DOSERANK: MEASURED RANK ALLOCATION FOR TRAINING-FREE MLA CONVERSION

Anonymous authors
Paper under double-blind review

ABSTRACT

The key-value (KV) cache is a major memory bottleneck in large language model serving. Multi-head latent attention (MLA) reduces this footprint by caching a low-rank latent representation per token instead of separate keys and values. However, converting existing multi-head attention (MHA) and grouped-query attention (GQA) checkpoints to MLA without expensive retraining remains challenging. Prior methods allocate rank uniformly across layers or use spectral information to guide allocation, yet a layer's spectrum can be a weak predictor of its contribution to accuracy. We introduce DoseRank, an accuracy-guided rank allocator that measures each layer's sensitivity to compression using a smoothed accuracy metric on a small calibration set. At matched KV-cache sizes, DoseRank improves accuracy over uniform rank allocation by up to 5.5, 2.9, 5.6, and 7.9 percentage points on gpt-oss-120B, gpt-oss-20B, Grok-2.5, and Llama-3.1-8B, respectively. On our CARE-basis port, evaluated without recovery training, DoseRank outperforms the CARE-E spectral scheduler by up to 3.4, 2.5, 5.5, and 5.8 percentage points, respectively. Among evaluated budgets where compression leaves room for accuracy recovery, item-paired 95% confidence intervals exclude zero at most budgets for uniform and spectrum-guided allocation. Calibration on a strict subset of tasks retains most of the gains on held-out tasks, suggesting that DoseRank captures compression sensitivity that transfers beyond the calibration tasks. These results demonstrate the value of allocating rank according to measured accuracy sensitivity rather than spectral information alone. Code will be released upon acceptance.

1 INTRODUCTION

The key-value (KV) cache is a major memory bottleneck in long-context LLM serving. Multi-head latent attention (MLA) (DeepSeek-AI, 2024) reduces this footprint by caching a compact latent representation per token, making it an attractive alternative to multi-head attention (MHA) and grouped-query attention (GQA). Adopting MLA in existing checkpoints, however, requires conversion methods that preserve model accuracy at an acceptable computational cost. MHA2MLA and X-EcoMLA rely on substantial recovery training (Ji et al., 2025; Li et al., 2025), whereas TransMLA (Meng et al., 2025) uses an orthogonal transform and truncated PCA without gradient updates. CARE (Zhou et al., 2026) uses a covariance-whitened SVD, assigns layer-specific ranks from spectral information, and applies recovery training on 1–3B tokens.

These methods differ in their factorization and training requirements but share a consequential design choice: how much latent capacity to retain in each layer. Existing approaches assign ranks uniformly or according to singular-value spectra, without directly measuring the effect of allocation on task accuracy. We show that spectral information is a weak predictor of a layer's sensitivity to compression and that allocating rank according to measured accuracy sensitivity substantially improves accuracy at a fixed KV-cache budget. Our method, DoseRank, makes these measurements on a small calibration set without updating model weights. On gpt-oss-120B, it reduces the KV cache 81% while remaining within one percentage point of the unconverted model's accuracy; uniform allocation at the same budget loses 6.5 percentage points.

Whatever the factorization, one global latent-rank budget must still be split across layers, and nothing in the architecture says how. The split that matters is the one minimizing downstream task loss,

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.

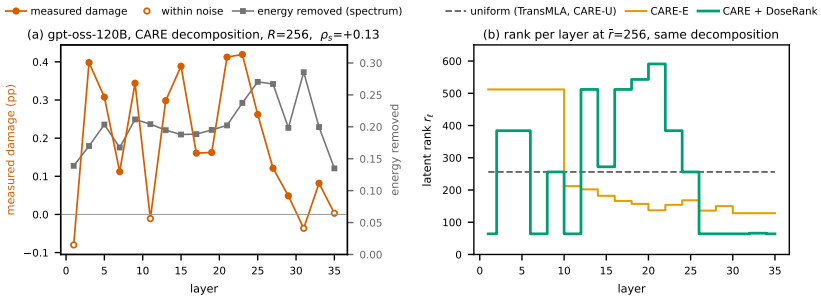


Figure 1: **Layer importance is not in the spectrum.** gpt-oss-120B on CARE's decomposition, R=256. (a) Cutting one layer at a time: the accuracy lost (orange, pp; hollow markers within noise) against the energy the spectrum the cut removes (gray). ρ_s is their rank correlation. There is low correlation between the energy in the removed and the actual damage to the accuracy. (b) Rank alocation per layer at $\bar{r}$=256: uniform, CARE's spectral scheduler (CARE-E), and CARE+DoseRank. CARE-E follows its depth prior; DoseRank follows the measured damage.

but that loss is a black-box benchmark score that does not decompose across layers, since truncation error injected at one layer propagates through the residual stream rather than being independently correctable downstream. It is also combinatorial: roughly 450^{18} candidate rank vectors at gpt-oss-120B's budget. Existing converters use uniform allocation or a spectral proxy. TransMLA gives every layer the same rank. CARE schedules ranks greedily from a score on each layer's spectrum. Neither directly measures the effect of rank changes on task accuracy.

The stand-in is the weak link, and Figure 1 shows it on one model: the layers whose spectrum calls a cut most expensive are among those the model loses nothing on, and the spectral scheduler spends its rank on them. Across four models, the Spearman correlation between the spectral energy a rank cut removes from a layer and the accuracy damage that same cut causes is weak on gpt-oss-120B, gpt-oss-20B and Grok-2.5 ($\rho_s = +0.32, +0.03, +0.16$) and inverted on Llama-3.1-8B ($\rho_s = -0.55$), and the picture is the same on CARE's own decomposition ($+0.13$ in Figure 1a); equalizing energy is not the same as protecting accuracy (Section 5). This is not a matter of solving the spectral objective better: fed to the same exact solver our method uses, two spectral costs buy at most $+1.9$ pp over uniform ranks on gpt-oss-120B and are statistically indistinguishable from it at five of seven budgets, where a measured cost buys up to $+5.5$ pp. Cheaper substitutes for measurement fail too, and they fail informatively. An allocator built on gradient sensitivity is KL-optimal against the unconverted model yet *loses* 2.8 pp to uniform ranks; ablating one layer at a time from the *full-rank* model produces no signal at all, with all 90 probes within ± 1 standard error. Sensitivity read at the uncompressed reference point ranks noise. What a layer protects is a property of the compressed model, and it has to be measured there.

We propose **DoseRank**, which treats a layer's rank as a dose and measures its response instead of assuming it. DoseRank builds the model at an operating point, cuts or restores one layer at a time over a small rank grid, and records how much each move shifts a logistic-smoothed accuracy statistic on calibration items. It then allocates the resulting per-layer dose-response curves under the budget with the same separable convex solver that makes the spectral stand-ins tractable (Section 3.3). No weight is updated. Probes use calibration items from training splits, or the validation split for MMLU. The curve-fitting procedure was selected using calibration/development data only and fixed before final evaluation; evaluation items were not used for method selection. Measurement is not free, and we report its price: 1.3–2.7 wall-clock hours per operating point on gpt-oss-120B, gpt-oss-20B and Llama-3.1-8B, and 11 hours (91 GPU-hours at tensor-parallel 8) on the 64-layer Grok-2.5.

Our contributions:

1. **DoseRank, A measured rank allocator.** DoseRank replaces the spectral stand-in with a per-layer damage curve measured at the compressed operating point and solves the same separable convex allocation the stand-ins solve (Section 3.3). Over seven zero-shot tasks at matched KV bytes it improves on uniform ranks at 24 of 26 budgets across four models, by up to $+5.5$, $+2.9$, $+5.6$ and $+7.9$ pp, and reaches uniform allocation's accuracy at

20–50% smaller rank (Section 4.2). On one basis, forward-pass token equivalents on the same model, one operating point costs 13–34M training-token equivalents. That is 29–77 times less than CARE's smallest healing budget of 1B tokens and 177–462 times less than TransMLA's 6B-token recipe(Section 4.1). The two exceptions are the highest budget on Grok-2.5 and on Llama-3.1-8B, where the fraction of single-layer probes distinguishable from calibration noise has already collapsed and DoseRank allocates against noise rather than signal; we diagnose them from calibration data alone, with no test-set access (Appendix I).

2. **The gain is a property of the allocation, not of one factorization.** On our CARE-basis port, evaluated without recovery training, DoseRank is ahead of the CARE-E spectral scheduler at all 26 budgets, at 24 of them with the interval clear of zero, by up to $+5.8$ pp (Section 4.2, Appendix F). Measured allocation improves either decomposition rather than depending on one.
3. **An account of why the free proxies fall short.** Using the probes DoseRank already collected, we show that spectral energy is a weak and sometimes inverted predictor of measured damage, that damage concentrates in a handful of layers on gpt-oss but spreads thinly on Grok-2.5 and Llama-3.1-8B, and that sensitivity estimated away from the compressed operating point carries no signal. The results suggest that measured compression sensitivity transfers across held-out task families within the seven-task suite: on gpt-oss-120B, calibration on the QA and commonsense subsets improves held-out accuracy by $+2.21$ pp and $+5.28$ pp, respectively, at $\bar{r}=128$, with one boundary case reported in Section 5.

2 Related Work

Conversion to MLA. DeepSeek-V2 introduced Multi-head Latent Attention and trained it from scratch (DeepSeek-AI, 2024); converting existing multi-query and grouped-query checkpoints (Shazeer, 2019; Ainslie et al., 2023) to it came later. TransMLA (Meng et al., 2025) and MHA2MLA (Ji et al., 2025) introduced post-hoc conversion concurrently. TransMLA rotates RoPE energy into a carried head, merges keys and values, and truncates a balanced PCA; its one-shot conversion takes no gradient steps and gives every layer the same latent rank. It is the pipeline we inherit and report under its own name. MHA2MLA uses contribution-aware partial RoPE and a joint SVD of the K and V weights, followed by recovery training on the order of billions of tokens, and X-EcoMLA (Li et al., 2025) upcycles attention into MLA by structured SVD and distills into the result. CARE (Zhou et al., 2026) is the only prior converter that gives layers different ranks: it replaces TransMLA's PCA with a covariance-whitened SVD, schedules ranks by a greedy rule on each layer's singular-value spectrum, and heals with 1–3B tokens of continued training and distillation, reporting one-shot and post-healing accuracy on Llama-3.1-8B and Qwen3-4B. We port CARE's decomposition to each of our four models, verify the port against the official code (Appendix F), and evaluate every allocator on it with no training, so Table 1 compares allocators on CARE's own basis and on our models, not against CARE's published numbers.

