BanditKV: Dynamic Inter-layer Compression for Memory-Efficient KV Cache in LLM Inference

Anonymous ACL submission

Abstract

The inference of large language models typically relies on autoregressive generation. Caching intermediate key-value (KV) pairs can eliminate redundant computations, yet the substantial memory overhead introduced by multi-layer KV cache imposes a new bottleneck. Existing compression approaches operate either within or across layers, respectively suffering from limited optimization flexibility and static configuration strategies. In this paper, we adopt Rényi entropy to characterize the information distribution in cached KV pairs, revealing significant and irregular fluctuations that vary across both inputs and layers. Motivated by this observation, we propose BanditKV, a dynamic inter-layer compression framework built on a two-phase optimization mechanism. A contextual bandit-based policy firstly adaptively selects the optimal layer-grouping configuration for each input. Secondly, Rényi entropy guides a non-uniform, layer-specific memory allocation scheme within each group. Besides, we introduce a lightweight randomized SVD that enables compressing factor matrices derived from KV tensors, rather than the original tensors, to further improve the compression ratio. Extensive experiments show that BanditKV achieves an overall success with up to $16\times$ compression ratio, $2.2\times$ speedup factor, and nearly zero-loss in inference quality.

1 Introduction

Large Language Models (LLMs), such as GPT-4 and the Llama family of models (Touvron et al., 2023a,b; Grattafiori et al., 2024), are now widely deployed to provide diverse inference services. To reduce inference latency, these models typically cache intermediate key-value (KV) pairs across network layers, thereby eliminating redundant computations during autoregressive output token generation. However, this caching mechanism introduces substantial memory overhead (Pope et al., 2023). For instance, running a 175-billion-parameter model with a batch size of 64 and a sequence length of 4K per input requires approximately 1,208 GB of memory for the KV cache alone—about 3.45 times the memory occupied by the model weights (Ouyang et al., 2022). This considerable memory demand highlights the imperative need for compression techniques that can reduce the KV cache footprint without compromising generation quality.

Most existing approaches focus on compressing data within individual layers, primarily through methods such as quantization (Sheng et al., 2023) and sparsity-based eviction (Yang et al., 2024). These techniques inevitably incur quality degradation due to information loss within the constrained *intra-layer* optimization space. Recently, a new direction has emerged that merges *inter-layer* redundant data (Liu et al., 2024) for a subset of layers, expanding the optimization space and thus improving output quality even under high compression ratios. However, we argue that its performance remains far from ideal, as the static layer-grouping framework and simplistic merging-and-filtering policy fail to adapt to the heterogeneous and dynamically varying contexts encountered during inference.

Following the *inter-layer* compression line, we demonstrate that, in contrast to the traditionally used Shannon entropy, Rényi entropy provides a more suitable measure for characterizing the distribution of information contained in cached KV pairs. Our analysis reveals significantly irregular fluctuations in information density across varying input datasets/tasks and network layers. These observations motivate the design of our dynamic compression framework BanditKV, that is capable of adaptively responding to both input-specific and layer-wise characteristics.

BanditKV adopts a two-phase optimization mechanism to maintain inference quality while enhancing the compression ratio. For a given in-

ference input, the first phase consists of an Input-Adaptive Grouping module, which dynamically determines an appropriate group size to partition all network layers (rather than a partial subset) into cohesive units. The *inter-layer* redundancy reduction is then performed within each group. We formulate this group-size decision as a contextual bandit problem that learns from input features to yield an adaptive grouping configuration.

The second phase executes an Entropy-Aware Memory Allocator. Based on the available memory capacity, BanditKV determines a memory budget for each group. By Rényi entropy, we know even within a group, information density varies across layers. Accordingly, we implement a non-uniform memory allocation policy where layers with lower entropy are aggressively compressed, and the saved budget is reallocated to other layers to preserve critical information.

Specifically, we apply a randomized SVD algorithm to each KV tensor. The resulting factor matrices (U, S) exhibit asymmetric similarity across layers and are therefore stored only once per group, that substantially improves the compression ratio. In contrast, the coefficient matrix X remains layer-specific and is not merged. Notably, the randomized SVD and reconstruction are runtime efficient, as only a subset of elements (determined by the allocated memory budget) are involved in computations, rather than the entire tensor.

Our contributions are summarized as follows:

- Utilizing Rényi entropy to characterize the information distribution of cached KV pairs and revealing significant, input- and layer-specific irregularity in its fluctuation.
- Proposing BanditKV, a dynamic inter-layer compression framework that adaptively performs input-specific layer grouping and entropy-aware memory allocation within groups. By merging SVD-decomposed factor matrices instead of the original tensors, it further enhances the compression ratio. Together, these optimizations ensure robust inference quality while achieving significant memory reduction and improved runtime efficiency.
- Extensive experiments on 14 *LongBench* datasets show that BanditKV achieves a compression ratio of up to 16× and a speedup factor of up to 2.2×, while maintaining nearly full inference quality.

2 Related Work

Efficient Inference and KV Cache Management. Optimization for LLM inference demands end-to-end coordination, ranging from system-level kernels to algorithmic adjustments. At the system level, frameworks like FlashInfer (Ye et al., 2025b) and Apt-Serve (Gao et al., 2025) optimize attention kernels and request scheduling to minimize latency, while algorithmic innovations like Speculative Beam decoding (Qin et al., 2025b) and prompt compression (Liskavets et al., 2025) reduce overhead by adjusting generation width or input length. Specific to KV cache, intra-layer approaches primarily focus on sparse eviction and quantization. For instance, Attention Sinks (Xiao et al., 2023) ensure stability in streaming inference via token preservation, while Cache-Craft (Agarwal et al., 2025) and CAKE (Qin et al., 2025a) employ hierarchical tiering and adaptive eviction policies. However relying on heuristic-based eviction or aggressive quantization, they inevitably discard critical long-tail information or introduce quantization noise, leading to irreversible precision degradation, especially in long-context retrieval tasks. Recognizing the limitations of intra-layer methods, recent research has pivoted toward exploiting redundancy across transformer blocks. To further reduce memory usage, cross-layer compression techniques like MiniCache (Liu et al., 2024) attempt to merge KV states from adjacent layers. Nevertheless, these inter-layer approaches generally adopt static merging strategies or uniform compression configurations, failing to adapt to the dynamic information density variance across inputs and layers.

Compression and Spectral Analysis. Recent research integrates adaptive fusion paradigms (Gu et al., 2025; Zhang et al., 2025; Lu et al., 2024; Ye et al., 2025a; Huang et al., 2024; Zhu et al., 2025; Du et al., 2024; Dziadzio et al., 2025; Qu et al., 2025; Li et al., 2025; Zhao et al., 2025) with spectral analysis via low-rank approximations (Hu et al., 2021; Wang et al., 2024, 2025b; Boullé and Townsend, 2022; Huang et al., 2025) (Aharon et al., 2006; Wang et al., 2025a; Chen et al., 2021). While primarily targeting parameter efficiency, these methods validate singular value truncation as a theoretical basis for compressing activation states.

