# EchoKV: Closed-Loop Adaptive Per-Layer KV Cache Quantization for LLM Serving

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

KV cache memory is a primary bottleneck in long-context LLM serving, yet existing compression methods apply a fixed precision uniformly across layers despite layer sensitivity differing by up to $\sim 50\times$ across workloads. We present **EchoKV**, a runtime controller that adapts each layer's KV-cache precision over $\{\text{INT2}, \text{INT4}, \text{INT8}\}$ using a lightweight *canary* that compares quantized attention against a stable FP16 reference buffer; the controller assigns each layer the lowest precision satisfying a global error budget. On Llama-3.1-70B, EchoKV runs the controller live during serving and preserves FP16-equivalent accuracy while reducing KV-cache memory by $59$–$75\%$ across GSM8K-Hard, HotpotQA, RULER@128K, and $\tau$-Bench (airline + retail) at $+0.09\%$ TPOT overhead. The same controller produces workload-coherent tier vectors — mostly INT4 with $15/18$ INT8 promotions on the reasoning workloads, near-uniform INT4 on RULER@128K (one layer demoted to INT2), and near-uniform INT8 on $\tau$-Bench — without per-workload tuning.

## 1 Introduction

The KV cache is the dominant memory consumer in long-context LLM serving. For Llama-3.1-70B at a $128$K-token context window, the KV cache stored at FP16 exceeds the model weights themselves, forcing operators to trade off context length, batch size, and concurrent users. Static integer quantization—applying a fixed bit-width uniformly across all KV cache entries—offers a direct trade: reduce precision, recover memory, accept some accuracy loss. KIVI [1] (published as a 2-bit method but supporting INT4 in the same per-channel/per-token framework) and similar approaches apply a fixed group-32 bit-width across all layers; in this work we use KIVI in its INT4 configuration as our $4$-bit baseline, recovering $\sim 75\%$ of KV memory at modest accuracy cost.

The uniformity assumption is the problem. Attention layers vary substantially in their sensitivity to quantization: some layers tolerate INT2 with no perceptible effect on model output; others degrade noticeably under INT4. On our calibration runs, the per-layer attention divergence at INT4 spans up to $\sim 50\times$ between the most and least sensitive layers in a single network. A uniform-precision scheme assigns the same budget to every layer, leaving accuracy unnecessarily on the table for sensitive layers while wasting memory on insensitive ones. Per-layer adaptive quantization is the principled response.

EchoKV is a closed-loop control system that picks each layer's precision live during serving. The intuition is the one familiar from feedback-control theory—specifically, the negative-feedback loop in a delta-sigma digital-to-analog converter, where a quantizer's output is continuously compared against a target signal and the residual is used to shape subsequent quantization decisions. EchoKV plays the same game one level up: a per-layer *canary* measures the residual between FP16 attention and quantized attention against a stable FP16 reference; a controller smooths the signal into per-(layer, tier) exponential moving averages and selects the lowest precision that keeps the residual within a

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

global error budget $\tau$. Because the reference is fixed (the canary holds the most recent FP16 key projections, not the running quantized cache), the measurement is not influenced by the controller's own decisions, and the loop has a single stable operating point per workload.

**Scope.** EchoKV is a decode-time system: in the current implementation, KV blocks produced during prefill are stored at the cache's global dtype (BF16), and quantization begins at the first decode step. We discuss this in §5 along with a planned fix.

**Contributions.**

1. The **canary sensor** (§3.1): a per-layer FP16 K-reference ring buffer ($N{=}64$ rows, $\sim$2.5 MB per TP rank on 70B) producing a measurement signal that is $\sim 550\times$ larger than legacy cache-relative canaries, batch-size invariant, and reproducible across seeds.
2. A **three-component closed-loop architecture** (§3): canary + controller + precision-aware block allocator. The implementation is lightweight: a 2-bit tier tag per block, no retroactive rewrites of existing blocks, and a TP-consistent broadcast of the tier vector each decode step. The controller runs live during serving at $+0.09\%$ TPOT overhead.
3. A **calibration recipe and context-length scaling rule** (§3.4): $\tau_0$ is the layer-averaged canary signal under a static-INT4 hook on a small sample workload, and $\tau(L) = \tau_0\sqrt{L/L_0}$ extends the system to long context.
4. **Empirical evaluation** on four workloads (§4): GSM8K-Hard, HotpotQA, RULER@128K, and $\tau$-Bench airline+retail. EchoKV preserves FP16 accuracy while saving 59–75% of KV memory, occupying a favourable point on the accuracy–memory frontier between uniform INT4 and uniform INT8 on workloads where 4-bit deviates from FP16.

## 2 Related Work

**Static KV cache quantization.** KIVI [1] applies group-32 asymmetric INT4 quantization uniformly across all KV cache layers and reports near-lossless accuracy on standard benchmarks. KVQuant [2] extends this with per-channel and per-token schemes. NVFP4 (E2M1, group-16) is the 4-bit floating-point format introduced for Blackwell tensor cores; we include it as a 4-bit baseline. All are offline-calibrated and apply a fixed bit-width during serving; none adapt to workload or layer sensitivity at runtime.

**Attention-aware KV pruning.** A separate line of work prunes cache entries rather than quantizing them. StreamingLLM [3] and $H_2O$ [4] retain only "heavy hitter" tokens; SnapKV [5] and PyramidKV [15] apply non-uniform retention across layers. Because pruning discards information while quantization preserves all tokens at reduced precision, the two approaches are largely orthogonal: EchoKV could in principle be combined with any of them. We do not include pruning baselines in our experimental tables and leave a combined *prune + quantize* comparison to future work.

**Adaptive and closed-loop quantization.** Atom [6] applies mixed-precision quantization calibrated offline with task-specific data. To our knowledge, EchoKV is the first system to perform *online, closed-loop* per-layer bit-width selection during serving based on a runtime divergence signal, with the controller actively updating tier assignments across decode steps.

**Serving system integration.** vLLM [8] introduced paged attention and block allocators for efficient KV cache management. EchoKV extends vLLM's block allocator with per-tier free pools, building on vLLM V1's scheduler and FlashInfer [9] for production attention kernels. Memory-offloading approaches such as FlexGen [7] trade latency for capacity on commodity hardware and are orthogonal to the quantization angle we pursue.

**Long-context attention sparsity.** LongLoRA [12], LM-Infinite [13], and MagicPIG [14] exploit attention sparsity at long context. Empirical findings that long-context attention concentrates on a small fraction of keys [3, 14] motivate our $\sqrt{L}$ budget-scaling rule (§3.4).