Budgeted allocation, and orthogonal KV reductions. Spending a compression budget unevenly across layers is established outside MLA conversion: ASVD (Yuan et al., 2023) searches per-layer truncation ranks by perplexity sensitivity, Palu (Chang et al., 2025) allocates low-rank KV budgets across heads by Fisher information, MatryoshkaKV (Lin et al., 2025) picks per-layer KV compression rates by greedy search on calibration data, OWL (Yin et al., 2024) and AlphaPruning (Lu et al., 2024) set layer-wise sparsity from outlier and spectral statistics, HAWQ (Dong et al., 2019) assigns mixed precision from Hessian sensitivity, and ShortGPT (Men et al., 2024) scores whole-layer importance from input-output cosine similarity. All rank layers by a proxy: weight-space error, a curvature estimate, a spectral statistic, or a perturbation of the otherwise uncompressed model. DoseRank measures the accuracy response of each layer at the compressed operating point, and Section 5 shows why that distinction matters. A separate line reduces KV cost without changing the allocation question: Eigen Attention (Saxena et al., 2024) and LoRC (Zhang et al., 2024) project the cache to a low-rank subspace inside the original architecture, SVD-LLM (Wang et al., 2025) compresses weights, token eviction keeps only influential entries (Zhang et al., 2023; Li et al., 2024), and quantization, whether calibrated (Liu et al., 2024; Hooper et al., 2024) or data-oblivious via random rotation (Zandieh et al., 2025), and paged memory (Kwon et al., 2023) reduce bytes per entry. Each

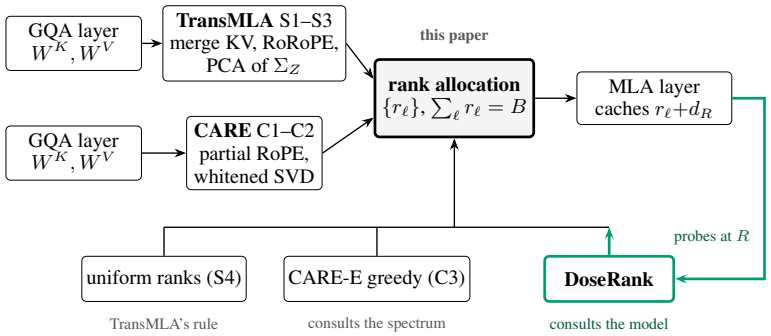


Figure 2: One allocation problem, two decompositions, three ways to solve it. Each decomposition fixes *which* subspace a layer keeps and hands the allocator a spectrum. The allocator fixes *how wide* that subspace is, and every arm on a decomposition differs only there. Dashed: the training stage both pipelines publish and this paper omits, so every arm is zero-training. DoseRank alone closes the green loop, probing the converted model at its operating point R and allocating on the damage it measures there.

composes with a latent-attention conversion rather than competing with it, and none addresses how a converted model's rank budget should be split.

3 METHOD

3.1 PRELIMINARIES: DECOMPOSITION AND ALLOCATION

Converting a GQA layer to MLA involves two separable decisions. The *decomposition* chooses which low-dimensional subspace of the layer's keys and values to keep. The *allocation* chooses how wide that subspace is, one latent rank r_ℓ per layer ℓ, under a total budget. This paper changes only the allocation and treats the decomposition as an input. We use the two published training-free decompositions, TransMLA's (Meng et al., 2025) and CARE's (Zhou et al., 2026), and compare allocators on each (Figure 2). Every arm in a comparison shares the decomposition and differs only in the rank vector $\boldsymbol{r} = (r_1, \ldots, r_L)$, so a difference in accuracy is attributable to allocation alone. A gain that appears on both decompositions is attributable to the allocator, not to the factorization it runs on.

TransMLA (Meng et al., 2025). We inherit TransMLA's conversion unchanged. For a GQA layer with n_{kv} key-value heads of dimension d, it merges the key and value projections into one down-projection (S1). It rotates the key's RoPE-dependent energy into C carried heads of total width d_R, so that the rest of the key is RoPE-free (S2). It estimates the joint activation covariance $\Sigma_Z = \text{Cov}([K_{\text{NoPE}}; V])$, of dimension $n_Z = 2n_{\text{kv}}d - d_R$, on calibration tokens and keeps its top r_ℓ eigenvectors as the layer's latent projection (S3). It folds biases and attention sinks into the converted weights (S4). A converted layer caches $r_\ell + d_R$ values per token against $2n_{\text{kv}}d$ for GQA, and sliding-window layers, where present, are not converted.

CARE (Zhou et al., 2026). We port CARE's conversion to our four models and verify the port against the official implementation (Appendix F). It handles RoPE as TransMLA does (C1). It chooses the per-layer ranks with its own scheduler, a greedy rule on the covariance-whitened weight spectra of the key and value branches whose score depends on the layer's depth and current rank (C2). It resizes the chosen ranks with one joint whitened SVD of the merged key-value operator, with KV-parity reparameterization and bias fusion (C3). A converted layer has the same cache geometry as TransMLA's.

3.2 THE ALLOCATION PROBLEM AND ITS SURROGATE

Each decomposition orders a layer's directions by a spectrum, the eigenvalues of Σ_Z on TransMLA's and the squared singular values of the whitened operator on CARE's (Section 3.1), and keeps the top r_ℓ directions.

Let $\lambda_{\ell,1} \geq \cdots \geq \lambda_{\ell,n_Z}$ be that spectrum for layer ℓ and $T_\ell = \sum_i \lambda_{\ell,i}$ its sum. Keeping the top r directions retains the fraction

$$E_\ell(r) \;=\; \frac{1}{T_\ell} \sum_{i \leq r} \lambda_{\ell,i} \tag{1}$$

of the layer's spectral energy. TransMLA ignores E_ℓ and gives every layer the same rank. CARE reads it, and spends more rank on layers whose E_ℓ rises slowly, through a greedy score on the spectrum. As Figure 1 shows, E_ℓ correlates weakly with the accuracy a layer's cut costs, and on one model inversely. To maximize the converted model's accuracy on the target task suite, the objective to allocate against is the task loss itself. We write it down first, then the surrogate every allocator, ours included, minimizes in its place. Let L be the number of converted layers and $\bar{r}$ the target average rank, so that the total budget is $B = L\bar{r}$, and let $r_{\min}$ and $r_{\max}$ bound the rank of any layer. Let $\mathcal{L}_{\text{task}}(\boldsymbol{r})$ be the downstream task loss of the model converted with rank vector $\boldsymbol{r}$. The allocation we would like is the one that minimizes that loss within the budget,

$$\boldsymbol{r}^\star \;=\; \operatorname*{arg\,min}_{\boldsymbol{r} \in \mathcal{R}(B)} \mathcal{L}_{\text{task}}(\boldsymbol{r}), \qquad \mathcal{R}(B) \;=\; \big\{ \boldsymbol{r} \in \mathbb{Z}^L : r_{\min} \leq r_\ell \leq r_{\max},\ \textstyle\sum_\ell r_\ell = B \big\}. \tag{2}$$

Two properties put it out of reach. (1) The search is combinatorial, on the order of $(r_{\max} - r_{\min})^L$ rank vectors, and evaluating one candidate costs a full benchmark run. (2) The loss does not decompose across layers: the accuracy lost of cutting one layer depends on how every other layer was cut. Every allocator therefore minimizes a surrogate in its place, one that splits across layers: TransMLA a constant, so that every layer receives the same rank, and CARE a score built from E_ℓ. We adopt the same form and change only the cost, and the question the paper asks is which cost tracks $\mathcal{L}_{\text{task}}$.

3.3 DoseRank: a measured cost

Let $\phi_\ell(r)$ be a per-layer cost, the penalty an allocator assigns to giving layer ℓ rank r, defined for $r_{\min} \leq r \leq r_{\max}$. The allocator solves

$$\min_{\boldsymbol{r} \in \mathcal{R}(B)} \sum_\ell \phi_\ell(r_\ell). \tag{3}$$

Equation 3 keeps the budget and the bounds of Equation 2 and drops only the coupling between layers, which is what makes it solvable. DoseRank measures the cost at an *operating point* $\mathbf{R} = (R, \ldots, R)$, the uniform allocation at rank R on the decomposition in use, at budget $B = LR$. A *probe* evaluates the loss defined next in a model that differs from $\mathbf{R}$ in the rank of one layer.

The DoseRank cost ϕ_ℓ. The cost of layer ℓ at rank r is the change in a measured loss D when that layer alone moves from R to r. We define D first. Every calibration item i is a multiple-choice question with K candidate answers, one of them correct. For each candidate k the model scores z_k, the sum of the log-probabilities of the candidate's tokens given the question, divided by the candidate's length in characters, and z_{gold} is the score of the correct one. The model's answer is the candidate with the largest score. Let $m_i(\boldsymbol{r}) = \max_{k \neq \text{gold}} z_k - z_{\text{gold}}$ be the signed margin in the model converted with rank vector $\boldsymbol{r}$, negative exactly when the item is answered correctly. Let $\tau = \text{median}_i\, |m_i(\mathbf{R})|$, so that every layer is scored on one scale. The per-item statistic $\sigma(m_i(\boldsymbol{r})/\tau)$, with σ the logistic function, is the smoothed probability of answering item i wrong. Let $\mathcal{T}$ be the calibration tasks and I_t the items of task t. The measured loss is the mean statistic per task, averaged over tasks,

$$D(\boldsymbol{r}) \;=\; \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{|I_t|} \sum_{i \in I_t} \sigma\big(m_i(\boldsymbol{r})/\tau\big), \qquad \sigma(x) \;=\; \frac{1}{1 + e^{-x}}. \tag{4}$$

The statistic is built so that a probe registers only when it changes what the model answers, not how confidently. Items near the decision boundary move it. Items the model gets confidently right or wrong saturate near 0 or 1. A raw likelihood fails this test: gold-NLL rewards cuts that make already-wrong items less wrong without flipping any answer (Section 5). Any per-item statistic can replace $\sigma(m_i/\tau)$ in Equation 4, and Section 5 compares three.