Existing paradigms, constrained by static heuristics and isolated processing, ignore the dynamic heterogeneity of attention patterns.

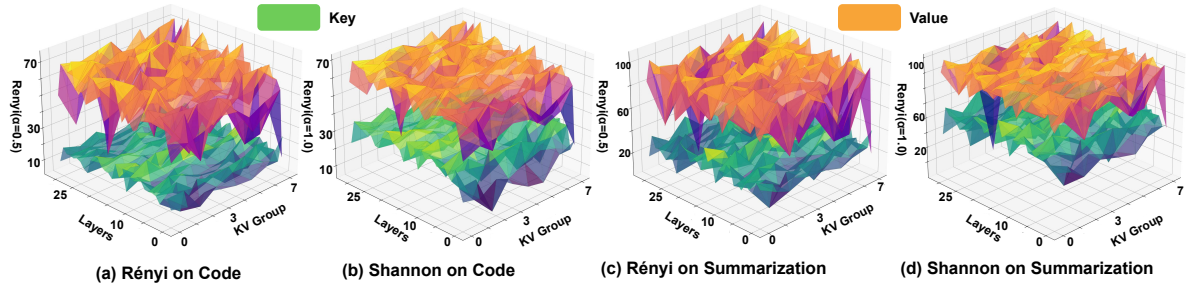

Figure 1: Rényi Entropy analysis of different datasets. The Shannon Entropy metric is shown in (b) and (d), while the Rényi Entropy metric is shown in (a) and (c).

3 Motivation

While low-rank compression (Wang et al., 2024) offers a promising pathway for reducing KV cache overhead, current cross-layer approaches(Liu et al., 2024) are limited by static assumptions. These methods enforce fixed grouping strategies and uniform rank configurations, ignoring the inherent heterogeneity of information across layers and tasks. In this section, we dismantle these limitations through empirical analysis, identifying three critical phenomena that drive the design of BanditKV.

3.1 Task-Dependent Information Dynamics

Current methods typically enforce a fixed grouping size, implicitly assuming that information flow remains invariant across tasks. However, our empirical findings contradict this assumption.

Before analyzing task-specific dynamics, we first identify the optimal metric for quantifying the information density of KV tensors. We compare the standard Shannon entropy ($\alpha = 1.0$) against Rényi entropy ($\alpha = 0.5$) based on the singular value distribution. As illustrated in Figure X, Shannon entropy tends to exhibit a flat, uniform distribution across layers, failing to capture the subtle structural sparsity of attention heads. In sharp contrast, Rényi entropy reveals distinct, high-variance fluctuations. By assigning greater weight to the dominant singular values, it serves as a more sensitive proxy for effective rank, accurately distinguishing information-rich layers from redundant ones.

Task-Specific Shifts. As shown in Figure 1, utilizing Rényi entropy, we analyzed the data flow of Llama3-8B-Instruct across distinct tasks. The results demonstrate that the distribution of entropy peaks shifts significantly depending on the task type and input length. A grouping strategy that is optimal for summarization may prove suboptimal for coding, leading to a trade-off where static strategies are either too aggressive or too conservative.

Data-Driven Dynamic Grouping. Given that the optimal grouping size is highly dynamic, we formulate the grouping decision as a Contextual Bandit problem. This enables the system to adaptively select the optimal configuration based on input features, efficiently leveraging cross-layer redundancy in a personalized manner.

3.2 Asymmetric Spectral Redundancy

To investigate the structural compressibility of KV tensors, we analyze the spectral properties of Key and Value matrices via Singular Value Decomposition (SVD). Taking the Value tensor as a representative example, we decompose the tensors from individual layers to extract their spectral components: the orthonormal token basis U, the singular value spectrum S, and the coefficient matrix X.

Our analysis reveals a striking dichotomy in the spectral redundancy of Value tensors across layers. As visualized in Figure 2 (with consistent results for Key tensors in Appendix Figure 6), the U and S matrices exhibit strong inter-layer correlations, indicated by high similarity scores. This suggests that the principal semantic subspaces are largely shared among adjacent layers. In sharp contrast, the coefficient matrices X display significantly lower similarity, appearing almost orthogonal between layers. This phenomenon implies a clear functional separation: the projection basis (US) encodes a global semantic alignment shared across the group, while the reconstruction coefficients (X) capture local, layer-specific nuances essential for distinct attention patterns.

This observation motivates our **structural decoupling** design. By storing a single shared copy of

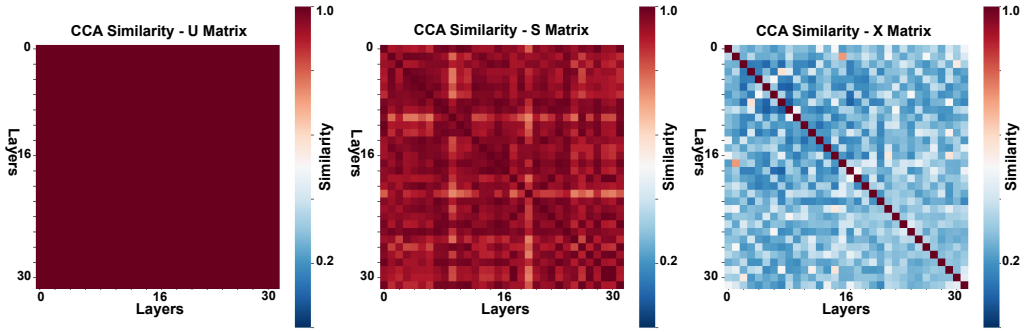

Figure 2: V Tensor Similarity Analysis Across Layers

the US basis per group while maintaining independent X matrices, we can achieve aggressive compression without sacrificing the representational diversity of individual layers.

3.3 Irregularity of Information Density

A critical limitation of existing methods is their reliance on static rank allocation, which implicitly assumes a uniform distribution of information across the network. However, our Rényi entropy analysis reveals a distinct *mountain-shaped* distribution, where intermediate layers exhibit significantly higher information density compared to the input and output layers (see Figure 1). Consequently, a uniform rank strategy incurs a double penalty: it squanders memory resources on low-entropy edge layers while simultaneously starving high-entropy intermediate layers, leading to irreversible semantic loss. Furthermore, we observe that Value tensors consistently exhibit higher entropy than Key tensors, necessitating a differential allocation of the rank budget. These findings motivate our density-driven strategy, which utilizes Rényi entropy as a dynamic proxy to funnel rank budgets toward information-dense regions, maximizing feature retention within a constrained global limit.

4 BanditKV

Building on the insights from Section 3, we introduce BanditKV, a dynamic compression framework designed to optimize the trade-off between memory footprint and generation quality. In this section, we detail its two core components the Contextual Bandit Decision Engine and the Dynamic Rank Allocation Engine.

4.1 Model Framework

As illustrated in Figure 3, BanditKV operates as a unified two-phase pipeline that harmonizes decision-making with execution.

