ROLE-AWARE QUANTIZATION FOR RELIABLE TRANSFORMER INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantization is widely used to reduce Transformer memory usage and accelerate inference. Prior work has commonly used uniform activation quantization, assigning the same bit-width across activations. However, Transformer components perform distinct functions, including query–key routing, value transport, feed-forward computation, and normalization. We show that these functional roles differ in sensitivity to quantization error and therefore require different precision levels to preserve predictions. To allocate precision efficiently under a fixed budget, we introduce Role-Aware Reliability (RAR) allocation. RAR quantizes one layer–role activation site at a time and measures the resulting change in the next-token distribution using Kullback–Leibler (KL) divergence. The resulting functional risk atlas pinpoints where low precision most strongly distorts predictions. Building the functional risk atlas for a 7B checkpoint requires only a small sample of calibration text and approximately one to two minutes. Our allocation rule, RAR-log, uses the atlas to assign higher precision to high-risk sites and lower precision to low-risk sites under a fixed cost-weighted average bit-width budget. By accounting for how quantization errors in different roles affect predictions, RAR complements SmoothQuant-style activation smoothing. Our theoretical analysis explains how quantization errors from different sites combine and proves bounds on the resulting changes in model predictions. With the same target precision budget, the combined policy, Smooth-RAR-log, removes an average of 94.1% of the language-modeling degradation caused by uniform 6-bit activations, compared with 87.4% for smoothing alone. The combined policy improves on smoothing alone in all 24 paired model–corpus evaluations, covering three corpora and eight Transformer checkpoints ranging from 124M to 7B parameters. Even with packed activations and 8-bit quantized weights, Smooth-RAR-log lowers next-token prediction loss compared with smoothing alone across the evaluated models. Smooth-RAR-log also improves average accuracy on multiple-choice benchmarks.

1 INTRODUCTION

Quantization has been widely used to accelerate Transformer inference and reduce memory use (Xiao et al., 2023; Frantar et al., 2023; Lin et al., 2024). In practice, memory traffic and key–value (KV) cache capacity are major bottlenecks in autoregressive inference (Pope et al., 2023; Kwon et al., 2023). Quantizing activations to lower precision reduces the memory needed for activation storage but introduces rounding errors. A common approach is uniform activation quantization, which assigns the same precision to every activation site using the same number of bits per value.

However, activation sites in Transformer language models (Vaswani et al., 2017) serve distinct functional roles. Query (Q) and key (K) activations, which together implement query–key (QK) routing, determine how much attention each token gives to tokens in the token's context. Value (V) activations carry the information that attention combines. Multilayer perceptron (MLP) layers transform features of each token and the token's context, while normalization outputs regulate feature scales. Quantization errors in QK distort the distribution of attention across tokens, while errors in V distort the information carried by the attended tokens. Uniform activation quantization ignores these functional differences and the corresponding differences in sensitivity to quantization error.

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.

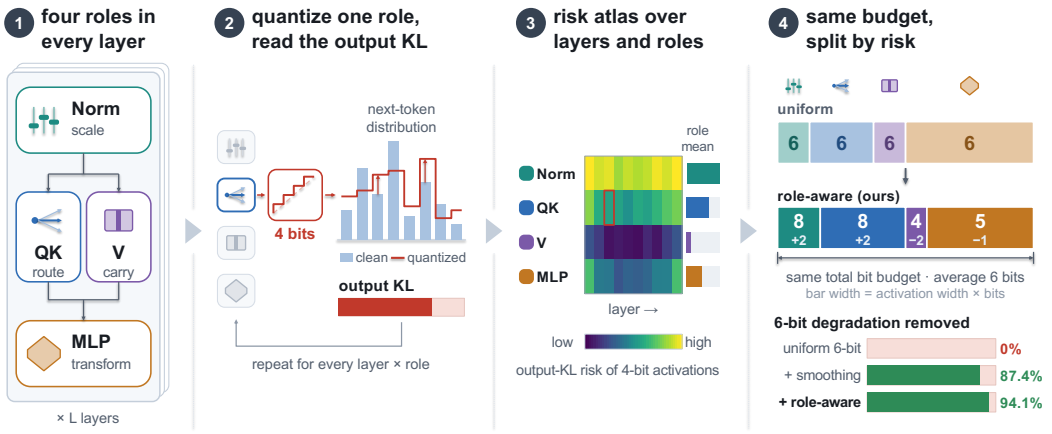

Figure 1: Pipeline of RAR. We identify functionally distinct activation sites, quantize one site at a time, and measure the KL divergence from the clean next-token distribution. The resulting risk atlas guides precision allocation under a fixed cost-weighted average bit-width budget.

In this study, we formulate activation quantization as a problem of matching precision to quantization sensitivity under a fixed budget. We introduce Role-Aware Reliability (RAR) allocation to measure quantization sensitivity at each layer–role activation site and allocate precision accordingly. RAR quantizes one site at a time and measures the Kullback–Leibler (KL) divergence between the clean next-token distribution and the perturbed distribution. The resulting single-site measurements form a functional risk atlas: a layer–role map showing where quantization changes predictions most. We then use the atlas to allocate activation precision under a cost-weighted average bit-width constraint. Figure 1 summarizes how RAR measures single-site output KL and allocates activation precision. RAR-log is the core allocation rule; combining RAR-log with activation smoothing gives Smooth-RAR-log, our main combined policy. Post-smoothing recalibration refines the atlas used by Smooth-RAR-log.

RAR complements post-training quantization (PTQ) methods that choose quantization boundaries or transform weights, activations, or caches (Xiao et al., 2023; Frantar et al., 2023; Lin et al., 2024; Shao et al., 2024; Ashkboos et al., 2024; Liu et al., 2025). At the activation boundaries used by these PTQ methods, RAR measures how quantization at each site changes predictions and uses the measured output changes to distribute the activation bit-width budget. Xu et al. (2026) similarly use output-facing perturbations for value-cache allocation in RippleKV, whereas RAR covers layer–role boundaries throughout the Transformer. Appendix B compares RAR with sensitivity-based mixed-precision allocation and reviews related PTQ, KV-cache quantization, and Transformer stability work.

Our experiments reveal three main patterns that demonstrate the value of role-aware precision allocation. First, sensitivity to quantization error differs across functional roles. Second, smoothing reduces quantization error but leaves heterogeneous residual risk, allowing further gains from allocation and post-smoothing recalibration. Third, the allocation benefit persists with packed integer activations and 8-bit weight quantization (W8). Our analyses in Secs. 4.2 and 5 show how QK errors in early layers alter attention routing, are amplified in later layers, and degrade free-running generation, where tokens generated by the model serve as context for subsequent predictions. Fixing QK activations at uniform precision and reallocating the protected QK bits to other roles largely eliminates the allocation benefit.

Our contributions are as follows.

- We introduce a functional risk atlas that pinpoints where activation quantization most strongly distorts predictions across Transformer layers and roles.
- We propose RAR-log, a mixed-bit allocation method that assigns higher precision to high-risk sites under a fixed cost-weighted budget. We combine RAR-log with activa-

tion smoothing in Smooth-RAR-log and refine the combined method's risk atlas through post-smoothing recalibration.

- We prove upper bounds on how much model predictions can change when multiple sites are quantized. Our analysis establishes when single-site risks add and quantifies the allocation gains from differences in risk per unit cost.
- We evaluate RAR across eight checkpoints from 124M to 7B parameters on language modeling (LM) and multiple-choice tasks, with matched-budget baselines, packed-activation kernels, and W8 PTQ evaluations. Smooth-RAR-log consistently lowers LM loss compared with smoothing alone across all 24 model–corpus evaluations and improves average multiple-choice accuracy at the same precision budget.

2 Functional Risk Atlas

2.1 Role-Specific Activation Sites

For an input sequence x of T tokens, a Transformer language model with parameters θ produces logits and next-token distributions at each position:

$$a_\theta(x) \coloneqq \{a_{\theta,t}(x)\}_{t=1}^{T}, \qquad p_{\theta,t}(\cdot \mid x) \coloneqq \mathrm{softmax}(a_{\theta,t}(x)). \tag{1}$$

We call each layer–role activation tensor a quantization site and index each site by $c \coloneqq (\ell, r, i_{\mathrm{seg}})$, where ℓ is a layer, r is a functional role, and i_{seg} optionally indexes a projection segment. Individual tensor dimensions are referred to as feature coordinates or feature channels. The layer–role activation sites cover the Transformer's different functional roles, including QK routing, value transport, attention and MLP outputs, MLP hidden states, and normalization. Appendix D.2 gives the detailed role sets and architecture-specific hook boundaries.

2.2 Quantization-Induced Risk

The goal is to identify where lower precision most strongly changes the model's next-token predictions. We compare predictions on the same input with and without quantizing a single activation site. Measuring the change at the output captures how the site's quantization error affects predictions after propagating through later computation.

Let $z_c(x)$ denote the activation at site c, and let p_θ^{c,Q_m} denote the output distribution when only site c is quantized and all other computation is unchanged. For activation quantization, the perturbation is $\delta_c^{(m)}(x) \coloneqq Q_m(z_c(x)) - z_c(x)$, where Q_m is symmetric per-token dynamic fake quantization to m bits: Activations are rounded to an m-bit grid and reconstructed in floating point (FP). The quantizer is a zero-point-free specialization of the affine integer quantization map described by Jacob et al. (2018). The single-site quantization risk is

$$A_c^{(m)} \coloneqq \mathbb{E}_{x\sim\mathcal{D}_{\mathrm{cal}}}\left[\frac{1}{T(x)-1}\sum_{t=1}^{T(x)-1} D_{\mathrm{KL}}(p_{\theta,t}(\cdot \mid x)\|p_{\theta,t}^{c,Q_m}(\cdot \mid x))\right]. \tag{2}$$

Eq. 2 averages the KL divergence over prediction positions and calibration inputs. A higher $A_c^{(m)}$ means a larger average change in the next-token distribution when site c is quantized to m bits. Collecting the single-site risk scores across layers and roles gives the functional risk atlas used for precision allocation in Sec. 3.

We use $A_c^{(4)}$ as the main allocation score and also measure risk curves from 3 to 8 bits. Throughout this paper, KL divergence and negative log-likelihood (NLL) use natural logarithms. We measure single-site quantization risks directly from forward passes. Appendix C.1 relates risk to numerical error and downstream sensitivity and explains why cross-site interactions can invalidate additivity.

The 2^{-2m} rate used in RAR-log's allocation objective (Eq. 3) follows the classical high-resolution squared-error quantization scaling described by Gray & Neuhoff (1998). In the allocation model, the 2^{-2m} rate means that each additional bit reduces a site's predicted risk by a factor of four. Our finite-bit analysis establishes a corresponding output-KL upper bound under downstream Lipschitz assumptions. Appendix C presents the finite-bit, decision-stability, sequential-policy, and multi-token choice-score results.

3 RELIABILITY ALLOCATION

Our RAR framework turns the functional risk atlas built from the single-site measurements in Eq. 2 into an activation-precision policy. The goal is to preserve the model's predictions by assigning higher precision to sites that are more sensitive to quantization, within a fixed bit budget. RAR-log is the core allocation rule, and Smooth-RAR-log combines RAR-log with activation smoothing. Post-smoothing recalibration updates the atlas by remeasuring each site's output KL after smoothing. Decision-aware extensions such as RAR-task also penalize increases in target-token NLL, reductions in the original top-ranked token's logit margin, and changes in which token ranks first.

3.1 COST-WEIGHTED AVERAGE BIT-WIDTH

Activation sites have unequal widths, so adding one bit costs more at a wider site: The extra bit must be stored for more activation values. To account for differences in activation-width cost, we define the cost-weighted average bit-width $\bar{m} \coloneqq (\sum_c \kappa_c m_c)/(\sum_c \kappa_c)$, where m_c is the assigned bit-width and κ_c is the activation-width cost. For grouped-query attention (GQA) and multi-query attention, κ_c uses the actual projection output width. Equivalently, a target average bit-width $\bar{m}$ corresponds to the total cost-weighted bit budget $B \coloneqq \bar{m} \sum_c \kappa_c$.

3.2 RAR-LOG

RAR-log allocates precision by balancing the predicted reduction in output distortion against the activation cost of extra bits.

Continuous allocation. To compare candidate allocations, we model how each site's risk falls as the site's precision increases. We combine the 2^{-2m_c} rate from the finite-bit single-site output-KL analysis with measured atlas coefficients. We fix the reference precision $m_0 \coloneqq 4$, write $A_c \coloneqq A_c^{(m_0)}$, and model the risk of site c at m_c bits as $A_c 2^{-2(m_c - m_0)}$. Under the single-site risk model, each additional bit reduces the predicted site risk by a factor of four. Summing the modeled single-site risks gives the relaxed allocation objective

$$\min_{\{m_c\}} \sum_c A_c 2^{-2(m_c - m_0)} \quad \text{s.t.} \quad \sum_c \kappa_c m_c \le B, \qquad m_{\min} \le m_c \le m_{\max}. \tag{3}$$

The next proposition gives the allocation that minimizes the additive risk surrogate under a fixed budget and shows why risk must be weighed against activation width.

Proposition 1 (Continuous RAR-log optimum)**.** *Suppose $A_c > 0$ and $\kappa_c > 0$ for every site and $m_{\min} \sum_c \kappa_c < B < m_{\max} \sum_c \kappa_c$. Then Eq. 3 has a unique minimizer. The minimizer uses the full budget, and for some scalar τ the minimizer is*

$$m_c^\star = \mathrm{clip}_{[m_{\min}, m_{\max}]} \left(\tau + \frac{1}{2} \log_2 \frac{A_c}{\kappa_c} \right), \tag{4}$$

where τ is chosen so that $\sum_c \kappa_c m_c^\star = B$. The endpoint budgets give the trivial all-$m_{\min}$ or all-$m_{\max}$ solutions.

Eq. 4 assigns precision according to measured risk per unit activation cost. Before clipping, a site with four times another site's A_c/κ_c receives one additional bit. The scalar τ sets the overall precision level: Raising τ shifts all proposed bit-widths upward, and lowering τ shifts the proposed bit-widths downward. We choose τ so that, after clipping, the cost-weighted average bit-width equals the target $\bar{m}$. For the additive risk surrogate, differences in A_c/κ_c make reallocation improve on uniform precision: Extra bits protect sites with higher risk per unit cost, funded by lower precision at sites with lower risk per unit cost.

Assigning integer bit-widths. To use the continuous solution in inference, we convert each site's assigned precision to an integer bit-width. In the RAR-log implementation, we replace A_c by $\max(A_c, 0) + \epsilon$, with $\epsilon > 0$, to keep the input to $\log_2(A_c/\kappa_c)$ in Eq. 4 strictly positive. We then round to integer bits in the range 3 to 8 and make a final cost-aware adjustment to closely match the target average bit-width. We call the resulting policy RAR-log and compare the RAR-log integer

allocation with the exact dynamic-programming (DP) solution in Eq. 34. In the empirical tables, a 6-bit budget therefore denotes the target budget; the maximum absolute realized drift is 0.0102 bits, and packed-payload accounting uses the realized assignments.

3.3 Combining RAR-log with Smoothing

We use activation smoothing, a feature-rescaling technique inspired by SmoothQuant (Xiao et al., 2023), to reduce quantization error from outliers. RAR-log allocates higher precision to sites where quantization errors most affect predictions. Combining activation smoothing with RAR-log addresses both the size of quantization error and the model's sensitivity to quantization error.

Smoothing transform and base-atlas allocation. We apply smoothing at the same activation boundaries used for risk measurement. Rescaling changes the quantization grid seen by each feature, and inverse rescaling reconstructs the activation immediately after rounding. We keep the weights fixed to isolate the effect of reducing activation quantization error. The positive per-feature scales s_c are estimated from calibration data. The quantized activation is

$$\widetilde{Q}_m(z_c) \coloneqq s_c \odot Q_m(z_c \oslash s_c). \tag{5}$$

Here, $\odot$ and $\oslash$ denote elementwise multiplication and division, respectively.

Smooth-RAR-log applies smoothing at evaluation while retaining the base atlas and the RAR-log allocation rule. Smooth-RAR-log is our main combined policy for LM, packed activations, and quantized weights.

Post-smoothing recalibration. Smoothing changes site risks by different amounts, so the sites that need the most protection can change. Recalibration redirects precision according to the output distortion that remains after smoothing. Remeasuring Eq. 2 with $\widetilde{Q}_4$ gives the residual atlas $A^{(4)}_{c,\mathrm{smooth}}$, which captures the output effect of the errors remaining after smoothing. Applying Eq. 4 to the residual atlas gives Smooth-Recalib-RAR-log, retaining the core output-KL objective.

4 Experiments

4.1 Experimental Setup

Models, data, and evaluation. We evaluate eight checkpoints: the small Generative Pre-trained Transformer 2 (GPT-2) (Radford et al., 2019), the 125M-parameter Open Pre-trained Transformer (OPT-125M) (Zhang et al., 2022), TinyLlama-1.1B (Zhang et al., 2024), and five 6.9B–7B checkpoints: Pythia (Biderman et al., 2023), Qwen2.5-7B and Qwen2.5-7B-Instruct (Yang et al., 2024), Mistral (Jiang et al., 2023), and Open Language Model 2 (OLMo-2) (OLMo et al., 2025; 2024). We shorten the two Qwen checkpoint names to Qwen and Qwen-Instruct. Risk atlases are calibrated on WikiText-2 train (Merity et al., 2017). The main 256-token LM evaluation uses WikiText-2, the Colossal Clean Crawled Corpus (C4) (Raffel et al., 2020), and FineWeb-Edu (Penedo et al., 2024); downstream evaluation uses custom conditional-likelihood scoring on multiple-choice benchmarks and standard `lm-evaluation-harness` tasks (Biderman et al., 2024). All mixed policies use the same cost-weighted average bit-width accounting and are compared with uniform, proxy-based, random, and SmoothQuant-style (Xiao et al., 2023) baselines. We report LM NLL in nats per token, accuracy in percent, and accuracy differences in percentage points. The clean reference uses unquantized activations under the weight configuration specified for each evaluation. Exact model and hook specifications, sample counts, hardware and timing, policy coverage, the packed-path and weight-PTQ protocols, and the additive DP oracle are in Appendix D.

4.2 LM Results

Aggregate 6-bit degradation reduction. Table 1 summarizes 6-bit LM results using NLL recovery: the fraction of uniform 6-bit NLL degradation removed by policy m. When $\mathrm{NLL}_{\mathrm{uniform6}} > \mathrm{NLL}_{\mathrm{clean}}$, we define the recovery fraction below and report $100 \times \mathrm{Reduction}(m)$:

$$\mathrm{Reduction}(m) \coloneqq \frac{\mathrm{NLL}_{\mathrm{uniform6}} - \mathrm{NLL}_m}{\mathrm{NLL}_{\mathrm{uniform6}} - \mathrm{NLL}_{\mathrm{clean}}}. \tag{6}$$

Table 1: Average 6-bit LM degradation reduction (%) by checkpoint, as defined in Eq. 6. Higher is better, and a negative entry means the policy is worse than uniform 6-bit. Checkpoint rows average WikiText-2, C4, and FineWeb-Edu. Aggregate rows are unweighted means over the indicated number of model–corpus evaluations and compare only policies evaluated on the same set. A dash denotes unavailable coverage. Table headers use smooth uniform (SmUni), Smooth-RAR-log (SmRAR), and Smooth-Recalib-RAR-log (SmRecalib).

Model	RAR-log	SmUni	SmRAR	SmRecalib
GPT-2	65.0	84.8	89.2	–
OPT-125M	74.3	87.9	89.1	–
TinyLlama-1.1B	68.5	84.9	91.5	–
Pythia-6.9B	85.0	95.3	97.1	97.5
Qwen2.5-7B	96.2	90.5	99.4	99.4
Qwen2.5-7B-Instruct	95.5	89.6	98.9	98.9
Mistral-7B	−38.7	75.3	90.7	91.1
OLMo-2-7B	75.5	90.6	96.8	97.1
Small-to-1B (9)	69.3	85.9	90.0	–
Matched 7B (15)	62.7	88.3	96.6	96.8
All (24)	65.2	87.4	94.1	–

Matched 7B contains the five checkpoints with post-smoothing atlases. The All row includes only policies evaluated for all eight checkpoints. Table 7 in Appendix E includes the decision-aware policies and margin-matched subset.

Averaged over the three corpora for each checkpoint, RAR-log removes between 65.0% and 96.2% of the uniform 6-bit degradation on the seven non-Mistral checkpoints. On Mistral-7B, RAR-log's reduction is -38.7%, indicating greater degradation than uniform 6-bit. Smooth uniform removes between 70.4% and 96.0% across individual model–corpus evaluations, with a mean of 87.4%. Smooth-RAR-log increases the mean reduction to 94.1% and achieves at least 88.2% in every evaluation. Smooth-RAR-log lowers NLL relative to smooth uniform in all 24 model–corpus pairs, with a mean reduction of 0.180 nats per token and a median reduction of 0.035. All eight checkpoint-level mean differences also favor Smooth-RAR-log; Appendix D.9 reports the statistical tests. The paired NLL reductions show that the remaining NLL degradation can be reduced by changing where activation bits are assigned under the same target budget.

Post-smoothing recalibration. For the five 7B-scale checkpoints with matched post-smoothing atlases, Smooth-Recalib-RAR-log and Smooth-Recalib-RAR-task both remove 96.8% of the uniform degradation across 15 model–corpus evaluations. For Smooth-Recalib-RAR-log, the additional NLL reduction over Smooth-RAR-log is small but positive on 12 of 15 paired LM evaluations, supporting calibration on the post-smoothing residual risk. Combining smoothing with the base atlas therefore achieves most of the measured LM performance recovery; recalibration provides a smaller gain by adapting the allocation to the remaining output distortion.

Across the six evaluations containing all three granularities, the smooth site-, role-, and layer-level policies remove 98.3%, 96.5%, and 97.5% of uniform 6-bit degradation, respectively. On the 15-evaluation recalibrated subset, the corresponding values are 96.8%, 93.1%, and 95.5%. Thus layer-only allocation is a strong baseline, while the layer–role atlas provides the highest mean recovery on both matched subsets. Resolving both layer and role lets the allocator protect a sensitive site without raising precision for every site in the same layer or role.