Write $\mathbf{R}_{\ell \leftarrow r}$ for the operating point with layer ℓ set to r. The cost is

$$\phi_\ell(r) \;=\; D(\mathbf{R}_{\ell \leftarrow r}) - D(\mathbf{R}). \tag{5}$$

This is Equation 2 sliced one layer at a time. Property (2) is not removed but frozen: each probe is taken in the compressed model, with the other $L - 1$ layers at R rather than at full rank, so

the interaction between layers is inside every measurement. Property (1) is met by the solver of Equation 3, given next. Each value of Equation 5 costs one probe, so we probe each layer at a grid $\mathcal{G}$ of ranks below and above R. The measurements are made non-increasing by a running minimum and completed into a curve on every rank in $[r_{\min}, r_{\max}]$ (Appendix C). This is $L\,|\mathcal{G}|$ forward passes in total, with no gradient and no weight update.

Minimizing the DoseRank cost. Let $\delta_\ell(r) = \phi_\ell(r) - \phi_\ell(r{+}1)$ for the saving that one more unit of rank buys layer ℓ. When every δ_ℓ is non-negative and non-increasing in r, so that ϕ_ℓ is non-increasing and convex, two equivalent rules solve Equation 3 exactly (Fox, 1966; Ibaraki & Katoh, 1988) with the argument restated in Appendix B. The greedy rule gives each next unit of rank to the layer that saves most from it,

$$r_\ell(\theta) = \operatorname*{arg\,min}_{r_{\min} \le r \le r_{\max}} [\phi_\ell(r) + \theta r], \tag{6}$$

$$\operatorname*{arg\,max}_{\ell}\ \delta_\ell(r_\ell). \tag{7}$$

The price rule charges every layer the same price $\theta \ge 0$ per unit of rank. Each layer then chooses its rank alone,

and θ is bisected until $\sum_\ell r_\ell(\theta) = B$. DoseRank bisects θ, then spends the remainder by Equation 7 (Algorithm 1). A measured ϕ_ℓ is non-increasing but not convex. The price rule then returns the exact minimizer over each layer's lower convex envelope, the largest convex function below ϕ_ℓ, and every rank we ship lies on its envelope (Appendix I).

4 Experiments

4.1 Setup

Models, arms and calibration. We convert four models from two families: gpt-oss-120B (18 of 36 layers converted, the rest sliding-window and left as is) and gpt-oss-20B (12 of 24) (OpenAI, 2025), Grok-2.5 (64 layers) (xAI, 2025) and Llama-3.1-8B-Instruct (32 layers) (Grattafiori et al., 2024). All have 8 KV heads, with head dimension 64 on gpt-oss and 128 on the other two. On each decomposition we compare its own allocator with DoseRank: uniform ranks on TransMLA's, and CARE's scheduler (CARE-E) on CARE's, where DoseRank is fitted on probes taken inside that decomposition. Every arm on a decomposition shares its calibration spectra. DoseRank's probes score items drawn from the training splits (constant seed), 2044 items on gpt-oss and Llama and 1029 on Grok, over the seven task families of the evaluation suite. Expert counts, RoPE settings, calibration corpora, and covariance details are in Appendices D.

Evaluation. Seven zero-shot loglikelihood tasks, ARC-Challenge and ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020) and WinoGrande (Sakaguchi et al., 2020), scored with `acc_norm` and macro-averaged over 31,237 items. Calibration and evaluation never share an item. Probes score items from each task's training split, or the validation split for MMLU, and the evaluation runs on the test split for ARC and MMLU and on the validation split for the other four. Every arm at every budget is exported through the same conversion pipeline and evaluated by the same harness, hardware and settings (Appendix E). Each contrast is an item-paired 95% bootstrap interval over 10,000 resamples, marked * when it excludes zero. Every reported number is the accuracy of the exported model on this suite, never the probe prediction of Equation 5. We report every contest we ran, nulls and losses included.

Cost of measurement. Summing the per-probe timers, one operating point costs 2.3–2.7 wall-clock hours on gpt-oss-120B, 1.6–1.9 on gpt-oss-20B and 1.4–1.6 on Llama-3.1-8B. On Grok-2.5 at $R{=}768$ it costs 11.3 hours, 91 GPU-hours at tensor-parallel 8.

All comparisons are at matched KV bytes. At every budget the allocators spend the same total rank, so they cache the same $\bar{r} + d_R$ values per token per converted layer, and a contrast between two arms differs only in the distribution of rank across layers. Table 1 lists the KV cut beside each budget. One probed operating point serves every target budget within a factor of two of it, for gpt-oss-120B

Table 1: Macro accuracy (%) of every arm at every tested budget. On each decomposition the two columns differ only in the rank vector: uniform ranks (TransMLA's allocator) against DoseRank on TransMLA's, and CARE's scheduler (CARE-E) against DoseRank on CARE's. Bold marks the better of the two allocators on each decomposition. KV cut is the per-converted-layer cache saving at that budget. Vanilla is the unconverted model. Item-paired confidence intervals for every contrast are in Appendix G.

gpt-oss-120B (vanilla 57.49)						gpt-oss-20B (vanilla 55.54)					
		TransMLA decomp.		CARE decomp.				TransMLA decomp.		CARE decomp.	
$\bar{r}$	KV cut	uniform	DoseRank	CARE-E	DoseRank	$\bar{r}$	KV cut	uniform	DoseRank	CARE-E	DoseRank
64	87.5%	48.90	**53.25**	48.90	**50.34**	64	87.5%	43.79	**45.87**	45.36	**45.85**
96	84.4%	50.01	**54.56**	49.92	**52.45**	96	84.4%	44.85	**47.08**	45.96	**47.45**
128	81.3%	50.99	**56.51**	51.69	**54.34**	128	81.3%	45.54	**47.21**	47.41	**48.20**
192	75.0%	53.87	**57.63**	53.28	**55.90**	192	75.0%	47.57	**50.47**	48.28	**50.29**
256	68.8%	56.07	**58.03**	54.20	**57.28**	256	68.8%	50.23	**52.50**	48.82	**51.15**
320	62.5%	56.90	**58.85**	54.66	**58.09**	320	62.5%	51.07	**53.45**	49.35	**51.80**
384	56.3%	57.63	**59.73**	55.25	**58.50**	384	56.3%	52.01	**54.40**	50.39	**52.50**
Grok-2.5 (vanilla 74.41)						Llama-3.1-8B-Instruct (vanilla 65.26)					
		TransMLA decomp.		CARE decomp.				TransMLA decomp.		CARE decomp.	
$\bar{r}$	KV cut	uniform	DoseRank	CARE-E	DoseRank	$\bar{r}$	KV cut	uniform	DoseRank	CARE-E	DoseRank
768	50.0%	65.89	**71.46**	69.01	**74.53**	128	90.6%	34.44	**35.73**	34.41	**34.60**
896	43.8%	68.29	**72.73**	71.19	**74.71**	192	87.5%	37.71	**43.09**	37.00	**38.64**
1024	37.5%	70.11	**73.07**	72.96	**74.67**	256	84.4%	41.31	**49.25**	38.87	**41.71**
1280	25.0%	73.01	**73.84**	73.77	**74.63**	384	78.1%	50.10	**56.32**	43.60	**46.84**
1536	12.5%	**73.94**	73.54	74.19	**74.63**	512	71.9%	55.66	**59.35**	47.89	**51.81**
						768	59.4%	60.94	**63.20**	53.03	**58.83**
						1024	46.9%	**63.44**	62.68	56.10	**61.75**

R=64 for $\bar{r}$=64, R=128 for 96 and 128, and R=256 for 192 to 384. No layer is assigned a rank below a floor $r_{\min}$ set per operating point, and $r_{\max}$ is the layer's full rank. Appendix G lists R and the floor beside every cell. Every number is a one-shot conversion, so absolute accuracy sits below the unconverted model and MMLU sits near its choice floor (Appendix H). Generation and long-context behavior are unmeasured, and the intervals cover item-sampling noise at one calibration seed (Section K).

4.2 Allocator comparison across four models at matched budget

TransMLA decomposition. DoseRank beats uniform ranks at 24 of 26 budgets, every one with an item-paired interval clear of zero, by $+0.8$ to $+7.9$ pp (Table 1, Figure 3). The margin is largest at the tight budgets and shrinks as the budget grows toward the spectra's capacity, which is the shape allocation should have: on gpt-oss-120B it runs from $+5.5$ pp at an 81% KV cut to $+2.1$ pp at 56%, and on Llama-3.1-8B from $+7.9$ pp at 84% to $+2.3$ pp at 59%.

Rank efficiency. Read horizontally, the same result quantifies how much rank measured allocation saves at fixed accuracy, a practical consideration for deployment. On gpt-oss-120B, DoseRank matches or beats uniform ranks at *half* the latent rank at four consecutive pairs ($\bar{r}$=64 vs. 128: 53.25 against 50.99; 96 vs. 192: 54.56/53.87; 128 vs. 256: 56.51/56.07; 192 vs. 384: 57.63/57.63), so measured allocation buys back roughly half the latent width. Elsewhere the saving is 1.25–1.5×: gpt-oss-20B at $\bar{r}$=192 beats uniform at 256 (50.47/50.23) and at 256 beats uniform at 384 (52.50/52.01), Grok-2.5 at 1024 beats uniform at 1280 (73.07/73.01), and Llama-3.1-8B at 384 beats uniform at 512 (56.32/55.66). These are point comparisons across budgets rather than the paired contrasts of Table 1, though they are scored on the same items.

