Value-Aware Stochastic KV Cache Eviction for Reasoning Models

Ting-Yun Chang[⋆σ] **Harvey Yiyun Fu**[⋆χ] **Deqing Fu**[⋆σ] **Chenghao Yang**[χ]
Jesse Thomason[σ] **Robin Jia**[σ]
[σ]University of Southern California [χ]University of Chicago
{tingyun,deqingfu,jessetho,robinjia}@usc.edu
{harveyfu,chenghao}@uchicago.edu

Abstract

Reasoning models improve accuracy through extended chains of thought, but their long outputs create a memory and compute bottleneck. KV cache eviction methods reduce this cost by evicting unimportant key-value pairs from the cache, yet they often yield worse accuracy than selection-based sparse attention alternatives, which keep the full KV cache. We identify key factors crucial to KV cache eviction accuracy. First, a small fraction of value states have abnormally large magnitudes, and evicting them causes catastrophic failure where models enter repetitive reasoning loops. Second, introducing stochasticity during eviction improves accuracy by increasing cache diversity. Based on these findings, we propose **V**alue-**a**ware **S**tochastic KV Cache **E**viction (VASE), a training-free recipe that protects large-magnitude value states and promotes diverse eviction decisions. Across six reasoning tasks, Qwen3 models using VASE with $4\times$ KV cache compression yield higher average accuracies than SOTA selection method at the same sparsity, while outperforming the strongest eviction method by more than 4%. Overall, VASE bridges the gap between efficiency and accuracy, supporting FlashAttention2 and enabling a static memory footprint for reasoning models.

1 Introduction

Reasoning models (OpenAI, 2024; Qwen, 2024; Guo et al., 2025a) leverage test-time compute (Snell et al., 2025) to improve accuracy by generating extended chains of thought before producing a final answer. While this approach yields high accuracy on complex tasks, it introduces an efficiency bottleneck at test time because the reasoning models tend to generate unnecessary long outputs. For example, Chen et al. (2025) show that models may overthink a simple arithmetic question, using more than 900 tokens to answer "2 + 3 = ?". As a consequence, an auto-regressive large language model that stores the key-value representations of every past token incurs substantial memory and computational overhead as the sequence length grows.

Sparse attention (Zhao et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020; Kitaev et al., 2020) addresses this bottleneck by attending to only a subset of previous tokens; such approaches fall into two broad categories. KV cache *selection* methods (Ribar et al., 2024; Tang et al., 2024; Yang et al., 2025b; Gao et al., 2026) keep the full KV cache in memory but activate only a sparse subset of KV pairs at each decode step. These methods reduce compute and memory movement, but their memory footprints scale linearly with sequence length. In contrast, KV cache *eviction* methods (Zhang et al., 2023; Ge et al., 2024; Devoto et al., 2024; Oren et al., 2024) permanently discard low-importance KV pairs once the cache reaches a predefined budget. While eviction methods (Li et al., 2024; Cai et al., 2025) suffer larger accuracy degradation on reasoning tasks compared to their selection-based counterparts, they can yield a static memory footprint and better throughput.

In this paper, we reveal two key findings to improve KV cache eviction methods on reasoning models. First, the magnitudes of value states show strongly skewed distributions, with a small fraction of

⋆ Equal contribution.

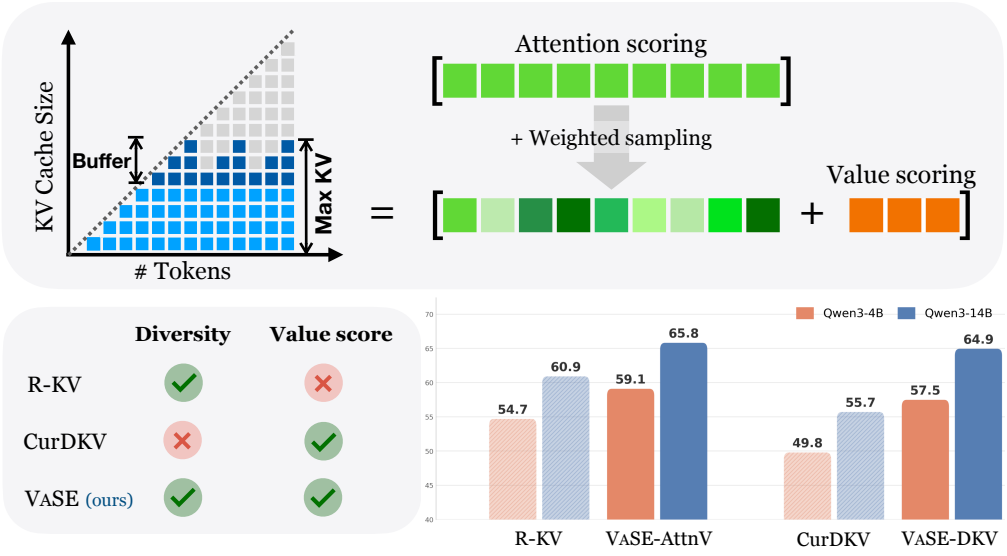

Figure 1: *Top*: VASE is a KV Cache eviction method that combines stochastic sampling with value-state magnitude scoring to retain diverse and important KV pairs under a fixed KV cache budget. *Bottom*: By integrating both stochasticity and value awareness, VASE outperforms baseline methods that use either signal alone, improving average pass@1 accuracy across various reasoning tasks.

tokens having abnormally large vector magnitudes (see Figure 2). Those large-magnitude value states are crucial for maintaining task accuracies: evicting them causes the models to enter repetitive loops, where they endlessly re-examine the context or generate nonsensical outputs without reaching a final answer (see Figure 10 for the examples). Second, we find that introducing stochasticity during KV cache eviction effectively improves reasoning task accuracies. We attribute this to improved diversity of the retained KV pairs, which together yield a more representative coverage of the full context. Our case study on GSM8K demonstrates that incorporating value scoring boosts Qwen3-4B's accuracy by as much as 16.2%, while introducing stochasticity yields an additional 4.7% improvement.

Building on these findings, we propose **V**alue-**a**ware **S**tochastic KV Cache **E**viction (VASE), a training-free KV cache eviction recipe that prioritizes large-magnitude value states and introduces stochasticity during eviction. We target the decoding step of reasoning models, by applying VASE to Qwen3 models (Yang et al., 2025a) and evaluating on six tasks spanning math, code generation, and science question answering. On average, VASE outperforms the previously SOTA eviction method R-KV (Cai et al., 2025) by 4.4% on Qwen3-4B, 4.9% on Qwen3-14B, and 1.1% on GLM-Z1-9B. When combined with CurDKV (Sengupta et al., 2025), stochastic sampling boosts its average accuracy by 7.7%, 9.2%, and 5.3% on the three models, respectively. VASE even slightly surpasses the average accuracy of SeerAttention-R (Gao et al., 2026), a strong selection method whose KV cache memory grows linearly with the sequence length, on both models while using static memory. In addition, we benchmark the actual throughput and peak memory of different methods, showing that VASE achieves higher throughput and a lower memory footprint than the R-KV eviction baseline across token budgets. Our results match closely with the theoretical memory compression ratio.

Furthermore, our insights generalize to other efficiency methods like KV cache quantization; we show that large-magnitude value states also lead to disproportionately large reconstruction errors under per-token quantization (Liu et al., 2024b). Together, we demonstrate that value-state magnitude and stochasticity are fundamental axes of KV cache management, with our VASE recipe bridging the efficiency-accuracy gap left by prior sparse attention methods.[1]

2 Problem Setup

We formalize the decode-phase KV cache eviction setting by drawing the distinction between selection-based and eviction-based methods, and introduce several representative eviction baselines.

[1] Our code is available at `https://github.com/terarachang/VaSE`.

Selection-based method. Selection-based sparse attention methods retain the full KV cache but activate only a sparse subset of KV pairs at each decoding step. A representative example is SeerAttention-R (Gao et al., 2026), which uses a trained attention gate to predict attention sparsity patterns and selectively activates different KV cache blocks during inference, thereby reducing computational complexity and memory movement. Compared to eviction-based methods, SeerAttention-R retains all KV pairs and the memory footprint scales linearly as $\mathcal{O}(T)$ with sequence length T.

Eviction-based method. KV cache eviction methods permanently discard low-importance key-value pairs once the cache size reaches a predefined budget, reducing memory cost at the risk of irrecoverable information loss. We specifically focus on eviction methods for the decoding step of autoregressive reasoning models, as these models can generate over $10{,}000$ tokens for a math question that has fewer than 200 prompt tokens (see Table 4 for statistics).