A free-running evaluation on 16 fixed prompts shows the same pattern of Qwen degradation and recovery with Smooth-RAR-log when generated tokens are fed back into the context (Appendix E.3). On the Mistral block trace, unsmoothed RAR-log lowers NLL relative to uniform precision on most blocks, but two severe failures make unsmoothed RAR-log's mean NLL worse. Reallocating bits alone does not prevent these two NLL spikes. Applying smoothing to the same allocation removes both NLL spikes. The block traces and finite-noise controls are in Appendix E.4. Appendix E provides checkpoint-level NLL and allocation-granularity results, 1024-token LM evaluations, and multiple-choice results under custom and standard-harness scoring.

Table 2: Full-dataset `lm-evaluation-harness` evaluation at the 6-bit activation budget. Entries are unweighted mean zero-shot accuracy (%) over ARC-Easy, ARC-Challenge, OpenBookQA, and WinoGrande. The policy columns are clean, uniform, SmUni, SmRAR, and Smooth-Recalib-RAR-task (SmTask).

Model	Clean	Uniform	SmUni	SmRAR	SmTask
Qwen2.5-7B	57.6	28.4	53.7	56.7	57.1
Qwen2.5-7B-Instruct	58.6	29.1	52.4	59.0	58.2
Mistral-7B	60.4	57.9	59.4	59.3	59.6
OLMo-2-7B	61.9	51.1	60.7	60.7	60.9
Mean	59.6	41.6	56.6	58.9	58.9

4.3 Downstream Evaluations

We evaluate Smooth-RAR-log to test whether preserving the next-token distribution through KL-based allocation also helps the model choose correct answers on multiple-choice tasks. The decision-aware comparison, Smooth-Recalib-RAR-task, combines output KL with target-token NLL increases, margin loss, and top-1 flips measured after smoothing on the same calibration text (Appendix D.1.2).

Full-dataset harness results. Table 2 reports `lm-evaluation-harness` (Biderman et al., 2024) zero-shot results for Qwen, Qwen-Instruct, Mistral, and OLMo-2 on the full Easy and Challenge splits of the AI2 Reasoning Challenge (ARC) (Clark et al., 2018) and on the full splits of OpenBookQA (Mihaylov et al., 2018) and WinoGrande (Sakaguchi et al., 2020), comprising 2,376, 1,172, 500, and 1,267 examples, respectively. Appendix E also reports BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), Massive Multitask Language Understanding (MMLU) (Hendrycks et al., 2021), and ReAding Comprehension from Examinations (RACE) (Lai et al., 2017).

Uniform 6-bit activations substantially reduce accuracy on both Qwen checkpoints, while Smooth-RAR-log and Smooth-Recalib-RAR-task restore accuracy to approximately the clean reference (Table 2). Smoothing alone recovers most of the degradation on Mistral and OLMo-2. Across all four checkpoints, Smooth-RAR-log improves mean accuracy by 2.34 percentage points over smooth uniform and yields higher accuracy in 12 of 16 task comparisons; Smooth-Recalib-RAR-task improves the mean by 2.39 percentage points and yields higher accuracy in 14 of 16 comparisons. The close mean accuracies of Smooth-RAR-log and Smooth-Recalib-RAR-task show that the core KL-based policy already captures most of the downstream gain in the full-dataset harness comparison. An allocation calibrated on next-token distributions can therefore improve answer selection without using benchmark answers to choose the bit-widths. Under the unpaired standard-error approximation, the 95% confidence intervals (CIs) for Smooth-RAR-log's gains over smooth uniform exclude zero on both Qwen checkpoints and include zero on Mistral and OLMo-2 (Appendix E.7). The clearest accuracy gains occur on the two Qwen checkpoints, where smoothing leaves a substantial gap for allocation to close.

4.4 Packed Activations and Quantized Weights

The allocation benefit persists with packed activations and W8 weight quantization. Smooth-RAR-log lowers NLL relative to smooth uniform in all 20 W8 LM comparisons and all six packed W8 comparisons (Table 19). The mean NLL reductions are 0.183 and 0.190 nats per token, respectively. The packed W8 comparisons use a lightweight variant of activation-aware weight quantization (AWQ) (Lin et al., 2024), termed AWQ-lite. The packed results show that the benefit survives storing activations as integers and reconstructing the activations for downstream computation with quantized weights. Appendix F reports the evaluation protocols and results, and Appendix F.3 summarizes the matched comparisons and payload accounting.

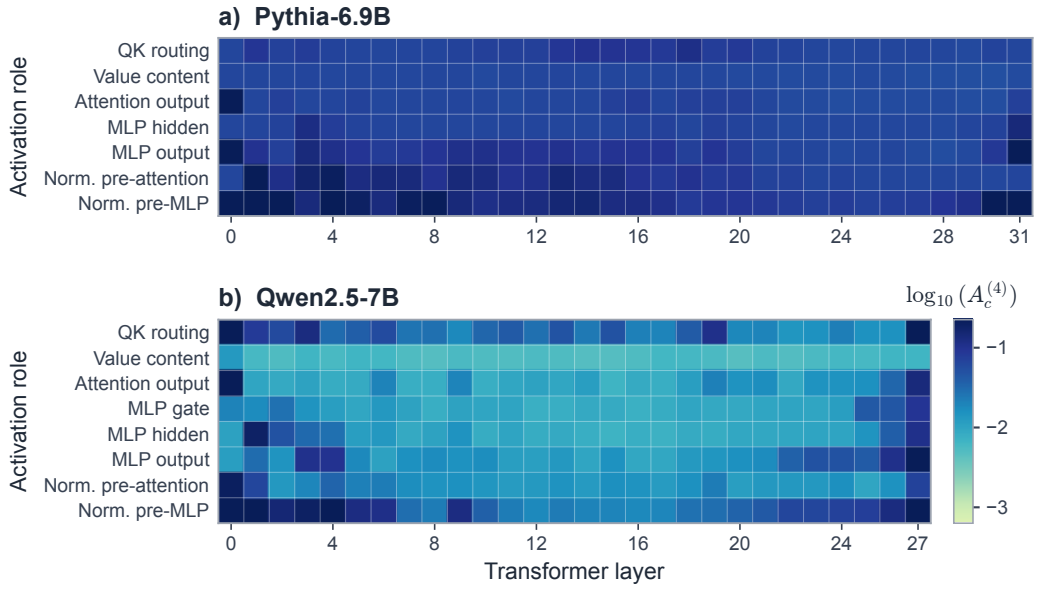

Figure 2: The risk atlas identifies sensitive roles and layers. Layer–role 4-bit risk atlases for Pythia-6.9B and Qwen2.5-7B. Color is $\log_{10}(A_c^{(4)})$ for each role and layer, on a scale shared with the remaining checkpoints in Appendix G. Pythia's uniformly dark heatmap therefore reflects Pythia's much larger absolute risk; Appendix G also lists the exact role shares.

5 Interpreting Functional Risk

We use the risk atlas to identify which layer–role computations need higher precision under a fixed bit budget and test the effect of removing that protection.

Activation size alone is not enough to choose precision. The rank correlation with measured 4-bit risk never exceeds 0.34 for activation root mean square (RMS) and peaks at 0.653 for maximum-to-RMS outlier scores (Table 20). Activation RMS and maximum-to-RMS outlier scores describe activation size and outliers, but do not measure how rounding changes the model's predictions. For example, smoothing reduces numerical error at Qwen's early QK sites, but the corresponding output KL remains substantial (Figure 6). Reducing error at a site can therefore leave a need for higher precision when the remaining error still changes predictions. The residual output distortion explains why RAR measures output KL and weighs the measured KL risk against activation cost: Extra bits should go to sites where additional precision preserves predictions most effectively per unit cost. Table 20 gives the complete proxy comparison.

Precision should protect sensitive computations. Figure 2 compares the layer–role risk atlases for Pythia and Qwen. Pythia's risks lie near the upper end of the shared color scale at most sites, whereas Qwen spans a wider range, with the highest risks in QK routing and in the first and last layers. In Pythia, MLP output and pre-MLP normalization contribute the largest shares of summed risk; in Qwen, QK routing contributes the largest share (Appendix G). The same RAR rule protects MLP computation and feature-scale regulation in Pythia, and context selection through attention in Qwen, by assigning more bits to sites with higher measured risk per unit activation cost.

Qwen needs precision in attention routing. Figure 3 tests whether reallocating the extra bits from Qwen's QK sites to other roles preserves the clean predictions equally well. Starting from unsmoothed RAR-log, we fix every site of one role at 6 bits and re-optimize all other sites at the same target budget. For QK, role neutralization removes the extra bits protecting high-risk routing sites and reallocates the released budget to other roles. Across four validation blocks, QK neutralization raises mean output KL from 0.367 to 6.472 nats, approaching the 7.545 nats of uniform 6-bit, and

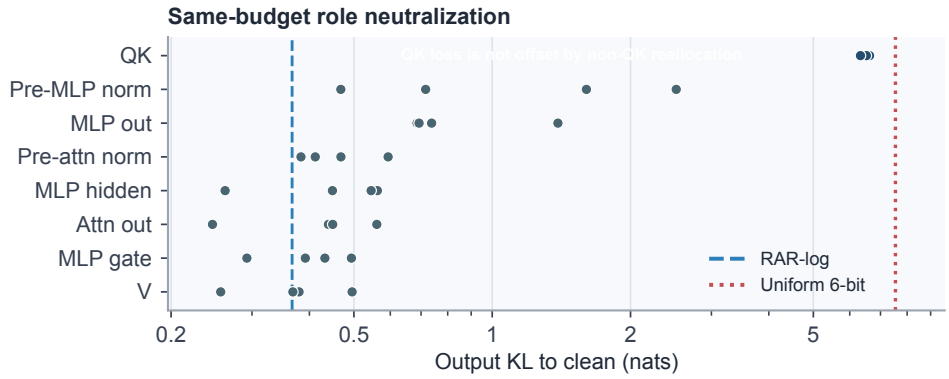

Figure 3: Removing QK precision protection nearly eliminates Qwen's allocation benefit. Each role-neutralization intervention fixes all sites of one role at 6 bits and re-optimizes the remaining sites at the same target budget. Bars show means, and dots show the four validation blocks. Role-wise budgets and bit assignments are in Appendix G.3.

lowers agreement with the clean top-1 prediction from 78.6% to 4.9%; realized budgets range from 6.000 to 6.012 bits. Within the evaluated RAR-log policy family, spending the released bits on other roles does not compensate for reduced routing precision.

QK errors change attention weights and thus the mixture of context information passed downstream. The attention traces in Appendix G.3 show the rerouting directly. Together with the role-neutralization intervention, the traces identify preserved routing as a key source of Qwen's allocation benefit: Extra precision in other roles does not provide the same protection.

The remaining heatmaps, role-risk profiles across all eight checkpoints, smoothing-risk reductions, exact role shares, and sensitivity analyses for calibration corpus, sample size, and post-smoothing risk rankings are in Appendix G.

6 Conclusion

We introduced RAR to allocate activation precision in Transformer language models according to each layer–role site's sensitivity to quantization error. RAR measures changes in the next-token distribution with KL divergence and assigns higher precision to sensitive layer–role sites under a fixed cost-weighted bit budget. The risk atlas identifies where quantization error most strongly disrupts predictions in each model. RAR complements activation smoothing, an existing approach to reducing quantization error from outliers. Our main combined policy, Smooth-RAR-log, combines the RAR-log allocation rule with SmoothQuant-style smoothing. Remeasuring the atlas after smoothing uses the bit budget more efficiently by identifying where the remaining errors matter most.

Our theoretical analysis explains how quantization errors from different parts of a Transformer interact to change model predictions. The theoretical results give upper bounds on how much predictions can change when activations are quantized throughout the model. Our theoretical comparison of RAR-log with uniform precision also shows why assigning extra bits to sites with higher risk per unit cost uses a fixed budget more efficiently.

Across eight checkpoints, Smooth-RAR-log removes an average of 94.1% of the LM degradation caused by uniform 6-bit activations, compared with 87.4% for smoothing alone. Smooth-RAR-log's advantage over smoothing alone persists with packed integer activations and W8 PTQ. Standard-harness evaluations across four checkpoints also show higher average multiple-choice accuracy than smoothing alone. Decision-aware extensions such as Smooth-Recalib-RAR-task also account for lower target-token likelihoods, smaller prediction margins, and changes in the top-ranked token. The decision-aware extensions achieve higher average task accuracy than smoothing alone at the same bit budget. The LM and multiple-choice results show why roles with different sensitivities to quantization error need different precision levels.

AI USE STATEMENT

Generative AI tools were used as assistive tools for writing and editing, mathematical proofreading and consistency checking, and code development and debugging. Any AI-assisted outputs incorporated into this work were reviewed and validated by the authors as appropriate. The authors take full responsibility for the final content of this work, including any text, claims, or artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies numerical precision allocation for Transformer inference using existing pretrained models and public benchmark datasets. The study does not involve human participants, private data, or newly collected personal information. We do not identify any additional ethical concerns specific to the methods and evaluations presented in this paper.

REPRODUCIBILITY STATEMENT

Secs. 2 and 3 describe risk measurement and allocation. Sec. 4.1 and Appendix D specify the experimental setup, hyperparameters, and evaluation protocols. Appendices D.3 and D.9 detail smoothing calibration, randomization, and statistical procedures.

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 *EMNLP*, pp. 4895–4901, 2023.

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Systematic Outliers in Large Language Models. In *ICLR*, 2025.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-Free 4-Bit Inference in Rotated LLMs. In *NeurIPS*, 2024.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *CoRR*, abs/1607.06450, 2016.

Aishwarya Bhandare, Vamsi Sripathi, Deepthi Karkada, Vivek Menon, Sun Choi, Kushal Datta, and Vikram A. Saletore. Efficient 8-Bit Quantization of Transformer Neural Machine Language Translation Model. *CoRR*, abs/1906.00532, 2019.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. In *ICML*, volume 202, pp. 2397–2430, 2023.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, Charles Lovering, Niklas Muennighoff, Ellie Pavlick, Jason Phang, Aviya Skowron, Samson Tan, Xiangru Tang, Kevin A. Wang, Genta Indra Winata, François Yvon, and Andy Zou. Lessons from the Trenches on Reproducible Evaluation of Language Models. *CoRR*, abs/2405.14782, 2024.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Understanding and Overcoming the Challenges of Efficient Transformer Quantization. In *EMNLP (1)*, pp. 7947–7969, 2021.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *NAACL-HLT (1)*, pp. 2924–2936, 2019.

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. *CoRR*, abs/1803.05457, 2018.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. GPT3.int8(): 8-bit Matrix Multiplication for Transformers at Scale. In *NeurIPS*, 2022.

Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V2: Hessian Aware trace-Weighted Quantization of Neural Networks. In *NeurIPS*, 2020.

Ali Edalati, Alireza Ghaffari, Mahsa Ghazvini Nejad, Lu Hou, Boxing Chen, Masoud Asgharian, and Vahid Partovi Nia. OAC: Output-adaptive Calibration for Accurate Post-training Quantization. In *AAAI*, pp. 16453–16461, 2025.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate Quantization for Generative Pre-trained Transformers. 2023.

Robert M. Gray and David L. Neuhoff. Quantization. *IEEE Trans. Inf. Theory*, 44(6):2325–2383, 1998.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding. In *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 *NeurIPS*, 2024.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *CVPR*, pp. 2704–2713, 2018.

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. *CoRR*, abs/2310.06825, 2023.

Jinuk Kim, Marwa El Halabi, Wonpyo Park, Clemens J. S. Schaefer, Deokjae Lee, Yeonhong Park, Jae W. Lee, and Hyun Oh Song. GuidedQuant: Large Language Model Quantization via Exploiting End Loss Guidance. In *ICML*, volume 267, 2025.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. I-BERT: Integer-only BERT Quantization. In *ICML*, volume 139, pp. 5506–5518, 2021.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. SqueezeLLM: Dense-and-Sparse Quantization. In *ICML*, volume 235, pp. 23901–23923, 2024.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *SOSP*, pp. 611–626, 2023.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In *EMNLP*, pp. 785–794, 2017.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the Limit of Post-Training Quantization by Block Reconstruction. In *ICLR*, 2021.

Chengxi Liao and Zeyi Wen. Channel-Aware Mixed-Precision Quantization for Efficient Long-Context Inference. In *The Fourteenth International Conference on Learning Representations*, 2026.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. In *MLSys*, 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: LLM Quantization with Learned Rotations. In *ICLR*, 2025.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen (Henry) Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A Tuning-Free Asymmetric 2bit Quantization for KV Cache. In *ICML*, volume 235, pp. 32332–32344, 2024.

Yuexiao Ma, Huixia Li, Xiawu Zheng, Feng Ling, Xuefeng Xiao, Rui Wang, Shilei Wen, Fei Chao, and Rongrong Ji. AffineQuant: Affine Transformation Quantization for Large Language Models. In *ICLR*, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer Sentinel Mixture Models. In *ICLR*, 2017.

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 *EMNLP*, pp. 2381–2391, 2018.

Ashitabh Misra, Madhav Agrawal, Arham Jain, and Tarek F. Abdelzaher. MixQuant: Adaptive Mixed-Precision Quantization for Large Language Models. *CoRR*, abs/2607.23047, 2026.

Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or Down? Adaptive Rounding for Post-Training Quantization. In *ICML*, volume 119, pp. 7197–7206, 2020.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 OLMo 2 Furious, 2024. URL https://arxiv.org/abs/2501.00656.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 OLMo 2 Furious. *CoRR*, abs/2501.00656, 2025.

Andrew Or, Apurva Jain, Daniel Vega-Myhre, Jesse Cai, Charles David Hernandez, Zhenrui Zheng, Driss Guessous, Vasiliy Kuznetsov, Christian Puhrsch, Mark Saroufim, Supriya Rao, Thien Tran, and Aleksandar Samardzic. TorchAO: PyTorch-Native Training-to-Serving Model Optimization. *CoRR*, abs/2507.16099, 2025.

Guilherme Penedo, Hynek Kydlícek, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin A. Raffel, Leandro von Werra, and Thomas Wolf. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. In *NeurIPS*, 2024.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently Scaling Transformer Inference. In *MLSys*, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *AAAI*, pp. 8732–8740, 2020.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally Calibrated Quantization for Large Language Models. In *ICLR*, 2024.

Noam Shazeer. GLU Variants Improve Transformer. *CoRR*, abs/2002.05202, 2020.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. Q-BERT: hessian based ultra low precision quantization of BERT. In *AAAI*, pp. 8815–8821, 2020.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive Activations in Large Language Models. *CoRR*, abs/2402.17762, 2024.

Philippe Tillet, Hsiang-Tsung Kung, and David D. Cox. Triton: an intermediate language and compiler for tiled neural network computations. In *MAPL@PLDI*, pp. 10–19, 2019.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models. *CoRR*, abs/2302.13971, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *NIPS*, pp. 5998–6008, 2017.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. DeepNet: Scaling Transformers to 1,000 Layers. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(10):6761–6774, 2024.

Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-Aware Automated Quantization With Mixed Precision. In *CVPR*, pp. 8612–8620, 2019.

Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. QDrop: Randomly Dropping Quantization for Extremely Low-bit Post-Training Quantization. In *ICLR*, 2022.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier Suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling. In *EMNLP*, pp. 1648–1665, 2023.

Guangxuan Xiao, Ji Lin, Mickaël Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. In *ICML*, volume 202, pp. 38087–38099, 2023.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *ICLR*, 2024.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On Layer Normalization in the Transformer Architecture. In *ICML*, volume 119, pp. 10524–10533, 2020.

Dongjie Xu, Kai Qian, Weijie Shi, Yuxuan Sun, Minghua Tang, Fenglei Jin, Hanchi Dong, Jiajie Xu, et al. RippleKV: Cross-Layer KV Cache Allocation via Perturbation Propagation. *arXiv preprint arXiv:2608.08684*, 2026.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report. *CoRR*, abs/2412.15115, 2024.

Zhangyang Yao, Haiyan Zhao, Haoyu Wang, Tianbo Huang, Lihua Zhang, and Xu Han. GAMMA: Global Bit Allocation for Mixed-Precision Models under Arbitrary Budgets. *CoRR*, abs/2605.18475, 2026.

Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V3: Dyadic Neural Network Quantization. In *ICML*, volume 139, pp. 11875–11886, 2021.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers. In *NeurIPS*, 2022.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *ACL (1)*, pp. 4791–4800, 2019.

Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization. In *NeurIPS*, pp. 12360–12371, 2019.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. TinyLlama: An Open-Source Small Language Model. *CoRR*, abs/2401.02385, 2024.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. OPT: Open Pre-trained Transformer Language Models. *CoRR*, abs/2205.01068, 2022.

APPENDIX TABLE OF CONTENTS

A Notation **16**

B Related Work **17**

C Additional Analytical Statements **18**
C.1 Local Output KL and Joint Risk 18
C.2 Finite-Bit, Sequential-Policy, and Choice-Score Certificates 19
C.3 Heterogeneity Dividend 21

D Detailed Experimental Setup and Allocation Evaluations **21**
D.1 Policy Hierarchy and Objective Variants 21
D.2 Activation-Site Boundaries and Cost Accounting 24
D.3 Smoothing Calibration and Residual-Risk Measurement 24
D.4 Reference Implementation of RAR 25
D.5 Models, Data, and Evaluation Protocols 26
D.6 Allocation Baselines and PTQ Controls 27
D.7 Hardware and Calibration Cost 28
D.8 Systems and Allocation Evaluations 28
D.9 Randomness and Statistical Protocol 30
D.10 Limitations 30

E Additional LM and Downstream Evaluations **31**
E.1 Checkpoint-Level 6-Bit LM Results 31
E.2 Constrained Post-Smoothing LM Comparison 32
E.3 Free-Running Generation 32
E.4 Blockwise Failure Regimes and Finite-Noise Controls 32
E.5 LM at 1024 Tokens 33
E.6 Downstream Evaluation with Custom Choice Scoring 33
E.7 Standard Harness Evaluations on Four 7B Checkpoints 35

F Packed-Activation and Weight-PTQ Evaluations **37**
F.1 Inference with Packed Activations 37
F.2 Activation Allocation after W8 Weight Quantization 37
F.3 Results across Evaluation Settings 40

G Additional Functional-Risk Atlas Analyses **40**
G.1 Numerical Proxies Partially Predict Functional Risk 40
G.2 Numerical Error, Functional Leverage, and Coordinate Erasure 40
G.3 Attention Routing and Precision Allocation 42
G.4 Additional Architecture and Smoothing Views 44
G.5 Sensitivity to Calibration Settings 45

H Proofs of Analytical Statements **49**
H.1 Proof of the Local Output KL Expansion 49
H.2 Proofs for the Joint Local Risk Decomposition 49
H.3 Proofs for the Finite-Bit Output KL Rate and Top-1 Barrier 50
H.4 Proofs for the Nonasymptotic Simultaneous-Policy Certificate 51
H.5 Proofs for the Multi-Token Choice-Likelihood Certificate 52
H.6 Proof of the Continuous RAR-log Optimum 53
H.7 Proofs for the Heterogeneity Dividend 53

Further details. Appendix C gives the single-site and sequential-policy KL bounds in Theorems C.5 and C.7, derives the exact gain over uniform precision when no bits are clipped, and extends the analysis to bit-width limits. Appendix H.6 gives the Karush–Kuhn–Tucker (KKT) proof of Proposition 1. Policy names and decision-aware objectives are defined in Appendix D.1; Appendices D.2, D.3, and D.4 collect architecture-specific site costs, smoothing and residual-risk calibration, and a compact Python reference implementation of single-site risk measurement.

