IDEAL-KV: INFERENCE-ALIGNED CODEBOOK LEARNING AND CALIBRATED STAGE ALLOCATION FOR KV-CACHE COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Residual-codebook quantization reduces key–value (KV) cache storage, but preserving prediction quality at high compression ratios depends on both the learned codewords and the allocation of residual stages. Reconstruction error alone does not fully characterize the predictive effects of these decisions. We propose IDEAL-KV, which combines inference-aligned codebook learning with continuation-calibrated global allocation. With the language-model backbone frozen, IDEAL-KV learns structured Key and Value codebooks by aligning attention maps, attention outputs, and continuation distributions with the dense model. It then measures the continuation KL induced by each layer–component–depth choice in an otherwise dense cache and uses the resulting additive scores to allocate residual stages under a global KV-payload budget. The selected offline policy retains all historical tokens and supports packed-code attention without materializing the full historical KV cache during decoding. Across three LLM backbones, IDEAL-KV achieves near-BF16 performance on LongBench, RULER-32K, and GSM8K at approximately $8\times$ historical-KV payload compression. At approximately $16\times$, LongBench and RULER-32K averages remain within 0.67 and 0.80 points of BF16, respectively. Code is available at `https://anonymous.4open.science/status/IDEAL-KV-7562`.

1 INTRODUCTION

Long-context autoregressive inference in large language models (LLMs) relies on a key–value (KV) cache whose storage grows with context length, batch size, and network depth. The resulting memory footprint and decoding memory traffic constrain the context lengths and concurrency that a deployment can support. KV-cache quantization reduces these costs through compact representations tailored to the characteristics of Key and Value states (Liu et al., 2024; Hooper et al., 2024; Duanmu et al., 2024). More broadly, efforts to preserve quality under limited cache budgets have explored predictive supervision and heterogeneous compression, including projection-based distillation (Lin et al., 2025), mixed-precision search (Li et al., 2025b), and quantizer-aware rate allocation (Zuo et al., 2026). These approaches improve the compression–quality trade-off by adapting the compressed representation, its allocation, or both.

At the representation level, codebook-based vector quantization encodes KV subvectors using indices into shared codebooks, exploiting structure across their coordinates. Residual codebooks approximate each subvector by a sum of codewords from successive stages, with the retained stage depth controlling the encoded payload. CommVQ (Li et al., 2025a) provides a structured construction that supports RoPE-compatible attention directly on compressed representations. Building on this representation and its compressed-domain algebra, we study how to preserve prediction quality at high compression ratios. Under tight payload budgets, this requires addressing two complementary problems: learning the K/V codewords and allocating their retained stages across the model.

With stage depths fixed, reconstruction-based fitting minimizes KV-vector error, but lower reconstruction error does not necessarily imply better prediction. Key errors affect attention logits through their projections onto queries, while Value errors are aggregated under attention weights that reconstructed Keys may already have changed. These effects interact and propagate through the remaining decoder,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

so reconstructions with similar geometric errors can have different predictive consequences. The first challenge is therefore to learn codewords that preserve the computations induced by the reconstructed states, rather than vector fidelity alone, while retaining the structured residual representation and hard discrete assignments used at deployment.

Once the codebooks are learned, the second challenge is to allocate residual stages across layers and Key/Value components under a shared payload budget. Prior work has shown that these components differ in their sensitivity to KV-cache compression (Li et al., 2025b; Lin et al., 2025). Within residual-codebook representations, each retained depth provides a reconstruction whose predictive effect depends on both the learned codewords and the component in which it is used. The benefit of additional stages can therefore vary across components and starting depths. Moreover, assigning more stages to one component leaves less budget for others, making depth selection a model-wide allocation problem. Addressing this problem requires characterizing the predictive distortion and encoded cost of the available depths, then using these trade-offs to select a configuration within the global budget.

We introduce IDEAL-KV (**I**nference-aligned co**de**book learning and calibrated stage **al**location for **KV**-cache compression). Starting from reconstruction-trained codebooks, we jointly optimize the structured Key and Value codeword parameters while freezing the entire LLM backbone. A composite objective aligns attention maps, attention outputs, and continuation distributions with those of the dense model, supervising attention routing, joint K/V aggregation, and downstream predictive effects. Reconstruction supervision and parameter anchoring regularize learning, while sampled uniform and mixed-depth configurations expose the codebooks to different stage choices before the deployment policy is selected. This procedure targets inference behavior while preserving the hard residual representation used for deployment.

We then freeze the learned codebooks and evaluate candidate depths through single-component compression interventions. Each candidate compresses only one layer's Key or Value cache in the original dense model, leaving all other cache components unquantized. Starting from shared dense-prefill states, we measure each intervention's continuation KL relative to the dense model. Together with encoded costs, these measurements form an empirical option table for selecting layerwise K/V depths under a global payload budget. Their sum is a tractable surrogate, rather than an exact model of simultaneous compression effects. The selected policy is fixed offline and remains compatible with the packed-code attention of CommVQ (Li et al., 2025a), retaining historical tokens without materializing the full historical KV cache during decoding.

We evaluate IDEAL-KV across three LLM backbones on 15 LongBench (Bai et al., 2024) tasks, five RULER-32K (Hsieh et al., 2024) tasks, and GSM8K (Cobbe et al., 2021). IDEAL-KV achieves better overall performance than the evaluated compression baselines, preserving near-BF16 quality at approximately $8\times$ KV-cache compression and remaining close to BF16 in long-context benchmark averages at approximately $16\times$. Ablations further support the effectiveness of both inference-aligned codebook learning and calibrated stage allocation.

2 Related Work

KV-cache representations and learning. KIVI (Liu et al., 2024), KVQuant (Hooper et al., 2024), and SKVQ (Duanmu et al., 2024) improve low-bit KV storage through tailored quantization granularities, outlier handling, and channel organization. CommVQ (Li et al., 2025a) learns structured codebooks through reconstruction-based fitting and supports RoPE-compatible attention on compressed representations. Functional information also guides quantizer design: AnTKV (Li et al., 2025c) uses gradient-weighted clustering, A^2ATS (He et al., 2025) constructs query-aware Key codebooks for KV retrieval, and NOVA-KV (Fernández-Menduiña et al., 2026) derives attention-preserving transforms for quantization. Beyond quantization, KVSculpt (Jiang & Jin, 2026) optimizes sequence-specific virtual KV pairs to preserve attention behavior.

Heterogeneous compression and allocation. KVTuner (Li et al., 2025b) searches layerwise K/V precision configurations using model accuracy and memory cost after sensitivity-based pruning. MatryoshkaKV (Lin et al., 2025) distills nested orthogonal projections for lower-dimensional continuous KV features and uses output KL to greedily evaluate further rank reductions from the current compressed configuration. RateQuant (Zuo et al., 2026) combines gradient sensitivities with cali-

brated, quantizer-specific reconstruction-distortion curves for bit allocation. RippleKV (Xu et al., 2026) maps output KL responses to controlled Value-cache noise into layerwise token budgets. Other rate-distortion formulations address joint eviction and quantization (Zhang et al., 2026) and attention-aware transform coding (Laus et al., 2026).

Positioning of IDEAL-KV. IDEAL-KV builds on CommVQ's structured residual representation (Li et al., 2025a), but trains its K/V codebooks with attention-map, attention-output, and continuation-distribution losses. For allocation, it evaluates each candidate depth with the remaining cache kept dense and optimizes the resulting additive cost–distortion objective under a global payload budget.

3 Progressive KV Codebook Compression

Consider an autoregressive Transformer f_θ with L layers. The cached state $\mathbf{z}_{\ell,c,t} \in \mathbb{R}^{d_{\ell,c}}$ belongs to token t, layer ℓ, and component $c \in \{\mathrm{K}, \mathrm{V}\}$. We split it into $G_{\ell,c}$ non-overlapping groups of width d_g, so that $d_{\ell,c} = G_{\ell,c} d_g$. Each group is encoded independently. We use $\mathbf{x} \in \mathbb{R}^{d_g}$ for one group and suppress its layer, component, group, and token indices.

Residual vector quantization represents $\mathbf{x}$ by a sequence of codewords. Stage r has parameters Θ_r and maps a discrete index $i \in \mathcal{I}$ to a codeword $\phi_{\Theta_r}(i)$. Let $\sigma > 0$ be the token's preprocessing scale, with $\sigma = 1$ for unnormalized inputs. Starting from $\mathbf{e}^{(0)} = \mathbf{x}/\sigma$, it selects the nearest codeword to the remaining residual and subtracts it:

$$\begin{aligned} i^{(r)}(\mathbf{x}) &= \arg\min_{i \in \mathcal{I}} \|\mathbf{e}^{(r-1)} - \phi_{\Theta_r}(i)\|_2^2, \quad \mathbf{e}^{(r)} = \mathbf{e}^{(r-1)} - \phi_{\Theta_r}(i^{(r)}(\mathbf{x})), \\ \hat{\mathbf{x}}^{(s)} &= \sigma \sum_{r=1}^{s} \phi_{\Theta_r}(i^{(r)}(\mathbf{x})). \end{aligned} \tag{1}$$

The *stage depth* s specifies how many codewords are retained. A deeper prefix adds refinement terms without changing earlier indices: one shared codebook family therefore provides multiple payload–fidelity operating settings. CommVQ (Li et al., 2025a) uses M learned parameter pairs and indices $\mathcal{I} = \{1, \ldots, M\}^2$ to produce M^2 stage codewords that support RoPE-aware attention directly on codes (Appendix A). Following this approach, we obtain the initial codebooks Θ^{rec} through reconstruction-based training (Algorithm 1 in Appendix E.1).