Formally, let N be the total KV cache budget, and d be the dimension of keys and values. As the KV cache budget fills up at the token index t, we have N KV pairs $(\boldsymbol{k}_i, \boldsymbol{v}_i)$ for $i = 1, \ldots, N$, where $\boldsymbol{k}_i \in \mathbb{R}^d, \boldsymbol{v}_i \in \mathbb{R}^d$. Given the current query $\boldsymbol{q}_t \in \mathbb{R}^d$, the attention head computes its output as a weighted sum over the N cached values,

$$\boldsymbol{o}_t \;=\; \sum_{i \le N} \alpha_i^{(t)}\, \boldsymbol{v}_i, \qquad \alpha_i^{(t)} \;=\; \frac{\exp\!\left(\boldsymbol{q}_t^\top \boldsymbol{k}_i / \sqrt{d}\right)}{\sum_{j \le N} \exp\!\left(\boldsymbol{q}_t^\top \boldsymbol{k}_j / \sqrt{d}\right)}, \tag{1}$$

where $\alpha_i^{(t)}$ is the attention weight assigned to token i, obtained by applying softmax to the scaled dot products between the current query and all cached keys (Vaswani et al., 2017).

Periodic eviction with budget K and buffer B. We adopt the periodic-eviction framework proposed by Cai et al. (2025) and Song et al. (2025), which targets decode-phase compression. The framework consists of a persistent budget of K tokens and a buffer of size $B \ll K$ holding the most recent tokens, capping the total cache size at $N = K + B$ tokens. Since each decoding step adds one KV pair to the cache, the buffer fills every B steps and triggers an *eviction step*: the eviction operator chooses B pairs to discard, clears the buffer, and restores the cache to size K (See the upper-left panel in Figure 1). Concretely, the most recent B KV pairs in the buffer are protected from eviction. A *scoring function* then assigns a scalar importance score to each of the remaining K candidates, and the B candidates with the lowest importance scores will be discarded. Below, we describe the scoring functions of three prior eviction methods, SnapKV (Li et al., 2024), R-KV (Cai et al., 2025), and CurDKV(Sengupta et al., 2025). Following common practice, KV pairs are scored independently for each attention head and layer. To simplify notation, we omit layer and head indices hereafter.

SnapKV. Li et al. (2024) rank candidate pairs by their average attention score over a window of recent queries, where we set the window size to be the same as the buffer size B.[2]

$$\bar{\alpha}_i \coloneqq \frac{1}{B} \sum_{\tau=t-B+1}^{t} \alpha_i^{(\tau)}, \tag{2}$$

where i is the KV pair index, t is the eviction timestep, and $\alpha_i^{(\tau)}$ is the attention score defined in Eq. 1. SnapKV uses keys for the attention computation, but does not consider values in the scoring function.

R-KV. Cai et al. (2025) specifically target redundant tokens in reasoning models. They augment the SnapKV score $\bar{\alpha}_i$ with a redundancy penalty r_i to promote diversity among retained KV pairs:

$$s_i^{\text{RKV}} \coloneqq \lambda \cdot \bar{\alpha}_i - (1-\lambda) \cdot r_i, \qquad \text{where} \quad r_i = \text{Softmax}\left(\frac{1}{K} \sum_{j \neq i} \text{CosSim}(\boldsymbol{k}_i, \boldsymbol{k}_j)\right), \tag{3}$$

where $\lambda \in [0, 1]$ is a hyperparameter. A high r_i indicates that token i has a high cosine similarity to many cached tokens; subtracting it discourages choosing redundant pairs.

CurDKV. Sengupta et al. (2025) observe the limitations of scoring KV pairs with attention alone, which ignores value states. They compute leverage scores for keys and values respectively via

[2]Following Li et al. (2024), we apply average pooling to $\bar{\alpha}_i$ to smooth the attention scores. Following the decode-phase SnapKV variant introduced by Cai et al. (2025), we cache the most recent queries and compute non-causal attention scores.

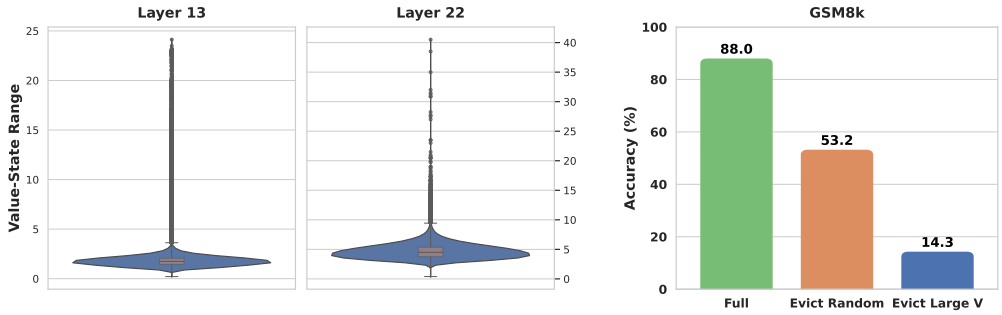

Figure 2: *Left*: Range distribution of the value states. The violin plots show the presence of extreme magnitude outliers at different layers. *Right*: Evicting the large magnitude outliers causes accuracy to collapse to 14.3%, greatly underperforming a random eviction baseline at the same token budget, suggesting that these large-magnitude value states are crucial to model accuracy.

approximate CUR matrix decomposition and combine them into a single importance score. Originally designed for the prefill stage, we extend CurDKV for the decode phase with the periodic eviction framework. Specifically, CurDKV draws a Gaussian projection $\boldsymbol{G} \in \mathbb{R}^{d\times r}$ at the start of generation, with each entry $G_{ab} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/r)$, and rank candidates by the product of key and value scores:

$$s_i^{\text{CUR}} := \ell_i^{(K)}(\boldsymbol{G}) \cdot \ell_i^{(V)}(\boldsymbol{G}), \qquad \text{where} \quad \ell_i^{(K)}(\boldsymbol{G}) := \|\boldsymbol{G}^\top \boldsymbol{k}_i\|_2^2, \quad \ell_i^{(V)}(\boldsymbol{G}) := \|\boldsymbol{G}^\top \boldsymbol{v}_i\|_2^2. \tag{4}$$

CurDKV considers value states during scoring, but it does not promote diversity among KV pairs.

3 Methodology

We hypothesize that incorporating value states and diversity into KV cache scoring is essential for preserving model accuracy under eviction. Inspired by the scoring function of CurDKV, which favors value vectors with larger magnitudes, and by the finding of Sun et al. (2024) that massive activations in LLMs are indispensable, we first study the effect of large-magnitude value states on eviction accuracy. Based on the analysis, we propose a framework that prioritizes those large-magnitude value states within the scoring function and introduces stochasticity to promote diversity.

3.1 Significance of Value States with Large Magnitudes

We first analyze the distribution of value-state magnitudes, where we extract the full KV cache from Qwen3-4B while generating responses to GSM8K examples (Cobbe et al., 2021). We define the magnitude[3] of a cached value vector $\boldsymbol{v} \in \mathbb{R}^d$ as:

$$\text{Range}(\boldsymbol{v}) := \max_{j\in[d]} v_j - \min_{j\in[d]} v_j \tag{5}$$

where v_j is the j-th entry of $\boldsymbol{v}$, and $\text{Range}(\boldsymbol{v})$ is chosen over $L_2(\boldsymbol{v})$ for its connection to quantization (see §4.4). Figure 2 shows that the distribution of $\text{Range}(\boldsymbol{v})$ has large outliers over layers (see Appendix A for more layers). We hypothesize that these large-Range outliers are critical to the model accuracy. Because attention output is computed as a weighted summation of values (Eq. 1), those values with larger magnitudes have a disproportionate influence on the output.

To validate this hypothesis, we purposefully evict the B largest KV pairs ranked by $\text{Range}(\boldsymbol{v}_i)$. This reduces GSM8K accuracy to just 14.3%, under-performing random eviction by 38.9% at the same 512-token budget (Figure 2; *Right*). Further inspection of the model outputs reveals that the model tends to enter endless loops of self-reflection and never reaches a conclusion; it also tends to generate non-grammatical sentences (see Figure 10). Prior work (Guo et al., 2024, 2025b) observes that low-magnitude values co-occur with attention sinks (Xiao et al., 2024), which are crucial to eviction methods. In this paper, we demonstrate that large-magnitude values are equally important and should be up-weighted within the scoring function (§4.1).

[3]We experiment with three variants, Range, L_2 norm, and variance, to capture the magnitude and variety of value vectors in Appendix A. We find that these variants are positively correlated and lead to comparable accuracies.

Method	Family	Key score	Value score	Diversity
SeerAttention-R (Gao et al., 2026)	Select	Attn Gate	–	–
SnapKV (Li et al., 2024)	Evict	$\bar{\alpha}_i$	–	–
R-KV (Cai et al., 2025)	Evict	$\lambda \cdot \bar{\alpha}_i - (1-\lambda) \cdot r_i$	–	redundancy r_i
CurDKV (Sengupta et al., 2025)	Evict	$\ell_i^{(\boldsymbol{K})} = \|(\boldsymbol{KG})_i\|_2^2$	$\ell_i^{(\boldsymbol{V})} = \|(\boldsymbol{VG})_i\|_2^2$	–
VASE-ATTNV(ours)	Evict	$\bar{\alpha}_i$	$\mathrm{Range}(\boldsymbol{v}_i)$	sample $\bar{\alpha}_i$
VASE-DKV(ours)	Evict	$\ell_{i,t}^{(\boldsymbol{K})} = \|(\boldsymbol{KG}_t)_i\|_2^2$	$\ell_{i,t}^{(\boldsymbol{V})} = \|(\boldsymbol{VG}_t)_i\|_2^2$	sample $\boldsymbol{G}_t$

Table 1: Design space of sparse attention methods. Among eviction methods, prior work covers at most two of the three axes (key-based scoring, value-based scoring, diversity); VASE is the first to combine all three, instantiated as VASE-AttnV and VASE-DKV.