A NOTATION

Table 3: List of notation used throughout the paper.

Symbol	Meaning
Functional-risk atlas	
x, θ	Input sequence; model parameters
$\mathcal{D}, \mathcal{D}_{\mathrm{cal}}$	Data distribution; calibration distribution
T, t	Sequence length; token position
$a_\theta(x), a_{\theta,t}$	Logits; position-t logits
$p_{\theta,t}, p_\theta^{c,\delta}$	Clean; site-perturbed output distribution
$\mathrm{softmax}, D_{\mathrm{KL}}$	Softmax; forward KL divergence
$c = (\ell, r, i_{\mathrm{seg}})$	Layer–role–segment site index
$\mathcal{C}, n$	Intervention-site set; cardinality
z_c	Activation at site c
$\delta_c, \delta_{c,\rho}, \xi_c, \rho$	Perturbation; scaled noise; noise direction; scale
$\mathcal{R}_c(x), \overline{\mathcal{R}}_c$	Input-level; dataset-level functional risk
$\mathbb{E}, \mathrm{Cov}, \Sigma_{c,0}, \Sigma_c$	Expectation; covariance and local covariance forms
F_{out}, J_c	Output Fisher matrix; site-to-logit Jacobian
g_x, J	Joint logit map; stacked site Jacobian
$\Omega_{cd,0}$	Cross second-moment block
q_c, b_{cd}	Site curvature; cross-site interaction
Quantization, decisions, and allocation	
$Q_m, \delta_c^{(m)}, A_c^{(m)}$	Quantizer; quantization error; single-site risk
$\mathcal{G}_c, g, u_g$	Dynamic-range groups; group; normalized range
$\sigma_j, q_m, \mathcal{E}_\sigma$	Reconstruction scale; grid factor; quantizer energy
$Q_{m,\sigma}, G_{c,t}, L_{c,t}$	Scaled quantizer; logit map; Lipschitz constant
$D_t^{c,m}, U_t^{c,m}$	Site KL; finite-bit KL upper bound
$w_t, v_t, \gamma_t, \chi_t$	Winner; runner-up; margin; flip-KL barrier
$\widetilde{\gamma}_t^{c,m}$	Quantized clean-winner margin
$M_c^{(m)}, \mathrm{flip}_c^{(m)}, \mathbb{I}$	Margin loss; flip rate; indicator
$\Delta\mathrm{NLL}_c^{(m)}, [u]_+$	Target NLL increase; positive part
$\mathrm{rank}, \mathrm{med}_+$	Percentile rank; positive-entry median
$S_c^{\mathrm{margin}}, S_c^{\mathrm{task}}$	Margin-aware; task-aware allocation score
$V_{j,c}, \lambda_j, G_c$	Task statistic; rescaling; auxiliary penalty
$\lambda_{\mathrm{task}}, \eta, \widehat{G}(m)$	Task coefficient; scan multiplier; task proxy
m, m_c, m_0	Bit policy; site bits; reference bits
$m_{\min}, m_{\max}, \bar{m}$	Bit bounds; target average bit-width
κ_c, B	Site cost; total bit budget
$m_c^\star, \tau$	RAR-log allocation; budget-shift scalar
K_κ, π_c, r_c	Total cost; cost weight; cost-normalized risk
M_r, G_r	Weighted arithmetic; geometric risk mean
$\Phi_A, m^{\mathrm{het}}, m^{\mathrm{uni}}$	Allocation objective; heterogeneous; uniform policy
$\epsilon, m^{\mathrm{DP}}$	Risk regularizer; exact DP policy
Smoothing and evaluation	
$u_{c,j}, s_{c,j}, s_c, \alpha, d_c$	Feature range; smoothing scales; exponent; width
$\widetilde{Q}_m, A_{c,\mathrm{smooth}}^{(4)}$	Smoothed quantizer; post-smoothing residual risk
$c_i, m_i, a_t^{(i)}, z_i^{(i-1)}$	Hook; bits; prefix logits; prefix activation
$E_i, L_{i,t}, D_t^m, U_t^m$	Prefix energy and sensitivity; joint KL and bound
$K, k, \mathcal{T}_k, y_{k,t}, w_{k,t}$	Candidates; scored positions, tokens, weights
$H_{k,t}, B_k, S_k^{(j)}$	Logit radius; score radius; choice score
$\mathrm{Reduction}(m), \mathrm{NLL}_m$	NLL recovery fraction; policy NLL
Proof-specific notation	
$h_t, h(\delta), F_t, \phi_t$	Proof logit increment; Fisher; local KL function
$q(\delta), r_\rho(\xi), O(\cdot)$	Taylor remainders; asymptotic order
p, q, Δ	Generic distributions; probability simplex
λ, ν, ψ	KKT multipliers; pairwise flip barrier
f_y, e_y, v	Target log-probability; basis vector; feasible direction

B Related Work

This appendix reviews sensitivity-based mixed-precision allocation, PTQ, KV-cache quantization, and Transformer stability.

Sensitivity-based mixed-precision allocation. Prior work has studied how to allocate precision across layers or groups. Layerwise and groupwise allocation methods use hardware resource feedback (Wang et al., 2019), Hessian-based sensitivity estimates (Dong et al., 2020; Shen et al., 2020), or constrained optimization (Yao et al., 2021) to select bit-widths. Among recent large language model (LLM) methods, SqueezeLLM (Kim et al., 2024) uses Fisher-based saliency for weight groups, GAMMA (Yao et al., 2026) reconstructs hidden states to select module precisions, and MixQuant (Misra et al., 2026) accounts for upstream quantization plans. RAR instead allocates precision to functional layer–role activation sites spanning QK routing, value transport, MLP computation, and normalization. RAR scores each site by the measured change in the next-token distribution under single-site quantization. Two sites with similar activation scale can therefore receive different precisions when the downstream functional effects of quantization differ.

Activation PTQ and calibration. Early Transformer PTQ established that quantization boundaries and nonlinear components such as softmax and normalization require care (Bhandare et al., 2019; Kim et al., 2021). Data-aware PTQ subsequently improved on round-to-nearest through learned rounding, block reconstruction, and activation-aware reconstruction (Nagel et al., 2020; Li et al., 2021; Wei et al., 2022). For LLMs, LLM.int8() (Dettmers et al., 2022) addresses emergent outliers, GPTQ (Frantar et al., 2023) performs one-shot weight quantization, and ZeroQuant (Yao et al., 2022) performs fine-grained weight–activation quantization. Output-adaptive Calibration (Edalati et al., 2025) and GuidedQuant (Kim et al., 2025) move the calibration objective closer to output or end-task distortion. RAR complements the PTQ and calibration methods by measuring the output consequence of quantizing each layer–role activation site and allocating a fixed activation-precision budget.

Outlier-aware PTQ. Activation range heterogeneity and structured outliers are central sources of PTQ error (Bondarenko et al., 2021). SmoothQuant (Xiao et al., 2023) transfers activation outlier difficulty into weights, AWQ (Lin et al., 2024) rescales weight channels by activation-derived importance, Outlier Suppression+ (Wei et al., 2023) and AffineQuant (Ma et al., 2024) use equivalent channelwise transformations, and OmniQuant (Shao et al., 2024) jointly learns clipping and equivalent transformations. Our weight-PTQ controls use AWQ-lite equalization from the outlier-aware PTQ family. Rotation-based methods such as QuaRot (Ashkboos et al., 2024) and SpinQuant (Liu et al., 2025) further distribute outliers across weights, activations, and caches. Rotation-based methods reduce exposure to quantization error, whereas RAR distributes residual activation precision. We therefore evaluate RAR with SmoothQuant-style smoothing and remeasure the atlas using the smoothed quantizer.

KV-cache quantization. KVQuant (Hooper et al., 2024) and KIVI (Liu et al., 2024) exploit the distinct numerical behavior of KV tensors through per-channel and per-token quantization and quantization before rotary position embeddings (RoPE). ChanMix (Liao & Wen, 2026) assigns mixed precision to individual KV channels, and RippleKV (Xu et al., 2026) measures the output KL effect of isolated value-cache perturbations to reallocate cache budget across layers. RippleKV uses a closely related output-facing intervention, but RAR targets transient layer–role activation boundaries throughout the Transformer rather than only the persistent KV cache.

Transformer stability. Prior work studies attention sinks (Xiao et al., 2024), normalization placement (Xiong et al., 2020) and residual scaling (Wang et al., 2024), and massive (Sun et al., 2024) or systematic (An et al., 2025) activation and attention outliers. The analyses of Transformer stability identify numerical mechanisms that motivate role-specific protection, but activation magnitude or a local instability need not determine the final prediction error. RAR therefore uses output-distribution KL to distinguish sites with high downstream functional consequence from sites whose perturbations are largely absorbed by later computation.

C Additional Analytical Statements

C.1 Local Output KL and Joint Risk

Single-site risk. Let $z_c(x)$ be the activation at site c. If perturbation δ_c is injected at site c and all other computation is unchanged, denote the resulting output distribution by $p_\theta^{c,\delta}$. We use the forward KL divergence (Kullback & Leibler, 1951) and define input-level functional risk as

$$\mathcal{R}_c(x) \coloneqq \mathbb{E}_{\delta_c}\left[\frac{1}{T-1}\sum_{t=1}^{T-1} D_{\mathrm{KL}}(p_{\theta,t}(\cdot \mid x) \| p_{\theta,t}^{c,\delta}(\cdot \mid x))\right]. \tag{7}$$

The dataset risk is $\overline{\mathcal{R}}_c \coloneqq \mathbb{E}_{x\sim\mathcal{D}}\mathcal{R}_c(x)$. Throughout this paper, KL divergence and NLL use natural logarithms.

Let J_c be the site-to-logit Jacobian, F_{out} the position-averaged output Fisher matrix, and Σ_c the perturbation covariance. For small centered noise, under the smoothness and boundedness assumptions of Proposition C.1,

$$\mathcal{R}_c(x) \approx \frac{1}{2}\operatorname{Tr}(\Sigma_c(x) J_c(x)^\top F_{\mathrm{out}}(x) J_c(x)). \tag{8}$$

For noncentered noise, the second moment replaces the covariance. The local KL expansion motivates the atlas; we measure all reported risks directly from forward passes. Second-order additivity holds when Fisher-projected cross moments vanish, a condition that deterministic fake-quantization errors need not satisfy. The complete local and joint expansions below include the interaction terms and conditions for dataset averaging.

Rigorous local expansion. To state the small-noise approximation precisely, flatten z_c and the logits at the $T-1$ evaluated positions, and define

$$F_{\mathrm{out}}(x) \coloneqq \frac{1}{T-1}\bigoplus_{t=1}^{T-1}(\operatorname{diag}(p_{\theta,t}) - p_{\theta,t}p_{\theta,t}^\top), \qquad J_c(x) \coloneqq \frac{\partial(a_{\theta,1},\ldots,a_{\theta,T-1})}{\partial z_c}.$$

Proposition C.1 (Local output KL expansion)**.** *Fix x and let $\delta_{c,\rho} \coloneqq \rho\xi_c$, where $\mathbb{E}[\xi_c] = 0$, $\mathrm{Cov}(\xi_c) \coloneqq \Sigma_{c,0}(x)$, and ξ_c is almost surely bounded. If the downstream map from z_c to the stacked logits is three times continuously differentiable on a neighborhood containing the perturbed activations for all sufficiently small ρ, then, as $\rho \to 0$,*

$$\mathcal{R}_{c,\rho}(x) = \frac{\rho^2}{2}\operatorname{Tr}(\Sigma_{c,0}(x) J_c(x)^\top F_{\mathrm{out}}(x) J_c(x)) + O(\rho^3), \tag{9}$$

where $\mathcal{R}_{c,\rho}$ denotes Eq. 7 with perturbation $\delta_{c,\rho}$.

The proof is given in Appendix H.1. Writing $\Sigma_c(x) \coloneqq \mathrm{Cov}(\delta_{c,\rho}) = \rho^2\Sigma_{c,0}(x)$ and retaining the second-order term gives Eq. 8. For noncentered noise, the same expansion uses the second moment $\mathbb{E}[\delta_c\delta_c^\top]$ in place of the covariance. Eq. 8 motivates the atlas; reported quantization risks are measured directly from forward passes on calibration examples. Random-perturbation risks additionally average over sampled noise.

Joint risk and additivity. The single-site atlas is used to choose a policy that perturbs many sites at once. The following result gives the corresponding joint expansion and isolates exactly where additivity can fail.

For a finite set $\mathcal{C}$ of n intervention sites, let $g_x((u_c)_{c\in\mathcal{C}})$ be the stacked evaluated logits when an additive intervention u_c is applied as the forward pass reaches each hook c. Thus $g_x(0)$ is the clean logit vector, and the partial derivative at zero is the single-site Jacobian $J_c(x)$ defined above. Fix x and suppose g_x is three times continuously differentiable near zero. Let $\mathcal{R}_{\mathcal{C},\rho}(x)$ denote the expected, position-averaged forward KL risk of the simultaneous intervention, and let $\delta_{c,\rho} \coloneqq \rho\xi_c$, where the stacked random vector $\xi \coloneqq (\xi_c)_{c\in\mathcal{C}}$ is almost surely bounded, and define the cross-second-moment blocks and the corresponding Fisher-projected coefficients by

$$\Omega_{cd,0} \coloneqq \mathbb{E}[\xi_c\xi_d^\top], \quad q_c \coloneqq \operatorname{Tr}(J_c^\top F_{\mathrm{out}} J_c \Omega_{cc,0}), \quad b_{cd} \coloneqq \operatorname{Tr}(J_c^\top F_{\mathrm{out}} J_d \Omega_{dc,0}).$$

Theorem C.2 (Joint local risk decomposition)**.** *Under the setup above, the expected risk of the simultaneous intervention satisfies*

$$\mathcal{R}_{\mathcal{C},\rho}(x) = \frac{\rho^2}{2} \sum_{c,d\in\mathcal{C}} b_{cd} + O(\rho^3), \tag{10}$$

as $\rho \to 0$, where $b_{cc} = q_c \geq 0$. The coefficients also satisfy $b_{cd} = b_{dc}$ for every c, d.

Corollary C.3 (Second-order interaction gap)**.** *Under the assumptions and notation of Theorem C.2, if $\mathcal{R}_{c,\rho}(x)$ denotes the one-site risk obtained from the marginal perturbation $\rho\xi_c$, then, as $\rho \to 0$,*

$$\mathcal{R}_{\mathcal{C},\rho}(x) - \sum_{c\in\mathcal{C}} \mathcal{R}_{c,\rho}(x) = \rho^2 \sum_{c<d} b_{cd} + O(\rho^3). \tag{11}$$

Corollary C.4 (Second-order additive regime)**.** *Under the assumptions and notation of Corollary C.3, if the Fisher-projected cross moments vanish, so that $b_{cd} = 0$ for $c \neq d$, then*

$$\mathcal{R}_{\mathcal{C},\rho}(x) = \sum_{c\in\mathcal{C}} \mathcal{R}_{c,\rho}(x) + O(\rho^3). \tag{12}$$

Second-moment block orthogonality, $\Omega_{cd,0} = 0$ for $c \neq d$, is a sufficient condition. For centered perturbations, the orthogonality condition is precisely block uncorrelatedness; independent centered perturbations are therefore sufficient.

Appendix H.2 gives the complete proofs. The theorem does not require centering for the joint-risk decomposition. Deterministic fake-quantization errors generally have nonzero cross-second moments, so the cross terms in Eq. 11 need not vanish. The sum of single-site risks is therefore exact to second order under Corollary C.4; otherwise, the cross terms in Eq. 11 quantify the leading deviation from additivity.

Averaging over x gives the dataset-risk expansion, provided that $\sum_{c,d} |b_{cd}(x)|$ is integrable and the absolute local remainder is bounded by $\rho^3 C(x)$ for an integrable function C. Both conditions hold automatically on a finite calibration set.

C.2 Finite-Bit, Sequential-Policy, and Choice-Score Certificates

To cover both the base and smoothed quantizers, partition the flattened activation coordinates into dynamic-range groups $g \in \mathcal{G}_c$. In our implementation, each group is a token. Let $\sigma_j > 0$ be a fixed reconstruction multiplier. Throughout the finite-bit results, m is an integer with $m \geq 2$. The base quantizer has $\sigma_j := 1$, while Eq. 5 has $\sigma_j := s_{c,j}$. Let u_g be the normalized dynamic range used for group g, allowing the numerical floor used in an implementation. Thus, for $j \in g$, define

$$u_g \geq \max_{k\in g} \frac{|z_{c,k}|}{\sigma_k}, \qquad q_m := 2^{m-1} - 1, \qquad \mathcal{E}_\sigma(z_c) := \sum_{g\in\mathcal{G}_c} u_g^2 \sum_{j\in g} \sigma_j^2. \tag{13}$$

The quantizer rounds $z_{c,j}/\sigma_j$ to the nearest multiple of u_g/q_m and multiplies the result by σ_j; denote the reconstruction by $Q_{m,\sigma}(z_c)$. Equality gives the ideal dynamic quantizer, and an identically zero activation group is represented exactly under either convention.

Theorem C.5 (Finite-bit output KL rate)**.** *Fix x, site c, and an integer $m \geq 2$. For every evaluated position t, let $G_{c,t}(z)$ denote the position-t logits when the hook activation is replaced by z. Suppose, for every such t, that $G_{c,t}$ is $L_{c,t}$-Lipschitz in Euclidean norm on the line segment joining z_c and the site activation's m-bit quantization. Then, for every evaluated position t,*

$$D_t^{c,m} := D_{\mathrm{KL}}(p_{\theta,t} \| p_{\theta,t}^{c,Q_{m,\sigma}}) \leq U_t^{c,m} := \frac{L_{c,t}^2 \mathcal{E}_\sigma(z_c)}{16 q_m^2} \leq L_{c,t}^2 \mathcal{E}_\sigma(z_c) 2^{-2m}. \tag{14}$$

Lemma C.6 (Categorical top-1 KL barrier)**.** *Let p_t be a strictly positive categorical distribution with unique winner w_t, and let $v_t \in \arg\max_{j\neq w_t} p_{t,j}$ be any runner-up. Define*

$$\chi_t := p_{t,w_t} \log \frac{2p_{t,w_t}}{p_{t,w_t} + p_{t,v_t}} + p_{t,v_t} \log \frac{2p_{t,v_t}}{p_{t,w_t} + p_{t,v_t}}. \tag{15}$$

Every categorical distribution q_t under which some competing category ties or exceeds the clean winner, $q_{t,w_t} \leq \max_{j\neq w_t} q_{t,j}$, satisfies $D_{\mathrm{KL}}(p_t \| q_t) \geq \chi_t$.

The proofs are in Appendix H.3. Since $q_m^{-2} = \Theta(2^{-2m})$, Eq. 14 supplies a nonasymptotic rate justification for the model in Eq. 3 whenever downstream Lipschitz constants are uniformly bounded over the relevant bit range. Measured risk-curve evaluations from 3 to 8 bits test the rate model across the allocation range. Writing the clean logit margin as γ_t gives

$$\chi_t = p_{t,w_t}\left[\log\frac{2}{1+e^{-\gamma_t}} + e^{-\gamma_t}\log\frac{2e^{-\gamma_t}}{1+e^{-\gamma_t}}\right]. \tag{16}$$

Thus a top-1 change must cross a position-specific positive KL barrier, while the barrier depends on the clean winner margin and probability mass. The position-specific KL barrier motivates the margin-aware diagnostics below.

The preceding finite-bit result concerns one quantized site, whereas an inference policy quantizes a sequence of sites whose later activations depend on earlier quantization. The next theorem gives a finite-noise guarantee for the sequential setting without a local expansion or assumptions of centering, independence, or moment orthogonality.

Fix x and list the n quantized hooks as $c_1, \dots, c_n$ in forward execution order. For a bit policy $m \coloneqq (m_1, \dots, m_n)$ with $m_i \geq 2$ for every i, let $a_t^{(i)}$ be the position-t logits in the hybrid forward pass that quantizes precisely the prefix $c_1, \dots, c_i$, so $a_t^{(0)}$ is clean and $a_t^{(n)}$ is the joint policy output. Let $z_i^{(i-1)}$ be the activation arriving at c_i in the $(i-1)$-prefix pass, let σ_i be the fixed positive reconstruction multipliers for $z_i^{(i-1)}$, and set

$$E_i \coloneqq \mathcal{E}_{\sigma_i}(z_i^{(i-1)}), \qquad q_{m_i} \coloneqq 2^{m_i-1} - 1,$$

with the dynamic-range groups and any numerical range floor defined as in Eq. 13. For each i and t, hold the first $i-1$ prefix quantizers fixed, leave later hooks unquantized, and view the final position-t logits as a function of the replacement activation at c_i.

Theorem C.7 (Nonasymptotic simultaneous-policy KL bound)**.** *Under the setup above, suppose the resulting final-logit map is $L_{i,t}$-Lipschitz on the line segment from $z_i^{(i-1)}$ to $Q_{m_i,\sigma_i}(z_i^{(i-1)})$. Then the joint output KL satisfies*

$$D_t^m \coloneqq D_{\mathrm{KL}}(\mathrm{softmax}(a_t^{(0)}) \| \,\mathrm{softmax}(a_t^{(n)})) \leq U_t^m \coloneqq \frac{1}{16}\left(\sum_{i=1}^{n}\frac{L_{i,t}\sqrt{E_i}}{q_{m_i}}\right)^2. \tag{17}$$

Appendix H.4 gives the complete proof. The prefix construction computes E_i from the activation that reaches hook c_i after earlier sites have been quantized. The finite-noise theorem uses a Lipschitz envelope and complements the local interaction theorem. The local interaction theorem resolves signed second-order cross terms, while the finite-noise theorem gives a nonasymptotic upper bound under the stated sequential sensitivity assumptions.