CARE decomposition. On CARE's decomposition, run with no training, DoseRank is ahead of CARE's own scheduler at all 26 budgets, at 24 of them with the item-paired interval clear of zero. The margin reaches $+3.4$, $+2.5$, $+5.5$ and $+5.8$ pp on gpt-oss-120B, gpt-oss-20B, Grok-2.5 and Llama-3.1-8B. The two exceptions are the loosest Grok-2.5 budget ($\bar{r}$=1536, $+0.44$ pp) and the tightest Llama-3.1-8B budget ($\bar{r}$=128, $+0.19$ pp). The gain surviving the change of decomposition is what the result rests on: measured allocation improves either decomposition rather than depending on one. Cross-decomposition comparisons are not claimed, and are not safe here, since the two differ

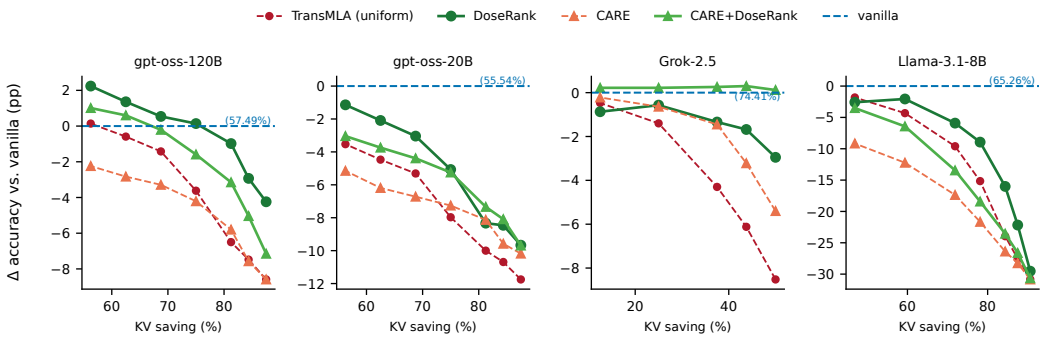


Figure 3: Accuracy against KV saving for all four models. Uniform ranks and DoseRank on TransMLA's decomposition (circles), CARE's scheduler and CARE+DoseRank on CARE's decomposition (triangles), with DoseRank solid and its baseline dashed. The y-axis is the change in macro accuracy from the unconverted model (pp), whose accuracy is printed on the dashed line. The x-axis is the per-converted-layer KV saving (%), so the most compressed budgets sit at the right. Every point is an exported model evaluated on the full suite.

by more than their allocators (on Llama-3.1-8B, TransMLA's decomposition is the stronger of the two at large budgets, by 5.3 pp at $\bar{r}$=1024).

Multiple comparisons. We report 26 contrasts per decomposition and the intervals in Table 1 are unadjusted. Most are far enough from zero that adjustment is immaterial. We take two-sided bootstrap p-values from the same 10,000 item-paired draws and apply a Holm correction at α=0.05 per column. Over the 26 TransMLA contrasts it leaves every significant cell significant, Grok-2.5 at $\bar{r}$=1280 (+0.83, p=0.025) being the closest to its corrected threshold. Over the 26 CARE contrasts it retains 23 of the 24 significant cells, gpt-oss-20B at $\bar{r}$=64 (+0.49, p=0.037) being the one that falls.

The bytes convert into throughput. At a 62.5% KV cut (r_{kv}=320), an DoseRank-converted gpt-oss-120B patched into SGLang (Zheng et al., 2024) sustains a measured 3.0× concurrency ceiling over vanilla GQA across three seeds on an 8-GPU node, gap got to 4.0× at 75% (r_{kv}=192), and serving accuracy matches offline evaluation at every tested rank (paired 95% CIs include zero), so the throughput is not purchased with a degraded numerical path (Appendix J). That configuration is a uniform-rank conversion, so it establishes the conversion pipeline at scale rather than the allocator, DoseRank spends the same bytes, and is served through the same path.

5 Ablations

Generalization across calibration tasks. Because DoseRank's statistic is a smoothed form of the evaluation metric, its allocation could depend on the tasks used for calibration. We assess this dependence by calibrating on task subsets and evaluating on tasks excluded from calibration. On the TransMLA basis of gpt-oss-120B, the QA and commonsense splits improve held-out performance over uniform ranks at $\bar{r}$=128 by +2.21 pp [+1.42, +3.02] and +5.28 pp [+4.48, +6.10], respectively. At $\bar{r}$=256, the QA split yields +1.63 pp [+0.86, +2.41]; calibration on MMLU alone yields +2.16 pp [+1.52, +2.83] across the six remaining tasks. An item-count control separates task coverage from sample size: at $\bar{r}$=256, distributing the matched item total across all seven tasks retains 95% of the full-calibration gain. Transfer is weaker for the commonsense split at $\bar{r}$=256: its aggregate gain remains significant, but the gain on the four held-out QA tasks is −0.16 pp [−0.97, +0.65], at a budget where the full-calibration margin is approximately 2 pp.

Table 2 reports the full results for both bases.

Table 2: Calibration-task robustness on gpt-oss-120B, both bases, with the full per-task breakdown. DOSE calibrated on task subsets, graded on all seven tasks. Δ_{base} (macro, real item-paired 95% CI; * = excludes zero) is DOSE's paired difference vs. that basis's own uniform-rank baseline (TransMLA, or CARE-U on CARE's decomposition); the seven task columns are the same per-task delta as point estimates (no per-task CI: too few items per task for a meaningful bootstrap at this granularity). **Bold** marks a task that calibration set did *not* include the cells the generalization claim is actually about. Retained is the share of the full-calibration gain kept (macro).

$\bar{r}$	calibration set	Δ_{base} [95% CI]	AE	AC	HS	MMLU	OBQA	PIQA	WG	ret.
TransMLA basis										
128	full (7 tasks)	$+5.52\,[+4.92, +6.12]^*$	$+9.26$	$+8.79$	$+7.34$	$+5.35$	$+3.60$	$+3.38$	$+0.95$	100%
128	H1 (ARC-e/c, OBQA, MMLU)	$+3.95\,[+3.35, +4.55]^*$	$+7.83$	$+5.55$	$\mathbf{+3.46}$	$+4.42$	$+3.20$	$\mathbf{+1.90}$	$\mathbf{+1.26}$	71%
128	H2 (HellaSwag, PIQA, WG)	$+4.73\,[+4.16, +5.31]^*$	$\mathbf{+6.02}$	$\mathbf{+7.17}$	$+7.74$	$\mathbf{+4.13}$	$\mathbf{+3.80}$	$+2.99$	$+1.26$	86%
128	items control (125/task)	$+4.31\,[+3.71, +4.90]^*$	$+8.50$	$+6.83$	$+4.73$	$+4.24$	$+4.60$	$+1.80$	-0.55	78%
128	MMLU only	$+3.40\,[+2.80, +4.00]^*$	$\mathbf{+8.92}$	$\mathbf{+5.12}$	$\mathbf{+2.22}$	$+4.15$	$\mathbf{+2.20}$	$\mathbf{+1.74}$	$\mathbf{-0.55}$	62%
256	full (7 tasks)	$+1.96\,[+1.41, +2.52]^*$	$+1.90$	$+4.44$	$+2.27$	$+1.15$	$+0.40$	$+1.30$	$+2.29$	100%
256	H1 (ARC-e/c, OBQA, MMLU)	$+1.95\,[+1.41, +2.51]^*$	$+2.15$	$+4.35$	$\mathbf{+1.29}$	$+2.29$	$+0.00$	$\mathbf{+1.31}$	$\mathbf{+2.29}$	100%
256	H2 (HellaSwag, PIQA, WG)	$+0.73\,[+0.16, +1.28]^*$	$\mathbf{-1.01}$	$\mathbf{+0.17}$	$+2.69$	$\mathbf{+0.59}$	$\mathbf{-0.40}$	$+0.60$	$+2.45$	37%
256	items control (125/task)	$+1.87\,[+1.33, +2.40]^*$	$+1.56$	$+4.69$	$+2.38$	$+0.97$	$+0.80$	$+0.54$	$+2.13$	95%
256	MMLU only	$+2.50\,[+1.93, +3.07]^*$	$\mathbf{+2.19}$	$\mathbf{+5.03}$	$\mathbf{+1.71}$	$+4.55$	$\mathbf{+0.40}$	$\mathbf{+1.47}$	$\mathbf{+2.13}$	127%
CARE basis										
128	full (7 tasks)	$+3.32\,[+2.77, +3.87]^*$	$+7.49$	$+5.63$	$+3.24$	$+3.23$	$+2.40$	$+1.31$	-0.07	100%
128	H1 (ARC-e/c, OBQA, MMLU)	$+3.46\,[+2.89, +4.03]^*$	$+7.66$	$+5.72$	$\mathbf{+3.15}$	$+3.07$	$+2.40$	$\mathbf{+1.58}$	$\mathbf{+0.63}$	104%
128	H2 (HellaSwag, PIQA, WG)	$+1.95\,[+1.37, +2.51]^*$	$\mathbf{+4.29}$	$\mathbf{+3.16}$	$+1.29$	$\mathbf{+2.48}$	$\mathbf{+1.00}$	$+0.87$	$+0.55$	59%
128	items control (125/task)	$+2.12\,[+1.57, +2.68]^*$	$+6.57$	$+2.99$	$+1.54$	$+2.09$	$+1.20$	$+0.87$	-0.39	64%
128	MMLU only	$+1.94\,[+1.41, +2.48]^*$	$\mathbf{+5.93}$	$\mathbf{+2.65}$	$\mathbf{+0.97}$	$+1.25$	$\mathbf{+2.00}$	$\mathbf{+1.41}$	$\mathbf{-0.63}$	58%
256	full (7 tasks)	$+2.14\,[+1.69, +2.63]^*$	$+3.41$	$+3.93$	$+0.73$	$+4.48$	$+1.40$	$+1.47$	-0.40	100%
256	H1 (ARC-e/c, OBQA, MMLU)	$+2.10\,[+1.63, +2.58]^*$	$+3.75$	$+4.52$	$\mathbf{+0.50}$	$+3.59$	$+1.40$	$\mathbf{+1.31}$	$\mathbf{-0.39}$	98%
256	H2 (HellaSwag, PIQA, WG)	$+2.07\,[+1.61, +2.55]^*$	$\mathbf{+3.16}$	$\mathbf{+3.75}$	$+0.96$	$\mathbf{+4.22}$	$\mathbf{+1.60}$	$+1.36$	-0.55	97%
256	items control (125/task)	$+2.13\,[+1.67, +2.60]^*$	$+3.45$	$+3.75$	$+0.32$	$+4.16$	$+1.80$	$+1.90$	-0.47	99%
256	MMLU only	$+1.49\,[+1.01, +1.97]^*$	$\mathbf{+2.48}$	$\mathbf{+2.56}$	$\mathbf{+0.01}$	$+1.66$	$\mathbf{+1.80}$	$\mathbf{+1.58}$	$\mathbf{+0.32}$	69%

Results. Across both rank budgets, every restricted calibration set yields a significant aggregate gain over TransMLA, retaining 37% to 127% of the full-calibration gain. MMLU-only calibration retains 62% at $\bar{r}$=128 and 127% at $\bar{r}$=256, exceeding full calibration at the latter budget.