3.2 VASE: Value-Aware Stochastic Eviction

We propose VASE (Value-aware Stochastic Eviction), a training-free eviction recipe designed around two principles: upweighting large-magnitude value states in the scoring function, and introducing stochasticity to promote diversity in the retained cache. We propose two VASE variants that apply our recipe to SnapKV and CurDKV, respectively.

VASE-ATTNV. VASE-ATTNV reserves part of the token budget for large-magnitude value states and applies stochastic sampling on top of SnapKV. Recall that K is the persistent budget and B is the buffer size (§ 2). We first rank each KV pair by $\mathrm{Range}(\boldsymbol{v}_i)$ (Eq. 5), where $i \in [K]$ indexes the non-buffer tokens. Given a value reservation budget $N_v < K$, we select the N_v candidates with the largest $\mathrm{Range}(\boldsymbol{v}_i)$ and unconditionally retain them as the reserved set $\mathcal{R}_{\boldsymbol{V}}$. We denote the set of tokens in the buffer as $\mathcal{R}_B$, forming the full reserved set $\mathcal{R} = \mathcal{R}_{\boldsymbol{V}} \cup \mathcal{R}_B$. The remaining $K - |\mathcal{R}|$ slots are then filled by sampling from the SnapKV attention distribution, with weights proportional to $\bar{\alpha}_i$ (Eq. 2). This preserves SnapKV's core idea of favoring high-attention tokens, while replacing its hard topk selection with a soft, probabilistic one.

Stochastic sampling offers a key advantage over topk in terms of token retention flexibility. Let $\pi_i^{(t)}$ denote the probability of token i being retained at step t. After T eviction steps, the probability that token i is kept in the KV cache can be factorized as $\prod_{t=1}^{T} \pi_i^{(t)}$. Under topk, any token below the cutoff is evicted with certainty ($\pi_i^{(t)} = 0$). In contrast, under sampling, every factor is strictly positive, $\pi_i^{(t)} \geq 1 - (1 - \bar{\alpha}_i^{(t)}/Z_t)^{K-|\mathcal{R}|} > 0$, where $Z_t = \sum_{j \in \mathcal{C}_t} \bar{\alpha}_j^{(t)}$ and $\mathcal{C}_t = [K] \setminus \mathcal{R}_{\boldsymbol{V}}^{(t)}$ is the candidate set for sampling; therefore, each token has a nonzero probability of being retained across all T steps.

VASE-DKV. VASE-DKV inherits value-state awareness directly from CurDKV's product score (Eq. 4), where the value score $\ell_i^{(V)}(\boldsymbol{G})$ upweights large-magnitude values. On top of this, we introduce stochasticity through the Gaussian matrix $\boldsymbol{G}$ at each eviction step t to promote diversity among the retained KV pairs. Specifically, at step t we resample a random projection $\boldsymbol{G}_t \in \mathbb{R}^{d \times r}$ independently of all past projections, and compute the leverage scores following Eq. 4. This improves upon CurDKV, which uses a fixed $\boldsymbol{G}$ across all steps: tokens with certain representations are then consistently assigned low scores and deterministically evicted. Resampling $\boldsymbol{G}$ avoids this failure mode by applying an independent scoring criterion at each eviction step such that no token would be permanently disadvantaged by a single projection.

4 Experiments

4.1 Isolating the Effects of Value-Awareness and Stochasticity

We perform a case study on GSM8K to isolate the individual contributions of value-awareness and stochasticity to the accuracy of Qwen3-4B.

Value Awareness. We experiment with different reservation budgets N_v to evaluate the efficacy of our value-scoring function $\mathrm{Range}(\boldsymbol{v})$ (Eq. 5). To isolate the effect of our value scoring, we implement VASE-ATTNVwithout sampling. Specifically, we fill the remaining budget with KV pairs ranked highest by SnapKV's scoring function $\bar{\alpha}_i$ (Eq. 2). Figure 3 shows that when increasing the number of

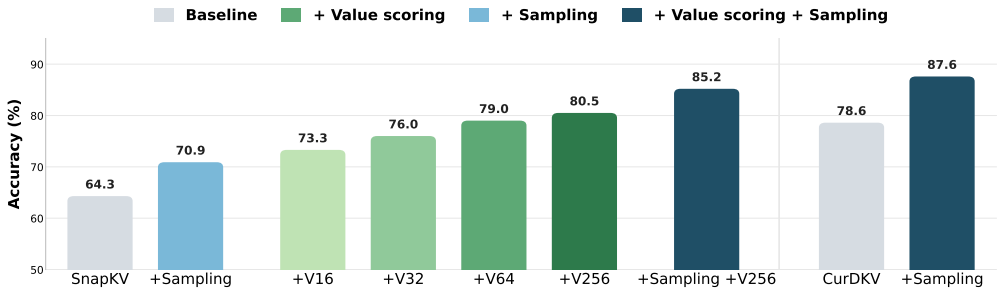

Figure 3: Accuracy of Qwen3-4B on GSM8K with a KV cache budget of K=512 ($\sim$4$\times$ compression). **+V16** indicates reserving 16 slots in the budget K for value scoring. By combining stochastic sampling with value scoring, our methods ■ achieve accuracy close to the full model (88.4%).

N_v from 16 to 256 (green bars), the accuracies on GSM8K increase consistently from 73.3%(+9.0%) to 80.5%(+16.2%), showing the importance of keeping large-range values from eviction.

Stochasticity. We introduce stochasticity into SnapKV and CurDKV to promote diversity in KV pair retention. For SnapKV (*leftmost* in Figure 3), replacing its deterministic topk selection with $\bar{\alpha}_i$-weighted sampling improves model accuracy from 64.3% to 70.9%. Similarly, for CurDKV (*rightmost*), resampling Gaussian matrix $\boldsymbol{G}_t$ at each eviction step t throughout generation improves accuracy from 78.6% to 87.6%. The teal bars ■ show that value scoring and stochasticity are complementary, with the combination of both strategies achieving the highest accuracies. Section 4.3 further quantifies the diversity of the value states discarded by attention-weighted sampling and deterministic topk selection.

4.2 Main Results

Setup. We evaluate VASE on Qwen3-4B, Qwen3-14B (Yang et al., 2025a), and GLM-Z1-9B (GLM et al., 2024) across five reasoning tasks: AIME25, AIME26 (Art of Problem Solving), HMMT25 (Feb and Nov splits combined; HMMT 2025), GPQA-Diamond (Rein et al., 2024), MATH (Hendrycks et al., 2021); with additional results on LiveCodeBench-v6 (Jain et al., 2025).[4] All eviction methods operate under the periodic-eviction framework (§2) with a shared recency buffer of $B = 64$ and a persistent budget K. For each task, we first measure the average number of tokens N_{avg} under the uncompressed full model and then set the eviction budget K to approximately $N_{\text{avg}}/4$, where $K \in \{1024, 2048, 4096\}$. This yields a nominal 4$\times$ compression ratio over the uncompressed baseline. SeerAttention-R is configured at the matching 25% activation ratio so that all sparse methods operate at the same effective attention sparsity. We use the models' default top-p configurations for all generations. We report pass@1 results, which are averaged over 16 runs for datasets under 100 examples, and 8 runs for larger datasets. We select the hyperparameters on GSM8K for each method and apply them to other tasks without further tuning (see Appendix B).

VASE-ATTNV achieves the best average accuracy. Table 2 shows that both VASE variants outperform prior eviction baselines on the per-model average and reach comparable accuracy with the selection method SeerAttention-R[5]. On Qwen3-4B, VASE-ATTNV achieves an average accuracy of 59.09%, edging out SeerAttention-R (58.81%) and surpassing the strongest eviction baseline R-KV (54.69%) by 4.4%; VASE-DKV reaches 57.48%, improving over CurDKV (49.78%) by 7.7%. The same pattern holds on Qwen3-14B: VASE-ATTNV (65.81%) matches SeerAttention-R (65.37%) and outperforms R-KV (60.90%) by 4.9%; VASE-DKV (64.94%) improves over the deterministic CurDKV baseline (55.70%) by 9.2%. On GLM-Z1-9B, both VASE variants and R-KV yield average accuracy close to the Full model, with VASE-ATTNV consistently performs the best and VASE-DKV greatly outperforms CurDKV. Across all evaluated models, VASE-ATTNV achieves the highest average accuracy among eviction methods; GPQA-Diamond is the only task on which it underperforms R-KV, possibly because its multiple-choice format requires different attention patterns. As SnapKV and R-KV baselines do not consider value states in their deterministic scoring, the

[4]We use the same evaluation scripts as SeerAttention-R: `https://github.com/microsoft/SeerAttention`

[5]Note that SeerAttention-R requires training, and its released checkpoints are limited to Qwen- and Llama-based models; therefore, we do not report SeerAttention-R results on GLM.