The preceding KL analysis concerns tokenwise output distributions. Downstream choice scorers instead add log likelihoods over several tokens, often on a different forward pass for every candidate. The next result directly covers multi-token choice scoring, allowing arbitrary nonnegative token weights and covering both total and length-normalized choice likelihood.

Fix a joint bit policy $m \coloneqq (m_1, \dots, m_n)$ with $m_i \geq 2$ for every i, and $K \geq 2$ candidate continuations. Candidate k is evaluated on a set of scored positions $\mathcal{T}_k$ with target tokens $y_{k,t}$ and weights $w_{k,t} \geq 0$. Let $a_{k,t}^{(0)}$ and $a_{k,t}^{(n)}$ be candidate k's clean and jointly quantized logits, respectively. For the forward pass of candidate k, define the realized prefix energy $E_{k,i}$ and segmentwise downstream Lipschitz constant $L_{k,i,t}$ at hook c_i exactly as in Theorem C.7, and assume the candidate forward pass satisfies the segmentwise Lipschitz premises. Set

$$H_{k,t}(m) \coloneqq \frac{1}{2}\sum_{i=1}^{n}\frac{L_{k,i,t}\sqrt{E_{k,i}}}{q_{m_i}}, \quad B_k(m) \coloneqq \sqrt{2}\sum_{t\in\mathcal{T}_k} w_{k,t}H_{k,t}(m), \tag{18}$$

$$S_k^{(j)} \coloneqq \sum_{t\in\mathcal{T}_k} w_{k,t}\log\mathrm{softmax}(a_{k,t}^{(j)})_{y_{k,t}}, \qquad j \in \{0, n\}. \tag{19}$$

Theorem C.8 (Multi-token choice-score perturbation bound)**.** *Under the setup above, without any independence, centering, or small-noise assumption,*

$$|S_k^{(n)} - S_k^{(0)}| \leq B_k(m) \qquad \textit{for every } k. \tag{20}$$

Because weighted target-token NLL is $-S_k$, the same $B_k(m)$ bounds the absolute NLL change. Appendix H.5 gives the complete proof. The choice-score radius in Eq. 18 accumulates explicit $q_{m_i}^{-1} = O(2^{-m_i})$ factors, whereas the joint-policy KL certificate in Eq. 17 squares the accumulated logit-error amplitude. The different choice-score and output-KL rates clarify the distinction between objectives: Minimizing the mean-output-KL surrogate need not minimize the bound on sequence-score perturbation. The choice-score bound applies directly to target-token NLL and the candidate-likelihood scorers used in our multiple-choice evaluations. The choice-score bound can be loose when signed tokenwise score changes cancel.

C.3 Heterogeneity Dividend

The following exact comparison quantifies the value of nonuniform allocation by isolating role-risk heterogeneity after normalizing by activation-width cost. For positive coefficients A_c and costs κ_c, write

$$K_\kappa \coloneqq \sum_c \kappa_c, \qquad \overline{m} \coloneqq \frac{B}{K_\kappa} = \bar{m}, \qquad \pi_c \coloneqq \frac{\kappa_c}{K_\kappa}, \qquad r_c \coloneqq \frac{A_c}{\kappa_c}, \tag{21}$$

and define the weighted arithmetic and geometric means

$$M_r \coloneqq \sum_c \pi_c r_c, \qquad G_r \coloneqq \prod_c r_c^{\pi_c}. \tag{22}$$

Let $\Phi_A(m) \coloneqq \sum_c A_c 2^{-2(m_c - m_0)}$ denote the objective in Eq. 3.

Theorem C.9 (Optimal heterogeneous allocation)**.** *Under the setup above, for the equality-budget relaxation of Eq. 3 without box constraints, the unique optimum is*

$$m_c^{\mathrm{het}} = \overline{m} + \frac{1}{2}\log_2 \frac{r_c}{G_r}. \tag{23}$$

The heterogeneous allocation's objective value is

$$\Phi_A(m^{\mathrm{het}}) = K_\kappa G_r 2^{-2(\overline{m} - m_0)}. \tag{24}$$

For comparison, let the uniform policy be $m_c^{\mathrm{uni}} \coloneqq \overline{m}$. Direct substitution gives

$$\Phi_A(m^{\mathrm{uni}}) = K_\kappa M_r 2^{-2(\overline{m} - m_0)}. \tag{25}$$

Corollary C.10 (Exact heterogeneity dividend over uniform precision)**.** *Under the assumptions and notation of Theorem C.9, the exact improvement over the uniform policy is*

$$\frac{\Phi_A(m^{\mathrm{uni}})}{\Phi_A(m^{\mathrm{het}})} = \frac{M_r}{G_r} \geq 1, \qquad 1 - \frac{\Phi_A(m^{\mathrm{het}})}{\Phi_A(m^{\mathrm{uni}})} = 1 - \frac{G_r}{M_r}. \tag{26}$$

Equality holds if and only if the cost-normalized risks A_c/κ_c are constant.

Proposition C.11 (Box-constrained heterogeneity gain)**.** *Under the assumptions and notation of Theorem C.9, if the vector in Eq. 23 lies in $[m_{\min}, m_{\max}]^{\mathcal{C}}$, the vector is also the optimum of the boxed inequality-budget problem in Eq. 3. More generally, if $m_{\min} < \overline{m} < m_{\max}$, the boxed RAR-log optimum is no worse than uniform and is strictly better whenever the r_c are not all equal, even when some coordinates of Eq. 23 are clipped.*

Appendix H.7 gives the complete proofs. In the interior regime, the exact heterogeneity dividend M_r/G_r shows that concentrated cost-normalized role risk creates a large potential gap between uniform and role-aware precision, whereas an almost flat atlas leaves little allocation gain. For the additive surrogate, the exact-oracle comparison in Table 6 finds that the implemented RAR-log policy is within 0.1% of the exact additive optimum after matched budget normalization.

D Detailed Experimental Setup and Allocation Evaluations

D.1 Policy Hierarchy and Objective Variants

RAR names the measurement-and-allocation framework. RAR-log is the framework's core output-KL allocation rule (Sec. 3.2), and Smooth-RAR-log is the main combination with activation smoothing (Sec. 3.3). Post-smoothing recalibration updates the risk atlas. RAR-task, RAR-margin, and

Table 4: Components of the policy-name convention. Prefixes compose from left to right; the final component names the allocation objective.

Component	Meaning	Example
no prefix	Base activations and base risk atlas	RAR-log
Smooth-	Smoothing at evaluation, base atlas reused	Smooth-RAR-log
Smooth-Recalib-	Smoothing plus a residual atlas	Smooth-Recalib-RAR-log
-log	Output KL risk objective	RAR-log
-task	KL plus decision-instability terms	RAR-task
-margin	Rank-normalized KL and decision terms	RAR-margin
-constrained	Minimum KL subject to the task proxy	RAR-constrained

RAR-constrained extend the allocation objective to decision stability (Appendix D.1.2); RAR-NLL, RAR-role, and RAR-layer are calibration-metric and granularity ablations.

Policy names follow a transform–atlas–objective convention. Table 4 decodes the components used throughout the paper. The -task, -margin, and -constrained suffixes change the allocation objective; the Smooth- and Smooth-Recalib- prefixes specify the transform and calibration atlas. For example, Smooth-Recalib-RAR-task combines smoothing, a post-smoothing atlas, and the task-aware objective. When an objective suffix is omitted from Smooth-Recalib-RAR, the name refers to the family of recalibrated variants.

Output KL is the core allocation signal and measures changes in the full next-token distribution. When the two most likely tokens have similar probabilities, even a small distribution change can switch the top-ranked token. The same single-site forward passes also provide target-token NLL, margin loss, and top-1 flips for the decision-aware extensions in this section, which defines the decision statistics and allocation objectives.

D.1.1 Decision Statistics and Margin-Aware Score

Average output KL is not identical to discrete decision stability. Lemma C.6 in Appendix C makes the pointwise gap precise: A top-1 change requires KL at least as large as the position-specific threshold χ_t. A small corpus-average KL can therefore coexist with top-1 changes at positions with small thresholds. For each prediction position, let w_t be the clean top-1 token and let

$$\gamma_t \coloneqq a_{\theta,t,w_t}(x) - \max_{j \neq w_t} a_{\theta,t,j}(x) \tag{27}$$

be the clean top-1 margin. Under site quantization, define the margin retained for the clean winner

$$\widetilde{\gamma}_t^{c,m} \coloneqq a_{\theta,t,w_t}^{c,Q_m}(x) - \max_{j \neq w_t} a_{\theta,t,j}^{c,Q_m}(x). \tag{28}$$

We record top-1 flip rate, positive margin loss

$$M_c^{(m)} \coloneqq \mathbb{E}_{x,t}[\max(0, \gamma_t - \widetilde{\gamma}_t^{c,m})], \tag{29}$$

and target-token NLL increase. More explicitly, using one fixed deterministic tie-breaking convention and the same example–position averaging as above,

$$\mathrm{flip}_c^{(m)} \coloneqq \mathbb{E}_{x,t}\left[\mathbb{I}\left[\arg\max_j a_{\theta,t,j}^{c,Q_m}(x) \neq w_t\right]\right],$$

$$\Delta\mathrm{NLL}_c^{(m)} \coloneqq \mathbb{E}_{x,t}[-\log p_{\theta,t}^{c,Q_m}(y_{t+1} \mid x) + \log p_{\theta,t}(y_{t+1} \mid x)].$$

The margin-aware score used in experiments is a rank-normalized mixture. For a nonnegative site-indexed vector v, let $\mathrm{rank}(v_c) \in (0, 1]$ denote the increasing empirical percentile rank of v_c across sites, using average ranks for ties; if every entry is zero, all ranks are set to one. Writing $[u]_+ \coloneqq \max\{u, 0\}$, the score is

$$S_c^{\mathrm{margin}} \coloneqq 0.5\,\mathrm{rank}(A_c^{(4)}) + 0.2\,\mathrm{rank}(\mathrm{flip}_c^{(4)}) + 0.2\,\mathrm{rank}(M_c^{(4)}) + 0.1\,\mathrm{rank}([\Delta\mathrm{NLL}_c^{(4)}]_+).$$

The margin-aware score combines output KL with top-1-related quantities for evaluation on discrete decision metrics. The margin quantities are highly correlated with KL risk and are measured from the same single-site quantized forward passes. The margin statistics complement the atlas by characterizing the sensitivity of discrete predictions.

D.1.2 Decision-Aware Extensions and Ablations

RAR-log allocates precision to preserve the next-token distribution. Decision-aware extensions additionally account for which token or answer the model selects. RAR-task combines output KL with target-token NLL increases, margin loss, and top-1 flips measured on the same calibration text. We use the post-smoothing form of RAR-task, Smooth-Recalib-RAR-task, as the main decision-aware extension in our downstream comparisons.

RAR-margin and RAR-constrained explore alternative ways to incorporate the token-level decision statistics. RAR-margin combines percentile ranks of KL risk and decision-instability statistics, while RAR-constrained minimizes predicted KL subject to a bound on a task-instability proxy. The decision-aware extensions retain the same activation sites and cost-weighted budget and change the allocation objective. The definitions of RAR-margin and RAR-constrained, along with the calibration-metric and granularity ablations, are given below.

Task-aware extension: RAR-task. RAR-task extends the core output-KL objective with penalties for target-token NLL increases, margin loss, and top-1 flips on the calibration distribution, retaining relative magnitudes within each statistic:

$$S_c^{\text{task}} := A_c^{(4)} + 0.5\lambda_{\text{nll}}[\Delta\text{NLL}_c^{(4)}]_+ + 0.25\lambda_{\text{margin}}M_c^{(4)} + 0.25\lambda_{\text{flip}}\text{flip}_c^{(4)}. \tag{30}$$

Let $\text{med}_+(v)$ denote the median of the strictly positive entries of a site-indexed vector, with value one by convention when no entry is positive, and set

$$V_{\text{nll},c} := [\Delta\text{NLL}_c^{(4)}]_+, \quad V_{\text{margin},c} := M_c^{(4)},$$
$$V_{\text{flip},c} := \text{flip}_c^{(4)}, \qquad \lambda_j := \frac{\text{med}_+(A^{(4)})}{\max\{\text{med}_+(V_j), 10^{-12}\}}, \quad j \in \{\text{nll}, \text{margin}, \text{flip}\}.$$

Thus each nonzero auxiliary statistic is rescaled to the positive median of $A_c^{(4)}$; when an auxiliary vector is identically zero, the corresponding multiplier is immaterial. The rescaling puts the auxiliary decision-instability terms on a scale comparable to KL risk. We call the resulting policy RAR-task. The RAR-task coefficients are fixed from calibration statistics and are not optimized on validation data.

Let G_c denote the auxiliary task-instability part of Eq. 30 without the KL term. A coefficient-sensitivity analysis considers $\lambda_{\text{task}} \in \{0, 0.125, 0.25, 0.5, 1, 2, 4, 8\}$ in $A_c^{(4)} + \lambda_{\text{task}}G_c$ at the 6-bit budget. The default $\lambda_{\text{task}} := 1$ reduces the additive task-instability proxy by 5.7% relative to RAR-log while increasing the additive KL proxy by 1.4%; over the five smoothed atlases the corresponding changes are 10.7% lower task proxy and 2.5% higher KL proxy.

Constrained alternative: RAR-constrained. RAR-constrained expresses the decision-aware objective as a constraint. For a candidate policy m, define the additive proxy $\widehat{G}(m) := \sum_c G_c 2^{-2(m_c-4)}$. RAR-constrained scans Lagrangian policies parameterized by $A_c^{(4)} + \eta G_c$ and selects the feasible candidate with the smallest predicted KL proxy subject to

$$\widehat{G}(m) \le \widehat{G}(m^{\text{uniform6}}). \tag{31}$$

On the base atlases, and on smooth policies that reuse the base atlas, the constraint is inactive at the 6-bit budget: RAR-log already satisfies Eq. 31. On the matched smoothed-risk atlases, the constrained scan selects the KL-minimizing feasible policy, which coincides with Smooth-Recalib-RAR-log but differs from the fixed-coefficient Smooth-Recalib-RAR-task policy at 37 to 49 sites on the four 7B checkpoints evaluated downstream.

Rank-based alternative: RAR-margin. RAR-margin uses Eq. 4 with $A_c := S_c^{\text{margin}}$. The rank-normalized RAR-margin score tests a different way of combining KL risk with decision-instability statistics.

Calibration-metric and granularity ablations. Three ablations assess the allocation signal. RAR-NLL uses the positive single-site target-token NLL increase in place of output KL. RAR-role

replaces each site's KL score with the mean KL risk over all sites of the same functional role, then applies the RAR-log allocation rule. RAR-layer averages over each layer and ignores role identity. The RAR-NLL, RAR-role, and RAR-layer ablations isolate the effects of the calibration metric and of resolving risk jointly by layer and role.

D.2 Activation-Site Boundaries and Cost Accounting

For a GPT-2-style decoder, the role set is

$$r \in \{\mathrm{QK}, \mathrm{V}, \mathrm{AttnOut}, \mathrm{MLPHid}, \mathrm{MLPOut}, \mathrm{Norm1}, \mathrm{Norm2}\},$$

where QK, V, $\mathrm{AttnOut}$, MLPHid, MLPOut, $\mathrm{Norm1}$, and $\mathrm{Norm2}$ denote QK routing, V transport, attention output, MLP hidden state, MLP output, and the first and second normalization outputs, respectively. Table 3 in Appendix A lists the notation used throughout the paper.

Checkpoints expose fused or separate projections, and normalization sites follow each architecture's pre- or post-sublayer placement. These architecture-specific layouts determine the exact hook boundaries. Checkpoints with separate Q and K projections, such as OPT-125M, expose Q and K as distinct sites. The GPT-NeoX architecture used by Pythia stores a fused query, key, and value (QKV) activation in head-major order, so we split QK from V by reshaping the output as $(n_{\mathrm{head}}, 3d_{\mathrm{head}})$ rather than using GPT-2's contiguous Q, K, and V segments. Checkpoints based on Large Language Model Meta AI (LLaMA) separate the MLP gate, up, and down projections, and OLMo-2 is post-sublayer normalized, so OLMo-2's normalization roles are post-attention and post-MLP outputs rather than pre-attention and pre-MLP inputs. The resulting site counts range from 84 for GPT-2 to 288 for Mistral-7B and OLMo-2-7B; Appendix D lists the site counts per checkpoint.

The normalization roles correspond to layer normalization (LayerNorm) (Ba et al., 2016) in GPT-2 and OPT and to root mean square layer normalization (RMSNorm) (Zhang & Sennrich, 2019) in the LLaMA-style checkpoints. We distinguish pre- and post-normalization roles because normalization placement has distinct Transformer stability behavior (Xiong et al., 2020).

Activation-width costs. Let H denote the model hidden size. Fused GPT-2 QK has cost $2H$, value and projection outputs have cost H, and MLP hidden activations have cost $4H$. For GQA (Ainslie et al., 2023) and multi-query attention architectures, κ_c uses the actual projection output width rather than the hidden size: TinyLlama K and V sites have width 256 and Qwen2.5-7B K and V sites have width 512, while the Q and output projections of both architectures retain hidden-width cost.

D.3 Smoothing Calibration and Residual-Risk Measurement

For a site with feature width d_c, we estimate a per-feature maximum over calibration examples and token positions and apply a positive numerical floor, $u_{c,j} \coloneqq \max\{\max_{x,t} |z_{c,t,j}(x)|, 10^{-8}\}$, before setting

$$s_{c,j} \coloneqq \mathrm{clip}_{[10^{-2},10^{2}]}\left(\left(\frac{u_{c,j}}{\exp(\frac{1}{d_c}\sum_k \log u_{c,k})}\right)^{\alpha}\right), \qquad \alpha \coloneqq 0.5. \tag{32}$$

Post-smoothing recalibration. For the 7B-scale checkpoints, we remeasure single-site 4-bit risk with $\widetilde{Q}_4$ after estimating s_c, giving

$$A^{(4)}_{c,\mathrm{smooth}} \coloneqq \mathbb{E}_{x\sim\mathcal{D}_{\mathrm{cal}}}\left[\frac{1}{T(x)-1}\sum_{t=1}^{T(x)-1} D_{\mathrm{KL}}(p_{\theta,t}(\cdot \mid x) \| p^{c,\widetilde{Q}_4}_{\theta,t}(\cdot \mid x))\right]. \tag{33}$$

Applying Eq. 4 to $A^{(4)}_{c,\mathrm{smooth}}$ yields Smooth-Recalib-RAR-log. Using the smoothed auxiliary statistics in Eq. 30 yields Smooth-Recalib-RAR-task.

D.4 Reference Implementation of RAR

Listings 1 and 2 implement the single-site measurement in Eq. 2. Each `ChannelSpec` stores a site key, module path, functional role, projection segment, and activation-width cost. The imported architecture adapters implement the hook boundaries in Appendix D.2, including the head-major GPT-NeoX QKV layout. The shared `fake_quantize` utility implements symmetric per-token quantization with optional feature scales from Eq. 5.

Calibration uses equal-sized, fixed-length token blocks, so averaging batch means gives the example–position average in Eq. 2. Passing `smooth_scales=None` measures the base atlas; supplying feature scales measures the post-smoothing atlas. The full implementation also records flip, margin-loss, and target-NLL statistics from the same forward passes.

Listing 1: Single-site quantization hook.

```python
from contextlib import contextmanager

import torch
import torch.nn.functional as F

from rar.channels import (
    channel_specs, get_submodule, model_hidden_size
)
from rar.data import move_batch
from rar.instrument import (
    fake_quantize, replace_segment, split_segment
)


@contextmanager
def quantize_channel(
    model, spec, bits, hidden_size, smooth_scales=None
):
    """Quantize exactly one layer-role site during a forward pass."""
    module = get_submodule(model, spec.module_name)

    def hook(_module, _inputs, output):
        base = output[0] if isinstance(output, tuple) else output
        _, segment = split_segment(base, spec.segment, hidden_size)
        feature_scale = (
            smooth_scales.get(spec.key) if smooth_scales else None
        )
        quantized = fake_quantize(
            segment, bits, feature_scale=feature_scale
        )
        replaced = replace_segment(
            base, spec.segment, hidden_size, quantized
        )
        if isinstance(output, tuple):
            return (replaced,) + tuple(output[1:])
        return replaced

    handle = module.register_forward_hook(hook)
    try:
        yield
    finally:
        handle.remove()
```

Listing 2: Single-site quantization-risk measurement.

```
@torch.no_grad()
def measure_quantization_risk_atlas(
    model, loader, device, bits=4, max_batches=None, smooth_scales=None
):
    specs = channel_specs(model)
    hidden_size = model_hidden_size(model)
    totals = {spec.key: 0.0 for spec in specs}
    counts = {spec.key: 0 for spec in specs}

    for step, batch in enumerate(loader):
        if max_batches is not None and step >= max_batches:
            break
        input_ids = move_batch(batch, device)
        clean_logp = F.log_softmax(
            model(input_ids).logits[:, :-1, :].float(), dim=-1
        )
        for spec in specs:
            with quantize_channel(
                model, spec, bits, hidden_size, smooth_scales
            ):
                quantized_logp = F.log_softmax(
                    model(input_ids).logits[:, :-1, :].float(), dim=-1
                )
            risk = F.kl_div(
                quantized_logp, clean_logp,
                reduction="none", log_target=True
            ).sum(dim=-1).mean()
            totals[spec.key] += float(risk.detach().cpu())
            counts[spec.key] += 1

    atlas = {
        key: totals[key] / max(counts[key], 1) for key in totals
    }
    return specs, atlas
```

The resulting atlas supplies $A_c^{(4)}$ to the RAR-log rule in Eq. 4; at evaluation time, the assigned bit-widths are applied through one grouped forward hook per module. The packed-forward evaluation reconstructs the quantized activations through an integer pack–unpack–dequantize round trip using the same atlas and allocation procedure.