Phase I: Dynamic Group Size Selection. The workflow initiates with the *Contextual Bandit Decision Engine*, acting as the system controller. Notably, this engine operates in a cold-start mode, eliminating the need for offline pre-training by learning optimal policies directly from online interactions. Conditioned on input Tasks and Environment states, the agent predicts an optimal grouping size G (e.g., $G = 8$). This step adaptively partitions the N transformer layers into cohesive groups to balance merging granularity with semantic fidelity.

Phase II: Entropy-Aware Memory Allocation. Within each group, the system executes a dual-stage compression strategy. First, in the Group-Level Coarse Truncation (First Cut-off), concatenated KV tensors are decomposed via *Randomized SVD*. Based on the group's overall entropy, the shared bases (U, S) and all layer-specific coefficients (X) are initially truncated to a unified group rank budget. Subsequently, the Layer-Wise Fine-Grained Allocation (Second Cut-off) redistributes this budget based on internal layer importance. Specifically, X matrices are physically truncated to layer-specific ranks to minimize storage, while U and S retain the group rank, undergoing implicit secondary truncation only during the runtime reconstruction phase.

4.2 Contextual Bandit Decision Engine

The core innovation of BanditKV is treating the layer grouping strategy not as a fixed hyperparameter, but as a sequential decision-making problem. We employ the LinUCB algorithm (Li et al., 2010) to learn a personalized grouping policy.

Problem Formulation. We formulate the layer

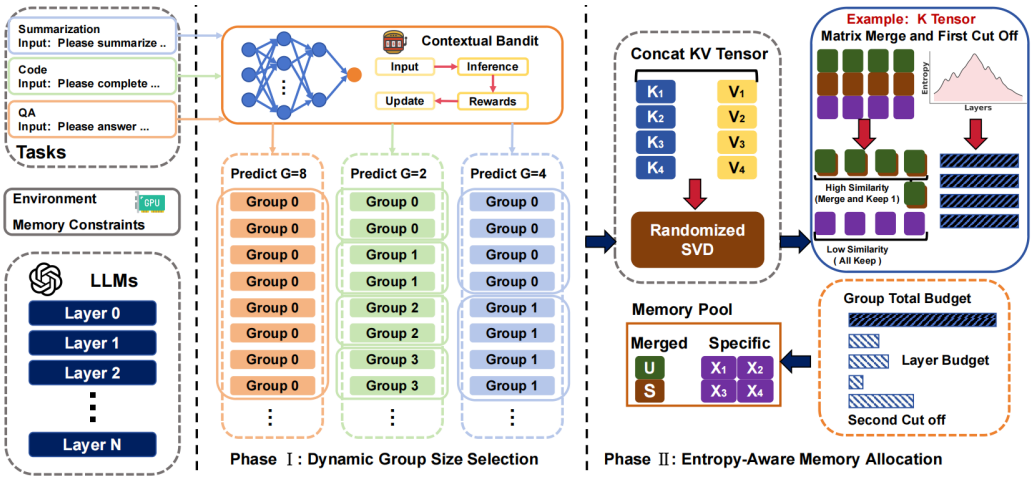

Figure 3: Framework of BanditKV

grouping decision as a contextual bandit problem characterized by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R} \rangle$. The State Space ($\mathcal{S}$) is represented by a context vector x_t capturing input-specific features, including sequence length, task semantics, and model configuration. The Action Space ($\mathcal{A}$) comprises a discrete set of candidate grouping sizes (e.g., $\mathcal{A} = \{1, 2, 4, 8\}$), where selecting an action a entails merging every a adjacent layers into a cohesive unit. Finally, the Reward ($\mathcal{R}$) is defined as a composite metric designed to optimize the multi-objective trade-off among generation accuracy, compression ratio, and inference latency.

Decision Mechanism. Unlike standard UCB which ignores context, LinUCB assumes the expected reward of an action a is linear in its features: $\mathbb{E}[r_{t,a}|x_t] = x_t^\top \theta_a$. At each step t, the agent selects the action that maximizes the Upper Confidence Bound to balance exploration and exploitation. The standard update rules and theoretical guarantees are detailed in Appendix A.1.

Composite Reward Function. To align the agent's objective with system efficiency, we design a scale-invariant reward function. We define the relative changes in accuracy (ΔAcc), compression (ΔCR), and latency (ΔLat) as Equation 1

$$
\begin{aligned}
r_{\text{acc}} &= \frac{\text{Acc}_{\text{new}} - \text{Acc}_{\text{base}}}{\text{Acc}_{\text{base}}}, \\
r_{\text{comp}} &= 1 - \frac{1}{\text{CR}_{\text{new}}}, \\
r_{\text{lat}} &= \frac{\text{Lat}_{\text{base}} - \text{Lat}_{\text{new}}}{\text{Lat}_{\text{base}}}
\end{aligned}
\tag{1}
$$

The final reward R_t is a weighted sum of these components as Equation 2

$$
R_t = w_{\text{acc}} \cdot r_{\text{acc}} + w_{\text{comp}} \cdot r_{\text{comp}} + w_{\text{lat}} \cdot r_{\text{lat}} \tag{2}
$$

where weights $w_{\text{acc}}, w_{\text{comp}}, w_{\text{lat}}$ control the preference for performance versus efficiency (see Appendix A.3 for settings). This design enables the agent to autonomously learn policies such as prioritize accuracy for coding tasks or maximize compression for long-context retrieval, effectively solving the local optimality issue of static strategies.

4.3 Dynamic Rank Allocation Engine

The Dynamic Rank Allocation Engine is responsible for distributing the global compression budget across layers based on their information density. It consists of three stages budget initialization, structural compression via Randomized SVD, and entropy-guided allocation.

4.3.1 Global Rank Budget Initialization

Prior to decomposition, we determine the total rank budget B_{total} to satisfy the target compression ratio (CR). Let the original KV cache size be $M_{\text{orig}} \propto L_{\text{seq}} \cdot N_{\text{layer}} \cdot D_{\text{model}}$. The compressed format consists of shared bases (US) and layer-specific coefficients (X) for each group.

To ensure the final memory footprint meets the target CR (i.e., $M_{\text{orig}}/M_{\text{comp}} \approx$ CR), we reverse-engineer the allowable average rank r_{avg} per group. Considering the storage cost of shared bases ($L_{\text{seq}} \times r$) and layer-specific coefficients ($N_{\text{layer}} \times D_{\text{model}} \times r$), the average rank is calculated as Eq 3

$$
r_{\text{avg}} = \frac{N_{\text{layer}} \cdot L_{\text{seq}} \cdot D_{\text{model}}}{\text{CR} \cdot (N_g \cdot L_{\text{seq}} + N_{\text{layer}} \cdot D_{\text{model}})} \tag{3}
$$

The total rank budget is then defined as Eq 4:

$$
B_{\text{total}} = N_g \times r_{\text{avg}} \tag{4}
$$

where N_g is the number of groups determined by the Bandit engine. This B_{total} acts as the global resource pool for the subsequent dynamic allocation.