Model	Method	AIME25	AIME26	HMMT25	GPQA-D	MATH	Avg.
Qwen3-4B	Full	66.41	62.71	45.94	56.19	93.93	65.04
	SeerAttention-R	58.59	60.42	40.42	49.94	84.67	58.81
	SnapKV	47.92	53.33	35.31	36.87	72.30	49.15
	R-KV	49.58	54.58	36.98	**49.05**	83.28	54.69
	CurDKV	48.75	54.58	33.23	37.94	74.42	49.78
	VASE-DKV	57.29	61.88	39.58	43.94	**84.72**	57.48
	VASE-ATTNV	**59.17**	**62.08**	**44.38**	47.98	81.82	**59.09**
Qwen3-14B	Full	70.21	76.04	54.79	65.25	95.22	72.30
	SeerAttention-R	64.79	65.62	48.65	61.68	86.12	65.37
	R-KV	54.58	60.21	44.58	**59.09**	86.05	60.90
	CurDKV	53.75	61.25	39.58	48.48	75.42	55.70
	VASE-DKV	63.33	68.75	49.38	56.76	**86.47**	64.94
	VASE-ATTNV	**63.96**	**72.29**	**50.00**	57.39	85.42	**65.81**
GLM-Z1-9B	Full	58.33	67.50	46.46	54.55	94.67	64.30
	SnapKV	52.50	64.38	47.71	51.64	84.45	60.14
	R-KV	56.46	67.71	47.60	**54.86**	88.67	63.06
	CurDKV	52.08	61.46	43.33	49.05	84.80	58.14
	VASE-DKV	57.71	67.71	48.02	52.71	**91.08**	63.45
	VASE-ATTNV	**59.38**	**69.38**	**49.06**	53.35	89.68	**64.17**

Table 2: Reasoning-task accuracy (%) of sparse attention methods on Qwen3-4B, Qwen3-14B, and GLM-Z1-9B with $\sim 25\%$ of full KV activated ($4\times$ compression for eviction methods). **Bold** marks the best eviction method of each task. Our eviction methods VASE achieve comparable average accuracy to SeerAttention-R, a strong selection method.

results of VASE-ATTNV show the effectiveness of combining value awareness with stochasticity in the eviction scoring function. The substantial improvements of VASE-DKV over CurDKV further highlight the importance of stochasticity. Despite VASE being an eviction method, we demonstrate that the VASE recipe can recover accuracy without paying the memory cost of selection methods.

Code generation. Figure 6 (*Left*) reports pass@1 of Qwen3-4B on LiveCodeBench-v6-Medium under a 2048-token budget ($\sim 20\%$ of full KV size). R-KV (62.6%), VASE-DKV (61.9%), and VASE-ATTNV (63.5%) achieve comparable performance, while CurDKV only yields 34.6%. Surprisingly, SeerAttention-R (45.3%) underperforms eviction methods, with the exception of CurDKV. This is likely because the learning-based attention gate of SeerAttention-R struggles to generalize under domain shifts, while the training-free eviction methods are less affected.

Token Budgets. Figure 4 reports pass@1 accuracy of Qwen3-14B on AIME26 and HMMT25 as the KV cache budget increases from 2048 to 6144 tokens. At the tightest budget of 2048 tokens, which corresponds to roughly $7.5\times$ compression of the full cache on AIME26 and $8.7\times$ on HMMT25, both VASE variants show clear improvement over R-KV and CurDKV baselines. As the budget increases, the performance gap diminishes with the accuracy of all methods recovering. Nevertheless, VASE methods (teal lines) consistently outperform the baselines. We observe this pattern in both datasets: the accuracy gains from value awareness and stochastic eviction are largest under aggressive compression and diminish gracefully when the cache capacity is abundant (see Table 6 in the appendix for results on GPQA-Diamond).

4.3 Quantifying KV Cache Diversity

To validate our hypothesis that stochasticity improves diversity, we quantify the diversity of the value states discarded at each eviction step. Let $\mathcal{E}_t$ be the set of B evicted value states at step t (§2). We define their diversity D_t as the mean pairwise cosine distance:

$$D_t \;:=\; 1 - \frac{\sum_{\substack{i,j \in \mathcal{E}_t \\ i \neq j}} \mathrm{CosSim}(\boldsymbol{v}_i, \boldsymbol{v}_j)}{B(B-1)}. \tag{6}$$

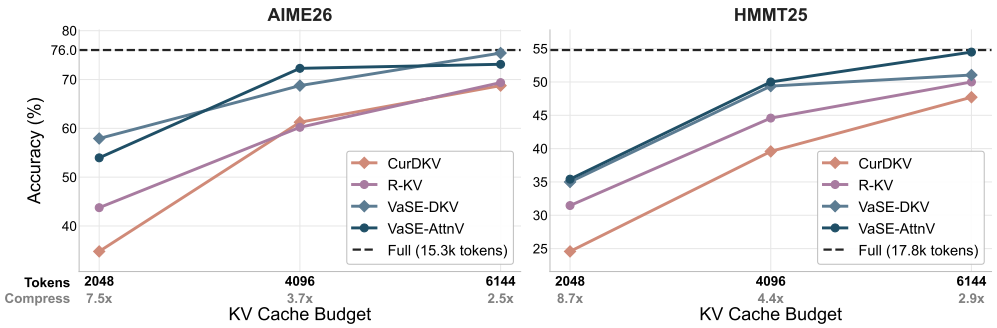

Figure 4: Pass@1 accuracy of Qwen3-14B under varying KV cache budgets. The full-cache model uses an average of 15.3k and 17.8k tokens on AIME26 and HMMT25, respectively. Each x-axis label reports the number of budget tokens (e.g., 2048) and its corresponding KV cache compression ratio to the full model (e.g., 7.5×).

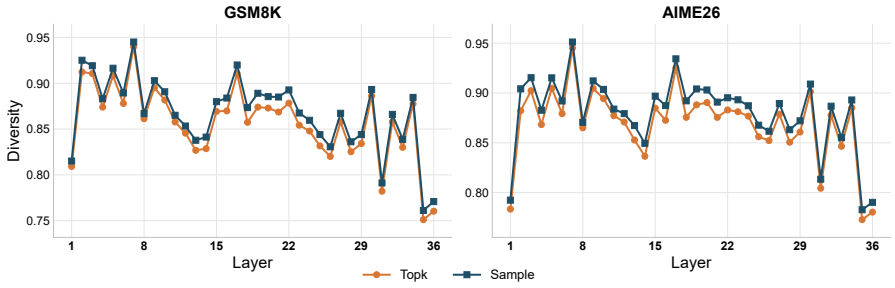

Figure 5: Diversity (mean pairwise cosine distance) among evicted value states across layers of Qwen3-4B. On both GSM8K and AIME26 datasets, stochastic sampling yields higher diversity among evicted value states than deterministic topk selection.

For each layer, we compute D_t at every eviction step on the GSM8K and AIME26 datasets with 16 runs per example. We measure the average D_t over steps within a sequence and then over dataset examples. We compare attention-weighted sampling, the stochastic selection strategy used in VASE-ATTNV, with the deterministic topk selection used in SnapKV. Figure 5 shows that on both datasets, sampling yields higher value-state diversity scores than topk across all 36 layers. These results show that stochastic sampling evicts a less similar set of value states than deterministic topk, supporting our hypothesis that sampling encourages diversity over the deterministic topk strategy.

4.4 Large-Range Values Correlate with KV Cache Quantization Error

Our findings regarding large-Range values have a deep connection with per-token KV cache quantization (Liu et al., 2024b; Su and Yuan, 2025), an alternative way to compress the KV cache. Formally, in asymmetric b-bit linear quantization, the range of the input value state $\boldsymbol{v}$, $[\min \boldsymbol{v}, \max \boldsymbol{v}]$, is mapped to the full range of the quantized integer space, $[0, 2^b - 1]$. The b-bit integer quantization-dequantization process can be expressed as:

$$Q(\boldsymbol{v}) = \left\lfloor \frac{\boldsymbol{v} - z_{\boldsymbol{v}}}{s_{\boldsymbol{v}}} \right\rceil, \quad \boldsymbol{v}' = Q(\boldsymbol{v}) \cdot s_{\boldsymbol{v}} + z_{\boldsymbol{v}}, \tag{7}$$

where $z_{\boldsymbol{v}} = \min \boldsymbol{v}$ is the zero point, $s_{\boldsymbol{v}} = (\max \boldsymbol{v} - \min \boldsymbol{v})/(2^b - 1) = \mathrm{Range}(\boldsymbol{v})/(2^b - 1)$ is the scaling factor, $\lfloor \cdot \rceil$ is the rounding operation, and $\boldsymbol{v}'$ is the reconstructed vector of $\boldsymbol{v}$ after dequantization. The scaling factor $s_{\boldsymbol{v}}$ represents the step size of quantization: if $\mathrm{Range}(\boldsymbol{v})$ is small, $s_{\boldsymbol{v}}$ is small, leading to fine-grained quantization, and the reconstructed value $\boldsymbol{v}'$ is close to the original value $\boldsymbol{v}$; on the other hand, if $\mathrm{Range}(\boldsymbol{v})$ is large, a large $s_{\boldsymbol{v}}$ results in a large and coarse step size, causing $\boldsymbol{v}'$ to deviate significantly from $\boldsymbol{v}$, which is known as the outlier issue (Dettmers et al., 2022) in LLM quantization.