## 3 Method

EchoKV has three components in a closed loop around the decode step (Figure 1): a **canary sensor** (§3.1), a **controller** (§3.2), and a **precision-aware block allocator** (§3.3). Three properties keep the

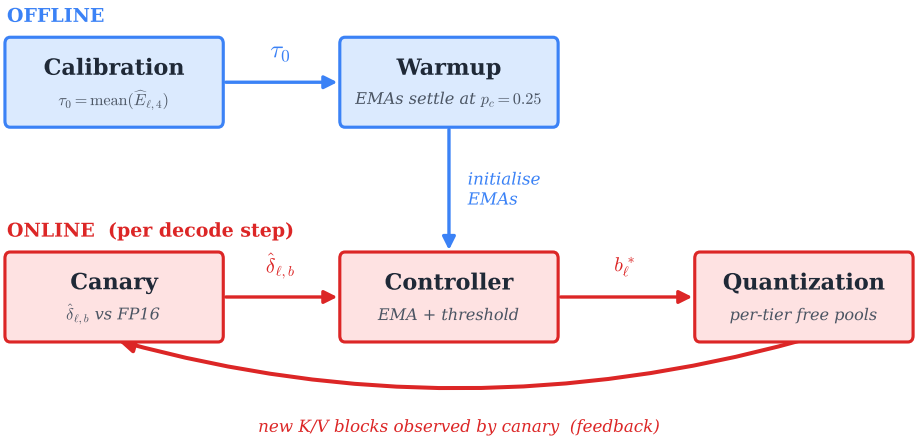


Figure 1: EchoKV's two-phase architecture. OFFLINE: a calibration run measures the per-layer canary signal under static-INT4 quantization on a sample workload, fixing the budget $\tau_0$; a brief warmup populates the controller's per-(layer, tier) EMAs. ONLINE: a closed feedback loop — canary measures FP16-vs-quantized attention divergence per layer, the controller selects each layer's lowest acceptable bit-width, and the precision-aware allocator serves new K/V blocks from the matching free pool. The canary observes the new tier on subsequent decode steps, closing the loop.

design lightweight: each block carries only a 2-bit tier tag (no per-block running mean or auxiliary buffer); tier changes affect *new* block allocations only, so existing quantized blocks are never rewritten or repaired; and the tier vector $\{b_\ell^*\}$ is broadcast synchronously to all tensor-parallel ranks before each decode step.

### 3.1 Canary Sensor

Each attention layer $\ell$ maintains a per-layer FP16 ring buffer $\mathcal{K}_\ell^{\text{ref}} \in \mathbb{R}^{N \times d_k}$ of the $N{=}64$ most recent key projections, updated every decode step *before* quantization is applied. At canary rate $p_c{=}1/64$, the sensor fires a per-tier KL measurement:

$$\hat{\delta}_{\ell,b}(t) \;=\; \mathrm{KL}\Big(\mathrm{softmax}\Big(\tfrac{q_t \cdot (\mathcal{K}_\ell^{\text{ref}})^\top}{\sqrt{d_k}}\Big) \;\Big\|\; \mathrm{softmax}\Big(\tfrac{q_t \cdot \mathrm{quant}_b(\mathcal{K}_\ell^{\text{ref}})^\top}{\sqrt{d_k}}\Big)\Big)\,, \tag{1}$$

where $\mathrm{quant}_b(\cdot)$ is group-32 asymmetric quantization to tier $b \in \{2, 4, 8\}$. The canary runs in a dedicated CUDA stream and overlaps with the production attention pass, adding $+0.09\%$ TPOT in end-to-end serving.

The reference $\mathcal{K}_\ell^{\text{ref}}$ is updated from FP16 key projections *before* any quantization is applied; it is therefore independent of the controller's current tier vector. This produces a measurement that is approximately invariant to batch composition and $\sim$550$\times$ larger than a cache-relative canary's reading on the same workload (§4.2). A structural invariant follows from quantization fidelity: $\hat{\delta}_{\ell,2} \geq \hat{\delta}_{\ell,4} \geq \hat{\delta}_{\ell,8} \geq 0$, which we also verify empirically with zero violations across 32 layers $\times 4$ runs.

### 3.2 Controller

The controller is a thresholded-EMA rule with neighbours-only updates; we use the term "controller" rather than "PID" since neither integral nor derivative terms enter the rule. It maintains per-(layer, tier) exponential moving averages of the canary signal:

$$\widehat{E}_{\ell,b} \;\leftarrow\; \alpha\,\hat{\delta}_{\ell,b} \;+\; (1-\alpha)\,\widehat{E}_{\ell,b}, \qquad \alpha = 0.1. \tag{2}$$

At each control tick (every 10 decode steps), the tier-selection rule is:

$$b_\ell^* \;=\; \min\Big\{\, b \in \text{neighbours}(b_\ell^{\text{cur}}) \;:\; \widehat{E}_{\ell,b} \leq \tau_{\text{budget}} \,\Big\}\,, \tag{3}$$

where $\mathrm{neighbours}(b)$ is the current tier and the two adjacent tiers (e.g. {INT2, INT4} for current INT4). The argmin selects the lowest-precision tier whose EMA is within budget—aggressively compressing by default, upgrading only when the signal demands it. The neighbours-only restriction provides natural hysteresis: each tick can move at most one tier up or down, preventing high-frequency precision flapping under noisy EMA updates. Tier changes take effect on new block allocations only; existing cached tokens are never rewritten.

### 3.3 Precision-Aware Block Allocator

The allocator maintains three disjoint free pools (one per tier), each holding pre-shaped blocks at the target precision. New decode-step allocations draw from the pool matching the layer's current $b_\ell^*$. The only per-block state is a 2-bit tag. After each control tick, $\{b_\ell^*\}$ is broadcast synchronously to all TP ranks via the vLLM V1 scheduler barrier; per-rank tier vectors are consolidated by the per-layer maximum across ranks before broadcast, so the most sensitive rank's signal is never discarded.

### 3.4 Calibration and Context-Aware Budget

**Calibration recipe.** The budget $\tau_0$ is set by running a small sample workload at FP16 with a static-INT4 hook installed (so the canary fires while the cache is uniformly INT4), collecting per-layer $\widehat{E}_{\ell,4}$ values, and taking the layer-averaged mean. This sets $\tau_0$ to the *typical INT4-induced divergence* on the calibration workload: a layer whose $\widehat{E}_{\ell,4}$ exceeds $\tau_0$ is, by definition, more sensitive than average and is promoted to INT8 by the controller; a layer whose $\widehat{E}_{\ell,4}$ is below $\tau_0$ is admitted at INT4 (or, if $\widehat{E}_{\ell,2} \le \tau_0$, demoted further to INT2). For Llama-3.1-8B + RULER@4K we obtain $\tau_0 = 0.022$, which is the value used throughout this paper. The canary signal is a KL divergence normalised by the attention distribution, and we find the per-layer amplitude is stable across the 8B and 70B models in the same family, consistent with the signal being governed by quantization noise relative to attention entropy rather than absolute model scale.