4.3.2 Structural Compression via Randomized SVD

Standard SVD is computationally prohibitive for high-dimensional KV tensors ($O(\min(m,n)^3)$). To address this, we adopt Randomized SVD (Boullé and Townsend, 2022), which reduces the complexity to $O(L_{seq} \cdot (|G|d) \cdot k)$, offering a theoretical 3–10× speedup, see complexity analysis in Appendix A.2 and we provide a schematic illustration of this process in Appendix Figure 5. Our framework introduces a Split-Storage Mechanism to maximize cross-layer redundancy as follow. Based on the bandit-selected size G, we concatenate KV tensors from adjacent layers into a joint matrix $\mathbf{Z}_G \in \mathbb{R}^{L_{seq} \times (|G| \cdot d)}$.

We decompose $\mathbf{Z}_G$ into a low-rank approximation $\mathbf{Z}_G \approx U_r S_r X_r^\top$. Crucially, leveraging the asymmetric redundancy identified in Section 3.2, we decouple the representation into two distinct components. The product of the left singular vectors and singular values constitutes the *Shared Basis* ($\mathbf{M}_{\text{shared}} = U_r S_r$), which encapsulates the common semantic subspace spanned by all layers in the group; consequently, we persist only a single copy of $\mathbf{M}_{\text{shared}}$ per group to minimize redundancy. Conversely, the right singular vectors form the *Layer-Specific Coefficients* ($X_r^\top$), capturing unique local variations. We partition $X_r^\top$ along the feature dimension into $|G|$ distinct blocks, formulated as $X_r^\top = [X_1, X_2, \ldots, X_{|G|}]$, where each sub-matrix X_i is exclusively mapped to an individual layer.

During decoding, the system reconstructs the specific KV tensor for layer l_i via a single matrix multiplication as Equation 5

$$\hat{Z}_{l_i} = \mathbf{M}_{\text{shared}} \times X_{l_i} \tag{5}$$

This decoupling reduces memory usage from storing N independent tensors to storing 1 shared basis plus N smaller coefficient matrices, significantly lowering the memory footprint.

4.3.3 Entropy-Based Information Quantification

To guide the allocation of the limited rank budget, we require a robust metric to quantify the *information density* of each compressed group. While standard methods often rely on the variance (squared singular values), we argue that for LLM attention, the information is highly concentrated in the top singular vectors.

To accentuate the dominant semantic components while suppressing the noise from the long tail of the spectrum, we propose a Cubic-Weighted Rényi Entropy. For a group g with singular values $\{\sigma_i^{(g)}\}$, we first construct a normalized probability distribution $p^{(g)}$ utilizing cubic weighting as Equation 6

$$p_i^{(g)} = \frac{(\sigma_i^{(g)})^3}{\sum_j (\sigma_j^{(g)})^3} \tag{6}$$

The information complexity score H_g is then computed using Rényi entropy ($\alpha = 0.5$) as Eq 7

$$H_g = H_\alpha(p^{(g)}) = -\log\left(\sum_i (p_i^{(g)})^\alpha\right) \tag{7}$$

This metric (H_g) effectively serves as a proxy for the representational richness of the group, enabling the system to distinguish between information-dense intermediate layers and sparse edge layers.

4.3.4 Complexity-Aware Rank Distribution

Based on the quantified entropy H_g, the Dynamic Rank Allocation Engine distributes the total budget B_{total} (derived in Section 4.3.1) proportionally.

We adopt a linear scaling strategy where the rank assigned to group g is directly proportional to its relative information complexity. The allocation weight w_g and the final rank k_g are calculated as Equation 8

$$w_g = \frac{H_g}{\sum_{j=1}^{N_g} H_j}, \quad k_g = \lfloor B_{\text{total}} \cdot w_g \rfloor \tag{8}$$

Reflecting our observation in Section 3.3 that Value tensors carry significantly higher entropy than Key tensors, we perform this allocation process independently for Key and Value matrices.

To prevent information collapse in extremely low-entropy layer, we enforce a Minimum Rank Threshold ($k_{\min}$). The final rank is clipped: $k_g^* = \max(k_g, k_{\min})$. This ensures theoretical robustness while maximizing memory efficiency for high-entropy layers.

5 Experiments

We evaluate BanditKV on the 14 Datasets of *LongBench* benchmark (Bai et al., 2024) using Llama3-8B-Instruct and Mistral-7B. All experiments were conducted on a single NVIDIA A6000 GPU. Setup details are provided in Appendix 3.

Table 1: Main Results: Performance Retention on LongBench. We report the relative performance retention (%) normalized against the FP16 baseline (set as 100%) to unify evaluation metrics across diverse tasks.

Methods(Compression Ratio)	Single-Doc QA			Multi-Doc QA			Summarization			Few-Shot			Code		Avg
	NaQA	QAs	MFQA	HPQA	WKQA	Mus	GovR	Qmsum	Mnews	Trec	TrivQA	Ssum	Lcc	RPB	
Llama3-8B-instruct															
Baseline(1.00x)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
StreamingLLM(8.00x)	68.17	63.31	64.08	80.30	80.74	79.01	88.51	91.96	86.40	93.09	93.78	92.80	92.31	97.79	83.73
H2O(7.98x)	92.47	85.88	86.93	93.50	94.01	91.99	90.21	93.72	88.06	86.92	87.57	86.65	94.22	**99.81**	90.85
PyramidKV(7.96x)	67.74	62.91	63.68	77.73	78.15	76.48	66.40	68.99	64.82	74.62	75.18	74.40	62.29	65.99	72.81
CAKE(8.01x)	96.52	89.64	90.73	97.82	98.35	96.25	93.15	96.77	90.92	98.05	98.79	97.75	**94.29**	98.72	96.68
BanditKV(8.01x)	**99.68**	**92.57**	**93.70**	**101.27**	**101.82**	**99.64**	**96.67**	**100.43**	**94.36**	**98.78**	**99.52**	**98.48**	83.71	93.63	**98.20**
Mistral-7B															
Baseline(1.00x)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
StreamingLLM(8.00x)	67.07	65.56	64.77	81.58	78.75	81.09	88.60	90.89	89.60	96.15	95.67	89.01	98.11	96.97	84.55
H2O(7.99x)	89.96	86.61	85.56	94.36	91.09	93.80	90.20	92.52	91.21	90.33	89.88	83.63	99.07	98.15	91.16
PyramidKV(7.93x)	67.16	64.65	63.87	77.62	74.94	77.16	65.98	67.68	66.72	75.10	74.73	69.53	67.06	66.28	69.89
CAKE(8.00x)	95.04	91.49	90.38	96.60	93.26	96.03	92.00	94.38	93.04	**98.99**	98.50	91.65	**99.47**	**98.89**	94.98
BanditKV(8.01x)	**101.20**	**97.44**	**96.25**	**96.79**	**93.45**	**96.22**	**97.39**	**99.92**	**98.47**	93.36	**100.84**	**100.35**	96.68	95.58	**97.42**

Table 2: Ablation Results comparing BanditKV components against baselines.