D.5 Models, Data, and Evaluation Protocols

Models. We evaluate GPT-2 small (Radford et al., 2019), which has 124M parameters, 12 layers, and 84 sites, and OPT-125M (Zhang et al., 2022), which has 12 layers and 96 sites. We also evaluate the TinyLlama-1.1B intermediate checkpoint (Zhang et al., 2024). TinyLlama is a LLaMA-style model (Touvron et al., 2023) with 22 layers, hidden size 2048, 32 Q heads, and 4 KV heads. TinyLlama uses RoPE (Su et al., 2024), RMSNorm, Swish-gated linear units (Shazeer, 2020), and 198 sites.

Pythia-6.9B (Biderman et al., 2023) is a GPT-NeoX model with 32 layers and 224 sites. Mistral-7B (Jiang et al., 2023) has 32 layers, hidden size 4096, 32 Q heads, 8 KV heads, GQA and sliding-window attention, and 288 sites. Qwen (Yang et al., 2024) has 28 layers, hidden size 3584, 28 Q heads, 4 KV heads, intermediate size 18944, and 252 sites. Qwen-Instruct is the instruction-tuned checkpoint with the same architecture and hook set. We refer to OLMo-2-1124-7B (OLMo et al., 2025; 2024) as OLMo-2-7B below. OLMo-2-7B has 32 layers, hidden size 4096, post-sublayer normalization, separate Q, K, and V projections, and 288 sites.

TinyLlama, Mistral, and Qwen test modern GQA architectures (Ainslie et al., 2023) with larger hidden dimensions than GPT-2 and OPT; Pythia adds a 7B-scale architecture with multi-head attention

and a different fused-QKV implementation; OLMo-2 adds a fully open 7B-scale post-normalized architecture with multi-head attention.

Risk calibration. Risk atlases are calibrated on WikiText-2 train (Merity et al., 2017). GPT-2 uses 64 calibration blocks and 16 risk batches. OPT-125M uses 48 blocks and 8 risk batches. TinyLlama uses 16 blocks and 4 risk batches. All five 7B-scale checkpoints use 32 calibration blocks and four 4-bit risk batches. Risk curves from 3 to 8 bits additionally use one batch for Pythia-6.9B and two batches for Qwen2.5-7B; the 3-to-8-bit risk curves are not measured for Qwen2.5-7B-Instruct, Mistral-7B, or OLMo-2-7B.

Matched smoothed 4-bit risk atlases for Pythia-6.9B, Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B support LM evaluation of Smooth-Recalib-RAR-log, Smooth-Recalib-RAR-task, Smooth-Recalib-RAR-role, and Smooth-Recalib-RAR-layer. Smooth-Recalib-RAR-margin is evaluated on Pythia-6.9B, Qwen2.5-7B, Qwen2.5-7B-Instruct, and Mistral-7B; Smooth-Recalib-RAR-constrained is evaluated on Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B. The custom-scored downstream evaluation with up to 512 examples uses clean, uniform 6-bit, smooth uniform, Smooth-RAR-log, and Smooth-Recalib-RAR-task; the full MMLU evaluation additionally includes Smooth-Recalib-RAR-constrained.

LM evaluation. The main LM evaluations for all eight checkpoints use sequence length 256; the custom multiple-choice scorer uses a maximum sequence length of 512. LM is evaluated on WikiText-2 validation, C4 validation (Raffel et al., 2020), and a FineWeb-Edu streamed sample (Penedo et al., 2024). GPT-2 uses 512 evaluation blocks, OPT uses 256 blocks, TinyLlama and Pythia use 128 blocks, and Mistral, OLMo-2, and both Qwen2.5 checkpoints use 256 blocks for each corpus.

Downstream evaluation. Downstream tasks include the Easy and Challenge splits of ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), BoolQ (Clark et al., 2019), OpenBookQA (Mihaylov et al., 2018), WinoGrande (Sakaguchi et al., 2020), MMLU (Hendrycks et al., 2021), and RACE (Lai et al., 2017). In addition to the standard-harness evaluations below, we use a custom conditional-likelihood scorer for an evaluation of Qwen2.5-7B with up to 512 examples on ARC-Challenge, HellaSwag, MMLU, and RACE, and for Mistral-7B on ARC-Challenge, HellaSwag, and MMLU. ARC-Challenge uses all 299 available validation examples, while the other listed model–task pairs use the first 512. We score each choice by conditional log-likelihood given the prompt. We report accuracy under both total and length-normalized choice scores. The full MMLU validation evaluation uses all 1,531 examples and six policies on Qwen, Qwen-Instruct, Mistral, and OLMo-2, with the same custom choice-likelihood scorer. The RACE evaluation uses up to 512 of the 4,887 long-document validation examples. We report custom-scored results separately from the standard `lm-evaluation-harness` evaluations.

Longer-context evaluation. A separate LM evaluation considers Qwen2.5-7B, Mistral-7B, and OLMo-2-7B at sequence length 1024 on WikiText-2 and C4, using 64 blocks per corpus and the clean, uniform 6-bit, smooth uniform, Smooth-RAR-log, and Smooth-Recalib-RAR-task policies. The 1024-token LM results are excluded from the 256-token LM aggregate.

D.6 Allocation Baselines and PTQ Controls

Policy comparisons. Using RAR-log results, we examine whether allocating activation bits according to output KL risk per unit cost preserves predictions better than uniform precision at the same target budget. Comparing Smooth-RAR-log with smooth uniform measures the allocation benefit after activation smoothing, and Smooth-Recalib-RAR-log tests the refinement from remeasuring the atlas. Decision-aware extensions examine alternative allocation objectives for downstream tasks, while calibration-metric and granularity ablations probe the allocation signal.

Core policies, extensions, and ablations. We compare RAR-log with the clean reference and uniform activation policies from 3 to 8 bits. Smooth-RAR-log is the main combined policy, and Smooth-Recalib-RAR-log tests the additional effect of updating the atlas after smoothing. RAR-task is the main decision-aware extension, with RAR-margin and RAR-constrained providing alternative

objectives. RAR-NLL, RAR-role, and RAR-layer are ablations of the calibration metric and allocation granularity. Additional mixed-bit baselines use activation RMS, maximum-to-RMS outlier scores, attention entropy, random assignments, or measured-curve greedy allocation.

We also evaluate smooth uniform and the log, margin, task, NLL, and layer variants of Smooth-RAR. Post-smoothing recalibration uses the log, margin, task, constrained, role, and layer variants of Smooth-Recalib-RAR. The allocation objectives are defined in Appendix D.1. All mixed policies use the same cost-weighted average bit-width accounting. At each budget, the random mixed-policy control uses three independently sampled allocations; the randomization protocol is described in Appendix D.9. For the remaining policy variants, table headers use Smooth-RAR-margin (SmMargin) and Smooth-Recalib-RAR-constrained (SmConstr).

Weight-PTQ controls. A W8 compatibility evaluation applies symmetric groupwise 8-bit quantization to every role-bearing `torch.nn.Linear` weight matrix, using 128-column groups and 16-bit floating point (FP16) scale accounting, then evaluates the same activation policies on top. We evaluate an activation-aware clipped-and-equalized variant for OPT-125M, Qwen2.5-7B, Mistral-7B, and OLMo-2-7B. Calibration RMS values weight the reconstruction error while scanning clipping ratios $\{1.0, 0.95, 0.9, 0.85\}$ and AWQ-lite (Lin et al., 2024) equalization exponents $\{0, 0.25, 0.5, 0.75\}$. Each equalized linear module computes $(x/s)Q(Ws)^\top$, selecting $s_j \propto \mathrm{rms}_j^\rho$ by activation-weighted reconstruction error. The resulting W8 setup is the reference against which residual activation degradation is measured.

The same clipped-and-equalized AWQ-lite W8 setup is evaluated with packed activations on Qwen2.5-7B, Mistral-7B, and OLMo-2-7B. In the packed-activation evaluation, each selected activation site is encoded to a dense uint8 bitstream, unpacked, dequantized, and then fed to the next module. The packed W8 evaluation uses WikiText-2 and C4, sequence length 128, and 32 evaluation blocks per corpus. For task-level PTQ evidence, the same clipped-and-equalized AWQ-lite W8 path is evaluated through `lm-evaluation-harness` (Biderman et al., 2024) on the full BoolQ validation set for Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B. The task-level PTQ evaluation uses all 3,270 examples, sequence length 1024, and the clean, uniform 6-bit, smooth uniform, Smooth-RAR-log, Smooth-Recalib-RAR-task, and Smooth-Recalib-RAR-constrained activation policies; results appear in Table 18.

Module-path control. An int8 weight-only control using PyTorch Architecture Optimization (TorchAO) (Or et al., 2025) evaluates activation allocation with quantized linear modules on Qwen2.5-7B, Mistral-7B, and OLMo-2-7B. The TorchAO path converts all role-bearing `torch.nn.Linear` modules with TorchAO's int8 weight-only configuration, producing `AffineQuantizedTensor` weights, and then evaluates the same activation policies on WikiText-2 and C4 with sequence length 128 and 32 evaluation blocks.

D.7 Hardware and Calibration Cost

Hardware. Experiments used an Intel accelerator system with 12 Intel Data Center GPU Max 1550 devices, with independent model evaluations assigned to separate devices. Model and dataset caches were stored locally.

Calibration cost. Table 5 reports the measured calibration timings. With four risk batches, the single-site 4-bit pass takes 64.5 seconds on TinyLlama and between 59.4 and 99.2 seconds on the 6.9B- and 7B-scale checkpoints. The core atlas time covers the activation proxy pass, the single-site 4-bit risk pass, and smoothing-scale estimation. The optional curve pass, 94.4 seconds for one Pythia-6.9B batch and 158.7 seconds for two Qwen2.5-7B batches, supports the measured risk curves from 3 to 8 bits; the default RAR-log policy uses only the 4-bit atlas.

D.8 Systems and Allocation Evaluations

Packed-path evaluation. A standalone accelerator kernel written in Triton (Tillet et al., 2019) tests device-level unpacking and dequantization of mixed-bit activation payloads for Qwen2.5-7B, Mistral-7B, and OLMo-2-7B. Activations are grouped by the bit-width assigned by each policy. The kernel unpacks the uint8 bitstream, applies per-row dynamic scales, and is compared with the ref-

Table 5: Calibration cost on the Intel accelerator system. All timed rows use four 4-bit risk batches and 256-token calibration blocks; TinyLlama uses 16 blocks and the 7B-class rows use 32. Core atlas time excludes the optional risk-curve pass from 3 to 8 bits.

Checkpoint	Sites	Calib. tokens	4-bit risk (s)	Core atlas (s)	Curve (s)
TinyLlama-1.1B	198	4096	64.5	66.9	–
Pythia-6.9B	224		67.4	70.9	94.4
Qwen2.5-7B	252		59.4	64.1	158.7
Qwen2.5-7B-Instruct	252	8192	60.1	65.0	–
Mistral-7B	288		65.6	71.1	–
OLMo-2-7B	288		99.2	105.0	–

erence PyTorch unpack-and-dequantize path. Across uniform 6-bit, smooth uniform, and Smooth-RAR-log for the three 7B checkpoints, all nine evaluations have zero encoded round-trip mismatches and zero maximum dequantization error. The measured standalone kernel processes 2.48B values per second on average, with between 10.35 and 11.39 gibibytes per second of effective traffic when counting packed payload reads, 32-bit FP scale reads, and 32-bit FP output writes.

A pack–unpack–replace forward evaluation for Qwen2.5-7B, Mistral-7B, and OLMo-2-7B materializes the packed uint8 payload, unpacks and dequantizes the packed payload, and feeds the restored activation to downstream layers. On 32-block WikiText-2 and C4 evaluations at sequence length 128, the packed path preserves the same qualitative ordering as the fake-quantized policy evaluation. On Qwen, uniform 6-bit produces the largest NLL increase; smoothing substantially reduces the increase, and the smoothed RAR policies reduce the remaining NLL gap further. Smooth-RAR-log reduces NLL relative to smooth uniform on all six model–corpus pairs, by 0.199 nats per token on average. The mean reductions for Smooth-Recalib-RAR-task and Smooth-Recalib-RAR-constrained are both 0.203 nats per token. Payload ratios remain between 37.52% and 37.57% of FP16 after dynamic scales, so the allocation benefit persists after integer packing and reconstruction at the target budget.

Weight-PTQ compatibility. Plain W8 weight-PTQ compatibility evaluations for OPT-125M, Qwen2.5-7B, Mistral-7B, and OLMo-2-7B use dequantized 8-bit groupwise weights in the role-bearing linear modules. We then evaluate unquantized activations, uniform 6-bit activations, smooth uniform, Smooth-RAR-log, and Smooth-Recalib-RAR policies on WikiText-2 and C4 at sequence length 128 with 64 evaluation blocks. For the same four checkpoints, we also evaluate activation-aware clipping and AWQ-lite equalization selected from 16 WikiText-2 calibration blocks. A matched clipped-and-equalized Pythia-6.9B evaluation uses 32 evaluation blocks per corpus. The theoretical W8 weight payload is 50.0% of FP16 weights, and per-group scales raise the W8 payload ratio to 50.8%.

Smoothing and recalibration. To assess whether allocation should be calibrated before or after smoothing, smoothed single-site 4-bit risk atlases are measured for the five 7B-scale checkpoints using the same four risk batches. Smoothing reduces total single-site risk by between 46.2% and 76.6% depending on the checkpoint, but the residual risk is still not uniform, which motivates the Smooth-Recalib-RAR policies evaluated below. Appendix G reports the per-checkpoint and per-role reductions.

Exact additive oracle. The measured-curve greedy baseline uses empirical risk curves, whose two-batch estimates are noisy on Qwen. To assess the allocation rule independently of curve-estimation noise, we solve the additive objective optimized by RAR-log exactly. For each evaluated atlas, we solve the exact integer DP

$$m^{\mathrm{DP}} \in \arg\min_{m_c \in \{3,\dots,8\}} \sum_c A_c 2^{-2(m_c - m_0)} \quad \text{s.t.} \quad \frac{\sum_c \kappa_c m_c}{\sum_c \kappa_c} = 6, \tag{34}$$

where $A_c \coloneqq A_c^{(4)}$ is the measured 4-bit site KL and κ_c is the activation-width cost.

The search uses the same calibration atlas as RAR-log and isolates the effect of integer rounding and budget adjustment. Table 6 reports the ratio of each policy's additive objective to the exact DP

Table 6: Exact additive-oracle allocation comparison. The DP ratio is the budget-normalized additive objective ratio relative to the exact DP optimum in Eq. 34. The uniform ratio is the corresponding ratio relative to uniform 6-bit. Hamming denotes the fraction of sites whose bit-width differs from the DP policy. Reported values are averages over the checkpoints indicated in the first column.

Atlas	Policy	DP ratio	Uniform ratio	Hamming
Base, 8 checkpoints	Uniform 6-bit	3.593	1.000	0.686
	RAR-log	1.000	0.337	0.019
	RAR-task	1.015	0.342	0.130
Smoothed residual, 5 checkpoints	SmUni	2.876	1.000	0.597
	SmRecalib	1.000	0.469	0.015
	SmTask	1.026	0.480	0.185

optimum. Because several implemented heuristic policies have cost-weighted drift below 0.011 bits from the target, a policy with realized average $\overline{m}_p$ is normalized by multiplying the policy's additive objective by $2^{-2(6-\overline{m}_p)}$ before taking the reported 6-bit ratio.

Under the additive objective, RAR-log is within 0.1% of the exact integer optimum on the base atlas and within 0.02% on the post-smoothing residual atlas. The gap between uniform 6-bit and the DP policy quantifies the benefit of heterogeneous allocation under the additive risk objective. Theorem C.9 and Corollary C.10 give the continuous counterpart of the uniform–DP allocation comparison. Dispersion in cost-normalized role risks produces an exact weighted arithmetic–geometric mean (AM–GM) gain whenever the ideal allocation is not clipped. The task-aware policies change a modest fraction of assignments to account for decision-stability terms; the added KL cost is about 1.5% on the base atlas and 2.6% after smoothing.

D.9 Randomness and Statistical Protocol

Run-level randomness. Each experiment initializes the Python, NumPy, and PyTorch random-number generators, including the accelerator generator when applicable, with one fixed base seed. The fixed base seed determines the calibration and evaluation order, policy construction, and uncertainty resampling for the recorded configuration. We report seed counts in the paper; the metadata and launch configurations record the seed values needed to reproduce each evaluation.

Random mixed-policy control. For every bit budget, we construct three independently sampled mixed-bit allocations that satisfy the same cost-weighted budget as the corresponding RAR and proxy policies. Each realization is recorded as a distinct policy, separate from uncertainty resamples. The random-policy realizations test whether an allocation informed by the atlas improves on a chance assignment at the same resource budget.

Uncertainty estimates. Bootstrap CIs use deterministic streams derived from the run-level base seed. Bootstrap resamples quantify finite evaluation-set uncertainty; the resamples are not additional random-policy realizations or independent experimental seeds. Per-run metadata records the seed and resampling settings alongside the reported metrics.

LM policy comparisons. For the 24 model–corpus pairs in Table 1, Smooth-RAR-log lowers NLL relative to smooth uniform in every pair. Treating the 24 pairs as independent gives one-sided sign-test and paired Wilcoxon p-values of 5.96×10^{-8}; we regard the pair-level sign and Wilcoxon tests as descriptive because corpora within a checkpoint are not independent. All eight checkpoint-level mean differences also favor Smooth-RAR-log, giving a one-sided sign-test p-value of $2^{-8} = 0.0039$.

D.10 Limitations

The experiments cover eight Transformer checkpoints from 124M to 7B parameters, three LM corpora, and selected multiple-choice benchmarks. The evaluation spans several architectures and precision settings within a defined range of model sizes, data domains, and context lengths. Broader

Table 7: Full policy comparison for Table 1: average 6-bit LM degradation reduction (%) by checkpoint, as defined in Eq. 6. Higher is better, and a negative entry means the policy is worse than uniform 6-bit. Checkpoint rows average WikiText-2, C4, and FineWeb-Edu. Aggregate rows are unweighted means over the indicated number of model–corpus evaluations and compare only policies evaluated on the same set. A dash denotes unavailable coverage.

Model	RAR-log	SmUni	SmRAR	SmRecalib	SmTask	SmMargin
GPT-2	65.0	84.8	89.2	–	–	89.5
OPT-125M	74.3	87.9	89.1	–	–	89.2
TinyLlama-1.1B	68.5	84.9	91.5	–	–	90.0
Pythia-6.9B	85.0	95.3	97.1	97.5	97.6	96.9
Qwen2.5-7B	96.2	90.5	99.4	99.4	99.4	99.1
Qwen2.5-7B-Instruct	95.5	89.6	98.9	98.9	98.9	–
Mistral-7B	−38.7	75.3	90.7	91.1	91.4	–
OLMo-2-7B	75.5	90.6	96.8	97.1	96.9	–
Small-to-1B (9)	69.3	85.9	90.0	–	–	89.6
Matched 7B (15)	62.7	88.3	96.6	96.8	96.8	–
Margin matched (15)	77.8	88.7	93.3	–	–	92.9
All (24)	65.2	87.4	94.1	–	–	–

Matched 7B contains the five checkpoints with post-smoothing atlases. Margin matched contains GPT-2, OPT-125M, TinyLlama, Pythia, and Qwen. The All row includes only policies evaluated for all eight checkpoints.

evaluation on larger models and more diverse tasks and datasets would further clarify how RAR's allocation benefits extend to other inference settings.

E Additional LM and Downstream Evaluations

E.1 Checkpoint-Level 6-Bit LM Results

NLL differences below are measured in nats per token relative to clean activations. Triples follow the order WikiText-2, C4, and FineWeb-Edu.

TinyLlama extends the comparison to a GQA architecture using the same cost-weighted activation budget. Across WikiText-2, C4, and FineWeb-Edu, RAR-log removes 67.6%, 70.8%, and 67.0% of uniform 6-bit degradation, while Smooth-RAR-log removes 92.6%, 90.5%, and 91.5%. The allocation effect is therefore not confined to GPT-2- or OPT-style decoder blocks.

Pythia-6.9B has uniform-6-bit NLL increases of 4.30, 4.47, and 4.66 across the three corpora. The corresponding residuals are 0.76, 0.62, and 0.62 for RAR-log and 0.16, 0.11, and 0.11 for Smooth-RAR-log. Post-smoothing recalibration raises average recovery to between 97.5% and 97.7%. Smooth-RAR-role, the smoothed role-level ablation of RAR-log, reaches 97.4%, showing that coarser allocation can be competitive when risk is strongly role-concentrated.

Qwen2.5-7B exhibits substantial degradation at the evaluated hook boundaries: Uniform 6-bit raises NLL by 6.71, 6.44, and 6.78. RAR-log reduces the NLL residuals to 0.24, 0.26, and 0.25, and Smooth-RAR-log to 0.023, 0.044, and 0.046, for 99.4% average recovery. Post-smoothing recalibration is effectively tied. The layer-only smooth ablation remains close, whereas averaging over layers within each role loses the concentrated early-layer signal.

Qwen2.5-7B-Instruct retains the pattern of degradation under uniform precision and recovery with RAR after instruction tuning. The uniform NLL increases are 7.13, 7.04, and 7.32; RAR-log leaves 0.28, 0.34, and 0.34, while Smooth-RAR-log leaves 0.058, 0.093, and 0.092. The best average recovery is 98.9% for Smooth-Recalib-RAR-task.

Mistral-7B illustrates the importance of reducing outlier-induced quantization error. The uniform NLL increases are only 0.23, 0.20, and 0.20, but unsmoothed RAR-log leaves 0.30, 0.22, and 0.35. Smooth uniform recovers 75.3% on average, Smooth-RAR-log recovers 90.7%, and recalibrated policies reach between 91.1% and 91.4%. The atlas is therefore most effective after smoothing reduces outlier-induced quantization error.

OLMo-2-7B, a post-sublayer-normalized architecture with separate projections, has uniform NLL increases of 0.75, 0.51, and 0.57. RAR-log leaves 0.20, 0.12, and 0.13; Smooth-RAR-log leaves 0.023, 0.016, and 0.020; and recalibration raises average recovery to 97.1%. Together, the checkpoint-level LM results show that residual allocation remains useful after smoothing, while the relevant roles depend on the architecture.