4 IDEAL-KV

4.1 Motivation and Problem Formulation

For one attention head, write $\mathbf{A} = \mathrm{softmax}(\mathbf{Q}\mathbf{K}^\top/\sqrt{d_h})$ and $\mathbf{O} = \mathbf{A}\mathbf{V}$. At fixed Query $\mathbf{Q}$, a Key error changes the logits through its projection onto the Query, not through its norm alone. The resulting attention-output error decomposes as

$$\hat{\mathbf{O}} - \mathbf{O} = \underbrace{(\hat{\mathbf{A}} - \mathbf{A})\mathbf{V}}_{\text{Key-induced routing error}} + \underbrace{\hat{\mathbf{A}}(\hat{\mathbf{V}} - \mathbf{V})}_{\text{Value aggregation error}}. \tag{2}$$

Thus, Key and Value errors interact before propagating through later layers. Similar reconstruction errors can have different predictive effects, and similar local attention improvements can have unequal downstream value across layers or starting depths. Local reconstruction error does not directly evaluate the effect of a depth choice on continuation prediction. This motivates inference-aligned codebook learning and direct measurement of candidate-depth effects for budget allocation.

For each layer and component, let $\mathcal{S}_{\ell,c}$ be its legal depths and $C_{\ell,c}(s)$ the encoded bits per historical token at depth s. A policy $\pi = \{s_{\ell,c}\}_{\ell,c}$ is shared across historical tokens and selects one legal depth per layer and Key/Value component. Its global KV-cache cost is $C(\pi) = \sum_{\ell,c} C_{\ell,c}(s_{\ell,c})$, measured in bits per token. Given a budget B in the same units, we seek codebooks Θ and a policy π that preserve predictive behavior subject to $C(\pi) \le B$. IDEAL-KV first learns one shared codebook family across multiple stage depths by aligning compressed inference with dense-model computations. We then freeze these codebooks and evaluate each candidate depth by compressing only its target layer

and Key/Value component while keeping the remaining cache dense. The resulting continuation-KL scores guide stage allocation across all layers and Key/Value components under B. The selected policy is fixed before inference and determines the packed-cache layout used by attention.

4.2 Inference-Aligned Progressive Codebook Learning

We initialize Θ with reconstruction-trained codebooks Θ^{rec} and jointly update the Key and Value codebooks while keeping the entire LLM backbone θ frozen. We optimize the codebooks used at deployment with a composite inference-alignment objective evaluated under sampled stage-depth configurations. Each training policy $\pi \sim \mathcal{P}_{\text{train}}$ specifies the retained depths. Sampling both homogeneous and mixed-depth configurations exposes the shared codebooks to multiple rates and non-uniform allocations, without requiring the final deployment policy. Smaller depths reuse prefixes of the same residual encoding. Assignments remain hard and are treated as stop-gradient. Gradients update the selected codeword parameters without differentiating through the index-selection decision, preserving the discrete representation and stage order.

For the same input and teacher-forced continuation, a dense teacher and a quantized student produce attention maps $\mathbf{A}^0, \mathbf{A}^\pi$, attention outputs $\mathbf{O}^0, \mathbf{O}^\pi$, and next-token distributions $\mathbf{p}^0_\tau, \mathbf{p}^\pi_\tau$ at temperature τ. The composite inference-alignment objective is

$$\mathcal{L}_{\text{align}} = \tau^2 \mathbb{E}\, D_{\text{KL}}(\mathbf{p}^0_\tau \| \mathbf{p}^\pi_\tau) + \lambda_{\text{attn-map}} \mathbb{E}\, D_{\text{KL}}(\mathbf{A}^0 \| \mathbf{A}^\pi) + \lambda_{\text{attn-out}} \mathbb{E} \frac{\|\mathbf{O}^\pi - \mathbf{O}^0\|_2^2}{\|\mathbf{O}^0\|_2^2}. \tag{3}$$

The three terms supervise different points in the computation: continuation distillation targets the effects of KV reconstruction on next-token predictions, attention-map alignment targets the routing induced by reconstructed Keys, and attention-output alignment targets the combined effect of reconstructed Keys and Values. In Eq. (3), the expectations average over continuation positions, supervised attention rows, and output vectors, respectively. The continuation comparison uses the dense teacher's full token distribution.

To preserve reconstruction quality across compression rates, we penalize reconstruction error at every training depth. For sequence b, layer ℓ, and component c, let $\mathbf{Z}_{b,\ell,c}$ be a matrix of sampled KV vectors in their original scale, and let $\hat{\mathbf{Z}}^{(s)}_{b,\ell,c}$ be its depth-s reconstruction obtained by concatenating the group reconstructions. For the set of training depths $\mathcal{S}^{\text{train}}_{\ell,c}$, we define

$$\mathcal{L}_{\text{rec}}(\Theta) = \mathbb{E}_{b,\ell}\left[\sum_{c\in\{\text{K},\text{V}\}} \frac{1}{|\mathcal{S}^{\text{train}}_{\ell,c}|} \sum_{s\in\mathcal{S}^{\text{train}}_{\ell,c}} \frac{\|\mathbf{Z}_{b,\ell,c} - \hat{\mathbf{Z}}^{(s)}_{b,\ell,c}\|_F^2}{\|\mathbf{Z}_{b,\ell,c}\|_F^2}\right]. \tag{4}$$

Here $\|\cdot\|_F$ denotes the Frobenius norm. The normalization accounts for differences in KV magnitudes across layers and components. This loss covers all training depths, independently of the policy sampled for inference alignment.

We also use $\mathcal{L}_{\text{anchor}}(\Theta) = \|\Theta - \Theta^{\text{rec}}\|_2^2 / \|\Theta^{\text{rec}}\|_2^2$ to limit drift from the reconstruction-trained initialization. Combining these regularizers with inference alignment gives the full learning objective:

$$\Theta^\star = \arg\min_\Theta \Big\{\mathbb{E}_{\pi\sim\mathcal{P}_{\text{train}}}[\mathcal{L}_{\text{align}}(\Theta;\pi)] + \lambda_{\text{rec}}\mathcal{L}_{\text{rec}}(\Theta) + \lambda_{\text{anchor}}\mathcal{L}_{\text{anchor}}(\Theta)\Big\}. \tag{5}$$

The learned codebooks provide a shared family of stage prefixes for different compression rates and are then frozen for capacity allocation. Appendix B gives the detailed loss definitions and optimization procedure, and Appendix E.1 specifies the training configuration.

4.3 Continuation KL for Global Allocation

With $\Theta^\star$ frozen, candidate $o = (\ell, c, s)$ compresses only component c at layer ℓ to s stages in the original dense model. All other cache components remain unquantized. Each candidate starts from the same dense-prefill states and keeps its depth fixed throughout the teacher-forced continuation.

Let $\mathbf{p}^0_t(x)$ and $\mathbf{p}^{\ell,c,s}_t(x)$ be the dense and intervened next-token distributions at scored continuation position t. The allocation score is

$$D_{\ell,c}(s) = \mathbb{E}_x\left[\frac{1}{H}\sum_{t=1}^{H} D_{\text{KL}}\Big(\mathbf{p}^0_t(x)\Big\|\mathbf{p}^{\ell,c,s}_t(x)\Big)\right]. \tag{6}$$

We compute full-vocabulary KL on shared observed continuations. It measures predictive distortion propagated through the complete model. Lower scores are better. Fixed calibration samples come from the codebook-training corpus, with equal weighting across context lengths and no downstream benchmark data. Allocation uses KL alone, while codebook learning uses the composite objective in Eq. (3).

An option costs $C_{\ell,c}(s) = G_{\ell,c} s\, b^{\mathrm{idx}}_{\ell,c} + b^{\mathrm{aux}}_{\ell,c}$ bits per token, where the two terms account for stage indices and auxiliary payload. The measurements and encoded costs form an empirical option table for budgeted layerwise Key/Value allocation. We jointly select one candidate depth for each layer and component:

$$\pi^\star = \arg \min_{\{s_{\ell,c} \in \mathcal{S}_{\ell,c}\}} \sum_{\ell,c} D_{\ell,c}(s_{\ell,c}) \quad \text{s.t.} \quad \sum_{\ell,c} C_{\ell,c}(s_{\ell,c}) \le B. \tag{7}$$

The budget spans all layers and K/V components, with fixed sink costs accounted for when setting B. After removing dominated options, we approximately solve Eq. (7) by Lagrangian search and local budget refinement, yielding a feasible policy $\hat{\pi}$. The additive score is a surrogate for joint policy quality, evaluated through full-policy ranking in Section 5.5. Appendix C details the estimator, accounting, and solver.

The selected policy $\hat{\pi}$ fixes the number of residual codebook stages $s_{\ell,\mathrm{K}}$ and $s_{\ell,\mathrm{V}}$ retained at each layer. During decoding, we compute attention outputs directly from packed code indices and learned codebooks without materializing the full historical Key and Value tensors. This computation uses a compressed-domain formulation similar to that of CommVQ (Li et al., 2025a). We implement and optimize fused CUDA kernels for IDEAL-KV to accelerate these operations. Appendix D presents the operator equations, and Section 5.4 reports the measured decoding latency.

5 Experiments

We evaluate three backbones on long-context understanding, retrieval, and mathematical reasoning near $8\times$ and $16\times$ historical-KV payload compression. Relative to BF16's 16 bits per KV element per token in this historical region, these ratios correspond to approximately 2 and 1 equivalent bits per element, respectively.

5.1 Experimental Setup

Models and training. We use Qwen2.5-7B-Instruct (Qwen) (Qwen Team, 2024), Mistral-7B-Instruct-v0.3 (Mistral) (Jiang et al., 2023), and Llama-3.1-8B-Instruct (Llama) (Llama Team, 2024). With the backbone frozen, we align reconstruction-trained codebooks on FineWeb-Edu (Penedo et al., 2024), using 512-dimensional groups and 12-bit indices. All models and rates use $\lambda_{\text{attn-map}} = \lambda_{\text{attn-out}} = 0.5$, $\lambda_{\text{rec}} = 0.05$, $\lambda_{\text{anchor}} = 10^{-4}$, and training temperature $\tau = 2$. One codebook family supports both rates. Calibration uses fixed corpus samples without downstream benchmark data (Appendix E.1).