We validate the relationship between $\mathrm{Range}(\boldsymbol{v})$ and per-token quantization error on the value cache empirically, using the HQQ quantizer (Badri and Shaji, 2023) with different bit-widths and quantiza-

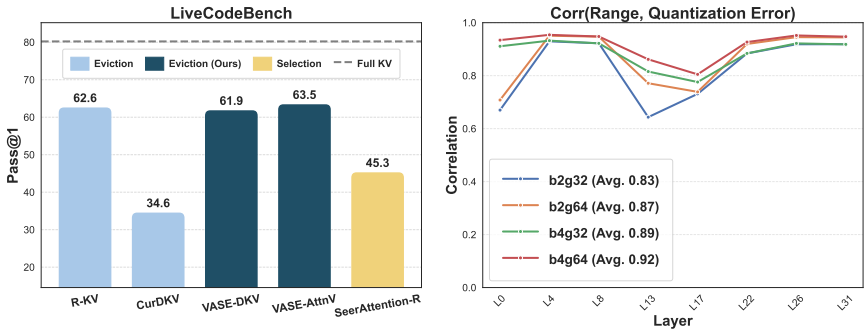

Figure 6: *Left*: Pass@1 results of Qwen3-4B on LiveCodeBench under a 2048-token budget ($\sim$20% of full KV). R-KV and our VASE methods achieve the strongest performance. *Right*: $\mathrm{Range}(\boldsymbol{v})$ and per-token value cache quantization errors are highly correlated under different quantization configurations, where b2g32 means 2-bit precision with a group size of 32.

tion group sizes. Quantization error is measured via the mean-squared reconstruction error between $\boldsymbol{v}$ and $\boldsymbol{v}'$ (Frantar et al., 2023). We extract the full value cache from Qwen3-4B during the generation of GSM8K examples, quantize them, and compute the per-token mean-squared error. Figure 6 (*Right*) demonstrates that $\mathrm{Range}(\boldsymbol{v})$ and the reconstruction error are highly correlated across layers under different quantization configurations, suggesting that large-Range value states lead to large information loss under per-token KV cache quantization.

5 Benchmark Throughput and Memory

In this section, we benchmark the actual decode-phase throughput and peak memory of different methods on the Qwen3-14B model using a single A100-80G GPU. We sweep over KV cache budgets $\{2048, 4096, 6144\}$ and total output tokens $\{16384, 32768\}$. All methods are benchmarked with FlashAttention2 kernels (Dao, 2024), without PagedAttention (Kwon et al., 2023). Our benchmark script builds on that of Song et al. (2025); we set the batch size to 16 and the input prompt length to 256 tokens across all experiments.

Figure 7 (*Left & Middle*) shows that all eviction methods achieve substantially higher throughput than the Full baseline, with a consistent ordering of VASE-DKV$>$ VASE-ATTNV$>$ R-KV$>$ Full across all settings. For example, at 16K output tokens and a KV cache budget of 2048, VASE-DKV is $3.1\times$ faster than Full (411 vs. 133 tokens per second). Our VASE-DKV is the fastest eviction method because it does not compute attention scores during eviction, and VASE-ATTNV outperforms R-KV because it avoids the additional redundancy-score computation used to promote diversity. For all eviction methods, the throughput decreases with increase of the token budget. On the other hand, the ordering of memory footprint (Figure 7, *Right*, $\downarrow$) is reversed: VASE-DKV$<$ VASE-ATTNV$<$ R-KV$<$ Full. Figure 12 in the appendix shows consistent ranking under different token budgets. Excluding the memory of the 14B model weights and prefilling (the hatched bars), our VASE methods achieve roughly $4\times$ theoretical KV cache compression (16K/4K) over the Full baseline.

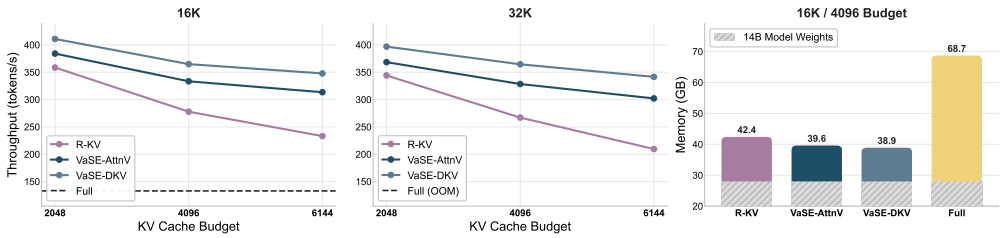

Figure 7: *Left & Middle*: Decode throughput ($\uparrow$) of the Qwen3-14B model on a single A100-80G GPU under different KV cache budgets {2048, 4096, 6144} and total output tokens {16K, 32K}. All eviction methods run well above the original Full method (dashed line; OOM at 32K), with VASE-DKV achieving the fastest throughput. *Right*: Peak GPU memory ($\downarrow$) at the 16K output tokens and 4096 budget; the 14B model weights (hatched) account for around 28GB of memory. All eviction methods have a much lower memory footprint than Full, with VASE-DKV using the least memory.

6 Related Work

KV Cache Compression Methods. Prior work reduces the cost of the KV cache through different approaches: reducing the number of tokens via sparse attention (Liu et al., 2023; Jiang et al., 2024; Singhania et al., 2024; Chari and Van Durme, 2025), reducing the precision via quantization (Hooper et al., 2024; Liu et al., 2024b; Kim et al., 2025; Zandieh et al., 2026), and reducing the hidden dimension via low-rank decomposition (Saxena et al., 2024; Liu et al., 2024a; Chang et al., 2025). In this paper, we focus on sparse attention methods. Our findings on the importance of large-magnitude values also carry profound implications for per-token KV cache quantization (Liu et al., 2024b).

Decoding Phase Compression. Inference proceeds in two phases. In the prefill phase, the model processes the T-token input prompt and computes T key–value pairs in parallel. In the decode phase, the model generates one token at a time and appends the corresponding KV pairs to the cache. Unlike the prefill phase, the decode phase is memory-bound, and the growing cache intensifies the memory bottleneck. For reasoning models, the KV pairs from the decode phase dominate the cost due to the long thinking traces. Therefore, several recent works (Cai et al., 2025; Song et al., 2025; Gao et al., 2026; Guo et al., 2026) have targeted KV cache compression during the decoding phase of reasoning models. Our paper also focuses on the decoding phase compression of reasoning models. Unlike Song et al. (2025), we do not distinguish between prompt tokens and generated tokens at the eviction step; therefore, prompt tokens may also be evicted in our experiments.

Leveraging Value-State Magnitude for KV Cache Scoring. Typically, sparse attention involves scoring key-value pairs by their importance. Because the output of attention is a weighted combination of values, Guo et al. (2024); Devoto et al. (2025) score KV pairs by weighting attention scores with the norm of their corresponding values. In this paper, we deeply investigate the importance of large-magnitude values. Rather than multiplying attention and value scores, we propose an alternative, VASE-ATTNV, which reserves dedicated slots in the token budget to ensure that large-magnitude values are always preserved in the cache.

7 Discussion and Conclusion

Implications for KV cache quantization. We show that large-Range value states are not only critical for maintaining KV cache eviction accuracy but also a major source of error under quantization. Therefore, a promising future direction is to investigate a mixed-precision approach (Liu et al., 2024b) to maintain the precision of large-Range value states. For example, they can be placed in a high-precision reserved cache, while the remaining KV cache is quantized to a lower bit-width. Once the high-precision budget is used up, the earliest value state in this reserved cache can be quantized and "evicted" into the low-precision cache.

Large-magnitude value states are crucial for reasoning progression. Our observation in Figure 10 suggests that large-magnitude value states play a special role in maintaining reasoning progression. After these values are evicted, generations often degenerate into repetitive loops, where the model repeatedly restates intermediate reasoning without making progress. One hypothesis is that these values help transition between latent reasoning steps, preventing the model from collapsing into self-reinforcing paths. Therefore, their removal traps the model in repetitive loops. One potential future direction is to analyze what information large-magnitude value states carry and how they influence long-form reasoning.

Conclusion. We present VASE, a training-free KV cache eviction framework designed to address the memory bottleneck of reasoning models. We identify two critical factors for maintaining accuracy during KV cache compression: protecting large-magnitude value states from eviction, and introducing stochasticity to improve KV cache diversity. Taking both factors into consideration, VASE provides a simple and effective recipe for improving existing eviction methods. More broadly, our findings highlight the importance of value states in long-form reasoning and suggest new directions for designing memory-efficient inference methods.

