities may exist for even more performant frequency-based inference by more deeply leveraging principles like signal locality or semantic clustering within the frequency domain. Finally, FAEDKV focuses on efficient KV cache management within a model's existing maximum context length and does not inherently extend this architectural limit,

8 Ethical Considerations

Large Language Models (LLMs), the systems our work aims to optimize, have well-documented broader ethical considerations. Our method, FAEDKV, is a technical contribution focused on improving computational efficiency via KV cache compression. As such, FAEDKV itself does not introduce new ethical dimensions beyond those inherent to the LLMs it is applied to, nor does it directly address these existing societal concerns.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore. Association for Computational Linguistics.

Yushi Bai, Xin Lv, Jiajie Zhang, Yuze He, Ji Qi, Lei Hou, Jie Tang, Yuxiao Dong, and Juanzi Li. 2024a. LongAlign: A recipe for long context alignment of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1376–1395, Miami, Florida, USA. Association for Computational Linguistics.

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. 2024b. Longbench: A bilingual, multitask benchmark for long context understanding. *Preprint*, arXiv:2308.14508.

Ruisi Cai, Yuandong Tian, Zhangyang Wang, and Beidi Chen. 2024a. Lococo: Dropping in convolutions for long context compression. *Preprint*, arXiv:2406.05317.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Wen Xiao. 2024b. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *Preprint*, arXiv:2406.02069.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed S. Abdelfattah, and Kai-Chiang Wu. 2024. Palu: Compressing kv-cache with low-rank projection. *Preprint*, arXiv:2407.21118.

DeepSeek AI. 2024. Deepseek-v2: A strong, economical, and open-source mixture-of-experts language model. Technical Report.

Yao Fu, Runchao Li, Xianxuan Long, Haotian Yu, Xiaotian Han, Yu Yin, and Pan Li. 2025a. Pruning weights but not truth: Safeguarding truthfulness while pruning llms. *arXiv preprint arXiv:2509.00096*.

Yao Fu, Xianxuan Long, Runchao Li, Haotian Yu, Mu Sheng, Xiaotian Han, Yu Yin, and Pan Li. 2025b. Quantized but deceptive? a multi-dimensional truthfulness evaluation of quantized llms. *arXiv preprint arXiv:2508.19432*.

RangRang Ge, ShiYe Song, Zhaorui Liu, Wei Liu, Yuesheng Wang, Dongling Wang, Bofang Zhou, Zhicheng Dou, and Ji-Rong Wen. 2023. A survey on KV cache compression for large language models. *arXiv preprint arXiv:2312.10546*.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. 2024. Model tells you what to discard: Adaptive kv cache compression for llms. *Preprint*, arXiv:2310.01801.

Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2024. Lm-infinite: Zero-shot extreme length generalization for large language models. *Preprint*, arXiv:2308.16137.

Ziwei He, Meng Yang, Minwei Feng, Jingcheng Yin, Xinbing Wang, Jingwen Leng, and Zhouhan Lin. 2023. Fourier transformer: Fast long range modeling by removing sequence redundancy with fft operator. In *Findings of the Association for Computational Linguistics: ACL 2023*, page 8954–8966. Association for Computational Linguistics.

Jushi Kai, Boyi Zeng, Yixuan Wang, Haoli Bai, Ziwei He, Bo Jiang, and Zhouhan Lin. 2025. Freqkv: Frequency domain key-value compression for efficient context window extension. *Preprint*, arXiv:2505.00570.

Byungchan Lee, Seokmin Lee, Donghyun Kim, and Beomseok Heo. 2024. Scaling FFNs for better transformer. *arXiv preprint arXiv:2403.15916*.

Heejun Lee, Geon Park, Jaduk Suh, and Sung Ju Hwang. 2025. Infinitehip: Extending language model context up to 3 million tokens on a single gpu. *Preprint*, arXiv:2502.08910.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2021. FNet: Mixing tokens with fourier transforms. *arXiv preprint arXiv:2105.03824*.

Mo Li, Songyang Zhang, Taolin Zhang, Haodong Duan, Yunxin Liu, and Kai Chen. 2024a. Needlebench: Can llms do retrieval and reasoning in information-dense context? *Preprint*, arXiv:2407.11963.

Shaoyi Li, Zhaowen Ni, Tiecheng Li, Hong Ma, and Zheng Wang. 2024b. AFFormer: Resolution-agnostic and frequency-adaptive recurrent transformer for image super-resolution. *arXiv preprint arXiv:2403.05088*.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024c. Snapkv: Llm knows what you are looking for before generation. *Preprint*, arXiv:2404.14469.

Liyue Liu, Shijie Li, Zhuo Chen, Tianyi Li, and Yu Wang. 2023a. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023b. Lost in the middle: How

language models use long contexts. *Preprint*, arXiv:2307.03172.

Zichang Liu, Jue Wang, Tri Zhao, Zirui Li, Yixin Bai, and Jeff Yu. 2024a. Deja vu: Contextual sparsity for efficient LLM inference. *arXiv preprint arXiv:2401.09486*.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen (Henry) Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. 2024b. Kivi: a tuning-free asymmetric 2bit quantization for kv cache. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Xianxuan Long, Yao Fu, Runchao Li, Mu Sheng, Haotian Yu, Xiaotian Han, and Pan Li. 2025. When truthful representations flip under deceptive instructions? *arXiv preprint arXiv:2507.22149*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

Sharan Murty, John D. Morris, Neel Nanda, Michael J. Li, Jacob Andreas, Michael I. Hudson, and Divya Misra. 2024. Exactly solving acrostic puzzles with a language model. *arXiv preprint arXiv:2403.04534*.

Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Jacob Steinhardt. 2023. Emergent linear representations in world models of self-supervised transformers. In *International Conference on Learning Representations*.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint*.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Sijun Tan, Xiuyu Li, Shishir Patil, Ziyang Wu, Tianjun Zhang, Kurt Keutzer, Joseph E. Gonzalez, and Raluca Ada Popa. 2024. Lloco: Learning long contexts offline. *Preprint*, arXiv:2404.07979.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2024. Long context compression with activation beacon. *Preprint*, arXiv:2401.03462.

Zhenyu Zhang, Ying Sheng, Tianyi He, Chen Tan, Yuesong Zhou, Lian Meng, Kai Yu, Aston Zhao, Haotong Chen, Jiaming Su, and 1 others. 2023a. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *arXiv preprint arXiv:2306.14048*.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. 2023b. H_2o: Heavy-hitter oracle for efficient generative inference of large language models. *Preprint*, arXiv:2306.14048.

Kaixuan Zhou, Jianing Wang, Zhiyuan Yin, Yan Zhou, Xin Cao, Yang Gao, and Sheng Zhao. 2024. SFM-LLM: A efficient long-term time series forecasting framework based on spectrum frequency mix. *arXiv preprint arXiv:2403.15912*.