Compression-rate convention. During decoding, our method and all compressed baselines except NOVA-KV (Fernández-Menduiña et al., 2026) retain the 128 most recent Key and Value states in BF16, whereas NOVA-KV retains 256 states. Similar to CommVQ (Li et al., 2025a), the reported $8\times$ and $16\times$ ratios exclude these method-specific exact buffers and refer only to the allocated historical KV payload. They are therefore not end-to-end reduction factors for the complete KV cache at arbitrary sequence lengths. The effective full-cache reduction is sequence-length dependent. Figure 1 separately reports measured peak total GPU memory, including weights, codebooks, cache storage, and intermediate tensors.

Benchmarks. We evaluate 15 LongBench (Bai et al., 2024) tasks covering all six task families. For RULER-32K (Hsieh et al., 2024), we evaluate three needle-in-a-haystack tasks (K1, K2, Val), variable tracking (VT), and question answering (QA2).

RULER-32K evaluates 500 examples for each of these five tasks. GSM8K (Cobbe et al., 2021) evaluates all 1,319 test questions using five fixed demonstrations and greedy decoding capped at 256 new tokens.

Table 1: LongBench results on 15 tasks. Avg.: task mean. Bold/red: best compressed score/average per model/rate block. VecInfer: no Qwen run.

Qwen2.5-7B-Instruct

Setting	Method	Single-doc. QA			Multi-doc. QA			Summarization			Few-shot			Retrieval	Code		Avg.
		Narr.	Qasper	MFQA	Hotpot	2Wiki	Mus.	Gov.	QMSum	News	TREC	Trivia	SAMSum	P.Ret.	LCC	Repo.	
Reference	BF16 / FullKV (1×)	29.66	43.74	52.84	58.35	46.88	30.20	31.73	23.38	23.80	72.00	89.29	45.97	100.00	60.25	66.90	51.67
≤ 8×	**IDEAL-KV (8×)**	**29.46**	**44.05**	50.31	57.72	45.60	29.58	33.61	23.93	**25.82**	**73.00**	89.65	**47.18**	99.00	**60.62**	64.39	**51.59**
	CommVQ (8×)	29.06	44.04	52.59	57.28	45.72	29.95	31.74	23.73	24.40	72.00	**90.19**	45.63	98.50	57.83	**65.48**	51.21
	CommVQ-G512 (8×)	29.29	43.65	52.90	**57.84**	**47.17**	**30.79**	31.93	23.65	24.53	72.00	90.15	45.38	99.00	58.15	64.51	51.40
	NOVA-KV (8×)	28.90	42.20	**53.42**	56.98	46.54	28.62	30.09	23.59	22.64	72.50	89.59	45.12	99.00	58.67	64.59	50.83
	KIVI (8×)	27.68	41.95	50.63	55.46	45.22	29.71	31.44	23.24	23.74	71.00	88.91	46.01	93.50	58.46	63.53	50.03
	KVQuant (8×)	23.53	33.06	37.48	40.93	28.72	14.84	28.52	21.47	22.24	60.50	85.31	43.32	70.50	58.36	50.59	41.29
	KVTuner (5×)	25.68	37.47	45.75	50.85	37.81	27.39	19.53	19.65	18.18	68.50	86.13	38.54	94.08	40.77	43.32	43.58
	MKV (1.25×)	28.38	42.31	53.03	56.55	46.78	30.49	**33.68**	**24.83**	25.65	71.50	90.18	45.67	**99.50**	55.00	60.85	50.96
	MKV+KIVI (8×)	27.16	37.96	46.61	53.22	42.46	27.24	27.22	24.20	22.89	70.50	89.12	43.17	89.50	49.60	54.38	47.02
	RateQuant (8×)	29.20	42.57	50.35	56.23	44.77	27.87	29.14	22.70	22.89	71.50	88.81	43.62	95.50	58.32	62.49	49.73
> 8×	**IDEAL-KV (16×)**	**30.30**	**43.84**	50.93	**57.39**	44.11	28.27	**33.10**	**23.97**	**25.73**	**72.50**	90.16	**47.09**	97.00	**58.51**	**62.11**	**51.00**
	Rec-Uni (16×)	29.69	42.40	51.40	56.60	44.98	29.12	30.41	23.13	23.91	72.00	90.30	45.43	**97.50**	51.23	58.48	49.77
	CommVQ (16×)	30.16	41.95	50.35	56.33	**46.12**	28.97	29.84	23.70	23.99	72.00	90.33	44.54	92.00	48.42	58.07	49.12
	CommVQ-G512 (16×)	29.21	42.86	**52.24**	56.59	45.01	**30.01**	31.54	23.74	24.42	71.50	**90.48**	45.65	97.00	50.69	58.66	49.97

Mistral-7B-Instruct-v0.3

Setting	Method	Single-doc. QA			Multi-doc. QA			Summarization			Few-shot			Retrieval	Code		Avg.
		Narr.	Qasper	MFQA	Hotpot	2Wiki	Mus.	Gov.	QMSum	News	TREC	Trivia	SAMSum	P.Ret.	LCC	Repo.	
Reference	BF16 / FullKV (1×)	27.05	38.14	49.83	51.15	38.97	28.44	34.44	25.49	26.54	76.00	88.64	47.20	98.00	58.86	60.70	49.96
≤ 8×	**IDEAL-KV (8×)**	26.56	39.20	51.38	51.44	38.46	27.45	33.76	**25.66**	26.44	76.00	88.84	46.41	**95.50**	58.81	**62.12**	**49.87**
	CommVQ (8×)	25.62	38.13	51.32	51.65	38.51	27.41	**34.15**	25.65	26.09	**76.50**	89.26	46.26	94.50	58.25	61.14	49.63
	CommVQ-G512 (8×)	25.30	38.85	50.51	51.83	38.40	27.91	33.70	25.18	26.05	76.00	89.14	46.22	95.00	57.79	60.46	49.49
	NOVA-KV (8×)	26.75	38.59	50.35	51.98	**40.06**	27.29	32.88	25.38	25.95	76.00	89.23	46.34	93.50	**59.86**	61.31	49.70
	KIVI (8×)	**28.74**	37.35	48.56	50.10	36.35	25.38	33.82	25.17	26.33	76.00	88.57	**47.53**	92.50	58.57	59.88	48.99
	KVQuant (8×)	21.30	28.53	42.08	38.58	28.31	19.14	30.44	22.96	26.76	73.50	86.00	44.12	61.50	55.24	53.20	42.11
	KVTuner (5×)	26.33	38.75	50.27	48.01	35.02	26.80	32.89	25.20	26.18	**76.50**	88.04	46.56	94.50	58.89	59.31	48.88
	MKV (1.25×)	25.82	**40.56**	**51.79**	52.03	38.41	**28.44**	32.15	24.10	**27.66**	76.00	**89.94**	45.23	94.00	56.45	54.90	49.17
	MKV+KIVI (8×)	25.68	36.86	50.43	50.23	37.14	28.16	28.81	24.50	26.02	76.00	89.34	42.64	77.50	52.46	51.28	46.47
	RateQuant (8×)	27.85	36.94	48.40	49.70	37.08	26.40	30.79	24.69	25.88	**76.50**	89.77	45.04	93.00	58.01	59.17	48.61
	VecInfer (8×)	26.60	36.22	48.05	**52.31**	38.21	26.13	30.97	24.53	25.82	76.00	89.06	46.68	90.00	58.87	60.83	48.69
> 8×	**IDEAL-KV (16×)**	26.14	**38.83**	50.80	50.90	**38.74**	27.27	**32.85**	**25.50**	**26.55**	76.00	89.09	**46.26**	**94.50**	58.36	**59.74**	**49.44**
	Rec-Uni (16×)	26.53	36.67	49.76	50.56	37.39	26.49	29.97	24.89	25.79	76.00	89.72	46.02	92.50	**58.68**	58.59	48.64
	CommVQ (16×)	26.41	36.42	**51.35**	**51.40**	37.65	26.98	29.67	25.11	25.10	**76.50**	89.34	45.44	91.50	55.83	57.37	48.40
	CommVQ-G512 (16×)	26.20	37.70	50.83	50.91	37.87	**28.15**	31.69	25.00	26.22	76.00	**89.77**	45.66	93.50	56.85	57.23	48.91
	VecInfer (11×)	**27.29**	34.70	45.24	50.00	36.66	25.42	27.44	23.85	24.80	74.50	89.12	45.69	85.00	58.26	59.51	47.17