Methods(Compression Ratio)	Single-Doc QA			Multi-Doc QA			Summarization			Few-Shot			Code		Avg
	NaQA	QAs	MFQA	HPQA	WKQA	Mus	GovR	Qmsum	Mnews	Trec	TrivQA	Ssum	Lcc	RPB	
Llama3-8B-Instruct															
Baseline(1.00x)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Minicache(5.02x)	92.12*	88.69*	96.34*	91.96*	98.26*	91.36*	93.46*	96.21*	89.19*	98.93*	100.00*	89.99*	**99.77***	99.65*	94.71*
StaticRank-Singlelayer(6.00x)	97.44	89.68	93.81	90.38	97.15	93.59	88.86	95.26	95.24	98.50	87.61	85.12	76.63	86.84	91.15
StaticRank-Doublelayer(5.98x)	68.77	89.65	84.22	87.47	91.48	79.80	74.40	90.43	97.21	98.12	86.83	84.03	73.81	78.91	84.65
DynamicRank-Singlelayer(6.00x)	97.62	96.38	95.96	93.10	97.21	99.15	81.13	96.41	99.10	99.50	96.31	88.94	91.66	98.05	95.03
DynamicRank-Doublelayer(5.99x)	98.31	96.36	97.96	98.20	95.58	92.36	97.51	97.53	98.73	99.34	95.21	90.97	**93.10**	95.62	96.19
BanditKV (6.01x)	**99.09**	**98.40**	**99.30**	**101.10**	**101.92**	**100.02**	**98.07**	**100.29**	**97.03**	**100.38**	**100.88**	**99.76**	87.63	**99.73**	**98.82**
Mistral-7B															
Baseline(1.00x)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Minicache(5.02x)	✗	✗	81.78	✗	96.59	✗	89.89	88.59	✗	98.50	✗	✗	**98.70**	**100.08**	93.44
StaticRank-Singlelayer(6.00x)	80.31	88.47	82.33	51.33	54.80	47.10	88.76	90.09	97.05	89.34	95.19	95.36	95.34	88.19	81.69
StaticRank-Doublelayer(5.99x)	90.44	90.57	90.68	55.81	71.96	48.04	93.66	91.98	98.08	90.35	96.43	98.07	95.98	89.91	86.28
DynamicRank-Singlelayer(6.00x)	100.97	97.34	99.10	95.38	96.38	95.72	95.85	99.46	98.05	93.95	99.35	99.77	96.20	95.18	97.33
DynamicRank-Doublelayer(5.99x)	100.71	97.81	82.33	96.44	96.13	**96.42**	97.47	99.27	98.53	94.46	98.55	100.02	96.57	94.69	96.38
BanditKV (Ours)	**101.94**	**98.10**	**100.09**	**96.47**	**96.66**	**96.42**	**98.21**	**99.48**	**98.65**	**94.50**	**101.34**	**100.16**	96.83	96.29	**98.22**

5.1 Experimental Setup and Baselines

We compare BanditKV against two categories of robust baselines: (1) Intra-layer Methods, including H2O (Zhang et al., 2023), StreamingLLM (Xiao et al., 2023), PyramidKV (Yang et al., 2024), and the state-of-the-art quantization-based CAKE (Qin et al., 2025a); and (2) Inter-layer Methods, specifically MiniCache (Liu et al., 2024).[1]

5.2 Overall Performance Analysis

Table 1 presents a comprehensive comparison of BanditKV against state-of-the-art baselines. As observed, BanditKV consistently achieves superior inference quality across diverse tasks. Given the diverse evaluation metrics employed by *LongBench* (e.g., ROUGE-L, F1, Accuracy), we report the Performance Retention Rate to facilitate a uniform comparison. Specifically, the performance of the uncompressed baseline is normalized to 100%, and all compression results are expressed as a percentage relative to this baseline.

BanditKV significantly outperforms the leading intra-layer quantization method, CAKE, improving average generation quality by 1.52% on Llama3-8B and 2.44% on Mistral-7B. The performance advantage is particularly pronounced in memory-intensive summarization tasks; for instance, on *Qmsum*, BanditKV achieves a peak improvement of over 5.5% compared to competitive baselines. This result highlights that optimizing the KV cache structure via dynamic inter-layer grouping is inherently more effective than aggressive quantization for preserving semantic integrity. Moreover, on retrieval-intensive QA tasks, our method surpasses static eviction strategies like H2O by a margin exceeding 10%. This substantial gap empirically proves that our entropy-aware allocation effectively safeguards long-tail information, which is frequently discarded by rigid, heuristic-based policies.

BanditKV demonstrates exceptional stability under aggressive compression regimes. Even at a 16× compression ratio, it retains over 99% of the

[1]Results for MiniCache are cited directly from the original paper due to the unavailability of official code.

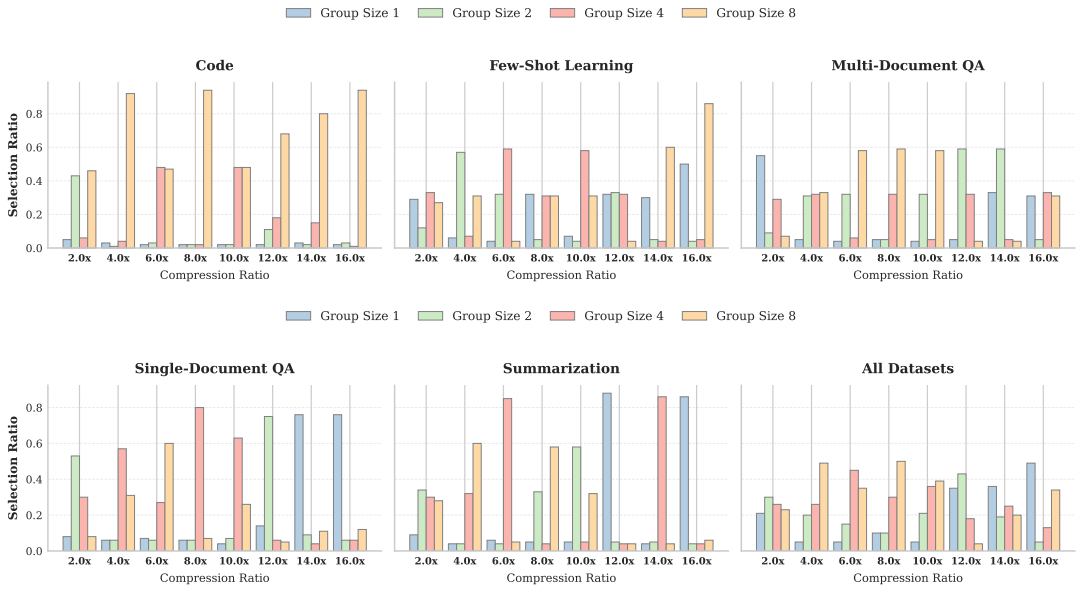

Figure 4: The distribution of Group Size (G) selections learned by BanditKV across different tasks and compression ratios.

original performance on multiple datasets (refer to Appendix Table 5). In contrast to static baselines that exhibit architectural sensitivity—performing well on one model but failing on another—our bandit-based decision engine dynamically adapts to model-specific variance. This adaptability ensures consistent high-fidelity retention across distinct architectures, such as Llama3 and Mistral, verifying the generalizability of our framework.