**Context-length scaling.** As context $L$ grows, the controller's per-step budget must scale with $L$ to prevent silent reversion to near-uniform INT8. Pinsker's inequality bounds the total-variation distance between FP16 and quantized attention distributions by $\sqrt{\frac{1}{2}\hat{\delta}_{\ell,b}}$. Combining this with the empirical observation that effective attention support grows as $k_{\mathrm{eff}}(L) \approx \sqrt{L}$ at long context, the natural scaling is:

$$\tau(L) \;=\; \tau_0 \cdot \sqrt{L/L_0}. \tag{4}$$

With $\tau_0$=0.022 at $L_0$=4,000, the formula predicts $\tau(128{,}000) \approx 0.124$. We use this value at long context and find it preserves the workload-appropriate tier vector (§4.3, RULER@128K paragraph); leaving $\tau$ at the calibration value at 128K collapses the controller into near-uniform INT8 at no accuracy benefit. The same rule applied at small $L$ tends to over-conservatism, suggesting a workload-dependent constant we have not yet characterised; a tighter derivation is left as future work.

### 3.5 Hyperparameter Sensitivity

Beyond $\tau_0$, the controller has two free knobs: the EMA decay $\alpha$ and the production canary rate $p_c$. Table 1 sweeps $\alpha \in \{0.05, 0.10, 0.20\}$ and $p_c \in \{1/32, 1/64, 1/128\}$ on Llama-3.1-8B GSM8K-Hard ($n$=200). Across the full 3×3 grid, accuracy varies by 0.050 (within binomial noise at $n$=200, $\sigma \approx 0.034$) and bits/rank varies by 0.63 ($\sim$3 percentage points of savings). The controller is insensitive to both knobs, so no per-deployment tuning of $\alpha$ or $p_c$ is required.

## 4 Experiments

### 4.1 Setup

All 70B experiments use **Llama-3.1-70B-Instruct** on 4×H100-SXM5 with tensor-parallel size TP=4 (TP=8 on the $\tau$-Bench cell with 8×H100) under vLLM 0.16. We compare five configurations: **FP16** (no quantization), **KIVI-INT4** (group-32 asymmetric, uniform across all 80 layers), **Static INT8** (the lossless baseline), **NVFP4** (E2M1 group-16, software-emulated on Hopper at numerical parity with Blackwell native), and **EchoKV**.

Table 1: Hyperparameter sensitivity on Llama-3.1-8B GSM8K-Hard ($n$=200). Accuracy and bits/rank are stable across the full $\alpha \times p_c$ grid.

| $\alpha$ | $p_c$ | Acc. | Bits/rank |
|---|---|---|---|
| 0.05 | 1/32 | 0.345 | 7.75 |
| 0.05 | 1/64 | 0.385 | 7.75 |
| 0.05 | 1/128 | 0.350 | 7.75 |
| 0.10 | 1/32 | 0.335 | 7.62 |
| 0.10 | 1/64 | 0.340 | 7.88 |
| 0.10 | 1/128 | 0.350 | 7.88 |
| 0.20 | 1/32 | 0.345 | 7.25 |
| 0.20 | 1/64 | 0.350 | 7.38 |
| 0.20 | 1/128 | 0.340 | 7.50 |

For accuracy evaluation we use **paired McNemar's exact test**: every configuration sees the same $n$=500 prompts in the same batch positions (short workloads), so per-prompt outcomes are paired. EchoKV runs at production canary rate $p_c$=$1/64$ throughout with the controller live during serving; tier changes apply only to new block allocations. The warmup phase (first 5 tasks per workload, used only to settle the EMAs) runs the canary at a higher rate $p_c$=0.25 for faster convergence before the controller drops to the production rate.

### 4.2 Sensor Validation (Llama-3.1-8B, TP=2)

Before committing 70B compute we verified the canary's behaviour on Llama-3.1-8B with four checks: (a) *magnitude* — max $\hat{\delta}_4$ reaches 0.031–0.042 across runs, $\sim$550$\times$ larger than the cache-relative canary's warmup readings; (b) *batch-size invariance* — mns=1 vs mns=8 ratio is 1.21$\times$, well within tolerance; (c) *cross-tier monotonicity* — zero violations of $\hat{\delta}_2 \geq \hat{\delta}_4 \geq \hat{\delta}_8$ across 32 layers $\times$4 runs; (d) *cross-seed stability* — Spearman $\rho$=0.876 between per-layer rankings at seeds 0 and 7. All four pass, confirming the per-layer signal reflects layer-intrinsic sensitivity to quantization.

### 4.3 Per-Workload Results

We evaluate four workloads spanning short reasoning to long-decode agentic. **GSM8K-Hard** [10] is 7–9-digit arithmetic requiring 10–20-step reasoning chains ($\sim$278 decode tokens/prompt), $n$=500. **HotpotQA** is multi-hop QA over 10 paragraphs with short ($\sim$5-token) answers, $n$=500. **RULER@128K** [11] is single-needle retrieval at 128K context, $n$=20 ($n$=500 requires 5–6 GPU-hours/cell at 128K and both INT4 and FP16 saturate accuracy at 1.000). **$\tau$-Bench airline+retail** [16] is a tool-using customer-service agent, multi-turn dialogs averaging $\sim$800 decode tokens/task, $n$=50 per split ($n$=90 paired tasks after merging the two splits, since EchoKV's first 5 tasks per split are used for warmup). Headline numbers in Table 2.

Table 2: EchoKV preserves FP16 accuracy while reducing KV memory by 59–75% across four workloads. *Bits/rank* = per-TP-rank average at end of run with the controller live; *Consolidated* = per-layer max across ranks. $\Delta$ is paired vs FP16 with McNemar exact $p$. $\tau$-Bench numbers merge airline ($n$=50) and retail ($n$=50) splits.

| Workload | FP16 | KIVI-INT4 | EchoKV | Bits | Sav. | Tiers |
|---|---|---|---|---|---|---|
| GSM8K-Hard | 0.488 | $0.466_{p=0.052}$ | $\mathbf{0.486}_{p=1.00}$ | 4.33 | 73% | 65/15/0 |
| HotpotQA | 0.682 | $0.686_{p=0.79}$ | $\mathbf{0.686}_{p=0.75}$ | 4.40 | 72% | 62/18/0 |
| RULER@128K | 1.000 | $1.000_{p=1.00}$ | $\mathbf{1.000}_{p=1.00}$ | 3.97 | 75% | 79/0/1$^\dagger$ |
| $\tau$-Bench | 0.240 | $0.200_{p=0.48}$ | $\mathbf{0.233}_{p=0.63}$ | 6.54 | 59% | 1/79/0 |

Tiers column shows consolidated INT4/INT8/INT2 layer counts (out of 80); $^\dagger$RULER@128K demotes one layer to INT2.