Llama-3.1-8B-Instruct

Setting	Method	Single-doc. QA			Multi-doc. QA			Summarization			Few-shot			Retrieval	Code		Avg.
		Narr.	Qasper	MFQA	Hotpot	2Wiki	Mus.	Gov.	QMSum	News	TREC	Trivia	SAMSum	P.Ret.	LCC	Repo.	
Reference	BF16 / FullKV (1×)	29.54	44.67	55.67	57.52	48.53	33.31	34.45	25.48	26.96	72.50	91.65	43.89	99.50	62.80	56.69	52.21
≤ 8×	**IDEAL-KV (8×)**	29.67	43.80	56.24	57.19	48.89	33.00	34.20	25.58	26.71	**72.50**	**92.55**	43.73	**100.00**	63.06	**56.41**	**52.24**
	CommVQ (8×)	29.72	44.14	56.87	57.92	48.17	**33.12**	33.21	**25.78**	26.35	**72.50**	92.36	42.63	99.50	63.15	55.55	52.06
	CommVQ-G512 (8×)	29.07	43.81	56.29	57.49	47.29	32.73	33.85	25.20	26.37	**72.50**	91.61	43.95	**100.00**	61.97	55.46	51.84
	NOVA-KV (8×)	30.18	43.60	**57.22**	**58.02**	47.30	32.47	30.89	25.58	25.68	**72.50**	92.17	41.84	**100.00**	61.92	55.45	51.65
	KIVI (8×)	29.18	43.51	54.39	54.58	43.86	30.92	**35.09**	25.24	27.07	**72.50**	91.76	43.32	99.00	61.91	54.78	51.14
	KVQuant (8×)	30.37	43.58	55.08	54.21	42.53	32.08	32.92	24.68	26.76	70.50	92.36	41.01	99.50	**63.22**	55.25	50.94
	KVTuner (5×)	30.40	44.11	54.59	53.68	43.56	31.69	31.46	25.24	26.53	**72.50**	91.51	44.04	97.00	57.88	51.64	50.39
	MKV (1.25×)	30.42	**45.55**	54.39	57.48	**50.37**	31.80	33.90	25.47	**27.79**	**72.50**	92.31	**45.15**	99.50	61.21	54.57	52.16
	MKV+KIVI (8×)	**30.45**	41.58	54.14	54.50	49.32	32.12	24.76	23.82	25.21	70.00	90.71	38.76	96.50	52.09	41.78	48.38
	RateQuant (8×)	27.41	42.90	54.62	57.17	49.24	32.81	31.00	24.74	25.28	**72.50**	91.81	41.33	99.50	61.49	53.30	51.01
	VecInfer (8×)	29.05	44.11	55.81	57.95	49.28	33.10	31.65	24.53	25.89	**72.50**	91.10	42.69	**100.00**	61.76	54.43	51.59
> 8×	**IDEAL-KV (16×)**	29.06	42.44	**56.31**	57.32	**48.73**	**33.18**	**33.39**	**25.67**	**25.83**	**72.50**	92.50	42.77	**100.00**	**61.20**	**53.23**	**51.61**
	Rec-Uni (16×)	**29.57**	**43.09**	56.20	56.40	48.58	32.56	29.59	24.91	24.95	**72.50**	92.13	42.36	**100.00**	59.89	50.54	50.88
	CommVQ (16×)	29.05	41.92	54.77	57.27	47.33	31.34	27.96	24.74	24.17	70.50	**92.66**	42.47	99.50	60.42	50.93	50.34
	CommVQ-G512 (16×)	29.09	42.16	55.82	56.91	46.97	31.91	28.52	25.23	25.25	**72.50**	91.37	**44.05**	99.50	58.17	50.63	50.54
	VecInfer (11×)	28.05	40.36	53.07	**58.25**	48.52	32.12	28.36	24.32	24.52	70.50	91.28	42.34	98.50	60.95	52.19	50.22

Baselines. We compare with CommVQ (Li et al., 2025a), NOVA-KV (Fernández-Menduiña et al., 2026), KIVI (Liu et al., 2024), KVQuant (Hooper et al., 2024), KVTuner (Li et al., 2025b), RateQuant (Zuo et al., 2026), VecInfer (Yao et al., 2026), and MatryoshkaKV (MKV) (Lin et al., 2025), including MKV+KIVI, alongside an uncompressed BF16 / FullKV reference. CommVQ-G512 follows the CommVQ method, setting its Key residual-codebook groups to 512 dimensions and stage indices to 12 bits to match IDEAL-KV, while retaining CommVQ's original Value compression configuration (Appendix E.2). Rec-Uni is our reconstruction-only control at 16× (Table 4). It shares IDEAL-KV's K/V representation and uses 42 stages for each layer's Key and Value components, without inference alignment or calibrated allocation.

Table 2: RULER-32K results on five tasks (500 examples each). Avg.: task mean. Bold/red: best compressed score/average per model/rate block.

Method	Qwen2.5-7B-Instruct						Mistral-7B-Instruct-v0.3						Llama-3.1-8B-Instruct					
	K1	K2	Val	VT	QA2	Avg.	K1	K2	Val	VT	QA2	Avg.	K1	K2	Val	VT	QA2	Avg.
BF16 / FullKV (1×)	99.00	95.80	88.90	29.48	48.00	72.24	87.00	61.80	88.15	80.24	44.00	72.24	99.80	99.20	98.00	87.16	52.40	87.31
≤ 8×																		
IDEAL-KV (8×)	**99.20**	92.20	88.15	**34.24**	**48.80**	**72.52**	87.00	**61.00**	87.85	81.96	**43.80**	**72.32**	**99.80**	**98.80**	98.75	**89.44**	**52.20**	**87.80**
CommVQ (8×)	98.80	**95.20**	84.85	29.84	**48.80**	71.50	85.40	52.00	88.85	**85.12**	**43.80**	71.03	99.60	**98.80**	96.45	87.64	51.80	86.86
CommVQ-G512 (8×)	98.80	93.60	85.85	29.40	47.80	71.09	85.40	53.60	**89.70**	78.20	43.40	70.06	**99.80**	98.60	98.00	86.28	52.00	86.94
NOVA-KV (8×)	98.60	95.00	79.00	30.88	47.00	70.10	86.40	56.20	82.60	74.68	43.20	68.62	**99.80**	98.00	98.20	85.56	51.00	86.51
KVQuant (4×)	98.40	94.00	86.90	20.64	48.60	69.71	**87.40**	58.80	87.75	83.60	42.40	71.99	98.80	98.00	97.45	86.32	50.60	86.23
KVTuner (5×)	84.60	66.80	30.75	20.88	43.00	49.21	85.00	55.80	87.00	79.44	43.00	70.05	**99.80**	97.80	92.70	76.96	**52.20**	83.89
MKV (1.25×)	97.60	89.20	**92.75**	23.88	45.20	69.73	75.60	35.80	86.90	84.28	41.60	64.84	**99.80**	97.00	**99.45**	69.64	**52.20**	83.62
MKV+KIVI (8×)	64.40	5.00	56.45	24.44	43.00	38.66	52.00	3.20	59.15	75.64	39.40	45.88	96.40	87.40	79.35	53.00	45.40	72.31
RateQuant (8×)	88.80	55.80	67.10	25.76	43.80	56.25	75.00	23.00	73.90	81.84	41.20	58.99	99.40	94.60	97.75	85.16	49.20	85.22
VecInfer (8×)	–	–	–	–	–	–	62.80	9.80	56.20	77.40	40.40	49.32	98.20	88.40	94.40	86.32	49.00	83.26
> 8×																		
IDEAL-KV (16×)	**98.80**	90.80	**84.95**	**35.44**	**47.20**	**71.44**	**87.40**	**56.40**	87.25	**83.44**	43.00	**71.50**	**99.80**	**98.60**	**98.15**	**90.16**	50.80	**87.50**
Rec-Uni (16×)	97.20	**91.40**	81.40	31.40	46.00	69.48	83.20	49.60	**87.45**	79.32	**43.60**	68.63	99.20	97.00	96.90	86.84	51.20	86.23
CommVQ (16×)	91.00	72.20	56.20	28.60	44.40	58.48	74.20	14.20	75.85	79.64	42.40	57.26	87.80	83.00	80.35	81.28	49.80	76.45
CommVQ-G512 (16×)	97.00	89.80	72.60	29.44	46.00	66.97	80.00	29.80	86.20	69.28	42.00	61.46	93.40	91.80	87.05	73.24	**51.80**	79.46
VecInfer (11×)	–	–	–	–	–	–	20.60	2.40	6.55	51.36	38.60	23.90	74.40	52.00	70.30	60.68	46.20	60.72

K1/K2: `niah_multikey_1`/`niah_multikey_2`; Val: `niah_multivalue`; VT: `vt`; QA2: `qa_2`. –: not evaluated.

5.2 Cross-Task Performance across Compression Levels

Tables 1–3 compare IDEAL-KV with BF16 and the evaluated compression baselines across three backbones. We use BF16 as the uncompressed quality reference and Rec-Uni as the representation-matched control at approximately 16× compression.

Table 3: GSM8K accuracy (%). Ratios follow Section 5.1.

Method	Qwen	Mistral	Llama
BF16 (1×)	84.69	47.69	76.35
CommVQ (8×)	82.64	44.96	74.53
CommVQ-G512 (8×)	82.03	47.01	71.49
KIVI (8×)	81.20	41.62	71.04
VecInfer (8×)	–	42.15	73.01
NOVA-KV (6×)	82.26	44.88	76.27
KVQuant (8×)	73.84	35.25	65.28
KVTuner (5×)	61.33	45.72	72.63
RateQuant (8×)	80.59	41.24	74.68
IDEAL-KV (8×)	**84.46**	**47.16**	**76.65**
CommVQ (16×)	68.46	36.85	62.09
CommVQ-G512 (16×)	76.19	41.17	54.28
Rec-Uni (16×)	75.51	40.03	61.11
IDEAL-KV (16×)	**80.44**	**42.61**	**72.25**

Near-BF16 quality at 8×. At approximately 8× compression, IDEAL-KV preserves near-BF16 performance across the evaluated benchmarks. Across Qwen, Mistral, and Llama, its task averages differ from BF16 by at most 0.09 points on LongBench and 0.49 points on RULER-32K, while GSM8K accuracy differs by at most 0.53 percentage points.

Improvements at 16×. At approximately 16× compression, IDEAL-KV achieves the highest reported LongBench and RULER-32K averages and GSM8K accuracy among the methods evaluated near this compression level, across all three backbones. To assess gains beyond representation choices, we compare against Rec-Uni, which uses the same K/V representation with reconstruction-only codebooks and uniform stage depths. IDEAL-KV improves LongBench averages by 0.73–1.23 points, RULER-32K averages by 1.27–2.87 points, and GSM8K accuracy by 2.58–11.14 percentage points. Unlike CommVQ-G512, which matches only the Key group size and index width while retaining CommVQ's original Value configuration, Rec-Uni controls for both Key and Value representations. These representation-matched gains support both inference-aligned learning and calibrated allocation, examined separately in Section 5.3.