The allocation-granularity comparison in Sec. 4.2 tests whether the allocator needs to distinguish layers within each role. RAR-role, the role-level ablation of RAR-log, averages measured KL risk across sites of the same role and uses the role-level averages in RAR-log's cost-aware allocation rule. RAR-role can therefore assign different bit-widths to different roles, whereas uniform quantization assigns the same bit-width to every site. On Pythia, Smooth-RAR-role removes 97.4% of uniform 6-bit LM degradation, compared with 97.1% for Smooth-RAR-log in Table 7. For Pythia, identifying the sensitive roles already captures much of the information needed for allocation. On Qwen, averaging within a role dilutes the high risk of specific early QK sites. Keeping separate scores for each layer–role site lets RAR-log protect the high-risk early QK sites without raising precision for every occurrence of the same role.

E.2 Constrained Post-Smoothing LM Comparison

The constrained post-smoothing comparison covers Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B. The constrained post-smoothing policy removes 96.7% of uniform degradation across 12 LM evaluations and reduces NLL relative to smooth uniform in all 12, by 0.333 nats per token on average. The constrained policy performs similarly to Smooth-Recalib-RAR-task: The constrained policy has lower NLL in 7 evaluations and higher NLL in 5, with a mean reduction of 0.0003 nats per token.

E.3 Free-Running Generation

The LM tables in Sec. 4.2 use teacher forcing, so every policy is evaluated on the same contexts. Figure 4 complements the teacher-forced comparison with a free-running evaluation in which each policy's generated tokens form the subsequent context. Each policy greedily generates 64 tokens from the first 32 tokens of the first 16 WikiText-2 validation blocks, and the aggregate curves include all 16 prompts.

Qwen's free-running generations also reflect the teacher-forced degradation. Smoothing eliminates fragmented character sequences but does not always prevent repetition, whereas Smooth-RAR-log closely matches the clean-reference NLL and repetition rate at the same 6-bit budget. Exact-prefix agreement eventually ends for every quantized policy in the 16-prompt evaluation, so we report exact-prefix agreement alongside continuation-level scores. The corresponding Mistral traces remain much closer together, consistent with the small clean–uniform gap in Table 1.

E.4 Blockwise Failure Regimes and Finite-Noise Controls

Checkpoint averages can obscure distinct patterns of quantization failure. Figure 5 traces the policies across individual validation blocks. The Qwen layerwise trace shows residual-stream corruption beginning in the first Transformer block under uniform precision, consistent with the concentrated layer-0 routing risk. On Mistral, unsmoothed RAR-log is better than uniform on 62 of the first 64 blocks, yet two large tail events reverse the mean. The control in Figure 5(c) assigns each site at least as many bits as RAR-log to test whether increasing precision at every low-precision site removes the failure. The failure persists under the higher-budget control, which is inconsistent with a bitwise-monotone additive risk model and suggests finite-noise interactions. Applying smoothing with the same RAR bit pattern removes the observed spike, motivating exposure reduction before or together with allocation.

In Figure 5(a), uniform 6-bit increases NLL by 7.01 nats per token, compared with 0.22 for RAR-log and 0.001 for Smooth-RAR-log. In Figures 5(b) and 5(c), the two Mistral spikes add 5.2 and 5.8 nats per token; applying smoothing to the same RAR allocation reduces the worst-block degradation to 0.03 nats per token.

(a) Qwen2.5-7B greedy continuations of the first WikiText-2 validation prompt at the 6-bit budget

Prompt 32-token prefix	*= Homarus gammarus = ¶ ¶ Homarus gammarus , known as the European lobster or common lobster , is a species of clawed lobster from the …*
Clean FP reference clean-ref. NLL 0.68	*… family Nephropidae . It is found in the North Atlantic Ocean , from Norway to the Mediterranean Sea , including the Baltic Sea and the Black Sea . It is the only species in the genus Homarus . It is a large lobster , reaching up to 90 cm (35 in) in length …*
Uniform 6-bit prefix match 0/64 clean-ref. NLL 5.21	*… onal ¶ .e ajar on on onon a a aE e fo0 foar tofo f oef oefof oof o fh homg gammm ommof om oomf e om f fomf ff fofmff hohom mof om …*
Smooth uniform prefix match 0/64 clean-ref. NLL 1.21	*… northern Atlantic Ocean. It is the only species in the genus Homarus native to the Atlantic coast of Europe. It is the only species in the family Nephropidae that is found in European waters.. It is the only species in the genus Homarus that is found in the Atlantic Ocean. It is the …*
Smooth-RAR prefix match 15/64 clean-ref. NLL 0.78	*… family Nephropidae . It is found in the North Atlantic Ocean from Norway to the Mediterranean Sea . It is a large lobster , reaching a maximum size of 90 cm (35 in) in length and 20 kg (44 lb) in weight . It is a long - …*

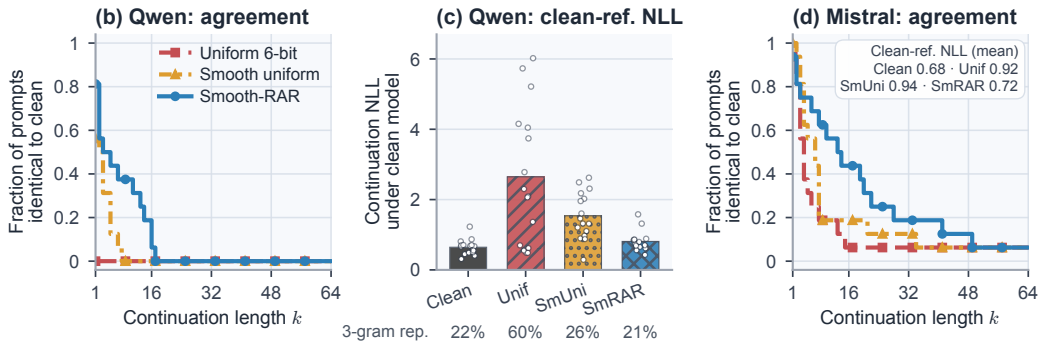

Figure 4: Matched-budget activation policies produce different free-running trajectories. Qwen2.5-7B continuations, exact-prefix agreement, and clean-reference NLL and repetition over 16 fixed prompts appear in (a) through (c), while (d) gives the corresponding Mistral-7B agreement curves. Smooth-RAR remains closest to clean decoding, whereas uniform 6-bit produces fragmented text and smooth uniform can produce repetitive continuations. Clean-reference NLL is a descriptive compatibility score rather than an independent fluency judgment. In this figure, Smooth-RAR and SmRAR denote Smooth-RAR-log.

E.5 LM AT 1024 TOKENS

The main LM evaluation uses sequence length 256. A 1024-token evaluation for Qwen2.5-7B, Mistral-7B, and OLMo-2-7B on WikiText-2 and C4 tests the same activation policies with four times the context length on validation corpora. Table 8 averages the two corpora for each model. Smooth-RAR-log improves on smooth uniform in all six model–corpus pairs. The gain is largest on Qwen, where uniform 6-bit causes the most severe degradation. Smooth uniform removes 86.9% of the degradation, while Smooth-RAR-log removes 99.2%. Mistral and OLMo-2 have smaller uniform degradation, but the same ordering holds. Across all six pairs, smooth uniform removes 87.4% of uniform degradation, Smooth-RAR-log removes 95.2%, and Smooth-Recalib-RAR-task removes 95.5%.

E.6 DOWNSTREAM EVALUATION WITH CUSTOM CHOICE SCORING

The evaluation with up to 512 examples reveals different patterns for Qwen and Mistral. Accuracy differences below are reported in percentage points. On Qwen2.5-7B, across ARC-Challenge, HellaSwag, MMLU, and RACE, uniform 6-bit lowers mean accuracy from 46.6% to 24.3%. Smooth uniform recovers 91.0% of the uniform NLL degradation for the correct answer choice and improves accuracy by 15.6 points over uniform. Smooth-RAR-log recovers 99.4% and improves accuracy by 22.4 points; Smooth-Recalib-RAR-task recovers 99.2% and improves accuracy by 21.7 points.

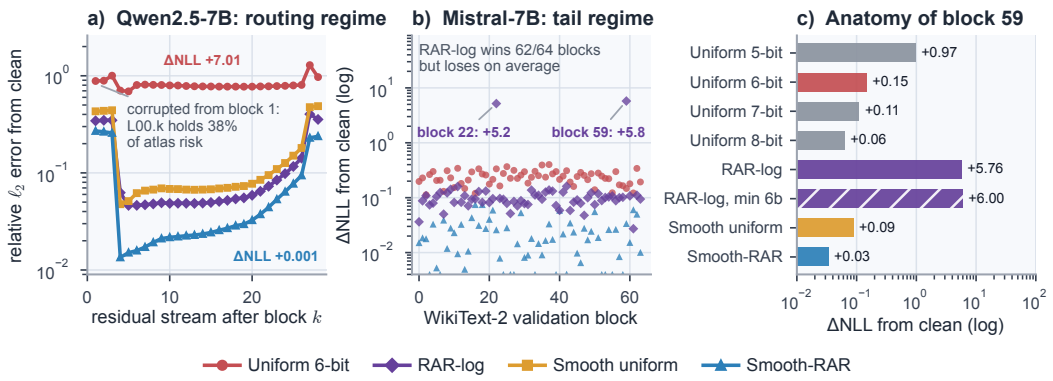

Figure 5: Architecture-dependent failure regimes under activation quantization. In (a), Qwen2.5-7B exhibits immediate residual-stream corruption under uniform 6-bit precision. On Mistral-7B, two tail events reverse RAR-log's otherwise favorable blockwise comparison in (b). The control in (c) uses at least as many bits at every site but does not remove the worst event. Smoothing removes the observed spikes. In this figure, Smooth-RAR denotes Smooth-RAR-log.

Table 8: 1024-token LM extension of Table 1. Values average the two 64-block corpus evaluations per model. Uniform ΔNLL is the increase relative to clean activations; the final column gives the NLL decrease from smooth uniform to Smooth-RAR-log. Both NLL differences are in nats per token.

		Degradation reduction (%)			
Model	Uniform ΔNLL	SmUni	SmRAR	SmTask	NLL decrease
Qwen2.5-7B	7.371	86.9	99.2	99.2	0.912
Mistral-7B	0.144	83.7	90.1	90.9	0.009
OLMo-2-7B	0.422	91.6	96.1	96.6	0.019
All six pairs	–	87.4	95.2	95.5	0.313

Table 9: Full MMLU validation with the custom multiple-choice scorer at the 6-bit activation budget. Each checkpoint uses all 1,531 validation examples. Entries are accuracies (%) under total conditional log-likelihood scoring.

Model	Clean	Uniform	SmUni	SmRAR	SmTask	SmConstr
Qwen2.5-7B	44.2	25.9	39.6	42.1	42.1	43.1
Qwen2.5-7B-Instruct	42.7	25.6	39.5	43.3	43.1	43.6
Mistral-7B	44.6	41.9	43.6	43.2	43.8	43.9
OLMo-2-7B	44.3	38.3	43.6	44.6	43.6	44.3
Mean	43.9	32.9	41.6	43.3	43.2	43.7

On Mistral-7B, across ARC-Challenge, HellaSwag, and MMLU, uniform 6-bit lowers mean accuracy only from 47.1% to 44.7%. Smooth uniform recovers 89.0% of NLL degradation and improves accuracy by 1.5 points, Smooth-RAR-log recovers 90.2% and improves accuracy by 1.9 points, and Smooth-Recalib-RAR-task recovers 89.3% and improves accuracy by 1.2 points. Qwen is highly sensitive to uniform activation precision and benefits strongly from role-aware allocation, whereas Mistral has smaller uniform degradation and gains most from smoothing.

The full MMLU evaluation in Table 9 extends the custom-scored downstream comparison to the full validation set. Smooth uniform recovers most of the four-checkpoint loss, and the role-aware policies reduce the remaining gap on average. The incremental gains are largest on the two Qwen checkpoints; Mistral is already close to clean after smoothing, while OLMo-2 shows a smaller but positive residual gain. The constrained policy attains the highest mean accuracy.

Table 10: `lm-evaluation-harness` evaluation with up to 512 examples per task at the 6-bit activation budget. Entries are unweighted mean zero-shot accuracy (%) over ARC-Easy, ARC-Challenge, HellaSwag, BoolQ, OpenBookQA, and WinoGrande. The policy columns are clean, uniform, smooth uniform, Smooth-RAR-log, and Smooth-Recalib-RAR-task, and the final column gives the largest gain among the quantized policies in percentage points relative to uniform 6-bit. OpenBookQA contributes all 500 test examples.

Model	Clean	Uniform	SmUni	SmRAR	SmTask	Best Δ
Qwen2.5-7B	61.4	30.1	56.0	60.2	59.6	+30.1
Qwen2.5-7B-Instruct	62.2	30.5	55.0	62.1	61.7	+31.5
Mistral-7B	62.3	60.2	61.7	61.4	61.4	+1.5
OLMo-2-7B	62.9	53.5	62.6	62.4	62.8	+9.3

Table 11: Extension of Table 10 to at most 1024 examples per task. Only the per-task cap changes; OpenBookQA evaluation still uses the full test set.

Model	Clean	Uniform	SmUni	SmRAR	SmTask	Best Δ
Qwen2.5-7B	60.6	30.7	55.8	60.1	59.9	+29.5
Qwen2.5-7B-Instruct	61.6	30.3	55.6	61.6	61.6	+31.3
Mistral-7B	62.6	60.8	62.1	62.1	62.3	+1.5
OLMo-2-7B	63.0	55.1	61.8	62.0	62.3	+7.2

Table 12: Full BoolQ extension of Table 10, using all 3,270 validation examples. Entries are zero-shot accuracies (%); the mean row averages the four checkpoints.

Model	Clean	Uniform	SmUni	SmRAR	SmTask
Qwen2.5-7B	84.9	39.6	74.8	83.2	83.2
Qwen2.5-7B-Instruct	85.8	39.1	74.0	84.5	84.6
Mistral-7B	84.5	81.4	83.7	84.2	83.8
OLMo-2-7B	80.0	74.6	79.5	80.6	79.1
Mean	83.8	58.7	78.0	83.1	82.7

E.7 Standard Harness Evaluations on Four 7B Checkpoints

Accuracy differences in this section are reported in percentage points. The custom-scored evaluations above report NLL for the correct answer choice and accuracy under total and length-normalized conditional-likelihood scoring. Standard `lm-evaluation-harness` zero-shot evaluations cover Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B on ARC-Easy, ARC-Challenge, HellaSwag, BoolQ, OpenBookQA, and WinoGrande. The 512-example evaluation applies a per-task cap to the standard task definitions; OpenBookQA contributes all 500 test examples. The 1024-example evaluation uses the same four checkpoints, with the OpenBookQA cap set to the test-set size. Table 2 reports results on the complete test sets for ARC-Easy, ARC-Challenge, and OpenBookQA and the complete validation set for WinoGrande, containing 2,376, 1,172, 500, and 1,267 examples, respectively. We refer to this four-task evaluation as the full-core set.

The 512-example results in Table 10 show the same pattern on the checkpoints most affected by quantization: Smooth-RAR-log nearly restores both Qwen checkpoints, and the recalibrated task policy is strongest on OLMo-2. Mistral again has a small uniform drop, leaving little residual improvement beyond smoothing.

The 1024-example evaluation in Table 11 shows architecture-dependent recovery with a larger sample. Smooth-RAR-log removes 97.2% of the Qwen2.5-7B clean–uniform gap and matches the sampled clean mean on Qwen2.5-7B-Instruct. Smoothing accounts for most of the recovery on OLMo-2, and the smoothed policies achieve similar accuracy on Mistral.

Table 13: Full HellaSwag extension of Table 10, using all 10,042 validation examples. Entries are zero-shot accuracies (%); the mean row averages the four checkpoints.

Model	Clean	Uniform	SmUni	SmRAR	SmTask
Qwen2.5-7B	59.0	26.1	50.9	57.8	57.8
Qwen2.5-7B-Instruct	60.7	25.9	52.2	59.6	59.7
Mistral-7B	61.6	59.9	61.6	61.4	61.7
OLMo-2-7B	60.4	54.6	59.7	60.0	59.9
Mean	60.4	41.7	56.1	59.7	59.8

The full BoolQ comparison in Table 12 comprises 130,800 log-likelihood requests across four checkpoints and five activation policies. Smooth-RAR-log substantially closes the residual Qwen gap and improves over smooth uniform on all four checkpoints, with a mean gain of 5.10 points. Mistral and OLMo-2 have smaller residual gaps because smoothing is already close to clean.

The full HellaSwag comparison in Table 13 comprises 803,360 log-likelihood requests across four checkpoints and five policies. The mean Smooth-RAR-log gain over smooth uniform is 3.64 points and is driven by the two Qwen checkpoints. OLMo-2 retains a smaller positive gain, whereas the Mistral policies remain effectively tied near clean accuracy.

Uncertainty in policy differences. We summarize policy differences across the tasks in each split using the task-level standard errors returned by `lm-evaluation-harness` and report 95% CIs. The calculation uses an unpaired approximation that treats the two policy accuracy estimates as independent, based on the aggregate task statistics retained for the harness evaluations. The unpaired approximation is conservative when policy accuracies on shared examples have nonnegative cross-policy covariance. The retained aggregate standard errors do not identify the cross-policy covariance.

Under the unpaired approximation, Smooth-RAR-log improves on smooth uniform for both Qwen checkpoints across all evaluated sample sizes. At 512 examples (Table 10), the gains are $+4.24$ points on Qwen2.5-7B and $+7.08$ points on Qwen2.5-7B-Instruct, with 95% CIs of $[+1.92, +6.57]$ and $[+4.74, +9.41]$, respectively.

In the 1024-example evaluation (Table 11), the Smooth-RAR-log gains over smooth uniform are $+4.33$ points for Qwen2.5-7B and $+6.04$ points for Qwen2.5-7B-Instruct, with 95% CIs of $[+2.56, +6.10]$ and $[+4.26, +7.83]$. Smooth-Recalib-RAR-task gives comparable gains of $+4.11$ and $+6.00$ points, whose 95% CIs are $[+2.33, +5.89]$ and $[+4.21, +7.79]$.

For the full-core evaluation in Table 2, under the unpaired standard-error approximation, the Smooth-RAR-log gains over smooth uniform are $+2.96$ percentage points on Qwen2.5-7B and $+6.54$ points on Qwen2.5-7B-Instruct, with 95% CIs of $[+0.90, +5.03]$ and $[+4.44, +8.63]$. The corresponding Mistral and OLMo-2 differences are -0.13 and -0.004 points, with 95% CIs of $[-2.18, +1.93]$ and $[-2.08, +2.08]$; both intervals include zero.

Full BoolQ (Table 12) yields the largest Qwen gains over smooth uniform: Smooth-RAR-log improves by $+8.38$ points on Qwen2.5-7B and $+10.46$ points on Qwen2.5-7B-Instruct, with 95% CIs of $[+6.41, +10.34]$ and $[+8.51, +12.41]$. The recalibrated task policy gives gains of $+8.38$ and $+10.58$ points, whose 95% CIs are $[+6.41, +10.34]$ and $[+8.63, +12.53]$.

Full HellaSwag (Table 13) shows the same pattern on a larger evaluation set: Smooth-RAR-log improves over smooth uniform by $+6.93$ points on Qwen2.5-7B and $+7.45$ points on Qwen2.5-7B-Instruct, with 95% CIs of $[+5.56, +8.31]$ and $[+6.08, +8.82]$. Smooth-Recalib-RAR-task gives gains of $+6.85$ and $+7.49$ points, whose 95% CIs are $[+5.48, +8.23]$ and $[+6.12, +8.86]$.

For Mistral and OLMo-2, the full BoolQ and full HellaSwag unpaired CIs include zero. On full BoolQ in Table 12, the Smooth-RAR-log gains for Mistral and OLMo-2 are $+0.52$ and $+1.04$ points, with 95% CIs of $[-1.26, +2.30]$ and $[-0.90, +2.98]$. The corresponding full-HellaSwag gains in Table 13 are -0.17 and $+0.36$ points, with 95% CIs of $[-1.52, +1.18]$ and $[-1.00, +1.71]$. RAR yields the largest standard-harness gains when uniform activation quantization induces a large

Table 14: Pack–unpack–replace forward evaluation. Values average 32-block WikiText-2 and C4 evaluations at sequence length 128; NLL is measured in nats per token after the dequantized packed activation is fed downstream, and payload ratios are relative to FP16.

Model	Policy	NLL	Payload + scales (%)
Qwen2.5-7B	uniform 6-bit	8.938	37.516
	smooth uniform	3.222	37.516
	Smooth-RAR-log	2.742	37.542
	Smooth-Recalib-RAR-task	2.735	37.566
	Smooth-Recalib-RAR-constrained	2.740	37.526
OLMo-2-7B	uniform 6-bit	3.412	37.518
	smooth uniform	2.750	37.518
	Smooth-RAR-log	2.706	37.552
	Smooth-Recalib-RAR-task	2.702	37.532
	Smooth-Recalib-RAR-constrained	2.700	37.528
Mistral-7B	uniform 6-bit	2.675	37.518
	smooth uniform	2.487	37.518
	Smooth-RAR-log	2.415	37.518
	Smooth-Recalib-RAR-task	2.413	37.537
	Smooth-Recalib-RAR-constrained	2.410	37.564

Qwen-style routing failure; smooth uniform is competitive when the uniform drop is small or mostly handled by smoothing.

Task-level results follow the same pattern. Smooth-RAR-log outperforms smooth uniform on all six tasks for both Qwen checkpoints in both the 512-example and 1024-example evaluations. Aggregated over all four 7B checkpoints, the corresponding task-win counts are 16 of 24 and 19 of 24 for the 512-example and 1024-example evaluations. On the full-core evaluation, Smooth-RAR-log outperforms smooth uniform on three of four Qwen2.5 tasks, all four Qwen-Instruct tasks, two of four Mistral tasks, and three of four OLMo-2 tasks. Smooth-Recalib-RAR-task outperforms smooth uniform on all four, all four, three of four, and three of four tasks, respectively. On the full BoolQ split, Smooth-RAR-log outperforms smooth uniform on all four checkpoints. On the full HellaSwag split, Smooth-RAR-log outperforms smooth uniform on three of four checkpoints and Smooth-Recalib-RAR-task outperforms smooth uniform on all four. The task-win counts show that the Qwen gains extend across multiple benchmarks.