Acknowledgment

This work was supported in part by a gift from the Capital One Center for Responsible AI and Decision Making in Finance (CREDIF), and by the National Science Foundation under Grant No. IIS-2403436. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Art of Problem Solving. AIME problems and solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions. 6

Hicham Badri and Appu Shaji. Half-quadratic quantization of large machine learning models, November 2023. URL https://dropbox.github.io/hqq_blog/. 8

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. 1

Zefan Cai, Wen Xiao, Hanshi Sun, Cheng Luo, Yikai Zhang, Ke Wan, Yucheng Li, Yeyang Zhou, Li-Wen Chang, Jiuxiang Gu, Zhen Dong, Anima Anandkumar, Abedelkadir Asi, and Junjie Hu. R-KV: Redundancy-aware KV cache compression for reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 1, 2, 3, 5, 10

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. Palu: KV-cache compression with low-rank projection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LWMS4pk2vK. 10

Vivek Chari and Benjamin Van Durme. Compactor: Calibrated query-agnostic kv cache compression with approximate leverage scores. *arXiv preprint arXiv:2507.08143*, 2025. 10

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=MSbU3L7V00. 1

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 4

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024. 9

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. GPT3.int8(): 8-bit matrix multiplication for transformers at scale. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=dXiGWqBoxaD. 8, 16

Alessio Devoto, Yu Zhao, Simone Scardapane, and Pasquale Minervini. A simple and effective l_2 norm-based strategy for KV cache compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18476–18499, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1027. URL https://aclanthology.org/2024.emnlp-main.1027/. 1

Alessio Devoto, Maximilian Jeblick, and Simon Jégou. Expected attention: Kv cache compression by estimating attention from future queries distribution. *arXiv preprint arXiv:2510.00636*, 2025. 10

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS. 9

Yizhao Gao, Shuming Guo, Shijie Cao, Yuqing Xia, Yu Cheng, Lei Wang, Lingxiao Ma, Yutao Sun, Tianzhu Ye, Li Dong, Hayden Kwok-Hay So, Yu Hua, Ting Cao, Fan Yang, and Mao Yang. Sparse attention adaptation for long reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=c5BOcHM6J8. 1, 2, 3, 5, 10

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uNrFpDPMyo. 1

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. Chatglm: A family of large language models from glm-130b to glm-4 all tools, 2024. 6

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a. 1

Tianyu Guo, Druv Pai, Yu Bai, Jiantao Jiao, Michael I. Jordan, and Song Mei. Active-dormant attention heads: Mechanistically demystifying extreme-token phenomena in LLMs. In *The Second Conference on Parsimony and Learning (Recent Spotlight Track)*, 2025b. URL https://openreview.net/forum?id=Zx6WUbE9J7. 4

Zhenyuan Guo, Tong Chen, Wenlong Meng, Chen Gong, Xin Yu, Chengkun Wei, and Wenzhi Chen. Dynamic thinking-token selection for efficient reasoning in large reasoning models. *arXiv preprint arXiv:2601.18383*, 2026. 10

Zhiyu Guo, Hidetaka Kamigaito, and Taro Watanabe. Attention score is not all you need for token importance indicator in KV cache reduction: Value also matters. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21158–21166, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1178. URL https://aclanthology.org/2024.emnlp-main.1178/. 4, 10, 18

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe. 6

HMMT. HMMT 2025, 2025. URL https://www.hmmt.org/. Accessed: 2025. 6

Coleman Richard Charles Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=0LXotew9Du. 10

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL. 6

Huiqiang Jiang, YUCHENG LI, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 10

Junhyuck Kim, Ethan Ewer, Taehong Moon, Jongho Park, and Dimitris Papailiopoulos. Not all bits are equal: Scale-dependent memory optimization strategies for reasoning models. *arXiv preprint arXiv:2510.10964*, 2025. 10

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgNKkHtvB. 1

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 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, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165. 9

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 1, 3, 5

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a. 10

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=JZfg6wGi6g. 10

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. In *International Conference on Machine Learning*, pages 32332–32344. PMLR, 2024b. 2, 8, 10

OpenAI. Learning to reason with LLMs. https://openai.com/index/learning-to-reason-with-llms, 2024. 1

Matanel Oren, Michael Hassid, Nir Yarden, Yossi Adi, and Roy Schwartz. Transformers are multi-state RNNs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18724–18741, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1043. URL https://aclanthology.org/2024.emnlp-main.1043/. 1

Qwen. QwQ: Reflect deeply on the boundaries of the unknown. https://qwenlm.github.io/blog/qwq-32b-preview/, November 2024. 1

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98. 6

Luka Ribar, Ivan Chelombiev, Luke Hudlass-Galley, Charlie Blake, Carlo Luschi, and Douglas Orr. SparQ attention: Bandwidth-efficient LLM inference. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 42558–42583. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ribar24a.html. 1

Utkarsh Saxena, Gobinda Saha, Sakshi Choudhary, and Kaushik Roy. Eigen attention: Attention in low-rank space for kv cache compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15332–15344, 2024. 10

Ayan Sengupta, Siddhant Chaudhary, and Tanmoy Chakraborty. Value-guided KV compression for LLMs via approximated CUR decomposition. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 2, 3, 5

Prajwal Singhania, Siddharth Singh, Shwai He, Soheil Feizi, and Abhinav Bhatele. Loki: Low-rank keys for efficient sparse attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=raABeiV71j. 10

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n. 1

Jiwon Song, Dongwon Jo, Yulhwa Kim, and Jae-Joon Kim. Reasoning path compression: Compressing generation trajectories for efficient LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 3, 9, 10

Zunhai Su and Kehong Yuan. KVSink: Understanding and enhancing the preservation of attention sinks in KV cache quantization for LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=gIqb6zWZoO. 8, 18

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=F7aAhfitX6. 4

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. 1

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. 4

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a. 2, 6

Lijie Yang, Zhihao Zhang, Zhuofu Chen, Zikun Li, and Zhihao Jia. Tidaldecode: Fast and accurate LLM decoding with position persistent sparse attention. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=EkfLaCJ7bk. 1

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020. 1

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. Turboquant: Online vector quantization with near-optimal distortion rate. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tO3ASKZlok. 10

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023. 1

Guangxiang Zhao, Junyang Lin, Zhiyuan Zhang, Xuancheng Ren, Qi Su, and Xu Sun. Explicit sparse transformer: Concentrated attention through explicit selection. *arXiv preprint arXiv:1912.11637*, 2019. 1

Appendix

A Variants for Capturing Value-State Magnitude and Variety 15
B Token Statistics and Hyperparameters 17
C Statistical Significance 17
D Additional Results 18
E Limitations 19
F Compute Resources 19
G Licenses 19
H Broader Impacts 19
⋆ NeurIPS Paper Checklist 20

A Variants for Capturing Value-State Magnitude and Variety

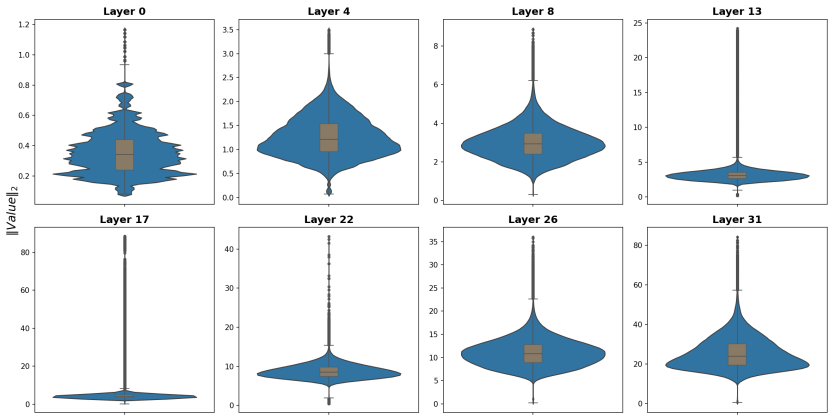

Figure 8: Layer-wise violin plots of $L_2(\boldsymbol{v}_i) := \|\boldsymbol{v}_i\|_2$, where the value vectors $\boldsymbol{v}_i$ are from the full KV cache of Qwen3-4B during GSM8K generation. The distribution shows outliers that have large L_2 norm.

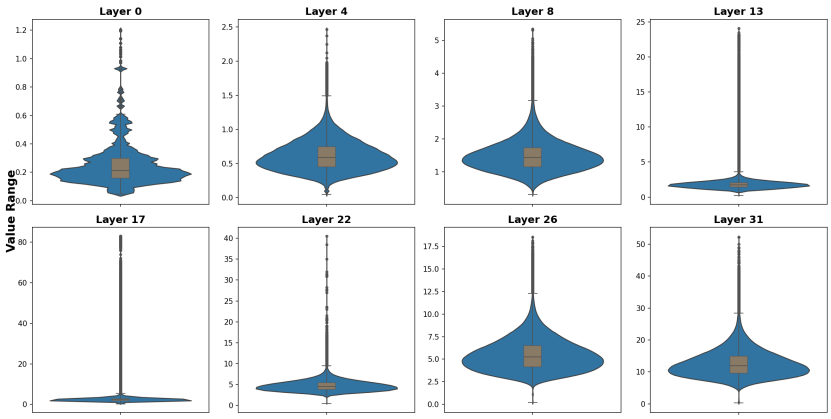

Figure 9: Layer-wise violin plots of $\mathrm{Range}(\boldsymbol{v}_i)$, where the values $\boldsymbol{v}_i$ are from the full KV cache of Qwen3-4B during GSM8K generation. The distribution shows outliers that have large Range.