Quality trade-offs under tighter budgets. Increasing compression from approximately 8× to 16× reduces IDEAL-KV's task averages by 0.43–0.63 points on LongBench and 0.30–1.08 points on RULER-32K. At 16×, these averages remain within 0.67 and 0.80 points of BF16, respectively, although individual tasks exhibit both gains and losses. On GSM8K, accuracy remains 4.10–5.08 percentage points below BF16 despite the improvements over Rec-Uni. Thus, the tighter setting preserves long-context benchmark averages more closely while retaining a larger accuracy gap on GSM8K.

5.3 Ablation of Codebook Learning and Allocation

Ablation design. Table 4 reports average scores for nine Mistral variants near 16× on the same task sets as Tables 1 and 2. Rec-Uni uses reconstruction-trained codebooks and assigns 42 stages to every layer's Key and Value. Align-Uni keeps these uniform depths but uses the inference-aligned codebooks of IDEAL-KV. Rec-KL applies continuation-KL allocation to Rec-Uni's codebooks to obtain layerwise K/V depths. KL-Rec-KL instead trains codebooks with continuation-KL, reconstruction, and anchoring losses, omitting both attention-alignment terms, before applying KL-based allocation. The remaining variants use the same inference-aligned codebooks as IDEAL-KV. Align-RE allocates by reconstruction error, whereas Align-AO uses all-layer attention-output error. Fisher-RE allocates Key and Value depths using their measured reconstruction-error curves weighted by gradient/Fisher sensitivity, following sensitivity-based allocation (Zuo et al., 2026). IDEAL-KV-1 uses the same aligned codebooks and global allocator as IDEAL-KV, but derives its policy from independent calibration samples.

Table 4: Mistral ablations.

Variant	LongBench	RULER
Rec-Uni	48.64	68.63
Rec-KL	49.14	70.21
Align-Uni	49.23	69.46
KL-Rec-KL	49.19	70.71
Align-RE	48.63	64.03
Align-AO	49.12	68.33
Fisher-RE	48.39	63.22
IDEAL-KV-1	49.36	71.14
IDEAL-KV	**49.44**	**71.50**

Table 4 supports five conclusions. (1) At uniform 42-stage depths, Align-Uni exceeds Rec-Uni by 0.59/0.83 points on LongBench/RULER, showing that inference alignment helps before layerwise allocation. (2) Rec-KL gains 0.50/1.58 points over Rec-Uni, while IDEAL-KV gains 0.21/2.04 over Align-Uni, showing that continuation-KL allocation helps under both codebook-training schemes. (3) With aligned codebooks fixed, IDEAL-KV exceeds Align-RE, Align-AO, and Fisher-RE by 0.81/7.47, 0.32/3.17, and 1.05/8.28 points, respectively, favoring continuation-KL allocation. (4) IDEAL-KV and IDEAL-KV-1 differ by only 0.08/0.36 points across two independent calibration sets, suggesting stability to calibration inputs. (5) The 0.25/0.79-point gain over KL-Rec-KL supports the contribution of its omitted attention-alignment losses.

5.4 Memory Footprint and Decoding Efficiency

Although our primary goal is to reduce KV-cache memory consumption, compression introduces additional encoding operations and changes how attention is computed during decoding. We therefore measure inference latency to assess whether these memory savings can be achieved while maintaining efficient generation. We benchmark Mistral with batch size 1 on an exclusive NVIDIA B200. For the latency comparison, we evaluate codebook-based methods at approximately 16× historical-KV payload compression, following the accounting convention in Section 5.1, with BF16 as the uncompressed reference. CommVQ$^+$ denotes CommVQ using our fused attention kernels. Table 5 reports median time to first token (TTFT) and time per output token (TPOT) at a 32K input length over 20 paired synthetic requests after warm-up. Each request decodes 256 tokens, with TPOT averaged over the 255 output steps after the first token.

Table 5: 32K latency.

Method	TTFT (ms)	TPOT (ms)
BF16	754.47	33.35
CommVQ	2891.33	62.36
CommVQ$^+$	1126.75	16.79
IDEAL-KV	1480.45	14.502

Memory footprint. Figure 1 reports peak allocated GPU memory from 8K to 128K inputs, including model weights, codebooks, cache storage, and intermediate tensors.

For the full-request peak, measured as the maximum over 20 requests, IDEAL-KV reduces memory from 21.15 to 16.99 GiB at 32K and from 44.08 to 23.61 GiB at 128K, corresponding to reductions of 19.7% and 46.4%, respectively. For the first decode step, measured as the maximum of three runs with the peak counter reset after prefill, memory decreases from 17.57 to 15.29 GiB at 32K and from 29.76 to 16.92 GiB at 128K. These results show reduced memory growth with context length while retaining all historical tokens.

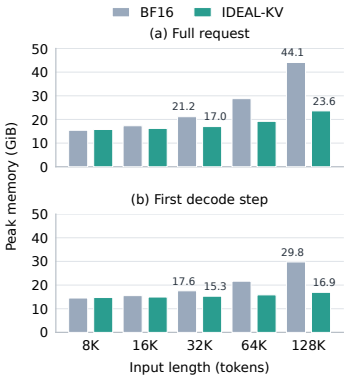

Figure 1: Mistral peak memory.

Decoding efficiency. IDEAL-KV computes attention directly from packed codes without materializing the full historical KV cache. At 32K, it achieves a TPOT of 14.502 ms/token, compared with 33.35 ms/token for BF16 and 16.79 ms/token for CommVQ$^+$. Using our fused kernels also reduces CommVQ's TPOT from 62.36 to 16.79 ms/token, highlighting the importance of the implementation. These measurements show that IDEAL-KV's storage savings are compatible with lower per-token decoding latency in the tested configuration. This benefit comes with additional prefill overhead: IDEAL-KV's TTFT is 1480.45 ms, compared with 754.47 ms for BF16 and 1126.75 ms for CommVQ$^+$. Thus, the method reduces memory consumption and decoding latency, but does not eliminate the upfront cost of KV encoding.

5.5 Full-Policy Ranking with the KL Surrogate

Table 6: KL ranking on the Mistral $16\times$ policy pool. Entries show 95% paired-bootstrap CIs over 144 shared validation texts, with L_{sel} reported in units of 10^{-3} nats/token. Full proxy results are in Appendix E.4.

Subset	Policies	ρ [95% CI] ↑	L_{sel} [95% CI] ↓
Full pool	50	$0.973\ [0.951, 0.980]$	$0.315\ [0.000, 0.972]$
Near-best†	13	$0.874\ [0.560, 0.923]$	$0.315\ [0.000, 0.972]$

†Post-hoc: NLL $\leq$ minimum $+0.005$, fixed for bootstrap.

To test whether additive KL tracks jointly compressed policies, we compare its ranking with held-out continuation NLL across 50 equal-cost Mistral policies near $16\times$, using 144 texts disjoint from calibration. Table 6 reports $\rho = 0.973$ [0.951, 0.980] on the full pool and $\rho = 0.874$ [0.560, 0.923] in the post-hoc 13-policy window within 0.005 NLL of the best policy in the evaluated pool. The KL proxy's selection loss relative to that policy is 0.315×10^{-3} nats/token. For the fixed KL-selected and full-validation best policies in the evaluated pool, the paired 95% CI ($[-0.273, 0.895]\times10^{-3}$) includes zero. Appendix E.4 reports the full proxy comparison and resampling protocol. KL ranks the full pool well, but ordering in the five-policy KL neighborhood remains unresolved.

6 Conclusion

We presented IDEAL-KV, which combines inference-aligned codebook learning with continuation-calibrated global allocation for residual-codebook KV-cache compression. With the language-model backbone frozen, we jointly optimize structured Key and Value codewords to preserve dense-model attention maps, attention outputs, and continuation distributions. After freezing the learned codebooks, we evaluate candidate depths through single-component compression and use their continuation KL scores as an additive surrogate for stage allocation under a shared payload budget. Across three backbones, IDEAL-KV achieves near-BF16 performance at approximately $8\times$ historical-KV payload compression and keeps LongBench and RULER-32K averages close to BF16 at approximately $16\times$, with larger accuracy losses on GSM8K at the tighter budget. The resulting offline policy remains compatible with packed-code attention without materializing the full historical KV cache. These results support aligning both codeword learning and residual-stage allocation with inference behavior, rather than relying on reconstruction fidelity alone.

AI Use Statement

Generative AI tools were used to assist with language editing and to improve the clarity and organization of the manuscript. The authors reviewed all AI-assisted outputs and take full responsibility for the final manuscript.

REFERENCES

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, 2024. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

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. URL https://arxiv.org/abs/2110.14168.

Haojie Duanmu, Zhihang Yuan, Xiuhong Li, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. SKVQ: Sliding-window key and value cache quantization for large language models. *arXiv preprint arXiv:2405.06219*, 2024. URL https://arxiv.org/abs/2405.06219.

Samuel Fernández-Menduiña, Amir Ziashahabi, Eduardo Pavez, Antonio Ortega, and Salman Avestimehr. Spend bits where queries look: KV cache vector quantization with attention-preserving transforms. *arXiv preprint arXiv:2608.04074*, 2026. URL https://arxiv.org/abs/2608.04074.

Junhui He, Junna Xing, Nan Wang, Rui Xu, Shangyu Wu, Peng Zhou, Qiang Liu, Chun Jason Xue, and Qingan Li. A^2ATS: Retrieval-based KV cache reduction via windowed rotary position embedding and query-aware vector quantization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12451–12463, 2025. doi: 10.18653/v1/2025.findings-acl.644. URL https://aclanthology.org/2025.findings-acl.644/.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *Advances in Neural Information Processing Systems*, volume 37, pp. 1270–1303, 2024. doi: 10.52202/079017-0040. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/028fcbcf85435d39a40c4d61b42c99a4-Abstract-Conference.html.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024. URL https://arxiv.org/abs/2404.06654.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. URL https://arxiv.org/abs/2310.06825.