At $\bar{r}$=128, reducing the number of calibration items while preserving all seven tasks reduces the gain by 22%. H2 uses the same item count across three tasks and loses only 14%, whereas H1 loses 29% despite using a third more items. Thus, reduced task coverage does not consistently impose a loss beyond that associated with sample size; the choice of calibration tasks also matters. The held-out results above show transfer beyond the calibration set, although the H2 result at $\bar{r}$=256 limits a stronger claim of task independence.

On CARE's decomposition, H1 retains approximately the full-calibration gain at both budgets (104% at $\bar{r}$=128 and 98% at $\bar{r}$=256). At $\bar{r}$=256, H2 and the items control retain 97–99%, while MMLU-only calibration retains 69%; at $\bar{r}$=128, these three settings retain about 60%. All restricted settings yield significant aggregate gains over CARE-U, and the mean gain within each held-out task block is positive at both budgets.

6 Conclusion

Every training-free GQA-to-MLA converter must split one latent-rank budget across layers, and until now each did so without asking the model. We showed that a layer's retained spectral energy predicts the accuracy it protects at $\rho \leq 0.32$ on three models and inversely on a fourth. Handing that spectrum to an exact solver buys at most $+1.9$ pp where a measured cost buys $+5.5$ pp. DoseRank measures the per-layer damage curve at the compressed operating point, with no weight update, and allocates it with the same separable convex solver. It beats uniform ranks at 24 of 26 budgets across gpt-oss-120B, gpt-oss-20B, Grok-2.5 and Llama-3.1-8B, reaching uniform's accuracy at half the latent rank on gpt-oss-120B. The measurement costs 13–34M training-token equivalents per operating point, 29 to 462 times less than the 1–6B tokens of recovery training the conversion literature relies on. It is what closes most of the gap to the unconverted model. On CARE's decomposition it is ahead of CARE's own scheduler at every budget, so measured allocation improves either decomposition rather than depending on one. Where it fails it says so first: at the highest budgets the probe signal falls below calibration noise, a condition visible in calibration data alone.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed S. Abdelfattah, and Kai-Chiang Wu. Palu: Compressing KV-cache with low-rank projection. In *International Conference on Learning Representations (ICLR)*, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ: Hessian AWare quantization of neural networks with mixed-precision. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Bennett Fox. Discrete optimization via marginal analysis. *Management Science*, 13(3):210–216, 1966. doi: 10.1287/mnsc.13.3.210.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, et al. The llama 3 herd of models, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

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 (NeurIPS)*, 2024.

Toshihide Ibaraki and Naoki Katoh. *Resource Allocation Problems: Algorithmic Approaches*. Foundations of Computing. MIT Press, Cambridge, MA, 1988. ISBN 978-0-262-09027-8.

Tao Ji, Bin Guo, Yuanbin Wu, Qipeng Guo, Lixing Shen, Zhan Chen, Xipeng Qiu, Qi Zhang, and Tao Gui. Towards economical inference: Enabling DeepSeek's multi-head latent attention in any transformer-based LLMs. *arXiv preprint arXiv:2502.14837*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023.

Guihong Li, Mehdi Rezagholizadeh, Mingyu Yang, Vikram Appia, and Emad Barsoum. X-EcoMLA: Upcycling pre-trained attention into MLA for efficient and extreme KV compression. *arXiv preprint arXiv:2503.11132*, 2025.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

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 (ICLR)*, 2025.

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 (ICML)*, 2024.

Haiquan Lu, Yefan Zhou, Shiwei Liu, Zhangyang Wang, Michael W. Mahoney, and Yaoqing Yang. AlphaPruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. ShortGPT: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*, 2024.

Fanxu Meng, Pingzhi Tang, Xiaojuan Tang, Zengwei Yao, Xing Sun, and Muhan Zhang. TransMLA: Migrating GQA models to MLA with full DeepSeek compatibility and speedup. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. arXiv preprint arXiv:2508.10925.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

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

Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations (ICLR)*, 2025.

xAI. Grok 2.5 open-weights release. `https://huggingface.co/xai-org/grok-2`, 2025. Model weights.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Gen Li, Ajay Jaiswal, Mykola Pechenizkiy, Yi Liang, Michael Bendersky, Zhangyang Wang, and Shiwei Liu. Outlier weighed layerwise sparsity (OWL): A missing secret sauce for pruning LLMs to high sparsity. In *International Conference on Machine Learning (ICML)*, 2024.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. ASVD: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*, 2023.

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. TurboQuant: Online vector quantization with near-optimal distortion rate. *arXiv preprint arXiv:2504.19874*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Rongzhi Zhang, Kuang Wang, Liyuan Liu, Shuohang Wang, Hao Cheng, Chao Zhang, and Yelong Shen. LoRC: Low-rank compression for LLMs KV cache with a progressive compression strategy. *arXiv preprint arXiv:2410.03111*, 2024.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Zhongzhu Zhou, Fengxiang Bie, Ziyan Chen, Zhenyu Zhang, Yibo Yang, Junxiong Wang, Ben Athiwaratkun, Xiaoxia Wu, and Shuaiwen Leon Song. CARE: Covariance-aware and rank-enhanced decomposition for enabling multi-head latent attention. In *International Conference on Learning Representations (ICLR)*, 2026.

A THE DOSERANK PROCEDURE

Algorithm 1 states DoseRank end to end (Section 3.3). Figure 2 places it inside both conversion pipelines of Section 3.1.

Algorithm 1 DoseRank: rank allocation from a measured cost

Require: operating point R and budget $B = L\bar{r}$; rank bounds $r_{\min} \le r_{\max}$; probe grid $\mathcal{G}$; calibration tasks $\mathcal{T}$ with items I_t; price $\theta \ge 0$ per unit of rank, found by bisection

build the model at $\mathbf{R} = (R, \dots, R)$; record every item's margin $m_i(\mathbf{R})$
$\tau \leftarrow \mathrm{median}_i\, |m_i(\mathbf{R})|$ {one scale for the whole run}
define $D(\cdot)$ by Equation 4 {measured loss}
for $\ell = 1$ **to** L {measure: $L\,|\mathcal{G}|$ forward passes, no gradient} **do**
  for $r \in \mathcal{G}$ **do**
    build $\mathbf{R}_{\ell \leftarrow r}$ and score the pool
    $\phi_\ell(r) \leftarrow D(\mathbf{R}_{\ell \leftarrow r}) - D(\mathbf{R})$ {Equation 5}
  end for
  complete ϕ_ℓ to a non-increasing curve on $[r_{\min}, r_{\max}]$ {Appendix C}
end for
repeat
  $r_\ell(\theta) \leftarrow \arg\min_{r_{\min} \le r \le r_{\max}} [\phi_\ell(r) + \theta r]$ for every ℓ {Equation 6}
  bisect θ
until $\sum_\ell r_\ell(\theta) \le B$ within bisection tolerance
spend the remaining $B - \sum_\ell r_\ell$ units one at a time on $\arg\max_\ell\, \delta_\ell(r_\ell)$ {Equation 7}
return $\boldsymbol{r} = (r_1, \dots, r_L)$

B EXACTNESS OF THE SOLVER

Section 3.3 solves Equation 3 with the greedy rule of Equation 7 or the price rule of Equation 6. This appendix states when the two rules return the same exact minimizer, and what the price rule returns when they do not.

Convex costs. Let every ϕ_ℓ be non-increasing and discretely convex on the integers of $[r_{\min}, r_{\max}]$, so that the saving $\delta_\ell(r) = \phi_\ell(r) - \phi_\ell(r+1)$ is non-negative and non-increasing in r. Start every layer at $r_{\min}$ and grant the remaining $B - Lr_{\min}$ units one at a time to the layer with the largest current saving. Suppose some allocation $\boldsymbol{r}'$ in $\mathcal{R}(B)$ had a strictly lower cost than the greedy result $\boldsymbol{r}$. Then some layer a has $r'_a > r_a$ and some layer b has $r'_b < r_b$. Moving one unit in $\boldsymbol{r}'$ from a back to b changes its cost by $\delta_b(r'_b) - \delta_a(r'_a - 1)$. By convexity $\delta_b(r'_b) \ge \delta_b(r_b - 1)$ and $\delta_a(r'_a - 1) \le \delta_a(r_a)$. The greedy rule granted unit r_b to layer b while layer a sat at r_a, so $\delta_b(r_b - 1) \ge \delta_a(r_a)$. The move therefore does not increase the cost, and repeating it turns $\boldsymbol{r}'$ into $\boldsymbol{r}$ without ever increasing the cost, contradicting the assumption. This is the standard argument for separable convex resource allocation (Fox, 1966; Ibaraki & Katoh, 1988). For the price rule, the first-order condition of $\phi_\ell(r) + \theta r$ at each layer is $\delta_\ell(r-1) \ge \theta \ge \delta_\ell(r)$. A common price θ therefore selects at every layer exactly the units whose saving exceeds θ, which is the set the greedy rule grants first. Bisection on θ finds the price at which these sets sum to B, and the two rules coincide.