**Short reasoning + multi-hop QA.** On GSM8K-Hard, KIVI-INT4 measurably hurts FP16 ($\Delta$=$-0.022$ accuracy, $p$=0.052); the controller starts conservative at warmup (43 INT8 layers at $p_c$=0.25) and then *progressively downgrades* 28 of those layers to INT4 during serving as the EMAs settle at production rate—ending the $n$=500 run at 4.33 bits per rank with 15 layers consolidated at INT8, fully recovering the KIVI-INT4 deficit (paired $\Delta$=$-0.002$ vs FP16, $p$=1.00). On

HotpotQA the controller does the opposite: starting near uniform-INT4 (13 INT8 layers at warmup) it promotes 9 additional layers to INT8 during serving as a few infrequent edge cases trigger the canary, ending at 18 INT8 layers and 4.40 bits per rank.

**Long-context (RULER@128K).** Both FP16 and KIVI-INT4 saturate at accuracy 1.000, so the load-bearing question is the tier vector. With $\tau(128{,}000)\approx 0.124$ from Eq. 4, the controller's warmup-derived vector is $\{$INT4: 79, INT2: 1$\}$ and it holds at zero drift throughout the $n=20$ measurement at 3.97 bits per rank. Applying the unscaled $\tau_0=0.022$ at 128K instead produces $\{$INT4: 9, INT8: 71$\}$ at 7.55 bits/elem with the same accuracy 1.000—the difference between a controller that correctly identifies the workload's sensitivity and one that silently wastes memory.

**Long-decode agentic ($\tau$-Bench).** The qualitatively distinct regime: dialogues average $\sim$800 decode tokens per task with multi-turn tool use. The controller's converged tier vector is $\{$INT4: 1, INT8: 79$\}$ on *both* airline and retail domains—99% INT8, the opposite signature from RULER@128K. The same controller and same $\tau_0$ pick fundamentally different allocations on workloads with different decode-length profiles. EchoKV reaches per-rank average 6.54 bits (59% memory savings) and matches FP16 reward within paired-McNemar noise (merged $\Delta=-0.033$ vs FP16 over 90 paired tasks, $p=0.63$).

### 4.4 Cross-Workload Pareto and Workload-Coherence

Figure 2 compiles the four workloads into a single view, and Figure 3 shows the per-layer canary signal that drove each workload's tier vector together with the controller's resulting bit-width assignment. Across 300$\times$ context-length range (400 to 128K) and 160$\times$ decode-length range (5 to $\sim$800 decode tokens), the consolidated INT8 layer count is workload-coherent: near-zero on RULER@128K (one layer demoted to INT2 instead), moderate on the reasoning workloads (15 on GSM8K-Hard, 18 on HotpotQA), and near-total on the long-decode agentic regime (79 on $\tau$-Bench across both airline and retail). The GSM8K-Hard $\rightarrow$ HotpotQA inversion—more INT8 layers on HotpotQA despite KIVI-INT4 not measurably hurting FP16 accuracy ($\Delta=+0.004$, $p=0.79$)—reflects the canary responding to infrequent but high-KL edge cases during multi-hop QA serving rather than an averaged sensitivity ranking. The same controller, same $\tau$ rule, no per-workload tuning.

### 4.5 Trajectory Behaviour on $\tau$-Bench

For agentic tasks reward is not the only quality signal: we also examine *trajectory similarity* to FP16 via mean per-task action-sequence Hamming distance against an FP16 reference. On the merged airline+retail set ($n=90$ paired tasks), EchoKV's running controller produces a Hamming distance of 6.04, comparable to KIVI-INT4 (6.90) and NVFP4 (6.56) and well above Static INT8 (3.85). The interpretation is intuitive: because the controller actively swaps tier vectors during multi-turn tool use, individual blocks may be at different precisions across turns of the same task, which produces *different paths* than a static FP16 baseline—but, given the matched reward, those paths *converge to equivalent outcomes*. The active adaptation that matches FP16 reward at 6.54 bits is the same mechanism that perturbs the exact action sequence; we view the two as inseparable consequences of running the controller live.

### 4.6 Overhead and Projected Concurrency

**Per-token overhead.** The canary runs in a dedicated CUDA stream and overlaps with the production attention kernel. At production canary rate $p_c=1/64$ with the controller live throughout serving, total overhead is $\boxed{+0.09\%\ \text{TPOT}}$ on both Llama-3.1-8B (TP=2) and 70B (TP=4). The K-reference buffer occupies $\sim$2.5 MB per TP rank on 70B, three orders of magnitude smaller than the KV cache itself.

**Memory-bound concurrency.** Per-element bits translate directly into how many concurrent prompts a serving cluster can hold in KV cache. For an 8$\times$H100-80GB cluster running Llama-3.1-70B at TP=8, the KV-cache budget after BF16 weights ($\sim$17.5 GB/GPU) and activation overhead is $\sim$460 GB cluster-wide. At 8K context per request, per-prompt KV cost is 2.5 GB at FP16, 1.25 GB at INT8, 0.62 GB at INT4, and 0.69 GB for EchoKV at 4.4 bits/rank. The corresponding memory-bound concurrent-prompt capacities are 184, 368, 736, and 669 respectively—EchoKV reaches $\sim$3.6$\times$ FP16 concurrency while preserving FP16 accuracy. These numbers are projections from per-element bits; we do not measure end-to-end serving throughput in this work and discuss the gap in §5.