Bo Jiang and Sian Jin. KVSculpt: KV cache compression as distillation. *arXiv preprint arXiv:2603.27819*, 2026. URL https://arxiv.org/abs/2603.27819.

Hannah Laus, Claudio Mayrink Verdun, Hao Wang, Flavio du Pin Calmon, and Felix Krahmer. KV cache compression through the lens of transform coding. *arXiv preprint arXiv:2608.14191*, 2026. URL https://arxiv.org/abs/2608.14191.

Junyan Li, Yang Zhang, Muhammad Yusuf Hassan, Talha Chafekar, Tianle Cai, Zhile Ren, Pengsheng Guo, Foroozan Karimzadeh, Colorado Reed, Chong Wang, and Chuang Gan. CommVQ: Commutative vector quantization for KV cache compression. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36831–36845, 2025a. URL https://proceedings.mlr.press/v267/li25du.html.

Xing Li, Zeyu Xing, Yiming Li, Linping Qu, Hui-Ling Zhen, Yiwu Yao, Wulong Liu, Sinno Jialin Pan, and Mingxuan Yuan. KVTuner: Sensitivity-aware layer-wise mixed-precision KV cache

quantization for efficient and nearly lossless LLM inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36451–36485, 2025b. URL https://proceedings.mlr.press/v267/li25dd.html.

Zeyu Li, Chuanfu Xiao, Yang Wang, Xiang Liu, Zhenheng Tang, Baotong Lu, Mao Yang, Xinyu Chen, and Xiaowen Chu. AnTKV: Anchor token-aware sub-bit vector quantization for KV cache in large language models. *arXiv preprint arXiv:2506.19505*, 2025c. URL https://arxiv.org/abs/2506.19505.

Bokai Lin, Zihao Zeng, Zipeng Xiao, Siqi Kou, TianQi Hou, Xiaofeng Gao, Hao Zhang, and Zhijie Deng. MatryoshkaKV: Adaptive KV compression via trainable orthogonal projection. In *International Conference on Learning Representations*, volume 2025, pp. 86669–86690, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/d7b351608d824a4680344a02b180a947-Paper-Conference.pdf.

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 *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32332–32344, 2024. URL https://proceedings.mlr.press/v235/liu24bz.html.

Llama Team. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi: 10.48550/arXiv.2407.21783. URL https://arxiv.org/abs/2407.21783.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems*, volume 37, pp. 30811–30849, 2024. doi: 10.52202/079017-0970. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/370df50ccfdf8bde18f8f9c2d9151bda-Abstract-Datasets_and_Benchmarks_Track.html.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. doi: 10.48550/arXiv.2412.15115. URL https://arxiv.org/abs/2412.15115.

Dongjie Xu, Kai Qian, Julius, Weijie Shi, Yuxuan Sun, Minghua Tang, Fenglei Jin, Hanchi Dong, and Jiajie Xu. RippleKV: Cross-layer KV cache allocation via perturbation propagation. *arXiv preprint arXiv:2608.08684*, 2026. URL https://arxiv.org/abs/2608.08684.

Dingyu Yao, Chenxu Yang, Zhengyang Tong, Zheng Lin, Wei Liu, Jian Luan, and Weiping Wang. VecInfer: Efficient LLM inference with low-bit KV cache via outlier-suppressed vector quantization. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31527–31543. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1454. URL https://aclanthology.org/2026.acl-long.1454/.

Junkai Zhang, Hang Guo, Luca Benini, and Yawei Li. RDKV: Rate-distortion bit allocation for joint eviction and quantization of the KV cache. *arXiv preprint arXiv:2605.08317*, 2026. URL https://arxiv.org/abs/2605.08317.

Fei Zuo, Zikang Zhou, Hao Cong, Xiaoyan Xi, and Ho Fai Leung. RateQuant: Optimal mixed-precision KV cache quantization via rate-distortion theory. *arXiv preprint arXiv:2605.06675*, 2026. URL https://arxiv.org/abs/2605.06675.

A Structured Residual Codebooks and Error Propagation

A.1 Groups, Parameters, and Discrete Indices

For token t, layer ℓ, and cache component $c \in \{\mathrm{K}, \mathrm{V}\}$, the cached vector is partitioned as

$$\mathbf{z}_{\ell,c,t} = \left[\mathbf{x}_{\ell,c,t,1}^{\top}, \ldots, \mathbf{x}_{\ell,c,t,G_{\ell,c}}^{\top}\right]^{\top}, \qquad \mathbf{x}_{\ell,c,t,g} \in \mathbb{R}^{d_g}. \tag{A.1}$$

Every group has its own sequence of codebook parameters $\Theta_{\ell,c,g,r}$, indexed by residual stage r. Suppressing these indices, the structured parameterization of CommVQ (Li et al., 2025a) uses $d_g = 2p$ and M learned pairs, $\Theta = \{(\mathbf{a}_j, \mathbf{b}_j) \in \mathbb{R}^p \times \mathbb{R}^p\}_{j=1}^{M}$. An ordered index pair $i = (u, v)$ chooses a codeword

$$\phi_\Theta(u, v) = \begin{bmatrix} \mathbf{a}_u - \mathbf{b}_v \\ \mathbf{b}_u + \mathbf{a}_v \end{bmatrix}, \qquad (u, v) \in \mathcal{I} = \{1, \dots, M\}^2. \tag{A.2}$$

The corresponding codeword set is $\mathcal{C}_{\ell,c,g,r} = \{\phi_{\Theta_{\ell,c,g,r}}(i) : i \in \mathcal{I}\}$. There are M^2 index choices, constructed from M parameter pairs. $\mathcal{I}$ denotes the legal indices, not a learned matrix. A fixed-width index therefore requires $\lceil \log_2(M^2) \rceil$ bits. For Keys, the two halves follow the paired-coordinate organization used by RoPE. This structure supports evaluating position-dependent query–codeword products directly from the indices and parameters, rather than first reconstructing a dense historical Key tensor. The representation and this compressed-attention property follow CommVQ, whereas IDEAL-KV changes codebook learning and stage allocation.

A.2 Progressive Reconstruction

Equation (1) encodes $\mathbf{x}/\sigma$, selects a nearest codeword, and applies the next stage to the residual. The scale $\sigma = \sigma_{\ell,c,t}$ is shared by all groups of a token. After stage r, $\mathbf{e}^{(r)} = (\mathbf{x} - \hat{\mathbf{x}}^{(r)})/\sigma$. Multiplying the codeword sum by σ restores the original scale. Retaining s stages stores the index sequence $(i^{(1)}, \dots, i^{(s)})$, while the codewords are shared across tokens. Changing the retained depth does not change the already selected prefix. Concatenating the restored groups gives the reconstructed KV state. Both the initialization metric and the fine-tuning regularizer in Eq. (4) compare state vectors without explicitly considering the Queries that will read them or the decoder computations that follow. Normalization, group sizes, and training settings are specified in Appendix E.1.

A.3 Why the Error Direction and Network Location Matter

For one attention head, with the causal mask understood, define

$$\mathbf{S} = \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}}, \qquad \mathbf{A} = \mathrm{softmax}(\mathbf{S}), \qquad \mathbf{O} = \mathbf{A}\mathbf{V}. \tag{A.3}$$

Holding $\mathbf{Q}$ fixed isolates the immediate effect of Key compression:

$$\hat{\mathbf{S}} - \mathbf{S} = \frac{\mathbf{Q}(\hat{\mathbf{K}} - \mathbf{K})^\top}{\sqrt{d_h}}. \tag{A.4}$$

Two errors of the same norm can have different projections onto the Query directions and therefore different logit errors. After softmax, adding and subtracting $\hat{\mathbf{A}}\mathbf{V}$ gives Eq. (2). Its first term changes which historical information is selected, whereas its second changes the information aggregated under that selection. The output projection, residual path, and later Transformer blocks propagate both effects to continuation predictions. In the full quantized model, Queries in later layers can also change. The fixed-Query argument is a local illustration, not an assumption made by the continuation objective or calibration. This motivates measuring the effect of candidate depths on continuation prediction in addition to evaluating their local reconstruction and attention errors.

B Inference-Aligned Learning Details

B.1 Optimization Variables and Shared Stage Prefixes

The trainable parameters are $\Theta = \{\Theta_{\ell,c,g,r}\}_{\ell,c,g,r}$. The backbone parameters θ, including embeddings, attention and MLP projections, normalization layers, and the output head, remain frozen. Reconstruction-trained codebooks Θ^{rec} initialize Θ. Hard nearest-codeword search is treated as a discrete stop-gradient operation. At each forward pass, the selected codewords are formed from the current parameters, and differentiation updates those parameters through the reconstruction and model-level losses. No gradient is taken through the index-selection decision itself.

A training policy is sampled from $\mathcal{P}_{\mathrm{train}}$, whose support includes both homogeneous and heterogeneous depth configurations. The latter are formed by perturbing layer–component depths without

using the final calibrated policy. For the same encoded state, evaluating its residual path to the largest required depth provides all smaller reconstructions by truncating the cumulative sum. Thus, different depths share assignments, stage ordering, and parameters rather than requiring independent rate-specific quantizers.

B.2 Components of the Composite Inference-Alignment Objective

Superscripts 0 and π denote the dense teacher and quantized student under the sampled policy. The following indexed terms are the components whose weighted sum gives $\mathcal{L}_{\text{align}}$ in Eq. (3). Both models use the same input, causal mask, RoPE positions, and teacher-forced continuation tokens. Dense outputs are fixed targets. At supervised layer ℓ, head h, and query position t, the attention-map objective is

$$\mathcal{L}_{\text{attn-map}}(\pi) = \mathbb{E}_{\ell,h,t} D_{\text{KL}}\left(\mathbf{A}^{0}_{\ell,h,t,:} \middle\| \mathbf{A}^{\pi}_{\ell,h,t,:}\right). \tag{B.1}$$

The KL divergence compares distributions over visible historical positions. For the corresponding attention-output vectors,

$$\mathcal{L}_{\text{attn-out}}(\pi) = \mathbb{E}_{\ell,t}\left[\frac{\|\mathbf{O}^{\pi}_{\ell,t} - \mathbf{O}^{0}_{\ell,t}\|_2^2}{\|\mathbf{O}^{0}_{\ell,t}\|_2^2 + \epsilon}\right]. \tag{B.2}$$

This term supervises the joint effect of reconstructed Keys and Values, rather than Value error under an assumed unchanged attention map. The numerical stabilizer $\epsilon > 0$ is omitted from Eq. (3).