5.3 Ablation Studies

To isolate the contributions of individual components, we compare BanditKV against MiniCache and various static configurations in Table 2.

The results unequivocally validate the necessity of our entropy-guided design. The *Dynamic-Rank* strategy consistently outperforms the uniform *Static-Rank* allocation, yielding an average accuracy gain of approximately 4.8% over the strongest static baseline (MiniCache). As shown in the degradation of *Static-Rank/Singlelayer* (e.g., dropping to 47.1% on Mistral summarization), uniform rank distribution fails to capture the heterogeneous information density described in Sec 3.3. By funneling the rank budget to information-rich layers, BanditKV effectively prevents this information collapse. Furthermore, the superiority of BanditKV over fixed grouping baselines confirms that the bandit-based policy successfully balances the trade-off between merging granularity and feature reconstruction.

5.4 Analysis of Adaptive Grouping Policy

Figure 4 illustrates the learned policy distribution, showing how BanditKV balances granularity and efficiency. The policy adapts to intrinsic task redundancy. For Code Generation, the model favors coarse grouping to exploit syntactic periodicity. Conversely, Multi-Document QA shifts significantly toward fine granularity, prioritizing precise token-level retention to capture dispersed *needle-in-a-haystack* retrieval cues. The policy dynamically adjusts to memory budgets. As compression ratios rise to $16\times$, redundant tasks shift to extreme grouping ($G = 8$). However, information-dense tasks exhibit protective behavior by retaining small groups ($G = 1$) to safeguard essential tokens. This demonstrates BanditKV's ability to autonomously balance semantic fidelity with strict system constraints.

6 Conclusion

We introduce BanditKV, a dynamic framework leveraging Rényi entropy to address the rigidity of static compression. By harmonizing contextual bandit grouping with adaptive rank allocation, it achieves up to $16\times$ compression and $2.2\times$ speedup with nearly zero quality loss. Our work highlights the efficacy of dynamic, structure-aware optimization for efficient long-context inference.

Limitations

While BanditKV excels in memory-intensive long-context scenarios, the overhead from contextual bandits and SVD renders it less suitable for short, compute-bound sequences. Future work will focus on integration with serving systems like vLLM (Kwon et al., 2023) and orthogonal quantization to further mitigate these overheads.

We utilized AI assistance (ChatGPT) solely for grammatical error checking and writing style refinement.

Ethics Statement

This work aims to improve the efficiency of Large Language Models (LLMs) by reducing memory usage and computational overhead. By enabling more efficient inference, our method contributes to Green AI initiatives, potentially lowering the energy consumption and carbon footprint associated with deploying large-scale models.

In our experiments, we utilized publicly available benchmark datasets (LongBench) and standard open-source models (Llama 3, Mistral). We did not collect, process, or release any private, sensitive, or demographic-specific personal data. We do not foresee any direct negative social impacts or specific ethical hazards resulting from this compression technique, although we acknowledge the general ethical considerations associated with the broader deployment of LLMs.

References

Shubham Agarwal, Sai Sundaresan, Subrata Mitra, Debabrata Mahapatra, Archit Gupta, Rounak Sharma, Nirmal Joshua Kapu, Tong Yu, and Shiv Saini. 2025. Cache-craft: Managing chunk-caches for efficient retrieval-augmented generation. *Proc. ACM Manag. Data*, 3(3).

Michal Aharon, Michael Elad, and Alfred Bruckstein. 2006. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311–4322.

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. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Nicolas Boullé and Alex Townsend. 2022. A generalization of the randomized singular value decomposition. page Poster Session 4.

Patrick Chen, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. 2021. Drone: Data-aware low-rank compression for large nlp models. *Advances in neural information processing systems*, 34:29321–29334.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, and Min Zhang. 2024. Parameter competition balancing for model merging. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Sebastian Dziadzio, Vishaal Udandarao, Karsten Roth, Ameya Prabhu, Zeynep Akata, Samuel Albanie, and Matthias Bethge. 2025. How to merge your multimodal models over time? In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20479–20491.

Shihong Gao, Xin Zhang, Yanyan Shen, and Lei Chen. 2025. Apt-serve: Adaptive request scheduling on hybrid cache for scalable llm inference serving. *Proceedings of the ACM on Management of Data*, 3(3):1–28.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Wangyun Gu, Qianghua Gao, Zhang Li-Xin, Xu Shen, and Jieping Ye. 2025. NeuronMerge: Merging models via functional neuron groups. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9015–9037, Vienna, Austria. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. 2024. Emr-merging: Tuning-free high-performance model merging. In *Advances in Neural Information Processing Systems*, volume 37, pages 122741–122769. Curran Associates, Inc.

Xinhao Huang, You-Liang Huang, and Zeyi Wen. 2025. Sola: leveraging soft activation sparsity and low-rank decomposition for large language model compression. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Haoyang Li, Fangcheng Fu, Hao Ge, Sheng Lin, Xuanyu Wang, Jiawen Niu, Yujie Wang, Hailin Zhang, Xiaonan Nie, and Bin Cui. 2025. Malleus: Straggler-resilient hybrid parallel training of large-scale models via malleable data and model parallelization. *Proc. ACM Manag. Data*, 3(3).

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670.

Barys Liskavets, Maxim Ushakov, Shuvendu Roy, Mark Klibanov, Ali Etemad, and Shane K Luke. 2025. 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.

Akide Liu, Jing Liu, Zizheng Pan, Yefei He, Gholamreza Haffari, and Bohan Zhuang. 2024. Minicache: Kv cache compression in depth dimension for large language models. *Advances in Neural Information Processing Systems*, 37:139997–140031.

Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. 2024. Twin-merging: Dynamic integration of modular expertise in model merging. In *Advances in Neural Information Processing Systems*, volume 37, pages 78905–78935. Curran Associates, Inc.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624.

Ziran Qin, Yuchen Cao, Mingbao Lin, Wen Hu, Shixuan Fan, Ke Cheng, Weiyao Lin, and Jianguo Li. 2025a. Cake: Cascading and adaptive kv cache eviction with layer preferences. *arXiv preprint arXiv:2503.12491*.

Zongyue Qin, Zifan He, Neha Prakriya, Jason Cong, and Yizhou Sun. 2025b. Dynamic-width speculative beam decoding for llm inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25056–25064.

Huaizhi Qu, Xinyu Zhao, Jie Peng, Kwonjoon Lee, Behzad Dariush, and Tianlong Chen. 2025. Uq-merge: Uncertainty guided multimodal large language model merging. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1401–1417.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. 2023. Flexgen: High-throughput generative inference of large language models with a single gpu. In *International Conference on Machine Learning*, pages 31094–31116. PMLR.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Jingcun Wang, Yu-Guang Chen, Ing-Chao Lin, Bing Li, and Grace Li Zhang. 2025a. Basis sharing: Cross-layer parameter sharing for large language model compression. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Xin Wang, Samiul Alam, Zhongwei Wan, Hui Shen, and Mi Zhang. 2025b. Svd-llm v2: Optimizing singular value truncation for large language model compression. *arXiv preprint arXiv:2503.12340*.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv*.