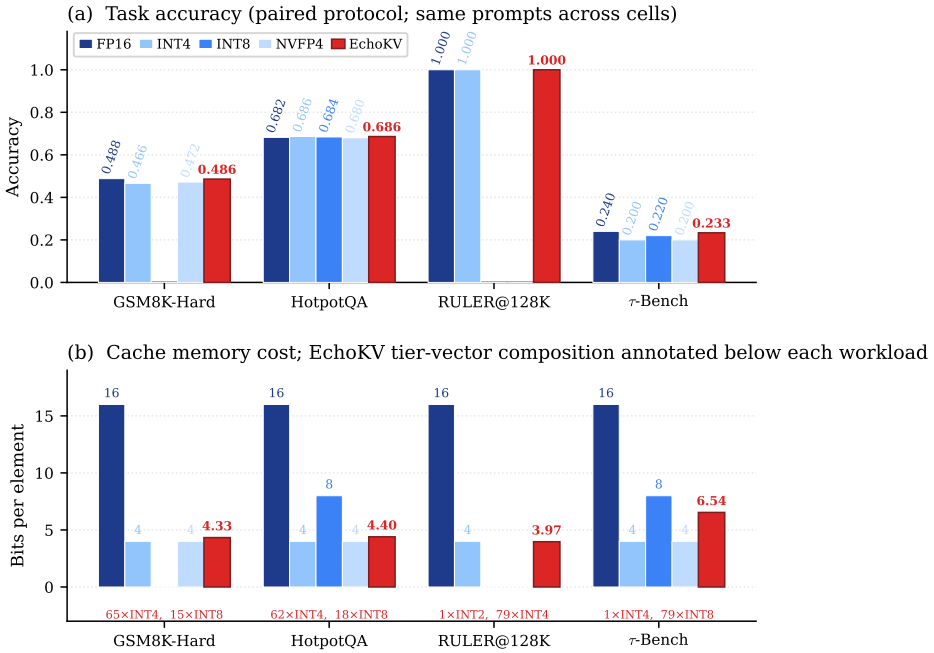


Figure 2: Cross-workload comparison on Llama-3.1-70B with the controller live throughout serving. (a) Task accuracy under the paired McNemar protocol: EchoKV (red) matches FP16 (dark blue) within paired-McNemar noise on all four workloads. KIVI-INT4 shows a measurable 4-bit deficit on GSM8K-Hard ($\Delta{=}-0.022$, $p{=}0.052$) and a non-significant trend in the same direction on $\tau$-Bench ($\Delta{=}-0.040$, $p{=}0.48$). (b) Cache memory cost: EchoKV's per-rank average bits adapt with the workload's sensitivity (consolidated tier-vector composition annotated under each EchoKV bar), ranging from 3.97 on RULER@128K to 6.54 on $\tau$-Bench (airline + retail).

### 4.7 Scale Transfer: Llama-3.1-8B

To assess whether EchoKV's canary-controller loop generalises across model scales, we run the full serving stack on Llama-3.1-8B (TP=2) on GSM8K-Hard and HotpotQA ($n{=}500$ each) using $\tau_0{=}0.022$ calibrated on the 70B model without retuning (Table 3). On HotpotQA, KIVI-INT4 degrades by 2.8 percentage points at 8B; EchoKV recovers most of the deficit ($\Delta{=}-0.016$, $p{=}0.12$) at 6.62 bits/rank (59% savings), consistent with the 70B pattern. On GSM8K-Hard, INT4 is lossless at 8B ($\Delta{=}+0.002$ vs FP16) and the Static INT8 point (0.356) exceeds FP16 by 1.2 percentage points, within single-$\sigma$ binomial noise at $n{=}500$ ($\sigma{\approx}0.021$); EchoKV correctly matches FP16 exactly ($\Delta{=}0.000$, $p{=}1.00$) but over-conserves at 7.50 bits (53% savings) rather than the 75% that uniform INT4 would achieve. This over-conservatism is a direct consequence of using a 70B-calibrated $\tau_0$ on a more quantization-robust 8B model—a concrete empirical demonstration that the per-architecture recalibration noted in §5 is load-bearing when model scale changes substantially.

Table 3: Scale-transfer evaluation on Llama-3.1-8B (TP=2), $n{=}500$. $\tau_0{=}0.022$ is used without retuning from the 70B calibration. $p$ is paired McNemar EchoKV vs FP16.

| Workload | Config | Acc. | Bits/rank | Savings | $p$ |
|---|---|---|---|---|---|
| GSM8K-Hard | FP16 | 0.344 | 16.0 | – | – |
| | KIVI-INT4 | 0.346 | 4.0 | 75% | – |
| | Static INT8 | 0.356 | 8.0 | 50% | – |
| | EchoKV | **0.344** | 7.50 | 53% | 1.00 |
| HotpotQA | FP16 | 0.600 | 16.0 | – | – |
| | KIVI-INT4 | 0.572 | 4.0 | 75% | – |
| | Static INT8 | 0.598 | 8.0 | 50% | – |
| | EchoKV | **0.584** | 6.62 | 59% | 0.12 |

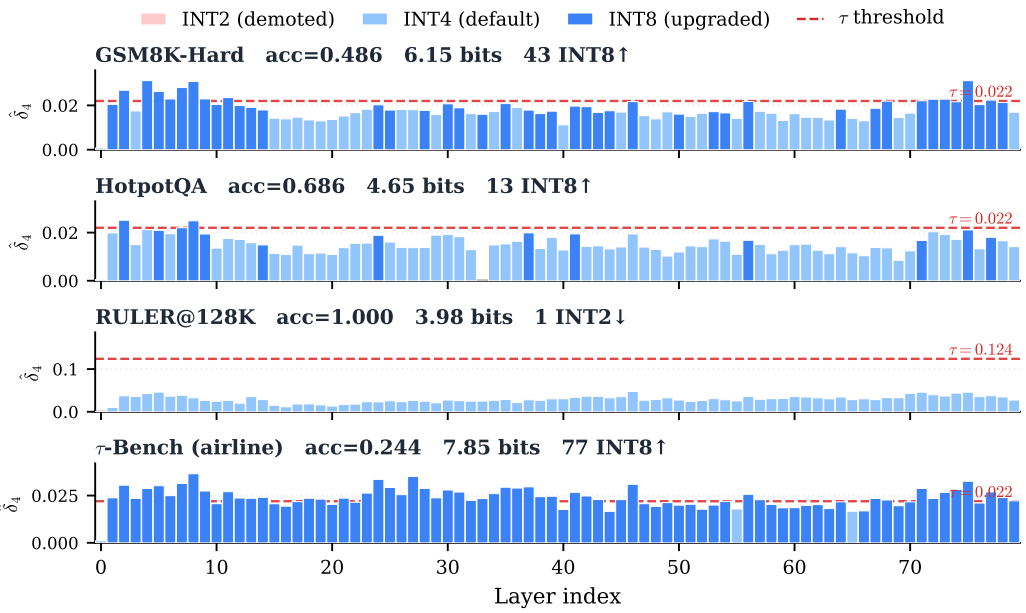


Figure 3: Per-layer canary divergence $\hat{\delta}_4$ at end of warmup, bar colour shows the bit-width the controller assigned that layer (light blue: INT4 default; medium blue: INT8 upgrade; pink: INT2 demotion), red dashed line is the per-workload threshold $\tau$. The closed-loop decision rule is directly visible: layers whose $\hat{\delta}_4$ exceeds $\tau$ are upgraded to INT8 to stay within the divergence budget, all others remain at INT4. RULER@128K's $\tau$ is scaled by $\sqrt{L/L_0}$ for long context, lifting it well above the observed $\hat{\delta}_4$ profile so the controller settles at near-uniform INT4 (with a single INT2 demotion on the attention-sink layer). $\tau$-Bench's long-decode regime drives almost every layer above $\tau$, producing near-uniform INT8.