Non-convex costs. A measured ϕ_ℓ is made non-increasing by the running minimum of Appendix C but is not convex in general. For each layer the price step $\arg\min_r [\phi_\ell(r) + \theta r]$ can return only

ranks at which ϕ_ℓ touches its lower convex envelope $\mathrm{env}(\phi_\ell)$, the largest convex function below it. A rank strictly above the envelope is dominated by a convex combination of two envelope points at every price. The procedure therefore returns an exact minimizer of $\sum_\ell \mathrm{env}(\phi_\ell)(r_\ell)$ over $\mathcal{R}(B)$, and it coincides with a minimizer of $\sum_\ell \phi_\ell(r_\ell)$ whenever every selected rank lies on its envelope. Appendix I reports, for every operating point, how much of each curve lies above its envelope and that every shipped rank lies on it.

Spectral costs as instances. With $\phi_\ell(r) = 1 - E_\ell(r)$ the cost is convex, since the spectrum is sorted in descending order, and the greedy rule is the trace-normalized greedy allocator of Section 5. Iso-energy equalizes $E_\ell(r_\ell)$ across layers and is a min-max rule rather than a summed cost, so it is tested alongside rather than as an instance. CARE's scheduler scores a layer by its depth and current rank, so it fixes no ϕ_ℓ and is not an instance of Equation 3.

C Completing the cost curve

Section 3.3 measures Equation 5 at the grid ranks and completes it into a curve on every integer rank. Let $g_1 < \cdots < g_K$ be the grid ranks together with R, at which ϕ_ℓ has been measured, with $\phi_\ell(R) = 0$ by definition. A probe is a noisy estimate, so the measured values can rise with rank, which the true cost cannot. A running minimum removes this,

$$\bar{\phi}_\ell(g_k) \;=\; \min_{j \le k} \phi_\ell(g_j), \tag{8}$$

so that more rank never costs more. The solver uses the curve that is linear in $\log r$ between grid ranks and flat beyond them,

$$\phi_\ell(r) \;=\; \begin{cases} \bar{\phi}_\ell(g_1) & r < g_1, \\ \bar{\phi}_\ell(g_k) + \big(\bar{\phi}_\ell(g_{k+1}) - \bar{\phi}_\ell(g_k)\big) \dfrac{\log r - \log g_k}{\log g_{k+1} - \log g_k} & g_k \le r \le g_{k+1}, \\ \bar{\phi}_\ell(g_K) & r > g_K. \end{cases} \tag{9}$$

The interpolation is in $\log r$ because the grid is spaced in rank halvings and doublings. The curve is flat beyond the grid so that no probe is extrapolated. The order matters: the running minimum is taken on the measured values first, and linear interpolation between non-increasing values is itself non-increasing, so the completed curve satisfies the condition the solver in Section 3.3 relies on. It is not convex in general. Appendix I reports, for every operating point, how much of the curve lies above its lower convex envelope and that every shipped rank lies on the envelope.

D Architecture and calibration

Table 3: gpt-oss-120B model and protocol. gpt-oss-20B shares the head configuration, RoPE settings, and calibration source with 24 layers (12 converted) and 32 experts (4 active).

parameters	116B MoE
layers (converted full-attn / SWA)	36 (18/18)
experts (total/active)	128/4
heads (query/KV), head dim	64/8, 64
RoPE	YaRN, θ=150000, factor 32
attention bias / sinks	yes / per-layer
$\dim \Sigma_Z = (2g-1)d$	960
eval metric	`acc_norm`
calibration	WikiText-2, 128×256=32,768 tokens (Σ_Z manifest `0d56c1a1`)

Grok-2.5 (64 layers, all converted; 8 experts, 2 active; 8 KV heads, head dimension 128) and Llama-3.1-8B-Instruct (32 layers, all converted; GQA with 8 KV heads, head dimension 128) each use their own architecture's conversion geometry and calibration. The covariance estimation passes its internal reconstruction gates on all converted layers (max eigendecomposition reconstruction error 6.1×10^{-5}). The same calibration tokens back every number in this paper, TransMLA and CARE included.

E EVALUATION SETTINGS

All arms are evaluated with the same harness build on the same hardware, in BF16 activations over the released checkpoints, with MXFP4 experts on gpt-oss. Prompts are zero-shot and the seed is 0. Checkpoint snapshots are pinned per model, and tokenizer parity between the converted and the released model is checked on gpt-oss. Batch size is fixed per model: 1 on gpt-oss-120B, 48 on gpt-oss-20B and 64 on Llama-3.1-8B. The evaluation splits are the test split for ARC-Easy, ARC-Challenge and MMLU and the validation split for HellaSwag, OpenBookQA, PIQA and WinoGrande, 31,237 items in total: 2376, 1172, 14042, 10042, 500, 1838 and 1267 in that order.

F CARE REPRODUCTION VERIFICATION

Our CARE decomposition is the MLA-pipeline CARE decomposition (joint whitened SVD of the latent read-out), ported from the official FutureMLS-Lab/CARE repository (commit `3b453464`) to each model's own geometry, with the paper's $\sqrt{\cdot}$ (p=0.5) scheduler weighting for CARE-E. We verified this port against the official code's own reported numbers rather than trusting the port by construction. Running the official repository directly on Llama-3.1-8B-Instruct at $\bar{r}$=512, its released zero-shot per-branch path (sqrt-whitened SVD of K and V separately, KV-parity) reproduces the CARE paper's own Table 1 at that budget (the paper's uniform-rank row 61.63 vs. the paper's 62.33; CARE-E 57.92 vs. 57.82). Its released MLA pipeline reproduces the CARE paper's TransMLA row (55.62 vs. 55.52) but its own CARE variants do not reproduce the paper under either weighting (38.6–46.8 vs. the paper's 57.0), which is why we port the MLA-pipeline decomposition but the paper's own scheduler weighting rather than the released code's default. Full-rank identity gates (CARE at full rank should match TransMLA at full rank, since both reduce to the unconverted model) passed on all four models. On Llama-3.1-8B specifically, CARE's full-rank per-task accuracy matched TransMLA's full-rank per-task accuracy within 0.3 pp on every task.

Our own pipeline's TransMLA cell at the same rank agrees with this official run on the tasks the two harnesses score identically: ARC-Easy, ARC-Challenge, PIQA, and OpenBookQA agree within 0.6 pp (66.62 vs. 66.08; 42.92 vs. 43.17; 74.65 vs. 74.92; 39.20 vs. 38.80). The remaining per-task gaps are protocol, not conversion, artifacts: HellaSwag differs by 1.4 pp from the calibration set (wiki spans in the official run vs. our random spans), and WinoGrande by 5.5 pp from an acc-vs-acc_norm harness offset present at a similar size (4.3 pp) between our own unconverted model and the CARE paper's own unconverted row. MMLU differs by 10.7 pp, but unlike WinoGrande this gap does *not* appear between the two unconverted models (they agree within 0.3 pp): the CARE paper's TransMLA row and its own original row evidently use different MMLU settings from each other. Our own MMLU numbers are internally consistent across every cell in this paper (`analysis/EXTERNAL_VALIDATION_2026-09-22.md` in the mla-allocators repository has the full cross-check).

G FULL ALLOCATOR TABLES

Table 4 gives the absolute macro accuracy of every arm behind Table 1, one pair of sub-tables per model, together with the operating point R each DoseRank cell was probed at and the rank floor $r_{\min}$ it was solved under. Iso-energy, the spectral rule tested in Section 5, is listed on both decompositions.

Table 4: Every arm behind Table 1: macro accuracy (%), one pair of sub-tables per model. R is the operating point whose probes the DoseRank cell was fitted from and $r_{\min}$ its rank floor (Section 4.1). Iso-energy is the spectral rule of Section 5. Δ is DoseRank against uniform ranks (TransMLA's decomposition) or against CARE's scheduler (CARE's decomposition), item-paired 95% CI, * when the interval excludes zero. Bold marks the higher of the decomposition's own allocator and DoseRank.

(a) gpt-oss-120B, vanilla = 57.49

TransMLA decomposition

$\bar{r}$	R	$r_{\min}$	uniform	iso	DoseRank	Δ [95% CI]
64	64	32	48.90	49.26	**53.25**	+4.35 [3.79, 4.91]*
96	128	32	50.01	50.88	**54.56**	+4.55 [3.94, 5.14]*
128	128	32	50.99	52.73	**56.51**	+5.52 [4.92, 6.12]*
192	256	64	53.87	54.19	**57.63**	+3.77 [3.20, 4.33]*
256	256	64	56.07	55.96	**58.03**	+1.96 [1.41, 2.52]*
320	256	64	56.90	57.00	**58.85**	+1.95 [1.37, 2.54]*
384	256	64	57.63	57.34	**59.73**	+2.10 [1.57, 2.64]*

CARE decomposition

$\bar{r}$	R	$r_{\min}$	CARE-E	iso	DoseRank	Δ [95% CI]
64	64	32	48.90	48.10	**50.34**	+1.45 [0.98, 1.91]*
96	128	32	49.92	49.78	**52.45**	+2.53 [1.98, 3.08]*
128	128	32	51.69	51.35	**54.34**	+2.64 [2.10, 3.20]*
192	256	64	53.28	53.49	**55.90**	+2.62 [2.10, 3.15]*
256	256	64	54.20	55.09	**57.28**	+3.08 [2.54, 3.62]*
320	256	64	54.66	56.26	**58.09**	+3.42 [2.86, 3.98]*
384	256	64	55.25	57.19	**58.50**	+3.25 [2.73, 3.79]*

(b) gpt-oss-20B, vanilla = 55.54

TransMLA decomposition

$\bar{r}$	R	$r_{\min}$	uniform	iso	DoseRank	Δ [95% CI]
64	64	32	43.79	44.22	**45.87**	+2.08 [1.59, 2.59]*
96	128	32	44.85	44.29	**47.08**	+2.24 [1.70, 2.78]*
128	128	64	45.54	46.21	**47.21**	+1.66 [1.19, 2.14]*
192	256	64	47.57	48.31	**50.47**	+2.90 [2.37, 3.44]*
256	256	64	50.23	49.80	**52.50**	+2.27 [1.77, 2.78]*
320	256	64	51.07	50.98	**53.45**	+2.37 [1.87, 2.88]*
384	256	64	52.01	51.34	**54.40**	+2.39 [1.89, 2.90]*