Dongjie Yang, Xiaodong Han, Yan Gao, Yao Hu, Shilin Zhang, and Hai Zhao. 2024. PyramidInfer: Pyramid KV cache compression for high-throughput LLM inference. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3258–3270, Bangkok, Thailand. Association for Computational Linguistics.

Peng Ye, Chenyu Huang, Mingzhu Shen, Tao Chen, Yongqi Huang, and Wanli Ouyang. 2025a. Dynamic

model merging with mixture of weights. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(8):7925–7939.

Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, and Luis Ceze. 2025b. Flashinfer: Efficient and customizable attention engine for llm inference serving. *Preprint*, arXiv:2501.01005.

Mingyang Zhang, Jing Liu, Ganggui Ding, Linlin Ou, Xinyi Yu, and Bohan Zhuang. 2025. Channel merging: preserving specialization for merged experts. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

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. 2023. H2o: heavy-hitter oracle for efficient generative inference of large language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Pinxue Zhao, Hailin Zhang, Fangcheng Fu, Xiaonan Nie, Qibin Liu, Fang Yang, Yuanbo Peng, Dian Jiao, Shuaipeng Li, Jinbao Xue, Yangyu Tao, and Bin Cui. 2025. Memo: Fine-grained tensor management for ultra-long context llm training. *Proc. ACM Manag. Data*, 3(1).

Didi Zhu, Yibing Song, Tao Shen, Ziyu Zhao, Jinluan Yang, Min Zhang, and Chao Wu. 2025. Remedy: Recipe merging dynamics in large vision-language models. In *The Thirteenth International Conference on Learning Representations*.

A Preliminaries and Algorithm Details

A.1 Standard Contextual Bandit Formulations

In this section, we provide the standard mathematical formulations for the Upper Confidence Bound (UCB) and Linear UCB (LinUCB) algorithms referenced in Section 4.2.

The classic UCB strategy selects the action a that maximizes the upper confidence bound to balance exploration and exploitation as Equation 9

$$a_t = \text{argmax}_{a \in \mathcal{A}} \left(Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right) \quad (9)$$

where $Q_t(a)$ denotes the average reward of action a, $N_t(a)$ is the selection count, and c is a confidence hyperparameter. The value estimation is updated incrementally as Equation 10

$$Q_{t+1}(a) = Q_t(a) + \frac{1}{N_t(a)} \left(r_t(a) - Q_t(a) \right) \quad (10)$$

In the Contextual Bandit setting, LinUCB assumes the expected reward of an arm a is linear in its feature vector $x_{t,a}$ with an unknown coefficient vector θ_a. The algorithm selects the arm with the highest potential reward upper bound:

$$a_t^* = \arg\max_{a \in \mathcal{A}} \left(\hat{r}(a) + \alpha\sqrt{x_{t,a}^\top A_a^{-1} x_{t,a}} \right) \quad (11a)$$

$$\text{where } \hat{r}(a) = \hat{\theta}_a^\top x_{t,a}, \quad \hat{\theta}_a = A_a^{-1} b_a \quad (11b)$$

Here, A_a represents the covariance matrix (initialized as I_d), b_a is the reward vector, and α controls the exploration intensity.

A.2 Randomized SVD: Procedures and Complexity

A.2.1 Standard Procedures

We employ Randomized SVD to efficiently approximate the low-rank structure of the concatenated KV tensors. Given an input matrix $X \in \mathbb{R}^{m \times d}$ and a target rank k, the standard procedure involves three steps:

1. Random Projection: To capture the dominant range of X, we apply a Gaussian random projection matrix $\Omega \in \mathbb{R}^{d \times k}$:

$$Y = X\Omega \quad (12)$$

This step maps the high-dimensional input into a lower-dimensional subspace $Y \in \mathbb{R}^{m \times k}$.

2. QR Decomposition: An orthonormal basis Q for the range of Y is obtained via QR decomposition:

$$Y = QR \quad (13)$$

where $Q \in \mathbb{R}^{m \times k}$ is an orthonormal matrix and R is upper-triangular.

3. SVD Computation: The original matrix is projected onto the subspace Q to form a smaller matrix $B = Q^\top X \in \mathbb{R}^{k \times d}$. Standard SVD is then performed on B:

$$B = \tilde{U}\Sigma V^\top \quad (14)$$

The final low-rank approximation of X is given by $U = Q\tilde{U}$, yielding $X \approx U\Sigma V^\top$.

A.3 Hyperparameter Settings

To facilitate reproducibility, we list the key hyperparameters used in the BanditKV decision engine and rank allocation module in Table 3.

Module	Parameter	Value
Contextual Bandit	Exploration α	0.1
	Feature Dim (x_t)	64
Reward Function	Accuracy Weight w_{acc}	10.0
	Compression Weight w_{comp}	1.5
	Latency Weight w_{lat}	0.5
Rank Allocation	Rényi Entropy α	0.5
	Min Rank Threshold k_{min}	16

Table 3: Hyperparameter settings for BanditKV experiments.

A.4 Detailed Analysis of High-Compression Robustness

A.4.1 Complexity Analysis

The efficiency gain of BanditKV stems from replacing the full SVD with Randomized SVD. For a concatenated KV tensor $X \in \mathbb{R}^{L \cdot B \times d}$ (where L is the number of layers in a group, B is batch size $\times$ sequence length, and d is hidden dimension), the computational complexity comparison is as follows:

- Full SVD: The complexity is typically $O(\min(m,d)^2 \cdot \max(m,d))$. Since $m \gg d$ in long-context scenarios, this approximates to $O(m \cdot d^2)$.
- Randomized SVD: The dominant cost is the matrix multiplication in the random projection step, with complexity $O(m \cdot d \cdot k)$, where k is the target rank.

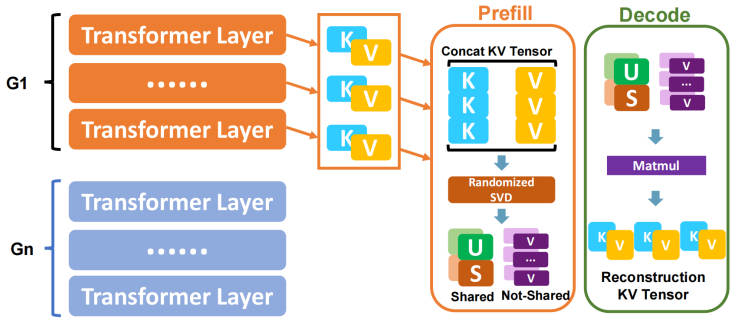

Figure 5: Randomized SVD of KV

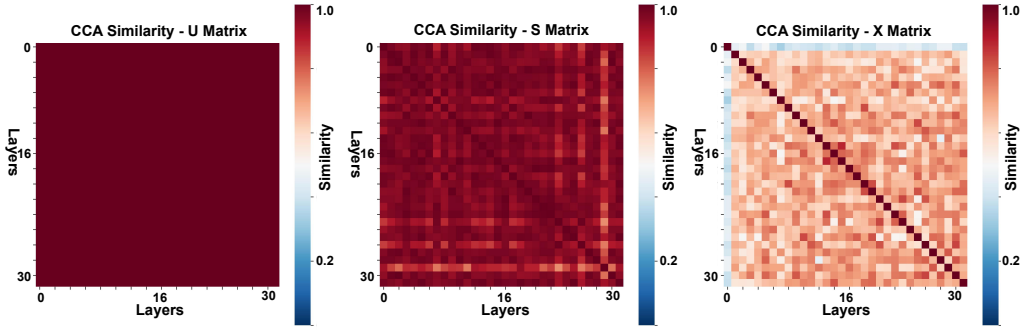

Figure 6: K Tensor Similarity Analysis Across Layers

Table 4: High Compression Experimental Results of BanditKV (Origin Scores)