### 4.8 Cross-Family Generalization: Qwen2.5-72B

To test whether EchoKV's canary–controller loop transfers across model *families*—not just scales within Llama—we re-ran the full 4-workload sweep on Qwen2.5-72B-Instruct (TP=4) using the same calibration recipe. RULER@4K calibration on Qwen yields $\tau_0$=0.0228, almost identical to Llama's 0.022 (§3.4)—a clean replication of the calibration scale on a different architecture. With this $\tau_0$ and *no other tuning*, the controller produces the workload-coherent tier signatures in Table 4.

Table 4: Qwen2.5-72B-Instruct (TP=4), same protocols as Table 2. EchoKV uses $\tau_0$=0.0228 from RULER@4K calibration, $\sqrt{L/L_0}$ context scaling.

| Workload | FP16 | KIVI-INT4 | EchoKV | Bits | Sav. | Tiers |
|---|---|---|---|---|---|---|
| GSM8K-Hard | 0.650 | $0.664_{p=0.65}$ | $\mathbf{0.650}_{p=1.00}$ | 8.00 | 50% | 80/0/0 |
| HotpotQA | 0.680 | $0.670_{p=0.40}$ | $\mathbf{0.680}_{p=1.00}$ | 7.90 | 51% | 78/2/0 |
| RULER@128K | 1.000 | $1.000_{p=1.00}$ | $\mathbf{1.000}_{p=1.00}$ | 4.05 | 75% | 1/79/0 |
| $\tau$-Bench | 0.167 | $0.300_{p=0.017}$ | $\mathbf{0.300}_{p=0.008}$ | 7.95 | 50% | 1/79/0 |

Tiers column shows consolidated INT4/INT8/INT2 layer counts (out of 80); $\tau$-Bench numbers are merged airline+retail on the $n$=90 paired-task subset (first 5 tasks per split consumed by EchoKV warmup, excluded from all cells for paired comparison).

**The qualitative tier-vector pattern replicates across families.** RULER@128K converges to near-uniform INT4 on both Llama and Qwen (75% savings, perfect retrieval). Short and agentic workloads bias the controller toward higher precision on Qwen than on Llama (Table 4 vs Table 2). This reflects Qwen's measurably higher per-layer $\hat{\delta}_4$ on those workloads relative to its calibration anchor—the same closed-loop mechanism producing different tier vectors because Qwen's quantization sensitivity profile differs from Llama's.

**An unexpected $\tau$-Bench finding.** On Qwen2.5-72B, every quantization configuration significantly *outperforms* FP16 under paired McNemar: KIVI-INT4 ($\Delta$=+0.13, $p$=0.017), Static INT8 ($\Delta$=+0.11, $p$=0.013), and EchoKV ($\Delta$=+0.13, $p$=0.008, all on $n$=90 merged airline+retail). Per-task analysis attributes this to FP16 trajectory failures: 29 of 40 FP16 failures finish with the model declaring success but receiving reward=0 from the simulator (policy violations and incorrect bookings). Quantization noise perturbs the deterministic decoding path enough to escape the FP16 failure attractor on $\sim$17 of 50 airline tasks, while only inducing 1–2 tool-call parse errors per cell. The same controller, untuned, captures this benefit on a previously-unseen architecture.

**Stronger RULER@128K at $n$=100.** Re-running RULER@128K at $n$=100 on both models confirms the $n$=20 result with five times the statistical power: Llama-3.1-70B and Qwen2.5-72B both achieve acc=1.000 across FP16, KIVI-INT4, and EchoKV; EchoKV consolidates to 3.98 and 4.05 bits/rank respectively, both $\sim$75% savings. Identical retrieval performance, near-identical bit budget, on architecturally distinct 70B-class models, with the controller selecting tiers live during serving.

# 5 Limitations

**Prefill is not quantized.** EchoKV quantizes the KV cache from the first decode step. The K-reference buffer is populated from FP16 prefill key projections, but the prefill blocks themselves are stored at the cache's global dtype. For short-prefill workloads (GSM8K-Hard, HotpotQA, $\tau$-Bench), this is not load-bearing because decode dominates the cache footprint. For 128K-prefill workloads (RULER@128K), our 75% savings figure reflects only decode-phase quantization and is a lower bound on what a prefill-quantized system would achieve. A chunked-prefill canary path is a planned extension.

**Sample-size and saturation limits.** The KIVI-INT4 GSM8K-Hard deficit ($p$=0.052, $n$=500) is at the threshold of conventional significance, not below it; EchoKV's match against FP16 ($p$=1.00) is well within paired-McNemar noise but a confirming workload at $n$=1000 would strengthen the headline. The merged $\tau$-Bench cell ($n$=90 paired tasks) is also borderline. RULER@128K is reported at $n$=20 in Table 2 (the headline cell) because 128K-context prefill costs 5–6 GPU-hours per cell at $n$=500; we additionally re-ran RULER@128K at $n$=100 on both Llama-3.1-70B and Qwen2.5-72B (§4.8) and confirmed acc=1.000 across all configurations with five times the statistical power. Both FP16 and KIVI-INT4 saturate at accuracy 1.000 on RULER@128K, so the row of Table 2 carries information only about the tier vector the controller selects, not about an accuracy comparison.

**No pruning baselines in the experimental tables.** Pruning-based KV-cache compression (StreamingLLM, $H_2O$, SnapKV, PyramidKV) is a distinct family of methods we discuss in §2 but do not evaluate here. A combined *prune + quantize* comparison—where EchoKV layers above an importance threshold also drop heavy-hitter keys—is the natural follow-up but requires non-trivial integration we have not done.