In this section, we explore different ways to compute the magnitude and variety of a value state $\boldsymbol{v} \in \mathbb{R}^d$ in KV cache: (1) $L_2(\boldsymbol{v}) = \sqrt{\sum_{j=1}^{d}(v_j)^2}$, (2) $\mathrm{Range}(\boldsymbol{v}) := \max_{j\in[d]} v_j - \min_{j\in[d]} v_j$, and (3) $\mathrm{Var}(\boldsymbol{v}) = \frac{1}{d}\sum_{j=1}^{d}(\boldsymbol{v}_j - \mu)^2$, where $\mu = \frac{1}{d}\sum_{j=1}^{d} v_j$.

First, we extract the value vectors $\boldsymbol{v}$ from the full KV cache during the generation of Qwen3-4B on GSM8K. Figures 8 and 9 present the layer-wise distributions of $L_2(\boldsymbol{v})$ and $\mathrm{Range}(\boldsymbol{v})$, respectively.

The violin plots show prominent outliers that have large $L_2(\boldsymbol{v})$ and $\mathrm{Range}(\boldsymbol{v})$ in different layers. We also compute the Pearson correlation between $L_2(\boldsymbol{v})$ and $\mathrm{Range}(\boldsymbol{v})$ and found that the two are highly correlated (> 0.8) over layers.

Next, we study whether preventing these outlier value states from eviction can maintain the accuracy under KV cache compression. We apply the framework introduced in Section 3 that pre-allocates dedicated value slots in the token budget. To decide which entries to keep in the slots, we experiment with the three value-scoring variants: keeping values with the largest $L_2(\boldsymbol{v})$, $\mathrm{Range}(\boldsymbol{v})$, or $\mathrm{Var}(\boldsymbol{v})$, respectively.[6] For the remaining budget, we apply stochastic sampling on top of the attention scores in SnapKV as described in Section 3. In Table 3, we compare the value-scoring variants on GSM8K and AIME25. The three variants demonstrate comparable accuracy gains over the SnapKV (attention-only) baseline.

Method	**GSM8K**	**AIME25**
SnapKV	64.25	45.80
+ $L_2(\boldsymbol{v})$	84.55	59.17
+ $\mathrm{Range}(\boldsymbol{v})$	84.45	59.19
+ $\mathrm{Var}(\boldsymbol{v})$	84.00	57.71

Table 3: Impact of value-state scoring variants on KV cache eviction accuracy. We report the results on Qwen3-4B at $4\times$ compression. The three variants, which capture different aspects of value-state magnitude and variety, achieve comparable performance gains over the SnapKV baseline.

We choose $\mathrm{Range}(\boldsymbol{v})$ as the value scoring function for our main approach due to its connection with quantization; specifically, $\mathrm{Range}(\boldsymbol{v})$ captures the extreme values $\max \boldsymbol{v}$ and $\min \boldsymbol{v}$ in the vector, which is associated with the *outlier* challenge in LLM quantization (Dettmers et al., 2022). We show the importance of large-Range value states in Section 3.1, where evicting them causes the model to enter a nonsensical reasoning loop:

[Example 1] Okay, let's see. Carla is trying to download a 200 GB file… First, let me figure out what 40% of the file is.... But wait, does the restart happen at the 80 GB mark, ... Wait, so does that mean she has to start over?… so she has to download the entire 200 GB again? Or does she have to restart... Wait, the total file is 200 GB? Wait, no. Wait, the problem says that the download is a 200 GB file? Wait, no, the problem says that the file is 200 GB? Wait, no. Wait, the problem says... Wait, no. Let me check again…

[Example 2] Okay, let's see. Janet has these ducks that lay 16 eggs every day. … **9 times 2 is 18. Yeah, that seems right**. But let me make sure I didn't misinterpret the question. It says she bakes muffins "with four." Does that mean four eggs per muffin or four muffins using four eggs? … Maybe it's four eggs? … Alternatively, maybe "with four" refers to four muffins? But that would be a bit unclear… Alternatively, maybe she uses four eggs per muffin? … Alternatively, maybe "with four" is a typo or mistranslation… Alternatively, maybe the problem is that she uses four eggs per day for muffins. So, 3 for breakfast, 4 for muffins, total 7, and 16 - 7 = 9. Alternatively, maybe the "with four" is per day. So, she uses four eggs for muffins each day. So, 3 + 4 = 7... Wait, no. Let me check again…

Figure 10: Examples of model outputs on GSM8K when evicting large-Range value states. In the first example, the model starts with reasonable thinking traces but falls into a repetitive loop, never reaching the correct answer. In the second example, the model generates the correct answer once (**18**), but then gets stuck in an endless loop of self-reflection and fails to reach a conclusion.

[6] Here, Range is the same as VASE-ATTNV in the main content.

B Token Statistics and Hyperparameters

	Prompt Tokens	Gen Tokens
AIME25	198	17,790
AIME26	148	16,642
HMMT25	127	18,189
MATH	94	5,281
LiveCodeBench-v6-medium	557	11,088

Table 4: The average token counts of prompt (prefill) and generation (decode) of reasoning tasks. The **Gen Tokens** column represents the average number of tokens generated by the full Qwen3-4B model. The decode phase yields a significantly higher number of tokens than the prefill phase.

Table 4 lists the average token counts of the prefill and decode phase of Qwen3-4B, respectively, showing that reasoning models may generate over $10{,}000$ tokens at decode steps for a math question that has fewer than 200 prefill tokens. This intensifies the memory-bound constraints of the decode phase, motivating our approach to bound KV cache memory usage to a fixed cost.

Based on the statistics of Table 4, we set the token budget $K = 4096$ for AIME25, AIME26, and HMMT25, $K = 1024$ for MATH, and $K = 2048$ for GPQA-Diamond, which roughly translates to a $4\times$ KV cache compression. For LiveCodeBench, we set $K = 2048$, which is $\sim 5\times$ compression.

All the eviction methods are training-free but have hyperparameters. We choose the best hyperparameters for each method on GSM8K and then apply them to our main experiments (Section 4.2) without further tuning. For R-KV, we experiment with different redundancy hyperparameters, $\lambda = \{0.1, 0.5, 0.9\}$, and set $\lambda = 0.5$. We set the Gaussian matrix rank $r = 20$ for CurDKV and VASE-DKV, and the value reservation budget $N_v = K/4$ for VASE-ATTNV.

C Statistical Significance

Method	AIME25	AIME26	HMMT25	GPQA-D	MATH
R-KV	54.6 ± 7.4	60.2 ± 7.4	44.6 ± 5.7	59.1 ± 3.0	86.1 ± 1.3
CurDKV	53.8 ± 7.6	61.3 ± 7.3	39.6 ± 5.3	48.5 ± 3.1	75.4 ± 1.7
VASE-DKV	63.3 ± 7.5	68.8 ± 6.9	49.4 ± 5.4	56.8 ± 3.0	86.5 ± 1.3
VASE-ATTNV	64.0 ± 7.3	72.3 ± 6.5	50.0 ± 5.6	57.4 ± 3.0	85.4 ± 1.4

Table 5: Reasoning-task accuracy (%) with standard errors for KV cache eviction methods on Qwen3-14B under $\sim 4\times$ cache compression.

Table 5 shows the average accuracy $\pm$ standard errors for KV cache eviction methods on Qwen3-14B under $\sim 4\times$ cache compression. For each problem indexed $s \in \{1, \ldots, S\}$, we sample R independent generations and compute the per-problem pass@1 accuracy:

$$p_s = \frac{1}{R} \sum_{r=1}^{R} \mathbf{1}[\text{sample } r \text{ of problem } s \text{ is correct}], \tag{8}$$

where $R = 16$ for datasets with fewer than 100 examples and $R = 8$ otherwise. The overall pass@1 accuracy and its standard error SE are then:

$$\bar{p} = \frac{1}{S} \sum_{s=1}^{S} p_s, \qquad \mathrm{SD} = \sqrt{\frac{1}{S-1} \sum_{s=1}^{S} (p_s - \bar{p})^2}, \qquad \mathrm{SE} = \frac{\mathrm{SD}}{\sqrt{S}}. \tag{9}$$

The variability captured by the standard errors reflects both the stochasticity of the model's sampling procedure and, for VASE methods, the randomness introduced by stochastic eviction. On tasks with larger standard errors (e.g., AIME25, AIME26), the small number of test problems (30 each) is the primary cause. While individual confidence intervals overlap due to small test sets.

D Additional Results

Recall that in Figure 3, we demonstrate that reserving N_v slots for large-range values greatly improves GSM8K accuracy. To verify the importance of these specific value states, we conduct an ablation study where we instead reserve N_v slots for randomly sampled KV pairs. Experimenting with $N_v = \{16, 32, 64\}$ yield accuracies of 65.6%, 63.4%, and 65.3%, respectively. These results are comparable to SnapKV (64.3%) but substantially underperform the green bars in Figure 3, confirming that replacing large-range values with random KV pairs is ineffective.