Compression Ratio (Speed Up Ratio)	Single-Document QA			Multi-Document QA			Summarization			Few-shot Learning			Code		Avg
	NaQA	QAs	MFQA	HPQA	WKQA	Mus	GovR	Qmsum	Mnews	Trec	TrivQA	Ssum	Lcc	RPB	
Llama3-8B-instruct															
1.0x(1.000x)	31.07	33.46	33.06	33.22	33.54	34.76	26.34	25.35	27.95	68.91	68.39	69.11	60.09	56.72	42.96
2.0x(1.065x)	31.05	35.06	33.67	33.84	34.57	34.74	26.33	25.42	28.38	68.85	70.63	70.35	59.99	56.82	43.55
4.0x(1.181x)	31.00	34.92	33.85	33.66	34.40	34.69	26.05	25.59	27.53	68.61	69.94	71.30	59.78	56.83	43.37
6.0x(1.319x)	30.78	32.92	32.82	33.58	34.18	34.82	25.83	25.42	27.19	69.17	68.99	69.94	52.65	56.67	42.45
8.0x(1.457x)	30.97	30.97	30.97	33.64	34.15	34.63	25.46	25.45	26.37	68.06	68.06	68.05	50.30	53.10	42.18
10.0x(1.598x)	30.73	28.79	29.07	33.34	33.11	34.65	25.43	25.25	26.11	68.71	68.53	67.96	47.10	51.08	40.75
12.0x(1.712x)	30.52	✗	✗	32.76	30.30	34.62	✗	24.98	✗	68.51	67.61	67.42	✗	✗	✗
14.0x(1.939x)	30.27	✗	✗	31.59	27.82	34.86	✗	25.24	✗	68.75	68.12	66.52	✗	✗	✗
16.0x(2.207x)	30.34	✗	✗	31.44	27.17	33.49	✗	24.82	✗	68.71	66.98	65.96	✗	✗	✗
Mistral-7B															
1.0x(1.000x)	32.09	31.56	34.66	32.82	33.94	32.96	27.14	24.95	28.15	69.92	67.39	70.01	62.39	57.02	43.12
2.0x(1.060x)	32.45	31.58	35.13	32.63	34.03	32.99	26.97	24.95	28.07	69.22	73.81	70.12	61.92	56.72	43.39
4.0x(1.826x)	32.59	31.35	34.69	32.42	33.73	32.56	26.86	24.97	28.07	67.12	71.21	70.46	61.26	55.55	42.96
6.0x(1.995x)	32.71	30.96	34.69	31.66	32.80	31.78	26.65	24.81	27.76	66.07	68.29	70.12	60.41	54.90	42.35
8.0x(2.211x)	32.47	30.75	33.36	31.76	31.71	31.71	26.43	24.93	27.71	65.27	67.95	70.25	60.31	54.49	42.00
10.0x(2.199x)	31.78	29.55	29.36	30.72	31.13	31.52	26.22	24.20	27.36	64.03	67.18	70.36	58.66	53.11	40.11
12.0x(2.223x)	31.68	26.83	28.18	28.36	30.21	29.79	26.15	24.52	27.17	62.92	66.81	69.18	58.07	52.93	39.53

Table 5: High Compression Experimental Results of BanditKV

Compression Ratio (Speed Up Ratio)	Single-Document QA			Multi-Document QA			Summarization			Few-shot Learning			Code		Avg
	NaQA	QAs	MFQA	HPQA	WKQA	Mus	GovR	Qmsum	Mnews	Trec	TrivQA	Ssum	Lcc	RPB	
LLaMa3-8B-instruct															
1.0x(1.000x)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
2.0x(1.065x)	99.96	104.79	101.87	101.87	103.10	99.95	99.97	100.28	101.54	99.92	103.28	101.80	99.84	101.18	101.38
4.0x(1.181x)	99.79	104.39	102.41	101.33	102.57	99.82	98.93	100.96	98.50	99.57	102.28	103.18	99.50	100.21	100.96
6.0x(1.319x)	99.09	98.40	99.30	101.10	101.92	100.02	98.07	100.29	97.03	100.38	100.88	99.76	87.63	99.73	98.82
8.0x(1.457x)	99.68	92.57	93.70	101.27	101.82	99.64	96.67	100.43	94.36	98.78	99.52	98.48	83.71	93.63	98.20
10.0x(1.598x)	98.91	86.07	87.96	100.39	98.73	99.70	96.55	99.61	93.43	99.72	100.21	98.34	78.39	90.06	94.86
12.0x(1.712x)	98.23	✗	✗	98.63	90.36	99.60	✗	98.55	✗	99.42	98.87	97.56	✗	✗	✗
14.0x(1.939x)	97.44	✗	✗	95.11	82.95	100.30	✗	99.60	✗	99.77	99.61	96.26	✗	✗	✗
16.0x(2.207x)	97.68	✗	✗	94.67	81.01	96.36	✗	97.93	✗	99.71	97.95	95.45	✗	✗	✗
Mistral-7B															
1.0x(1.000x)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
2.0x(1.060x)	101.15	100.07	101.37	99.45	100.27	100.10	99.38	100.03	99.72	99.00	109.54	100.16	99.26	99.48	100.64
4.0x(1.826x)	101.57	99.34	100.10	98.80	99.39	98.79	98.98	100.12	99.72	96.00	105.67	100.65	98.20	97.43	99.63
6.0x(1.995x)	101.94	98.10	100.09	96.47	96.66	96.42	98.21	99.48	98.65	94.50	101.34	100.16	96.83	96.29	98.22
8.0x(2.211x)	101.20	97.44	96.25	96.79	93.45	96.22	97.39	99.92	98.47	93.36	100.84	100.35	96.68	95.58	97.42
10.0x(2.199x)	99.06	93.64	84.73	93.61	91.73	95.66	96.63	97.02	97.20	91.58	99.69	100.51	94.03	93.15	93.07
12.0x(2.223x)	98.75	85.04	81.33	86.42	89.02	90.39	96.36	98.29	96.55	90.00	99.15	98.80	93.08	92.84	91.68

Since $k \ll d$ (typically $k \approx d/16$ for $16\times$ compression), the theoretical speedup ratio is approximately d/k.