**No measured end-to-end serving throughput.** The concurrent-prompt projections in §4.6 follow directly from per-element bits and the cluster's KV budget, but we do not report measured request/sec or batch-size scaling curves. A production deployment with native sub-byte INT4/NVFP4 attention kernels (KIVI's fused kernel or Blackwell native NVFP4 paths) would invert wall direction versus FP16; our BF16-emulated implementation does not.

**Three tiers, decode-only.** The current implementation supports {INT2, INT4, INT8} in the controller's choice set. NVFP4 is included as a static baseline but not as a controller option. Sub-INT2 formats and mixed-granularity (e.g., INT4 keys, INT8 values) are not explored.

**Memory measurement is projected, not on-GPU.** Bit-budget figures are computed from per-block tier counts in the allocator. A vLLM block-allocator fork that exposes on-GPU memory directly would let us report measured rather than projected savings.

**Model coverage.** Primary experiments use Llama-3.1-70B; §4.7 extends to Llama-3.1-8B (scale transfer) and §4.8 extends to Qwen2.5-72B-Instruct (cross-family generalisation), with all four headline workloads on each model. The RULER@128K cell is additionally re-run at $n$=100 on both 70B-class models for higher statistical power. Mistral, DeepSeek-V3, and multimodal variants remain future work. The canary design is model-agnostic, but $\tau_0$ recalibration per architecture is required,

as the 8B over-conservatism in §4.7 and the shifted Qwen tier signatures in §4.8 both concretely illustrate.

# 6 Future Work

**Native sub-byte attention kernels.** The headline result we *can't* report yet is end-to-end serving throughput against an FP16 baseline. The Triton emulation we use round-trips quantization through BF16 and adds wall-clock overhead per attention call; KIVI's fused INT4 attention kernel and Blackwell native NVFP4 paths would invert wall direction. Integrating EchoKV's per-layer dispatch with those kernels is the most consequential next step.

**Tighter $\tau(L)$.** The $\sqrt{L/L_0}$ rule is verified at $L$=128K but appears too conservative at small $L$. Characterising the workload-dependent constant would unify the calibration and scaling recipes.

**Larger agentic evaluation.** The merged airline+retail $\tau$-Bench cell ($n$=90 paired) keeps the EchoKV/FP16 comparison within paired-McNemar noise but does not cross conventional significance thresholds. A third $\tau$-Bench domain or SWE-Bench Verified would tighten the agentic-regime numbers further.

**NVFP4 as a fourth tier.** The current controller chooses among {INT2, INT4, INT8}. Adding NVFP4 as an option per layer would let the controller exploit format-specific strengths.

**Output-distance sensor.** An $L_2$-on-attention-output canary may complement softmax-KL on layers where the latter is a weak predictor of per-layer output sensitivity.

**SLO-coupled budget broadcaster and write-ahead allocator.** Two architectural extensions designed but not yet implemented: the former would couple $\tau$ to throughput/latency SLOs at request granularity; the latter would speculate the next step's tier vector to overlap allocator work with attention compute.

# 7 Conclusion

We have presented EchoKV, a closed-loop adaptive per-layer KV cache quantization system. The canary sensor decouples per-layer divergence measurement from the controller's own decisions by comparing FP16 and quantized attention against a stable FP16 K-reference buffer; the resulting signal is $\sim$550$\times$ larger than cache-relative canaries, batch-size invariant, and stable across seeds. A three-component closed-loop architecture—canary, controller, precision-aware block allocator—adapts per-layer bit-widths live during serving at $+0.09\%$ TPOT overhead. Across four workloads spanning short reasoning, multi-hop QA, 128K-token retrieval, and long-decode agentic interaction, EchoKV preserves FP16 accuracy while reducing KV memory by 59–75%, with workload-coherent tier vectors (15, 18, 0, and 79 INT8 promotions consolidated, ranging from 4.33 to 6.54 bits per rank) selected live by the same controller without per-workload tuning. On workloads where 4-bit deviates from FP16, EchoKV occupies a favourable point on the accuracy–memory frontier between uniform INT4 and uniform INT8, closing the gap between conservative INT8 serving and aggressive INT4 serving.

# References


[1] Z. Liu, J. Yuan, H. Jin, Z. Zhong, Z. Xu, V. Braverman, B. Chen, and X. Hu. KIVI: A tuning-free asymmetric 2-bit quantization for KV cache. In *Proc. ICML*, 2024.

[2] C. Hooper, S. Kim, H. Mohammadzadeh, M. W. Mahoney, Y. S. Shao, K. Keutzer, and A. Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *NeurIPS*, 2024.

[3] G. Xiao, Y. Tian, B. Chen, S. Han, and M. Lewis. Efficient streaming language models with attention sinks. In *ICLR*, 2024.

[4] Z. Zhang, Y. Sheng, T. Zhou, T. Chen, L. Zheng, R. Cai, Z. Song, Y. Tian, C. Ré, C. Barrett, Z. Wang, and B. Chen. $H_2O$: Heavy-hitter oracle for efficient generative inference of large language models. In *NeurIPS*, 2023.

[5] Y. Li, Y. Huang, B. Yang, B. Venkitesh, A. Locatelli, H. Ye, T. Cai, P. Lewis, and D. Chen. SnapKV: LLM knows what you are looking for before generation. In *NeurIPS*, 2024.

[6] Y. Zhao, C.-Y. Lin, K. Zhu, Z. Ye, L. Chen, S. Zheng, L. Ceze, A. Krishnamurthy, T. Chen, and B. Kasikci. Atom: Low-bit quantization for efficient and accurate LLM serving. In *MLSys*, 2024.

[7] Y. Sheng, L. Zheng, B. Yuan, Z. Li, M. Ryabinin, B. Chen, P. Liang, C. Ré, I. Stoica, and C. Zhang. FlexGen: High-throughput generative inference of large language models with a single GPU. In *ICML*, 2023.

[8] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with PagedAttention. In *SOSP*, 2023.

[9] Z. Ye, L. Chen, R. Lai, W. Lin, Y. Zhang, S. Wang, T. Chen, B. Kasikci, V. Grover, A. Krishnamurthy, and L. Ceze. FlashInfer: Efficient and customizable attention engine for LLM inference serving. In *MLSys*, 2025.

[10] M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. Le, E. Chi, D. Zhou, and J. Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of ACL*, 2023.

[11] C.-P. Hsieh, S. Sun, S. Kriman, S. Acharya, D. Rekesh, F. Jia, Y. Zhang, and B. Ginsburg. RULER: What's the real context size of your long-context language models? In *COLM*, 2024.

[12] Y. Chen, S. Qian, H. Tang, X. Lai, Z. Liu, S. Han, and J. Jia. LongLoRA: Efficient fine-tuning of long-context large language models. In *ICLR*, 2024.