Table 6 shows the Qwen3-14B results of GPQA-Diamond under different token budgets. While R-KV outperforms our VASE methods at 2048 token budget, the accuracy gap closes at 4096 tokens ($\sim 2\times$ compression). In this setting, both R-KV and our methods nearly recover the performance of the full model.

Token Budget **Compression**	**2048** $4\times$	**4096** $2\times$	**Full** $1\times$
R-KV	59.09	63.32	65.25
CurDKV	48.48	60.54	
VASE-DKV	56.76	63.38	
VASE-ATTNV	57.39	63.07	

Table 6: GPQA-D accuracy for eviction methods on Qwen3-14B under different token budgets.

In Figure 11, we extract the full value cache during the generation of Qwen3-4B on GSM8K examples and plot the distributions of $\mathrm{Range}(\boldsymbol{v})$ over different chunks of consecutive tokens. We put the first four sink tokens (Su and Yuan, 2025) into an individual chunk, which shows distinct distributions compared to the other chunks. Except for the first layer (Layer 0), the sink tokens have a lower median $\mathrm{Range}(\boldsymbol{v})$, which corresponds to the value-state drain phenomena (Guo et al., 2024). Excluding the sink tokens, the distributions of $\mathrm{Range}(\boldsymbol{v})$ are consistent as the sequence length increases.

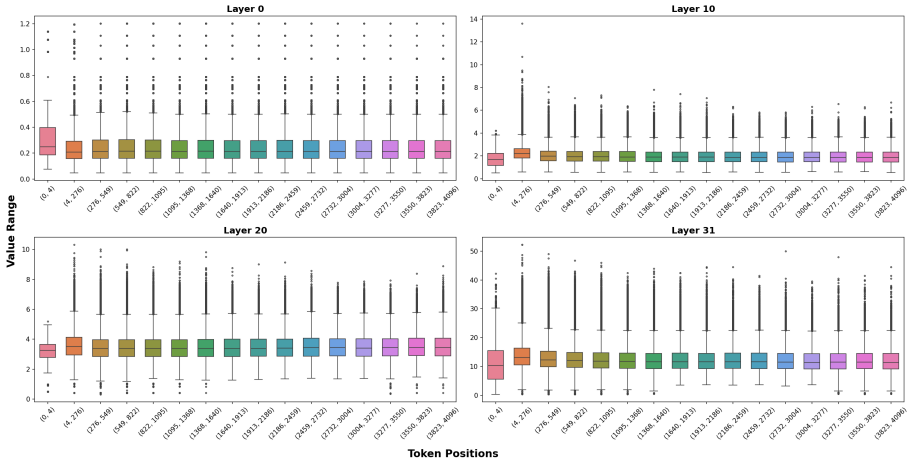

Figure 11: $\mathrm{Range}(\boldsymbol{v})$ over token position chunks, where each chunk consists of the value states of consecutive tokens. Boxplots illustrate the dynamic range (y-axis) of value states at specific layers. Token positions (x-axis) are bucketed to show how the range distribution evolves through the sequence. The first chunk contains sink tokens at position $(0, 4)$. Excluding the sink tokens, the distribution of $\mathrm{Range}(\boldsymbol{v})$ does not drift as the context window grows.

In Figure 12, we show the peak memory footprint of different methods across three KV-cache budgets $\{2048, 4096, 6144\}$ at 16K output tokens. Overall, VASE-DKV$<$ VASE-ATTNV$<$ R-KV$<$ Full, showing that our VASE methods are more memory-efficient than the R-KV baseline. At 32K output tokens, the memory footprints of different eviction methods remain the same under the same token budgets; however, the Full method can no longer fit a single GPU with 80GB memory.

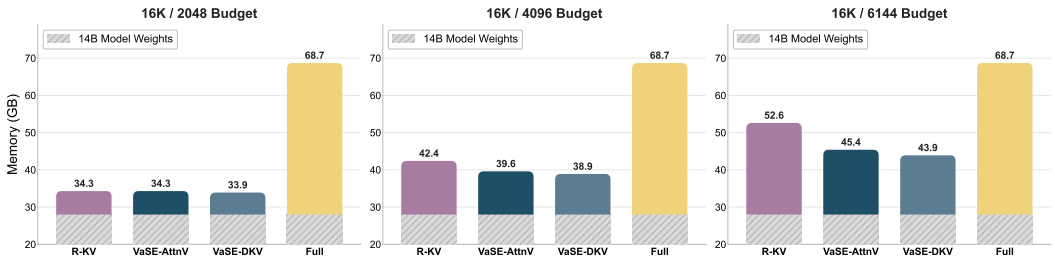

Figure 12: Peak GPU memory of Qwen3-14B at 16K output length across three KV-cache budgets $\{2048, 4096, 6144\}$. VASE-DKV is the most memory-efficient method at every budget.

E Limitations

Our evaluation focuses on reasoning models and the decode phase of generation, where eviction methods offer the greatest benefit. As a result, we do not evaluate on prefill-phase compression benchmarks. However, because our method is built on fundamental observations of value-state distributions and stochastic diversity, we believe the core methodology can be extended to the prefill phase for long-prompt compression.

We only implement our methods on Qwen3 models because our selection-based baseline SeerAttention-R only releases checkpoints for Qwen models. Nevertheless, since our value-scoring function relies on simple $\min$ and $\max$ statistics of the value cache, our method is model-agnostic and can be easily extended to other LLM architectures.

Lastly, while our findings link value-state magnitude to quantization errors, we focus on sparse attention, and quantization method development is beyond the scope of this paper. Future research could explore an integrated framework that combines KV cache eviction with outlier-aware quantization to push the limits of compression ratio.

F Compute Resources

Our experiments were conducted on NVIDIA A100 and H100 GPUs. Every experiment can be run on a single GPU with 80GB memory. Without PagedAttention and continuous batching, each task takes 10-72 GPU hours, depending on the dataset size and the average number of generated tokens.

G Licenses

As shown in Table 7, we include licenses for any artifacts used in this work. Copyright © MAA indicates that AIME problems are copyrighted by the Mathematical Association of America and are used here for evaluation purposes only.

H Broader Impacts

This work proposes a training-free KV cache eviction method that reduces the memory footprint of reasoning models during inference. The primary positive impact is enabling more efficient deployment of large language models, reducing hardware requirements and energy consumption. This can broaden access to reasoning-capable models for resource-constrained practitioners and organizations.

As a general-purpose inference optimization, VASE does not introduce new model capabilities or alter model outputs beyond the approximation inherent in cache compression. It therefore does not directly raise new ethical concerns beyond those already associated with the underlying language models. We do not foresee specific negative societal impacts arising from this work.

Asset	License	Source
Models		
Qwen3-4B	Apache 2.0	huggingface.co/Qwen/Qwen3-4B
Qwen3-14B	Apache 2.0	huggingface.co/Qwen/Qwen3-14B
Datasets & Benchmarks		
GSM8K	MIT	github.com/openai/grade-school-math
MATH	MIT	github.com/hendrycks/math
GPQA-Diamond	CC-BY-4.0	huggingface.co/datasets/Idavidrein/gpqa
LiveCodeBench-v6	MIT (code) / CC (data)	github.com/LiveCodeBench/LiveCodeBench
AIME 2025 / 2026	Copyright © MAA	artofproblemsolving.com/wiki
HMMT 2025	CC-BY-NC-SA 4.0	huggingface.co/datasets/MathArena/hmmt_feb_2025 huggingface.co/datasets/MathArena/hmmt_nov_2025
Code & Baselines		
SnapKV	CC-BY-4.0	openreview.net/forum?id=poE54GOq2l
R-KV	CC-BY-4.0	openreview.net/forum?id=2jwAjomEDB
CurDKV	CC-BY-4.0	openreview.net/forum?id=klmc4fwPLd
SeerAttention-R	CC-BY-4.0	openreview.net/forum?id=c5BOcHM6J8
HQQ	Apache 2.0	github.com/dropbox/hqq

Table 7: Assets and their licenses.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

Answer: [Yes]

Justification: Our claims are backed by our experimental results in § 4.

Guidelines:

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

2. **Limitations**

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

Answer: [Yes]

Justification: We talk about limitations in § E.

Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: [N/A]

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.

- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We disclose all of our experiment details in § 4 and in Appendix § B.

   Guidelines:

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

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

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

   Answer: [Yes]

   Justification: We will release codes upon acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: All experiment details are disclosed in § B.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We provide standard errors in Table 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).

- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We talk about compute details in § F.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The authors have read the NeurIPS Code of Ethics and made sure the paper follows the NeurIPS Code of Ethics in every aspect.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss broader impacts in § H.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This paper is about KV cache eviction which works on existing open-weight models, where we believe the safeguards have been done by the model providers.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We talk about licenses and datasets in § G.

    Guidelines:

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

13. **New assets**

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

    Answer: [N/A]

Justification: [N/A]

Guidelines:

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

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

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

Answer: [N/A]

Justification: [N/A]

Guidelines:

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

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

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

Answer: [N/A]

Justification: [N/A]

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: LLMs are used only for writing, editing, or formatting purposes.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.