CARE decomposition

$\bar{r}$	R	$r_{\min}$	CARE-E	iso	DoseRank	Δ [95% CI]
64	64	32	45.36	45.63	**45.85**	+0.49 [0.02, 0.94]*
96	128	32	45.96	46.45	**47.45**	+1.49 [0.96, 2.03]*
128	128	64	47.41	47.81	**48.20**	+0.78 [0.29, 1.29]*
192	256	64	48.28	49.57	**50.29**	+2.01 [1.48, 2.55]*
256	256	64	48.82	50.86	**51.15**	+2.33 [1.80, 2.88]*
320	256	64	49.35	51.74	**51.80**	+2.45 [1.93, 2.97]*
384	256	64	50.39	51.88	**52.50**	+2.10 [1.65, 2.56]*

(c) Grok-2.5 (C=2), vanilla = 74.41

TransMLA decomposition

$\bar{r}$	R	$r_{\min}$	uniform	iso	DoseRank	Δ [95% CI]
768	768	256	65.89	69.03	**71.46**	+5.57 [4.63, 6.50]*
896	768	256	68.29	71.14	**72.73**	+4.44 [3.57, 5.31]*
1024	768	256	70.11	72.16	**73.07**	+2.96 [2.16, 3.77]*
1280	1280	512	73.01	74.14	**73.84**	+0.83 [0.13, 1.56]*
1536	1280	512	**73.94**	74.33	73.54	−0.40 [−1.09, 0.27]

CARE decomposition

$\bar{r}$	R	$r_{\min}$	CARE-E	iso	DoseRank	Δ [95% CI]
768	768	256	69.01	73.80	**74.53**	+5.51 [4.66, 6.37]*
896	768	256	71.19	73.94	**74.71**	+3.53 [2.79, 4.29]*
1024	768	256	72.96	74.04	**74.67**	+1.71 [1.06, 2.37]*
1280	768	256	73.77	74.50	**74.63**	+0.86 [0.31, 1.39]*
1536	768	256	74.19	74.27	**74.63**	+0.44 [−0.03, 0.91]

(d) Llama-3.1-8B-Instruct, vanilla = 65.26

TransMLA decomposition

$\bar{r}$	R	$r_{\min}$	uniform	iso	DoseRank	Δ [95% CI]
128	128	32	34.44	33.61	**35.73**	+1.29 [0.73, 1.83]*
192	192	48	37.71	35.83	**43.09**	+5.38 [4.74, 6.04]*
256	256	64	41.31	39.11	**49.25**	+7.94 [7.25, 8.64]*
384	384	96	50.10	47.67	**56.32**	+6.22 [5.54, 6.91]*
512	512	128	55.66	53.76	**59.35**	+3.69 [3.04, 4.34]*
768	768	192	60.94	61.27	**63.20**	+2.26 [1.72, 2.80]*
1024	1024	256	**63.44**	63.41	62.68	−0.76 [−1.31, −0.21]*

CARE decomposition

$\bar{r}$	R	$r_{\min}$	CARE-E	iso	DoseRank	Δ [95% CI]
128	128	32	34.41	32.95	**34.60**	+0.19 [−0.29, 0.67]
192	256	64	37.00	33.42	**38.64**	+1.65 [1.15, 2.14]*
256	256	64	38.87	34.15	**41.71**	+2.84 [2.27, 3.39]*
384	256	64	43.60	38.56	**46.84**	+3.25 [2.63, 3.87]*
512	512	128	47.89	43.56	**51.81**	+3.92 [3.30, 4.55]*
768	512	128	53.03	53.46	**58.83**	+5.80 [5.17, 6.44]*
1024	1024	256	56.10	57.46	**61.75**	+5.65 [5.01, 6.29]*

H PER-TASK RESULTS

Table 4 (Appendix G) reports macro `acc_norm` for every arm. A per-task breakdown is in Table 2 for gpt-oss-120B's calibration-task ablation (both decompositions), and per-task columns for every arm, model, and budget are in the registry CSVs in the supplementary material. As background on the protocol: zero-shot loglikelihood `acc_norm` leaves MMLU near its 25% choice floor for these checkpoints on this suite, so macro deltas are typically driven mainly by the well-separated tasks (ARC, HellaSwag, WinoGrande). All allocator contests are paired per-item across the full suite, so near-floor tasks add variance but no bias.

Arms that exceed the unconverted model. On two models several converted arms score above vanilla on this suite: DoseRank is +0.54, +1.36 and +2.24 pp above vanilla gpt-oss-120B at $\bar{r}$=256, 320 and 384, uniform ranks are also above it at $\bar{r}$=384, and on CARE's decomposition every Grok-2.5 DoseRank cell sits within a few tenths of vanilla or above it. We report this and do not claim it.

Under the near-floor protocol above, small positive offsets against the unconverted model are within the range this protocol can produce for reasons unrelated to compression, and an identity-conversion control that would separate a genuine effect from an evaluation-path artifact is not part of this study. Nothing in Section 4.2 depends on it: every contrast we claim is between two arms that differ only in the rank vector.

Released-code CARE-E scheduler variant (`care_e_p4`). An early CARE-E run on this pipeline's CARE decomposition used the released code's default per-layer weighting, an H^4 power on the scheduler's importance score; cross-checking against the official CARE repository's own paper-reproduction run (Appendix F) showed this weighting does not reproduce the published CARE paper, which uses a $\sqrt{\cdot}$ (p=0.5) weighting instead. Table 4's CARE-E column uses only the corrected, paper-faithful scheduler; the released-code variant is reported here instead, macro accuracy (%): gpt-oss-120B $\bar{r}\in\{64, 96, 128, 192, 256, 320, 384\}$: $48.85, 50.09, 48.20, 53.82, 50.94, 55.98, 56.71$; gpt-oss-20B $\bar{r}\in\{128, 256\}$: $47.22, 51.04$ (other budgets not run under this variant).

I PROBE SIGNAL SATURATION AT HIGH BUDGETS

Table 1 shows DoseRank not winning at exactly one budget each on Grok-2.5 and Llama-3.1-8B: Grok-2.5 at $\bar{r}$=1536 (a non-significant loss) and Llama-3.1-8B at $\bar{r}$=1024 (a significant one). We diagnose this directly from the calibration probes, no test data: at each operating point, the fraction of single-layer cut/restore probes whose soft-accuracy change is distinguishable from calibration noise ($|\Delta|$ exceeds twice the item-paired standard error over the 2044 (Llama) or 1029 (Grok) calibration items), the mean restore gain, and the wrong-item signature that exposes gold-NLL's hedging failure: among cut probes the statistic scores as an apparent gain, the share of that gain coming from items the base model already gets wrong becoming *less* wrong, rather than from any real improvement (a share above 1 means base-*correct* items are being hurt on net).

Table 5: Probe-level diagnostics by operating point, computed from calibration probes only (item-paired weighted means across probe files where more than one covers a point). Grok-2.5's highest directly probed point is $\bar{r}$=1280; 1536 is a factor-of-1.2× extrapolation along the same trend.

model	$\bar{r}$	cut sig. frac.	restore sig. frac.	mean restore (pp)	app.-gain frac.	share base-wrong
Llama-3.1-8B	128	0.23	0.52	+0.10	0.25	1.62
Llama-3.1-8B	192	0.64	0.72	+0.24	0.06	1.13
Llama-3.1-8B	256	0.69	0.72	+0.33	0.07	3.81
Llama-3.1-8B	384	0.69	0.65	+0.29	0.08	4.44
Llama-3.1-8B	512	0.64	0.51	+0.22	0.10	1.13
Llama-3.1-8B	768	0.52	0.39	+0.09	0.17	10.27
Llama-3.1-8B	1024	0.42	0.29	+0.04	0.29	7.91
Grok-2.5 (C=2)	768	0.24	0.15	+0.05	0.47	2.06
Grok-2.5 (C=2)	1280	0.28	0.11	+0.01	0.32	11.64

Both losing budgets sit where the signal has already faded well below where it started: Llama-3.1-8B's restore-significant fraction falls from its peak near $\bar{r}\in\{192, 256\}$ (0.72) to 0.29 at op1024, its mean restore gain from $+0.33$ pp (op256) to $+0.04$ pp, while the apparent-gain fraction rises from 0.07 to 0.29 and that apparent gain is increasingly explained by base-wrong items rather than real recovery (share 7–10× at op768/1024). Grok-2.5's directly probed points show the same trend continuing past its own strongest signal: restore-significant fraction and mean restore gain both keep falling from op768 to op1280, with the apparent-gain share from base-wrong items reaching 11.64 at op1280 (2044/1029 calibration items, well above chance variation). At both models' losing budgets, DoseRank is effectively allocating rank against noise rather than a real per-layer damage signal.

Consistent with this, the teacher-KL statistic (`dose_kl`) ties TransMLA at both budgets instead of losing: Llama-3.1-8B at $\bar{r}$=1024, $+0.24$ pp, CI $[-0.17, +0.66]$; Grok-2.5 at $\bar{r}$=1536, -0.04 pp, CI $[-0.56, +0.50]$. We report this as corroborating evidence for the diagnosis above, not as a second method: the main tables and Figure 3 use soft accuracy (`dose_sa`) throughout, with no per-budget statistic substitution.

Table 6: Joint (multi-layer) vs. additive-sum predictions from the same single-layer probes, TransMLA decomposition, soft-accuracy statistic. This tests the separability the surrogate of Section 3.2 assumes. Several layers are cut or restored together from a uniform base and evaluated jointly; the additive prediction sums the corresponding single-layer probe costs, so a ratio of 1 means the joint effect matches the additive sum exactly. Predicted is that additive sum, signed as accuracy-point damage ($+100\times$ the predicted change in soft accuracy), so small values flag the least numerically stable ratios. Grok-2.5 C=1 is a distinct carry configuration from the main results (C=2), run only for this check; the gold-NLL companion statistic is below.