[13] C. Han, Q. Wang, H. Peng, W. Xiong, Y. Chen, H. Ji, and S. Wang. LM-Infinite: Zero-shot extreme length generalization for large language models. In *NAACL*, 2024.

[14] Z. Chen, R. Sadhukhan, Z. Ye, Y. Zhou, J. Zhang, N. Nolte, Y. Tian, M. Douze, L. Bottou, Z. Jia, and B. Chen. MagicPIG: LSH sampling for efficient LLM generation. In *ICLR*, 2025. Spotlight.

[15] Z. Cai, Y. Zhang, B. Gao, Y. Liu, T. Liu, K. Lu, W. Xiong, Y. Dong, B. Chang, J. Hu, and W. Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. In *COLM*, 2025.

[16] S. Yao, N. Shinn, P. Razavi, and K. Narasimhan. $\tau$-Bench: A benchmark for tool-agent-user interaction in real-world domains. arXiv:2406.12045, 2024.

# NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract claims (§Abstract) and introduction (§1) match the experimental results in §4: 59–75% KV-memory savings while preserving FP16 accuracy under paired McNemar across four workloads, +0.09% TPOT overhead (§4.6), and workload-coherent tier vectors (§4.4). Limitations are explicitly delineated in §5.

2. **Limitations**

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

   Answer: [Yes]

   Justification: §5 discusses seven limitations: prefill is not quantized; sample-size and saturation limits ($p$=0.052 on GSM8K-Hard, $n$=90 on $\tau$-Bench, RULER@128K saturated at 1.000); no pruning baselines; no measured end-to-end serving throughput (only projections); three tiers, decode-only; memory measurement is projected, not on-GPU; single model family (Llama-3.1, with 8B scale-transfer in §4.7).

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: The paper presents a systems contribution (a closed-loop runtime controller) and an empirical evaluation. We do not state formal theorems. The Pinsker-style $\sqrt{L}$ heuristic (§3.4) is motivated empirically and explicitly noted as an "approximate guide" rather than a tight bound.

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: §3 fully specifies the canary sensor (FP16 K-reference buffer, $N$=64, KL measurement), controller (EMA decay $\alpha$=0.1, threshold $\tau$, hysteresis), and allocator (three free pools, 2-bit tier tag). §3.4 gives the calibration recipe ($\tau_0$=0.022 from layer-averaged $\widehat{E}_{\ell,4}$ under static-INT4 hook). §4.1 specifies hardware (Llama-3.1-70B, TP=4/8 on $\tau$-Bench), inference engine (vLLM 0.16), workload sizes ($n$=500 short-context, $n$=20 RULER, $n$=50/split $\tau$-Bench), evaluation protocol (paired McNemar). §3.5 verifies the controller is insensitive to $\alpha$ and $p_c$ across a 3×3 grid.

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

   Answer: [No]

   Justification: Code release is pending camera-ready (anonymity-preserving). The method is described in sufficient detail (§3) to reimplement, and all benchmarks (GSM8K-Hard, HotpotQA, RULER@128K, $\tau$-Bench) are publicly available with cited references and standard licenses. Hyperparameter values are listed in §4.1 and §3.5.

6. **Experimental Setting/Details**

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

   Answer: [Yes]

Justification: All hyperparameters are specified: canary rate $p_c$=$1/64$ (production), $0.25$ (warmup); EMA decay $\alpha$=$0.1$; threshold $\tau_0$=$0.022$ at $L_0$=4K with $\tau(L)$=$\tau_0\sqrt{L/L_0}$; tick interval 10 decode steps; hysteresis $\pm 10\%$; min dwell 2 ticks. Calibration recipe in §3.4; sensitivity analysis in §3.5. Data splits: $n$=500 short, $n$=20 RULER, $n$=50/split $\tau$-Bench (§4.1); the same prompt batches are used for every configuration to enable paired comparisons.

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: All accuracy comparisons use *paired McNemar's exact test* on per-prompt outcomes. $p$-values are reported in Table 2 for each cell. §5 explicitly notes that the GSM8K-Hard $p$=$0.052$ result is at the threshold of conventional significance and that the $\tau$-Bench cell ($n$=90) is borderline. Sensor-validation also includes Spearman $\rho$ for cross-seed stability (§4.2). Binomial noise is acknowledged for the 8B scale-transfer Static-INT8 cell (§4.7).

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: §4.1 specifies $4\times$H100-SXM5 with TP=4 on Llama-3.1-70B, TP=8 on $\tau$-Bench, vLLM 0.16. §5 notes that RULER@128K costs 5–6 GPU-hours per cell at $n$=500, motivating the $n$=20 choice. The full 4-workload sweep (FP16, INT4, INT8, NVFP4, EchoKV) plus 8B scale-transfer plus 9-config ablation completed in $\sim$30 minutes wall-clock when run in parallel across 8 GPUs.

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics. The work uses publicly released model weights (Llama-3.1, under the Llama Community License), public benchmarks, and reports honest negative results (e.g., the 8B GSM8K-Hard over-conservatism in §4.7). No human subjects are involved. No new datasets are released.

10. **Broader Impacts**

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

    Answer: [N/A]

    Justification: This is a serving-efficiency paper: it lowers KV-cache memory cost so existing LLMs can serve more concurrent users at lower hardware cost. It does not produce a new model, dataset, or capability. Positive impact (lower carbon footprint of LLM serving) and negative impacts (faster/cheaper inference makes any underlying-model misuse marginally cheaper) are both indirect and not specific to this work.

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper does not release any pretrained models, image generators, or scraped datasets. The contribution is a runtime controller for KV-cache precision; safeguards are not applicable.

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All assets used are publicly released: Llama-3.1-70B/8B-Instruct (Llama 3.1 Community License); vLLM (Apache 2.0, [8]); FlashInfer (Apache 2.0, [9]); GSM8K-Hard ([10]); HotpotQA (CC-BY-SA 4.0); RULER ([11], Apache 2.0); $\tau$-Bench (MIT, [16]). All are credited via citation in §4.3 and §2.

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: The paper does not release new datasets or models. The runtime controller and Triton kernels will be released alongside the camera-ready as a separate code artefact.

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: No crowdsourcing or human-subjects research was conducted.

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: No human subjects research; IRB review is not applicable.

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?

Answer: [N/A]

Justification: LLMs are the *subject* of evaluation in this work, not a methodological component. The core method (canary sensor, threshold-EMA controller, precision-aware allocator) does not use LLMs as part of its algorithm. LLMs were used only for incidental writing assistance.