F Packed-Activation and Weight-PTQ Evaluations

F.1 Inference with Packed Activations

Table 14 evaluates inference in which activations are packed into uint8 payloads, unpacked, and dequantized before being passed to subsequent layers. The evaluation tests whether policy ordering persists beyond fake-quantization arithmetic. On Qwen2.5-7B, uniform 6-bit produces substantial degradation in the packed path. Smooth uniform reduces the NLL degradation, and the smoothed RAR policies achieve the lowest NLL. On Mistral-7B, uniform degradation is milder, and OLMo-2-7B provides a post-normalized checkpoint where the same Smooth-RAR-log ordering is preserved. Across the six paired model–corpus evaluations, Smooth-RAR-log reduces NLL relative to smooth uniform in all six cases, by 0.199 nats per token on average. The mean reductions for Smooth-Recalib-RAR-task and Smooth-Recalib-RAR-constrained are both 0.203 nats per token.

F.2 Activation Allocation after W8 Weight Quantization

RAR also allocates activation precision after weight PTQ. A compatibility evaluation first quantizes all role-bearing linear weights to groupwise W8 and then applies the activation policies. Table 15 reports the fraction of W8+uniform 6-bit activation degradation removed by each policy, using W8 with unquantized activations as the reference. As in the main activation fake-quantization evaluation,

Table 15: W8 weight-PTQ compatibility evaluation. Plain W8 uses groupwise 8-bit fake-quantized role-bearing linear weights; clipped-and-equalized W8 additionally selects activation-aware clipping ratios and AWQ-lite equalization exponents from calibration RMS. Uniform ΔNLL is the increase in nats per token relative to W8 with unquantized activations. Recovery is the percentage of the NLL increase removed by each policy; higher is better. Values are averaged over WikiText-2 and C4 at sequence length 128 with 64 evaluation blocks, except for the matched Pythia-6.9B evaluation, which uses 32 blocks.

		Activation policy on W8 weights			
			Recovery (%)		
Weight-PTQ	Model	Uniform ΔNLL	SmUni	SmRAR	SmTask
plain	OPT-125M	0.195	88.1	90.3	–
	Qwen2.5-7B	6.146	91.6	99.5	99.5
	Mistral-7B	0.305	71.6	92.0	93.4
	OLMo-2-7B	0.781	89.9	96.7	97.0
clipped+eq	Qwen2.5-7B	6.151	92.0	99.4	99.5
	Pythia-6.9B	4.489	95.3	97.1	97.4
	Mistral-7B	0.288	69.9	92.9	92.4
	OLMo-2-7B	0.779	90.2	96.6	96.9
	OPT-125M	0.188	86.6	89.0	–
plain all	–	–	85.3	94.6	96.6[†]
clipped+eq all	–	–	86.8	95.0	96.5[‡]
clipped+eq 6.9B and 7B	–	–	86.9	96.5	96.5

[†]The Smooth-Recalib-RAR-task policy is available for Qwen2.5-7B, Mistral-7B, and OLMo-2-7B in the plain W8 evaluation; the aggregate is over the matched Qwen, Mistral, and OLMo-2 rows. [‡]The clipped-and-equalized Smooth-Recalib-RAR-task aggregate is over the matched Pythia, Qwen, Mistral, and OLMo-2 rows because OPT-125M has no Smooth-Recalib-RAR policy.

smoothing removes most degradation, and role-aware allocation further reduces the remaining NLL gap.

Across eight model–corpus evaluations, W8+smooth uniform removes 85.3% of the W8+uniform activation degradation, while W8+Smooth-RAR-log removes 94.6%. The largest residual benefit remains on Qwen, but Smooth-RAR-log also improves over smooth uniform on OPT, Mistral, and OLMo-2. Relative to the strong W8+smooth uniform baseline, W8+Smooth-RAR-log increases degradation reduction by 9.3 percentage points overall; Smooth-Recalib-RAR-task adds 12.3 percentage points on the matched Qwen, Mistral, and OLMo-2 subset.

The activation-aware clipped-and-equalized W8 variant uses complete input-scale byte accounting and includes a matched Pythia-6.9B evaluation. Across OPT-125M, Pythia-6.9B, Qwen2.5-7B, Mistral-7B, and OLMo-2-7B, smooth uniform removes 86.8% of W8+uniform activation degradation, while W8+Smooth-RAR-log removes 95.0%. On the matched 6.9B and 7B subset, both W8+Smooth-RAR-log and Smooth-Recalib-RAR-task remove 96.5%, compared with 86.9% for smooth uniform. Storage-controlled comparisons use rows with complete auxiliary input-scale accounting.

As a complementary PTQ control, we convert every role-bearing linear module to TorchAO's int8 weight-only format (Table 16). The TorchAO conversion produces `AffineQuantizedTensor` weights for 196 modules on Qwen2.5-7B and 224 modules on Mistral-7B and OLMo-2-7B. The resulting weight payload plus scales is approximately 50.0% of FP16. Smooth-RAR-log outperforms TorchAO+smooth uniform on all six model–corpus pairs, with a mean NLL reduction of 0.179 nats per token and an average uniform-degradation recovery of 95.8%, compared with 83.4% for smooth uniform. The Smooth-Recalib-RAR-task policy is similar at 96.3%.

We also combine the clipped-and-equalized AWQ-lite W8 weights with packed activations to test whether the allocation benefit persists when downstream layers receive activations reconstructed from integer payloads. Table 17 reports results for Qwen2.5-7B, Mistral-7B, and OLMo-2-7B on WikiText-2 and C4, using sequence length 128 and 32 evaluation blocks per corpus.

Table 16: TorchAO int8 weight-only module-path evaluation. Role-bearing linear modules use TorchAO's int8 weight-only configuration. The LM protocol follows Table 14, and recovery follows the W8 reference convention in Table 15.

		Recovery (%)		
Model	Uniform ΔNLL	SmUni	SmRAR	SmTask
Qwen2.5-7B	6.242	93.1	99.9	99.8
Mistral-7B	0.280	67.8	91.5	92.5
OLMo-2-7B	0.714	89.5	95.9	96.6
All six pairs	2.412	83.4	95.8	96.3

Table 17: Packed AWQ-lite W8 activation evaluation. We apply the pack–unpack–replace path of Table 14 to the clipped-and-equalized W8 setting of Table 15; recovery follows the same W8 reference. The final column gives the NLL decrease from smooth uniform to Smooth-RAR-log, in nats per token.

	Activation policy on AWQ-lite W8 weights				
		Recovery (%)			
Model	Uniform ΔNLL	SmUni	SmRAR	SmTask	NLL decrease
Qwen2.5-7B	6.134	92.5	99.9	99.8	0.455
Mistral-7B	0.287	69.1	93.3	92.1	0.070
OLMo-2-7B	0.741	91.2	97.6	96.6	0.046
All six pairs	2.387	84.3	96.9	96.1	0.190

In the packed AWQ-lite W8 evaluation, W8+Smooth-RAR-log improves over W8+smooth uniform on all six model–corpus pairs, removes 96.9% of W8+uniform activation degradation, and increases degradation reduction by 12.7 percentage points over W8+smooth uniform. The measured packed-activation payload plus dynamic scales is 37.54% of the corresponding FP16 activation payload for Smooth-RAR-log, matching the 6-bit cost budget. Qwen supplies the largest absolute gain, while Mistral and OLMo-2 show smaller but positive gains. The allocation benefit therefore persists when both weights and activations use the evaluated low-precision representations.

To assess downstream task performance under weight PTQ, we also evaluate the same clipped-and-equalized AWQ-lite W8 path on the full BoolQ validation set for Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B. The W8 BoolQ evaluation uses all 3,270 BoolQ validation examples and sequence length 1024. Table 18 reports the resulting task-level comparison. W8+clean activation has 83.7% mean accuracy, whereas uniform 6-bit activation drops the mean to 58.2%. Smooth uniform reaches 77.5%, and all three role-aware policies exceed 82%. Smooth-RAR-log gains 4.89 percentage points over W8+smooth uniform on average, with the effect concentrated on the two Qwen checkpoints; Mistral and OLMo-2 retain the small residual differences observed in the non-W8 evaluations.

Under the same unpaired approximation as the non-W8 evaluations, the Qwen W8 BoolQ gains over W8+smooth uniform have CIs that exclude zero: Smooth-RAR-log gives $+9.02$ points on Qwen2.5-7B and $+9.91$ points on Qwen2.5-7B-Instruct, with 95% CIs of $[+7.03, +11.01]$ and $[+7.94, +11.88]$. Smooth-Recalib-RAR-task gives $+8.87$ and $+10.18$ points, whose 95% CIs are $[+6.88, +10.86]$ and $[+8.22, +12.15]$. The constrained policy gives $+9.02$ and $+10.24$ points, with 95% CIs of $[+7.03, +11.01]$ and $[+8.28, +12.21]$. The OLMo-2 and Mistral W8 BoolQ CIs include zero under the unpaired calculation: Smooth-RAR-log gives $+1.01$ points on OLMo-2-7B and -0.37 points on Mistral-7B, with 95% CIs of $[-0.97, +2.98]$ and $[-2.15, +1.42]$. As in the non-W8 harness evaluations, the largest accuracy gains occur on the Qwen checkpoints, consistent with Qwen's routing sensitivity. Mistral and OLMo-2 show smaller differences relative to smooth uniform.

Table 18: Full BoolQ AWQ-lite W8 extension of Table 12. The table uses the W8 setup of Table 15 and additionally reports Smooth-Recalib-RAR-constrained. Entries are zero-shot accuracies (%); the mean row averages the four checkpoints.

	Activation policy on AWQ-lite W8 weights					
Model	Clean	Uniform	SmUni	SmRAR	SmTask	SmConstr
Qwen2.5-7B	84.7	39.7	73.7	82.7	82.5	82.7
Qwen2.5-7B-Instruct	86.1	38.9	73.8	83.7	84.0	84.1
Mistral-7B	84.4	80.5	84.0	83.6	83.8	83.6
OLMo-2-7B	79.7	73.6	78.5	79.5	79.6	78.8
Mean	83.7	58.2	77.5	82.4	82.5	82.3

Table 19: Comparison across evaluation settings. Each unit is a complete model–dataset–split comparison group. SmUni wins counts improvements over uniform 6-bit; SmRAR wins counts improvements over smooth uniform, with no ties for SmRAR. Mean gains are NLL reductions in nats per token for LM rows and accuracy differences in percentage points for the harness row.

Setting	Units	SmUni wins	SmRAR wins	Mean SmRAR gain
LM NLL	15	15 of 15	15 of 15	0.282
1024-token LM NLL	6	6 of 6	6 of 6	0.313
Harness accuracy	72	69 of 72	54 of 72	+2.79
W8 LM NLL	20	20 of 20	20 of 20	0.183
Packed AWQ-lite W8 LM NLL	6	6 of 6	6 of 6	0.190

F.3 Results across Evaluation Settings

Aggregate comparison. Table 19 compares RAR with smooth uniform across evaluation settings. A comparison unit is a model–dataset–split group that contains all five of clean, uniform 6-bit, smooth uniform, Smooth-RAR-log, and Smooth-Recalib-RAR-task results; requiring the full set is why the LM row counts 15 units rather than the 24 model–corpus evaluations of Table 1. The W8 and AWQ-lite setups are specified in Appendix D. Smooth uniform improves on uniform quantization in every unit where uniform quantization causes a measurable degradation, and Smooth-RAR-log further improves NLL or accuracy in the LM, long-context, W8 LM, packed-W8 LM, and most standard-harness comparisons. The preceding packed-activation, fake-W8, TorchAO, and packed AWQ-lite W8 evaluations retain the LM allocation benefit with payload accounting based on the realized assignments. The full BoolQ W8 evaluation provides the corresponding task-level comparison.

G Additional Functional-Risk Atlas Analyses

G.1 Numerical Proxies Partially Predict Functional Risk

Table 20 compares site-level proxies with the measured 4-bit functional risk. Activation RMS is a weak predictor: The rank correlation with functional risk never exceeds 0.34 and is negative on OLMo-2-7B. The maximum-to-RMS outlier score is stronger on every checkpoint but still only moderately predictive, peaking at 0.653 on Mistral-7B. Gaussian perturbation risk, an output-facing measurement, is the most predictive proxy where available, but the cost of measuring Gaussian risk restricts the comparison to GPT-2 and OPT-125M. The site-level proxy correlations show that activation magnitude alone does not capture downstream functional sensitivity.

G.2 Numerical Error, Functional Leverage, and Coordinate Erasure

This section examines how numerical quantization error and the resulting changes in model outputs contribute to the results in Sec. 4.2.

Table 20: Spearman rank correlation between site-level proxies and the measured 4-bit risk $A_c^{(4)}$. Gaussian perturbation risk uses Gaussian noise in Eq. 7; a dash marks an unmeasured proxy.

Model	Activation RMS	Maximum-to-RMS	Gaussian risk
GPT-2	0.023	0.136	0.574
OPT-125M	0.223	0.310	0.508
TinyLlama-1.1B	0.338	0.369	–
Pythia-6.9B	0.071	0.417	–
Qwen2.5-7B	0.274	0.508	–
Qwen2.5-7B-Instruct	0.237	0.504	–
Mistral-7B	0.144	0.653	–
OLMo-2-7B	−0.079	0.413	–

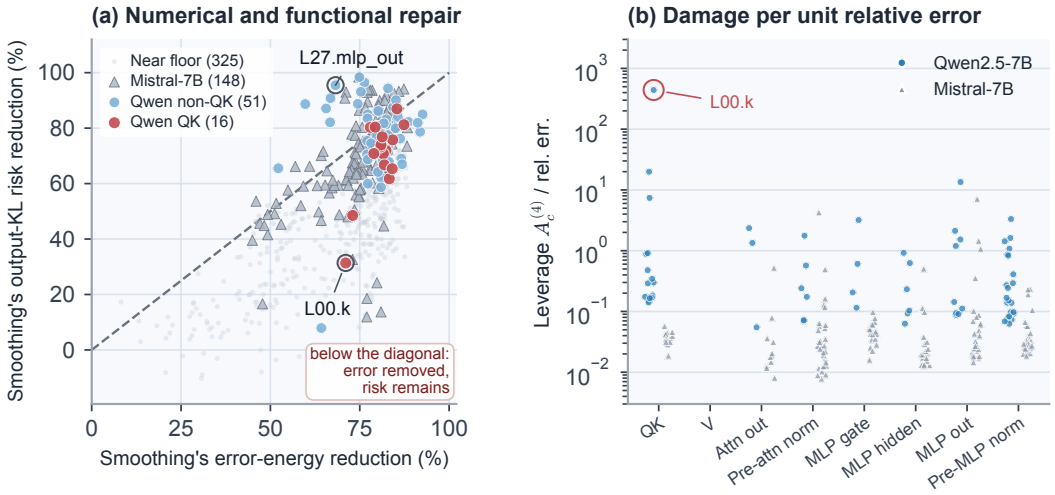

Figure 6: Quantization-error reduction and output sensitivity. In (a), numerical-error and output-KL risk reduction after smoothing are compared; early Qwen QK sites retain functional risk. In (b), empirical base-atlas leverage, the 4-bit site risk divided by the same site's relative error energy, is shown by role. The comparison in (a) includes near-floor sites for completeness; the accompanying text defines the threshold and explains the leverage ratio and coordinate-erasure analysis.

The LM comparisons show why smoothing and allocation work well together: Smoothing reduces numerical quantization error, and the assigned bit-widths directly reflect how strongly quantization error changes model predictions. RAR-log prioritizes sites with higher output KL risk per unit cost for extra bits.

We additionally measured each site's 4-bit quantization error before and after smoothing on Qwen and Mistral, and compared the measured quantization error with the corresponding output KL from the risk atlases (Figure 6). In Figure 6(a), early Qwen QK sites retain functional risk after numerical error at the early QK sites falls, whereas Mistral and most Qwen sites show comparable or larger KL reductions; Figure 6(b) shows that output KL per unit relative error energy differs sharply across roles and checkpoints.

Figure 6(a) compares the reduction in numerical quantization error with the reduction in output KL. Relative error energy is the sum of squared quantization errors divided by the sum of squared clean activation values over the measured calibration blocks. We call a site resolved when the site's base 4-bit risk is at least four times the fifth percentile of the checkpoint's base-atlas risks. The resolved sites account for 90.0% of summed Qwen risk and 93.1% of summed Mistral risk; the remaining near-floor sites are plotted but not interpreted. Smoothing removes much of the 4-bit error energy for nearly every resolved site. On Mistral and for most Qwen sites, output KL risk falls by a comparable or larger amount. In contrast, Qwen's early QK sites retain substantial functional

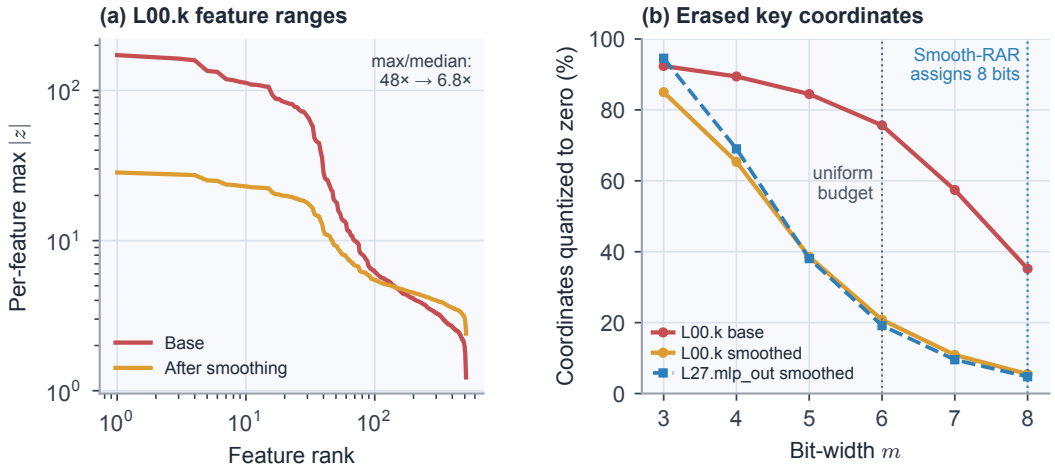

Figure 7: Feature ranges and coordinate erasure at Qwen's `L00.k`. In (a), feature ranges before and after smoothing are compared; (b) shows the fraction of all activation coordinates rounded from nonzero values to zero across bit-widths, with smoothed `L27.mlp_out` for comparison. Smoothing reduces feature-range imbalance, yet `L00.k` retains substantial coordinate erasure at 6 bits. Together with the high leverage of `L00.k` in Figure 6(b), the persistent coordinate erasure is consistent with the 8-bit precision assigned by Smooth-RAR. In this figure, Smooth-RAR denotes Smooth-RAR-log.

risk after smoothing reduces numerical error at the early QK sites, which motivates assigning the early QK sites higher precision after smoothing.

The leverage ratio in Figure 6(b) is an empirical finite-noise diagnostic, not a direct estimate of the local quadratic form $J_c^\top F_{\text{out}} J_c$ in Eq. 8: The leverage ratio also reflects the realized quantization-error direction and downstream nonlinearities. The leverage ratio connects numerical error to the allocation objective: The output KL atlas records how the remaining error affects predictions after propagating through the model. The corresponding site-level priority changes after smoothing are reported in Figure 10 below. Figure 7 examines the layer-0 key-projection site `L00.k`. Smoothing reduces the maximum-to-median feature-range ratio at `L00.k` from $48\times$ to $6.8\times$, yet the site's empirical leverage is 442 nats per unit relative error energy. We define coordinate erasure as rounding a nonzero activation value to zero and report the erased count as a fraction of all activation coordinates. With smoothing, the fraction of erased coordinates is 20.8% at 6 bits, compared with 5.5% at the 8 bits assigned by Smooth-RAR-log.

G.3 Attention Routing and Precision Allocation

To examine how high-risk Qwen sites affect the model, we compare the attention distributions produced by each activation policy (Figure 8) using Jensen–Shannon divergence (JSD) and clean top-attended-key agreement. In the first Transformer block, uniform precision moves probability mass to different keys. Smoothing largely restores concentration, while protecting the layer-0 pre-attention and QK path restores the attended targets. The recovery of attention routing persists when averaged over layers, heads, and validation blocks, connecting the output-facing atlas to changes in attention routing. Across the aggregate in Figure 8, mean routing JSD falls from 0.418 to 0.091 to 0.026, while clean top-1 attended-key agreement rises from 44.2% to 79.0% to 89.1% for uniform, smooth uniform, and Smooth-RAR-log, respectively.

The routing trace establishes what changes internally, but a policy-level comparison alone does not show whether the QK precision can be replaced by precision elsewhere. Figure 3 therefore uses a same-budget role-neutralization intervention. Starting from the unsmoothed Qwen RAR-log policy, the intervention fixes every site of one role at 6 bits and re-optimizes all other roles with the same atlas and total target budget. Figure 9 details the allocation underlying the role-neutralization intervention.

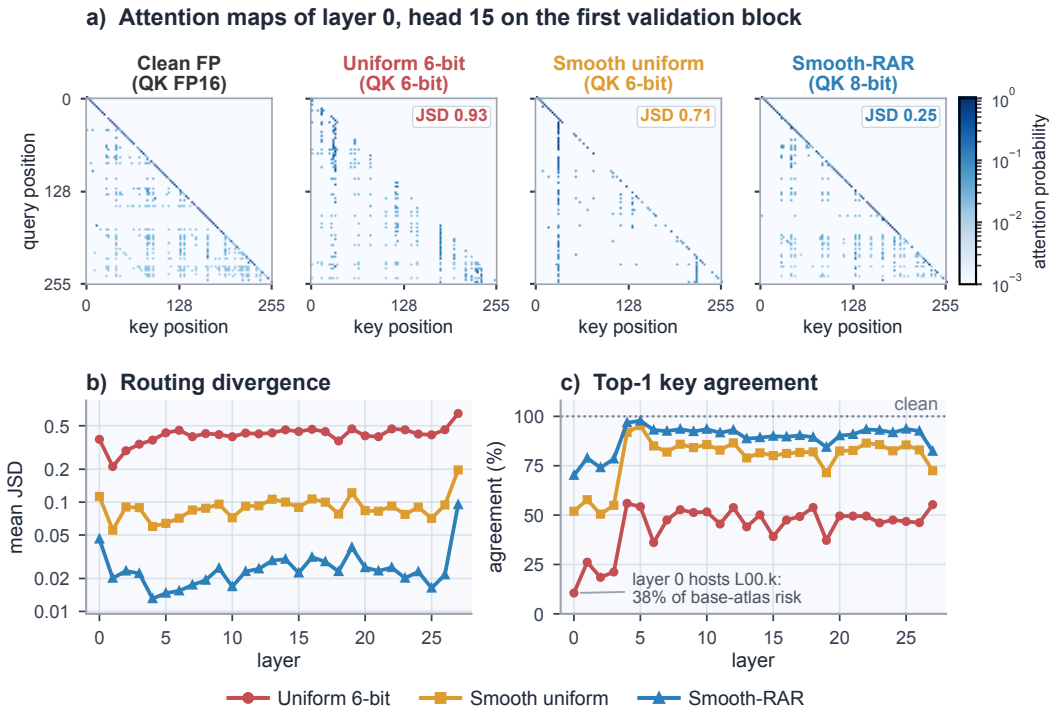

Figure 8: Uniform activation precision reroutes attention, whereas Smooth-RAR largely restores the clean attention routing. The deterministically selected layer-0 Qwen2.5-7B head with the largest uniform-6-bit routing divergence appears in (a), while (b) and (c) aggregate the first six 256-token validation blocks. Moving from uniform 6-bit to smooth uniform and then Smooth-RAR reduces routing divergence and increases agreement with the clean top-attended keys. In this figure, Smooth-RAR denotes Smooth-RAR-log.