For language-model logits $\mathbf{u}^0_t$ and $\mathbf{u}^\pi_t$, define $\mathbf{p}^0_{t,\tau} = \text{softmax}(\mathbf{u}^0_t/\tau)$ and $\mathbf{p}^\pi_{t,\tau} = \text{softmax}(\mathbf{u}^\pi_t/\tau)$. The continuation-distribution component is

$$\mathcal{L}_{\text{cont}}(\pi) = \tau^2 \mathbb{E}_{t\in\mathcal{T}_{\text{cont}}} D_{\text{KL}}\left(\mathbf{p}^0_{t,\tau} \middle\| \mathbf{p}^\pi_{t,\tau}\right), \tag{B.3}$$

where $\mathcal{T}_{\text{cont}}$ contains the supervised continuation positions. The quantized path passes through the remaining frozen decoder, so this loss captures downstream propagation beyond the locally supervised attention outputs. Like the allocation score, it compares the teacher's token distribution rather than scoring only the observed next token.

B.3 Geometric Regularization and Joint Optimization

The squared Frobenius ratio in Eq. (4) equals reconstruction MSE divided by mean target energy. The loss averages over training depths and supervised sequence–layer pairs, summing K/V components. In implementation, we floor the mean target energy at $\epsilon_{\text{rec}} = 10^{-12}$ for numerical stability but omit this floor from Eq. (4). Expanding the main-text parameter vector into blocks gives the stabilized anchoring loss:

$$\mathcal{L}_{\text{anchor}}(\Theta) = \frac{\sum_{\ell,c,g,r} \|\Theta_{\ell,c,g,r} - \Theta^{\text{rec}}_{\ell,c,g,r}\|_2^2}{\sum_{\ell,c,g,r} \|\Theta^{\text{rec}}_{\ell,c,g,r}\|_2^2 + \epsilon}. \tag{B.4}$$

Equation (5) combines the composite alignment loss with reconstruction and parameter regularization. Starting from Θ^{rec}, we jointly optimize the Key and Value codebooks using this full objective. Loss coefficients, temperature, and the supervised sampling pattern are fixed as reported in Appendix E.1, so the method does not require tuning a separate set of coefficients for each deployment rate. The learning ablation in Section 5.3 evaluates this composite training scheme against reconstruction-only learning.

C Continuation KL Calibration and Allocation Details

C.1 Single-Component KL Profiling

With the backbone and codebooks frozen, dense prefill supplies a shared KV snapshot. Each candidate restores an independent cache and quantizes only the historical core of $o = (\ell, c, s)$, leaving other components and exact-cache regions unquantized. The depth stays fixed as teacher-forced continuation updates the cache. Downstream states may change through propagation while retaining dense storage, so scoring captures the complete model's response to one compression intervention.

The next-token distribution produced from the prefill state, which predicts x_{m+1}, is not scored. After consuming the observed token x_{m+1}, we score the next H predictive distributions under teacher forcing, corresponding to targets $x_{m+2}, \ldots, x_{m+H+1}$. Let $\mathbf{p}_t^0(x)$ and $\mathbf{p}_t^o(x)$ be the dense and candidate distributions over the full vocabulary $\mathcal{V}$, obtained by softmax:

$$F_o(x) = \frac{1}{H} \sum_{t=1}^{H} \sum_{v \in \mathcal{V}} p_{t,v}^0(x) \log \frac{p_{t,v}^0(x)}{p_{t,v}^o(x)}. \tag{C.1}$$

The dense reference has zero KL, so the recorded candidate-minus-reference score equals $F_o(x)$. Scores are nonnegative in exact arithmetic but need not decrease monotonically with depth.

For prefix-length buckets $\mathcal{M}$ and fixed sample sets $\mathcal{D}_m^{\mathrm{cal}}$, we average within each bucket and then equally across lengths:

$$D_{\ell,c}(s) = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \frac{1}{|\mathcal{D}_m^{\mathrm{cal}}|} \sum_{x \in \mathcal{D}_m^{\mathrm{cal}}} F_{(\ell,c,s)}(x). \tag{C.2}$$

All options share samples, token positions, and dense references.

C.2 Payload Accounting and Discrete Allocation

For one layer and component, $C_{\ell,c}(s) = G_{\ell,c} s b_{\ell,c}^{\mathrm{idx}} + b_{\ell,c}^{\mathrm{aux}}$ counts stage indices and auxiliary metadata. With BF16 cost $C_{\mathrm{dense}} = 16 \sum_{\ell,c} d_{\ell,c}$, fixed sink fraction α, and target ratio r, the allocator uses

$$\alpha C_{\mathrm{dense}} + (1-\alpha) C(\pi) \le C_{\mathrm{dense}}/r, \qquad B = \frac{C_{\mathrm{dense}}/r - \alpha C_{\mathrm{dense}}}{1-\alpha}. \tag{C.3}$$

For IDEAL-KV, we keep four sink tokens exact and use an accounting length of 32K (32,768 tokens), giving $\alpha = 4/32{,}768$. The target-rate constraint includes their amortized BF16 cost but excludes the recent buffer used during decoding (Appendix D).

For each (ℓ, c), we remove a depth if another has no greater cost and no greater KL score, with at least one strict improvement. For multiplier $\lambda \ge 0$, the remaining choices separate across components:

$$s_{\ell,c}(\lambda) = \arg \min_{s \in \mathcal{S}_{\ell,c}} \left[D_{\ell,c}(s) + \lambda C_{\ell,c}(s) \right].$$

If the minimum-score policy exceeds B, we increase λ to find a feasible policy, then refine it by binary search. Local refinement chooses affordable upgrades with the largest positive KL reduction per added bit, while any necessary downgrades minimize KL increase per saved bit. The returned policy $\hat{\pi}$ satisfies $C(\hat{\pi}) \le B$ and remains fixed at deployment.

D Compressed-Domain Inference Details

The allocated policy is fixed for a given model and payload budget. Prefill attention uses dense BF16 Key and Value states, but after each layer's attention its historical states are immediately encoded into packed stage indices. A full-model dense KV cache is therefore never resident. Even stage depths let us pack each pair of 12-bit stage indices into exactly three bytes (24 bits), avoiding byte-alignment padding and preserving exactly 12 bits per index.

As in the compared KIVI (Liu et al., 2024), CommVQ (Li et al., 2025a), VecInfer (Yao et al., 2026), and other baselines with an exact recent region, we retain the 128 most recent KV states in BF16 during decoding. Our reported $8\times$ and $16\times$ compression ratios exclude this exact recent buffer and instead describe the allocated historical KV cache. The following identities use the codebook representation introduced in Section 3.

D.1 Key Logits from Encoded Stages

Keys are encoded before applying their absolute RoPE positions. For one Key group at historical position t,

$$\hat{\mathbf{k}}_{t,g}^{(s)} = \sum_{r=1}^{s} \phi_{\Theta^\star_{\ell,\mathrm{K},g,r}} \left(i_{t,g}^{(r)} \right). \tag{D.1}$$

Let $\mathcal{R}_t$ denote the Key-side positional transformation. By linearity,

$$\left\langle \mathbf{q}_g, \mathcal{R}_t\left(\hat{\mathbf{k}}_{t,g}^{(s)}\right)\right\rangle = \sum_{r=1}^{s}\left\langle \mathbf{q}_g, \mathcal{R}_t\left(\phi_{\Theta^\star_{\ell,\mathrm{K},g,r}}\left(i_{t,g}^{(r)}\right)\right)\right\rangle. \tag{D.2}$$

The attention logit can therefore be accumulated from stage indices and structured codebook parameters without reconstructing the full historical Key tensor.

D.2 Value Aggregation from Encoded Stages

Let ω_t be the attention weight assigned to historical token t and $\sigma_{\ell,\mathrm{V},t}$ its stored Value scale. For one Value group,

$$\begin{aligned}\sum_t \omega_t \hat{\mathbf{v}}_{t,g}^{(s)} &= \sum_t \omega_t \sigma_{\ell,\mathrm{V},t} \sum_{r=1}^{s} \phi_{\Theta^\star_{\ell,\mathrm{V},g,r}}\left(i_{t,g}^{(r)}\right) \\ &= \sum_{r=1}^{s}\sum_{i\in\mathcal{I}} H_{g,r,i}\,\phi_{\Theta^\star_{\ell,\mathrm{V},g,r}}(i),\end{aligned} \tag{D.3}$$

where

$$H_{g,r,i} = \sum_{t:i_{t,g}^{(r)}=i} \omega_t \sigma_{\ell,\mathrm{V},t} \tag{D.4}$$

is the scale-weighted occupancy of codeword i at stage r. Accumulating these statistics through the codebooks yields the Value contribution without materializing the dense historical Value tensor. Exact and compressed regions share one softmax over all visible tokens.

E Experimental Details

E.1 Codebook Training and Calibration

Representation and initialization. Each residual stage stores one 12-bit index per 512-dimensional group, with one group for Qwen and two for Mistral and Llama. Keys use $\sigma = 1$, whereas Values use the token-level ℓ_2 norm across groups, with a positive floor. To obtain the initial codebooks Θ^{rec}, we train stages sequentially on collected KV states, independently for each layer–component group with the backbone frozen. Each stage fits the residual left by frozen earlier stages, alternating soft assignments with structured weighted least-squares updates. We retain the iterate with the lowest hard-assignment MSE, $\mathrm{mean}[(\sigma\mathbf{e}^{(r)})^2]$, averaged over collected tokens and coordinates in the fitted group. We then freeze that stage and pass its remaining residual to the next stage. Algorithm 1 gives the procedure for one group.