Model	Base $\bar{r}$	Config	Layers	Predicted (pp)	Meas./pred. (soft acc)
gpt-oss-120B	256	cut 6 least-sensitive	6	-0.37	0.75
		cut 3 most-sensitive	3	$+1.96$	1.05
		restore 6 best	6	-3.69	0.92
		cut-least + restore-best	10	-2.55	1.18
Grok-2.5 C=1	768	cut 8 least-sensitive	8	-0.78	0.73
		cut 4 most-sensitive	4	$+3.71$	0.97
		restore 8 best	8	-2.55	0.97
		cut-least + restore-best	16	-3.34	0.70

Additivity under soft accuracy. We compare the additive sum of several single-layer probe costs against the same layers actually moved together and re-evaluated jointly (Table 6). gpt-oss-120B stays close to additive throughout (0.75–$1.18\times$). Grok-2.5 C=1 (a distinct carry configuration from the main results, run only for this check) is close to additive for the single-direction moves ($0.97\times$, $0.97\times$) and moderately sub-additive for the cut-least and combined moves ($0.73\times$, $0.70\times$); the sub-additive combined move on Grok is consistent with the saturation diagnosis above, though this small-scale check does not on its own establish that link. The table also lists the predicted effect sizes, which flag the least stable ratios.

Additivity under gold-NLL. Table 6 above reports the joint-vs.-additive check under the soft-accuracy statistic; the same check under gold-NLL, the raw per-item statistic soft accuracy is built to avoid (Equation 4 and its discussion), mostly agrees on direction but not on size. On gpt-oss-120B (base $\bar{r}$=256) gold-NLL agrees with soft accuracy's near-additive restore and most-sensitive-cut ratios (add6best $0.92\times$, cut3most $0.97\times$, both within 0.08 of soft accuracy) but is more sub-additive on the least-sensitive cut ($0.28\times$ vs. soft accuracy's $0.75\times$). On Grok-2.5 C=1 (base $\bar{r}$=768) gold-NLL is sub-additive at all four configurations, somewhat more so than soft accuracy throughout (0.58/0.90/0.73/$0.55\times$ vs. soft accuracy's 0.73/0.97/0.97/$0.70\times$). Gold-NLL's own hedging failure (Section 5) makes it a noisier ruler for this check specifically, not evidence that the soft-accuracy additivity pattern above is itself an artifact of the statistic.

Not a convexification artifact. The price rule of Section 3.3 allocates on the lower convex envelope of each measured cost ϕ_ℓ (Appendix B). If the two losses came from structure the envelope discards, that discarded structure would be largest at the losing budgets. It is smallest there (Table 7). On Llama-3.1-8B the summed gap between the curves and their envelopes falls from $1.66\,\mathrm{pp}$ at $\bar{r}$=384, its highest across all seven budgets, to $0.79\,\mathrm{pp}$ at $\bar{r}$=1024, the lowest, and the one losing budget. On Grok-2.5 it falls from $2.68\,\mathrm{pp}$ at the op-768 probes to $1.74\,\mathrm{pp}$ at op-1280, and the win at 1280 and the loss at 1536 are fit from the same curves, so no curve-shape metric can separate them. What does change is the signal itself: total measured damage falls from $18.1\,\mathrm{pp}$ to $9.5\,\mathrm{pp}$ on Llama-3.1-8B between those budgets, and the number of zero-signal layers on Grok-2.5 doubles from 8 to 17 of 64. Re-solving every reported allocation from the curves reproduces the shipped rank vectors exactly, and every chosen rank sits on its envelope, so the envelope costs nothing at the operating allocation. The solver is optimizing the curves it was given. At these budgets the curves have little left to say.

Layers with no measurable signal. A layer none of whose probes moves the measured loss beyond noise has, after the running minimum, a cost that is identically zero. The price step $\arg\min_r[\phi_\ell(r)+\theta r]$ then selects $r_{\min}$ for it at every positive price, so such layers are floored by the absence of a measurement rather than by evidence that they tolerate the floor. Table 7 counts them per operating point. A uniform-rank fallback for them is the untested fix named in Section 4.2.

Table 7: Lower-convex-envelope diagnostics, TransMLA decomposition, one row per model $\times$ operating point R (rows split where the same R's curves differ across the target budgets it serves). *Structure discarded* is $\sum_\ell \max_r(\phi_\ell(r) - \mathrm{env}(\phi_\ell)(r))$ over $r \in [r_{\min}, r_{\max}]$, the lower convex envelope taken in r; *frac. non-convex* is the share of layers where that max gap exceeds 10^{-9}; *zero-signal layers* are layers whose curve is identically zero; *total signal* is $\sum_\ell \max_r |\phi_\ell(r)|$. Re-solving every row's allocation from its curves reproduces the shipped rank vector exactly (agreement 1.000 on all 16 rows shown, and on all 26 model$\times$budget cells before this collapsing). Source: `mla-allocators/data/results/ablations/probe_signal_saturation/nonconvexity.csv`.

model	R	budgets served	DoseRank $-$uniform (pp)	structure discarded (pp)	frac. non-convex	zero-signal layers	total signal (pp)
gpt-oss-120B	64	64	$+4.35$	1.79	94%	1/18	13.39
gpt-oss-120B	128	96, 128	$+4.55, +5.52$	2.18	94%	1/18	10.18
gpt-oss-120B	256	192, 256, 320, 384	$+3.76, +1.96, +1.95, +2.10$	1.71	94%	1/18	7.15
gpt-oss-20B	64	64	$+2.08$	0.66	100%	0/12	7.47
gpt-oss-20B	128	96	$+2.23$	0.72	100%	0/12	6.94
gpt-oss-20B	128	128	$+1.67$	0.58	100%	0/12	6.86
gpt-oss-20B	256	192, 256, 320, 384	$+2.90, +2.27, +2.38, +2.39$	1.18	100%	0/12	5.55
Grok-2.5 (C=2)	768	768, 896, 1024	$+5.57, +4.44, +2.96$	2.68	70%	8/64	15.38
Grok-2.5 (C=2)	1280	1280, 1536	$+0.83, -0.40$	1.74	53%	17/64	14.12
Llama-3.1-8B	128	128	$+1.29$	0.58	97%	1/32	6.10
Llama-3.1-8B	192	192	$+5.38$	0.90	97%	1/32	13.33
Llama-3.1-8B	256	256	$+7.94$	1.32	97%	1/32	17.91
Llama-3.1-8B	384	384	$+6.22$	1.66	100%	0/32	18.14
Llama-3.1-8B	512	512	$+3.69$	1.55	91%	1/32	16.34
Llama-3.1-8B	768	768	$+2.26$	1.35	97%	1/32	12.37
Llama-3.1-8B	1024	1024	-0.76	0.79	84%	2/32	9.55

J SERVING CONFIGURATION

We patched MLA-converted gpt-oss-120B into SGLang (Zheng et al., 2024) with a per-layer KV pool and measured it on one 8-GPU node in data-parallel 8. Each request has a 60K-token prompt and one output token, which makes the run a KV-capacity test rather than a kernel benchmark.

Concurrency ceiling (120B). Figure 4 shows total token throughput versus offered concurrency for vanilla GQA and for the converted model at two budgets, $\bar{r}$=320 (62.5% KV cut) and $\bar{r}$=192 (75.0% KV cut, the DoseRank allocation of Table 1). Vanilla saturates its KV memory near 300 concurrent requests and collapses. The $\bar{r}$=320 model sustains roughly 900, a measured $3.0\times$ ceiling against an analytic byte ceiling of $2.67\times$. The $\bar{r}$=192 model sustains 1100 and collapses at 1200, a measured $4.0\times$ ceiling that matches its byte ceiling of $4.0\times$. We size the KV pool to the byte ceiling in both cases. Below saturation the prototype's slower kernels leave both converted models under vanilla. We have not isolated why the $\bar{r}$=320 ceiling exceeds the byte ratio. Paging granularity and scheduler admission are plausible contributors, and we flag the gap rather than explain it away.

Serving accuracy parity (120B). Accuracy evaluated through the serving engine at $\bar{r}$=320 matches offline evaluation, with paired 95% CIs that include zero, so the throughput numbers at that budget are not purchased with a degraded numerical path.

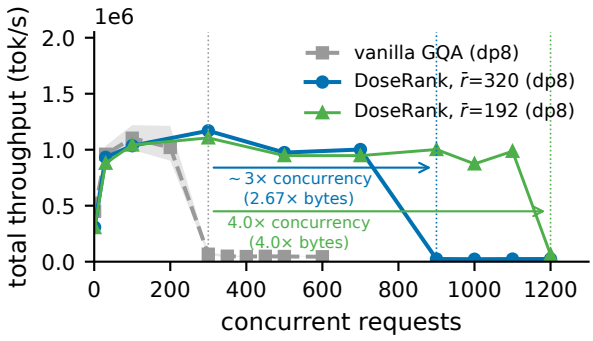


Figure 4: Serving-capacity cliff on gpt-oss-120B (data-parallel 8, 60K-token requests, three seeds, median with min to max band). Total token throughput stays near 1M tokens per second until the KV pool fills, then collapses. Vanilla GQA collapses near 300 concurrent requests, $\bar{r}$=320 near 900 (2.67× the KV bytes) and $\bar{r}$=192 near 1200 (4.0×).

K LIMITATIONS

Every result is a one-shot conversion, so absolute accuracy sits below the unconverted model, and whether the margin survives recovery training is untested. The suite is zero-shot multiple choice, so generation and long-context behavior are unmeasured. Intervals cover item-sampling noise at one calibration seed. They are unadjusted in Table 1, with the Holm-adjusted outcome reported in Section 4.2. The operating-point reuse rule and the rank floor of Section 4.1 are design choices rather than measured limits. The uniform-rank fallback for layers with no measurable signal is unevaluated. The CARE comparison uses CARE's released decomposition with the paper's scheduler weighting rather than its per-branch path (Appendix F). Serving throughput is measured only for a uniform-rank conversion.