Training data and objectives. We use FineWeb-Edu `sample-10BT`, reserving `train[:2048]` for validation and `train[2048:]` for training, capped at 512 and 120,000 documents, respectively. Training and data seeds are both 42. Tokenized documents are joined with boundary tokens and packed into 8,192-token sequences, discarding incomplete final blocks. The backbone remains frozen. Across models and rates, Eqs. (3) and (5) use $\lambda_{\text{attn-map}} = 0.5$, $\lambda_{\text{attn-out}} = 0.5$, $\lambda_{\mathrm{rec}} = 0.05$, and $\lambda_{\mathrm{anchor}} = 10^{-4}$, with unit continuation-KL weight and temperature $\tau = 2$. The legal candidate-rate set is $\{6\times, 8\times, 10\times, 12\times, 14\times, 16\times, 18\times\}$. Homogeneous rates are sampled from this set during training. Reconstruction samples 256 historical positions, and rotating autograd blocks supervise eight layers at a time.

Optimization. We use AdamW with an effective batch size of 8 and learning rates ranging from 10^{-5} to 10^{-4}. During optimization, we start from Θ^{rec} and jointly train the Key and Value codebooks using Eq. (5) for 500 updates, taking approximately four hours on eight NVIDIA H100 GPUs. We validate and save every 100 updates, selecting the checkpoint with minimum full-objective validation loss averaged equally over the seven rates. Selection uses no downstream benchmark scores. The selected codebooks are frozen for allocation and evaluation. The backbone uses BF16, whereas codeword parameters use FP32 for training and FP16 for assignment.

Algorithm 1 Sequential reconstruction-based codebook initialization

Require: Group vectors $\{\mathbf{x}_n\}_{n=1}^N$, preprocessing scales $\{\sigma_n\}_{n=1}^N$, maximum depth R
Require: Stagewise fitting schedules with temperatures $T_{r,k} > 0$
Ensure: Reconstruction-trained stage parameters $\Theta^{\mathrm{rec}} = \{\Theta_r^{\mathrm{rec}}\}_{r=1}^R$

$\mathbf{e}_n \leftarrow \mathbf{x}_n/\sigma_n$ for all n
for $r = 1, \ldots, R$ **do**
  Initialize structured parameters Θ_r from sampled current residuals
  $E_{\mathrm{best}} \leftarrow +\infty$
  for each fitting iteration k **do**
    $d_{ni} \leftarrow \|\mathbf{e}_n - \phi_{\Theta_r}(i)\|_2$ for all n and $i \in \mathcal{I}$
    $w_{ni} \leftarrow \exp(-d_{ni}/T_{r,k}) / \sum_{j\in\mathcal{I}} \exp(-d_{nj}/T_{r,k})$ ▷ Soft assignments
    $i_n \leftarrow \arg\min_{i\in\mathcal{I}} d_{ni}$ ▷ Hard assignments
    $E \leftarrow \mathrm{mean}_{n,\,\mathrm{coordinates}}[\{\sigma_n(\mathbf{e}_n - \phi_{\Theta_r}(i_n))\}^2]$
    if $E < E_{\mathrm{best}}$ **then**
      $(E_{\mathrm{best}}, \Theta_r^{\mathrm{rec}}) \leftarrow (E, \Theta_r)$
    end if
    $\Theta_r \leftarrow \arg\min_\vartheta \sum_{n=1}^N \sum_{i\in\mathcal{I}} w_{ni} \|\mathbf{e}_n - \phi_\vartheta(i)\|_2^2$ ▷ Structured weighted LS
  end for
  Freeze Θ_r^{rec}
  Recompute $i_n \leftarrow \arg\min_{i\in\mathcal{I}} \|\mathbf{e}_n - \phi_{\Theta_r^{\mathrm{rec}}}(i)\|_2^2$
  $\mathbf{e}_n \leftarrow \mathbf{e}_n - \phi_{\Theta_r^{\mathrm{rec}}}(i_n)$ for all n ▷ Residual for the next stage
end for
return Θ^{rec}

Continuation KL calibration. We fix 40 FineWeb-Edu samples per prefix length $\{2048, 4096, 6144, 8192\}$ for each model, without using downstream benchmark data. Every candidate uses the dense background and scores 256 continuation positions after one observed warmup token (Appendix C). Four sink tokens remain exact. Collecting the continuation-KL calibration statistics used to derive the allocation policy takes approximately 20 GPU-hours.

E.2 CommVQ-G512 Comparison

CommVQ-G512 follows the CommVQ (Li et al., 2025a) method with Key residual-codebook settings matched to IDEAL-KV: 512-dimensional groups and 12-bit stage indices. Keys are quantized before RoPE without inference-aligned learning. The Value compression module retains the original CommVQ configuration.

E.3 Rec-Uni Baseline

Rec-Uni uses reconstruction-trained codebooks with 42 residual stages for both Key and Value at every layer. It shares IDEAL-KV's representation and evaluation protocols but uses neither inference alignment nor calibrated allocation.

For GSM8K, these evaluations cover 1,319 questions with five-shot prompts, greedy decoding capped at 256 new tokens, and the same numeric answer scorer.

E.4 Full-Policy Ranking Protocol

Table 7: Full proxy results for Mistral 16×, with 95% paired-bootstrap CIs and L_{sel} in 10^{-3} nats/token.

Subset	Proxy	Spearman $\rho \uparrow$	Kendall $\tau_b \uparrow$	$L_{\mathrm{sel}} \downarrow$
Full pool (50)	RE	0.644 [0.517, 0.695]	0.478 [0.381, 0.515]	11.043 [8.678, 13.672]
	AttnOut	0.939 [0.899, 0.949]	0.799 [0.726, 0.820]	2.015 [1.100, 3.187]
	KL (ours)	0.973 [0.951, 0.980]	0.873 [0.820, 0.892]	0.315 [0.000, 0.972]
Near-best (13)†	RE	0.286 [−0.467, 0.302]	0.256 [−0.179, 0.308]	2.015 [1.100, 3.187]
	AttnOut	0.236 [−0.495, 0.269]	0.154 [−0.282, 0.231]	2.015 [1.100, 3.187]
	KL (ours)	0.874 [0.560, 0.923]	0.718 [0.333, 0.795]	0.315 [0.000, 0.972]

We evaluate a frozen pool of 50 equal-cost Mistral policies at $16\times$. It contains three proxy-optimized parents (RE, AttnOut-all, and continuation KL), four local equal-cost perturbations around each parent, three near-uniform policies, 16 random policies, and 16 K/V-budget-stratified policies. Candidate identities and family composition were fixed before held-out evaluation. The KL-optimized parent is used for Mistral's $16\times$ main results.

For proxy m, $S_m(\pi) = \sum_{\ell,c} D^m_{\ell,c}(s_{\ell,c})$ sums frozen single-component measurements. For IDEAL-KV, D^{KL} is the dense-teacher KL in Eq. (6) (dense baseline zero). Full-policy NLL $F(\pi)$ instead measures all components of π jointly on disjoint validation texts.

To match the main allocation protocol exactly, candidate construction and proxy scoring reuse the identical fixed calibration cohort used for the Mistral main experiments: 40 FineWeb-Edu samples at each prefix length in $\{2048, 4096, 6144, 8192\}$. Validation uses a disjoint set of 36 FineWeb-Edu samples per prefix length. No calibration and validation examples share text or raw-token input.

On an H100 80GB, the BF16 backbone and FP16 codebooks rebuild each policy cache from dense prefill, use Appendix D's decoding-cache settings with four exact sink tokens, and score 256 teacher-forced positions. All policies share 144 validation texts (36 at each length), yielding $50 \times 144 = 7{,}200$ policy–sample records and 1,843,200 scored tokens.

From frozen pool $\mathcal{P}$, each proxy selects $\hat{\pi}_m = \arg\min_{\pi\in\mathcal{P}} S_m(\pi)$. We define its selection loss relative to the best policy in the evaluated pool as

$$L^{(m)}_{\mathrm{sel}} = F(\hat{\pi}_m) - \min_{\pi\in\mathcal{P}} F(\pi).$$

We report $L^{(m)}_{\mathrm{sel}}$ in 10^{-3} nats/token in Tables 6 and 7. Spearman uses average ranks, Kendall uses τ_b, and proxy ties use lexicographic policy IDs. The full-validation best policy in the evaluated pool is a local KL variant.

We use 10,000 paired, context-stratified bootstrap replicates. Within each length, the 36 texts are resampled with replacement using identical indices for every policy, and the four length means remain equally weighted. Percentile 95% intervals therefore resample each text's 256-token mean NLL, not tokens or repeated policy evaluations. They condition on the frozen codebooks, pool, and validation cohort, excluding retraining, recalibration, model changes, and new candidate-pool generation.

The post-hoc 13-policy window in Tables 6 and 7 is defined by $F(\pi) \le \min_{\pi'} F(\pi') + 0.005$ on full-validation point estimates and then fixed during resampling. The construction-defined KL parent and its four local variants span only 0.000572 nats/token. Their KL Spearman is -0.600 [-0.900, 0.200] and Kendall τ_b is -0.400 [-0.800, 0.200]. Every proxy's CI crosses zero, so this cohort does not resolve the narrow neighborhood. The KL-selected parent (`p002`) minus the full-validation best policy in the evaluated pool (`p014`) is 0.000315 nats/token with paired 95% CI [-0.000273, 0.000895]. Bootstrap $L^{(\mathrm{KL})}_{\mathrm{sel}}$ is nonnegative because each replicate subtracts its best policy in the evaluated pool.