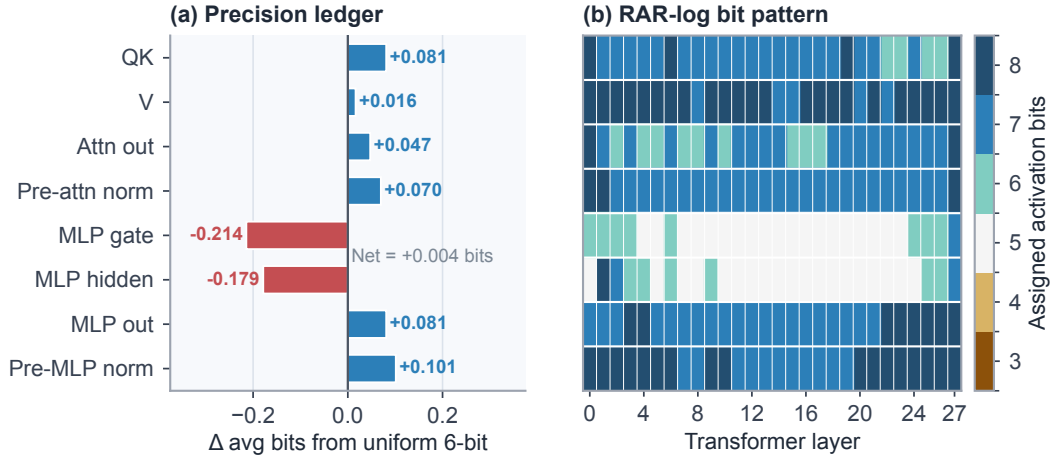

Figure 9: Precision allocation under Qwen RAR-log. Under the same 6-bit target, RAR-log transfers precision from wide MLP activations to narrower high-risk paths. Each role's contribution to the model-wide average bits relative to uniform 6-bit is given in (a), and the assigned bits per role and layer in (b). The corresponding role-neutralization results are shown in Figure 3.

The four individual blocks show the same large QK-neutralization gap. Within the tested atlas-driven policy family, the role-neutralization intervention shows that reallocating the released QK budget to non-QK sites according to the atlas does not substitute for protecting the routing computation.

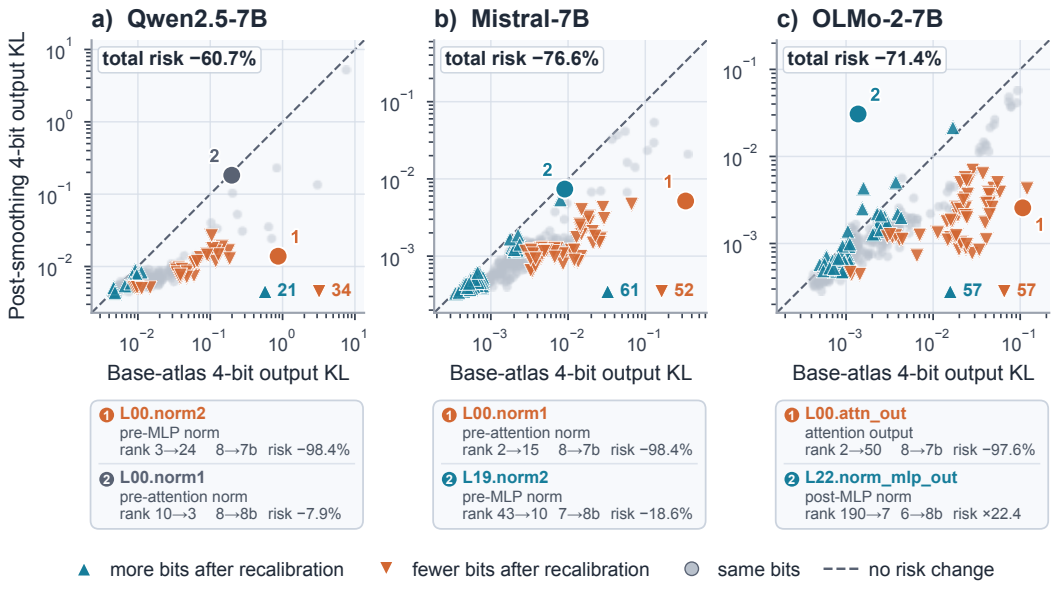

Figure 10: Smoothing reduces total risk and changes allocation priorities. Each point compares a site's base and post-smoothing 4-bit output KL; triangles mark sites whose assigned precision changes after recalibration. Summed single-site risk decreases for all three checkpoints, but unequal risk reductions change more than fifty bit assignments per checkpoint.

Together with Figure 8, the role-neutralization intervention links the measured QK risk to both the changed attention distributions and the specific budget decision that repairs attention routing. The realized budgets range from 6.000 to 6.012 bits. RAR-log has mean output KL 0.367 nats, compared with 7.545 for uniform; QK neutralization raises mean output KL to 6.472 and lowers clean-top-1 agreement from 78.6% to 4.9%.

G.4 Additional Architecture and Smoothing Views

Post-smoothing site reranking. Post-smoothing recalibration changes site priorities despite the modest aggregate improvement in LM performance. Figure 10 compares the base and post-smoothing risks of every matched site for Qwen, Mistral, and OLMo-2. Smoothing reduces the sum of single-site risks, but unequal reductions shift site rankings and change more than fifty bit assignments per checkpoint. For example, OLMo-2's `L22.norm_mlp_out` moves from base-atlas rank 190 to residual-atlas rank 7 and from 6 to 8 bits, whereas `L00.attn_out` moves from rank 2 to 50 and from 8 to 7 bits. Many affected sites have small residual risks, so individual rank changes remain sensitive to calibration sampling.

Figure 11 adds the matched atlases for Mistral-7B and OLMo-2-7B, where risk is concentrated in MLP and attention outputs rather than in QK routing. Figure 12 summarizes the corresponding role-risk profiles across all eight checkpoints and, when available, after smoothing. Each bar partitions a checkpoint's summed single-site risk by role. The role-risk shares distinguish Qwen's routing-dominated profile, Mistral's normalization and MLP profile, and OLMo-2's output-dominated profile. Smoothing redistributes the role-risk shares while leaving a nonuniform residual risk.

Table 21 quantifies the differences in role-risk profiles across checkpoints through the four largest role-level 4-bit risk sums for each checkpoint. GPT-2 concentrates risk in pre-attention LayerNorm, MLP hidden activations, and pre-MLP LayerNorm. OPT-125M is dominated by pre-MLP LayerNorm. TinyLlama shifts risk toward pre-attention RMSNorm and MLP output, with MLP gate and QK routing also substantial. Pythia-6.9B concentrates risk in MLP output and pre-MLP normalization, consistent with the competitive performance of Smooth-RAR-role, which allocates precision using role-averaged risk scores.

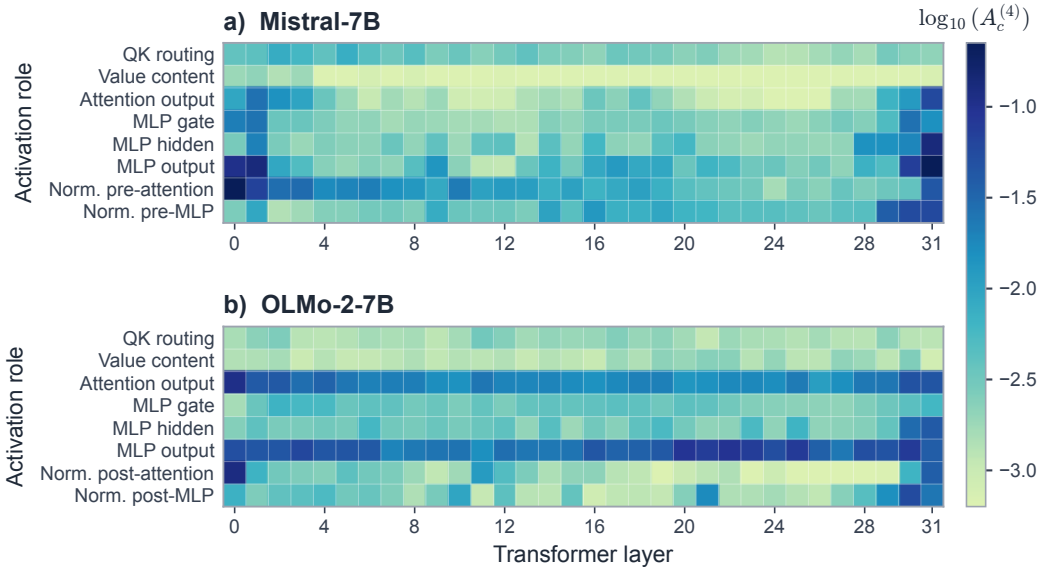

Figure 11: Layer–role 4-bit functional risk atlases for Mistral-7B and OLMo-2-7B. The color scale is shared with Figure 2.

On Qwen2.5-7B, QK routing accounts for the largest share of risk. Qwen2.5-7B-Instruct preserves the same QK-dominant profile, with slightly more risk in pre-MLP normalization. Mistral-7B concentrates risk in MLP output and pre-attention normalization rather than QK routing, and the failure of unsmoothed RAR-log on Mistral suggests that Mistral's high-risk roles also have strong outlier exposure. OLMo-2-7B concentrates risk in MLP output and attention output, while OLMo-2's post-attention and post-MLP normalization outputs account for smaller shares.

The smoothed atlases show how the architecture-specific role-risk profiles change after rescaling. Figure 13 shows that smooth scaling reduces total 4-bit single-site risk by 46.2% on Pythia-6.9B, 60.7% on Qwen2.5-7B, 59.7% on Qwen2.5-7B-Instruct, 76.6% on Mistral-7B, and 71.4% on OLMo-2-7B. The role shift is not uniform. On Pythia, smoothing removes 75.0% of MLP-output risk and 69.5% of pre-MLP-normalization risk, but value-content risk slightly increases. On Qwen, smoothing strongly reduces MLP-output, pre-MLP-normalization, and attention-output risk, but QK routing remains the largest residual mass.

On Mistral, the two roles that dominate the unsmoothed atlas also drop the most. Pre-attention normalization falls by 87.4% and MLP output by 81.8%. On OLMo-2, attention output falls by 91.8%, post-attention normalization by 83.6%, and MLP output by 67.3%, while QK routing and value-content risk are much smaller and drop less. The remaining smoothed risk is still concentrated in particular roles, consistent with the allocation gains in Table 1. Smoothing reduces outlier-related error, while role-aware allocation directs precision toward the sites with the highest residual risk.

G.5 Sensitivity to Calibration Settings

The GPT-2 calibration sensitivity experiment compares WikiText-2 train with a FineWeb-Edu sample and uses 16, 32, or 64 calibration blocks. Rank correlations are high. The minimum Spearman correlation is 0.992 for 4-bit KL risk, 0.986 for flip rate, and 0.993 for margin loss.

A complementary 7B-scale sensitivity analysis for Qwen2.5-7B, Qwen2.5-7B-Instruct, Mistral-7B, and OLMo-2-7B varies the same two corpora and compares 8 with 16 calibration blocks using one risk batch. With the smaller calibration budget, 4-bit KL-risk rankings remain stable: Qwen has mean and minimum Spearman correlations of 0.886 and 0.846 across the six setting pairs. The corresponding values are 0.888 and 0.834 for Mistral, 0.917 and 0.894 for Qwen2.5-7B-Instruct, and 0.951 and 0.927 for OLMo-2-7B. Margin-loss rankings are similarly stable, with mean and

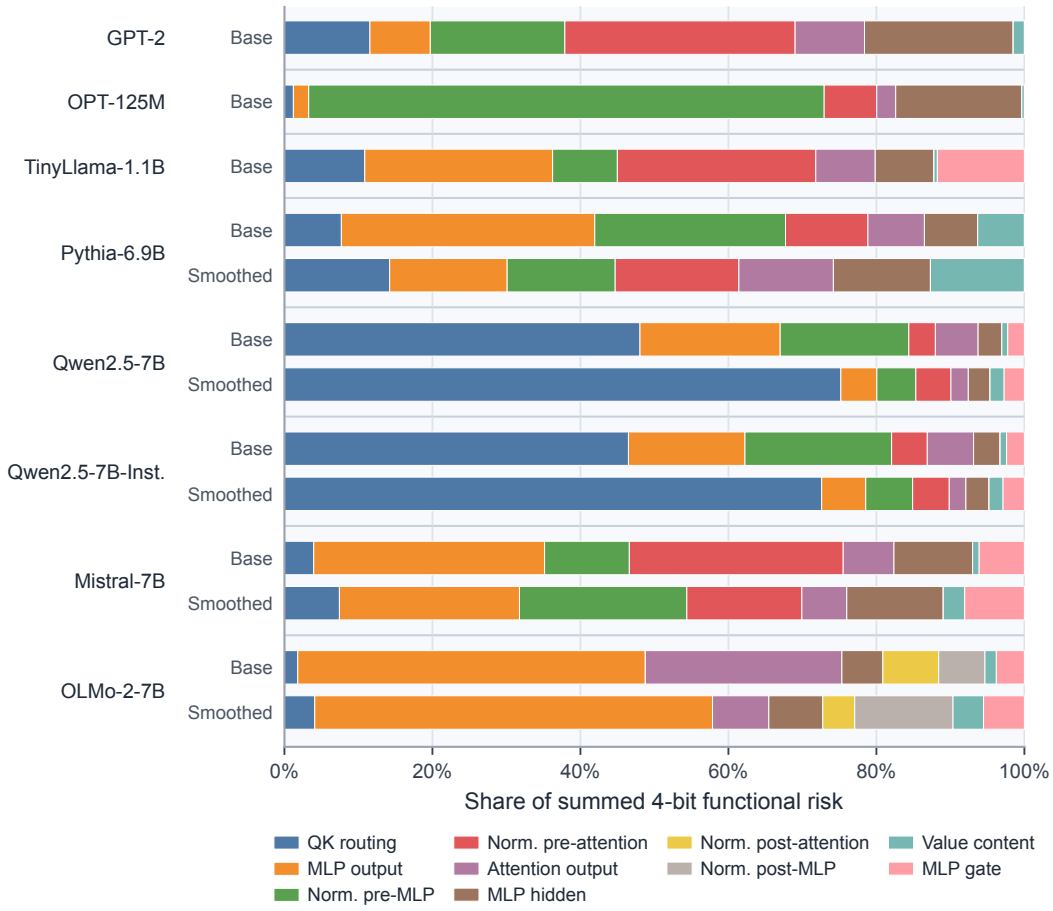

Figure 12: Architecture-level role-risk profiles. Each stacked bar shows the distribution of summed 4-bit functional risk across roles; smoothed bars use the matched post-smoothing residual atlas.

minimum correlations of 0.865 and 0.817 for Qwen, 0.872 and 0.811 for Mistral, 0.874 and 0.828 for Qwen2.5-7B-Instruct, and 0.940 and 0.912 for OLMo-2-7B.

Flip-rate rankings are less stable, especially for Mistral, OLMo-2, and the instruction-tuned Qwen checkpoint. The mean and minimum correlations are 0.637 and 0.505 for Mistral, 0.666 and 0.578 for OLMo-2, and 0.700 and 0.610 for Qwen-Instruct. The lower flip-rate stability is consistent with our downstream observation that argmax-level metrics are noisier than KL and NLL metrics.

The corpus-and-block sensitivity analysis is repeated after applying the SmoothQuant-style activation smoothing scales for Qwen2.5-7B-Instruct and OLMo-2-7B. The residual smoothed risk rankings are less stable, particularly on Qwen2.5-7B-Instruct. The mean and minimum Spearman correlations for 4-bit KL risk are 0.660 and 0.592 for the smoothed instruction-tuned Qwen checkpoint and 0.897 and 0.846 for smoothed OLMo-2. Margin-loss rankings remain more stable than flip-rate rankings after smoothing, with mean and minimum correlations of 0.588 and 0.489 for Qwen2.5-7B-Instruct and 0.850 and 0.777 for OLMo-2. In contrast, the mean and minimum flip-rate rank correlations drop to 0.250 and 0.157 for Qwen2.5-7B-Instruct and to 0.344 and 0.200 for OLMo-2. The comparisons across calibration corpora and sample sizes show high stability of base-atlas KL rankings, with lower stability for flip rates and some post-smoothing rankings. Flip rates and some post-smoothing rankings are therefore more sensitive to the choice and size of the calibration sample.

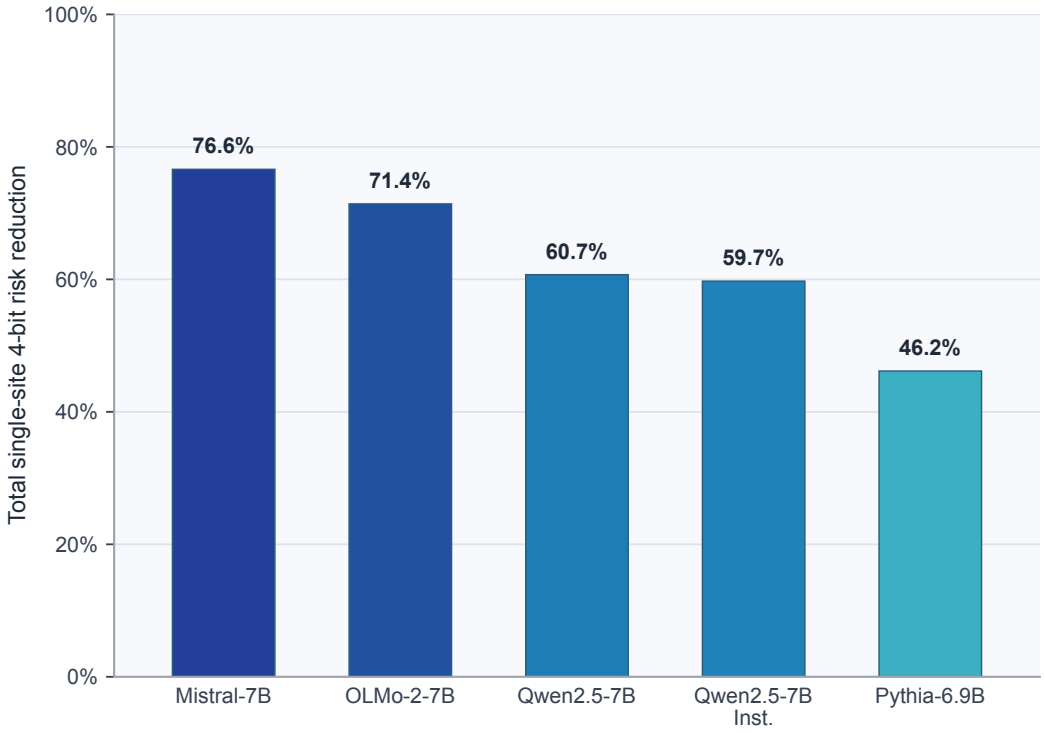

Figure 13: Reduction in summed single-site 4-bit risk after SmoothQuant-style activation smoothing. Bars show the percentage reduction for each checkpoint with a matched post-smoothing atlas.

Table 21: The four largest role-level 4-bit risk sums for each checkpoint. Risk sum is the sum of single-site output KL values, in nats, over the sites of a role. Share is the percentage of the checkpoint's total summed site risk.

Model	Role	Risk sum	Share (%)
GPT-2	pre-attention norm.	0.603	31.1
	MLP hidden	0.390	20.1
	pre-MLP norm.	0.353	18.2
	QK routing	0.224	11.6
OPT-125M	pre-MLP norm.	2.499	69.7
	MLP hidden	0.611	17.0
	pre-attention norm.	0.255	7.1
	attention output	0.092	2.6
TinyLlama-1.1B	pre-attention norm.	0.586	26.8
	MLP output	0.555	25.4
	MLP gate	0.257	11.8
	QK routing	0.237	10.9
Pythia-6.9B	MLP output	10.056	34.2
	pre-MLP norm.	7.569	25.8
	pre-attention norm.	3.270	11.1
	QK routing	2.259	7.7
Qwen2.5-7B	QK routing	9.456	48.0
	MLP output	3.729	18.9
	pre-MLP norm.	3.422	17.4
	attention output	1.137	5.8
Qwen2.5-7B-Instruct	QK routing	10.639	46.5
	pre-MLP norm.	4.536	19.8
	MLP output	3.604	15.7
	attention output	1.432	6.3
Mistral-7B	MLP output	0.790	31.2
	pre-attention norm.	0.732	28.9
	pre-MLP norm.	0.290	11.5
	MLP hidden	0.269	10.6
OLMo-2-7B	MLP output	1.420	47.0
	attention output	0.803	26.6
	post-attention norm.	0.228	7.6
	post-MLP norm.	0.189	6.2

H Proofs of Analytical Statements

H.1 Proof of the Local Output KL Expansion

Proof of Proposition C.1. Fix an evaluated position t, abbreviate the clean logit vector at position t by a_t and the clean distribution at position t by $p_t := \mathrm{softmax}(a_t)$, and let h_t be a logit increment. Define $\phi_t(h_t) := D_{\mathrm{KL}}(p_t \| \mathrm{softmax}(a_t + h_t))$. Expanding the definition of categorical KL and using $p_{t,j} = \exp(a_{t,j}) / \sum_k \exp(a_{t,k})$ gives the exact identity

$$\phi_t(h_t) = \log \sum_j e^{a_{t,j}+h_{t,j}} - \log \sum_j e^{a_{t,j}} - p_t^\top h_t.$$

Consequently,

$$\nabla \phi_t(0) = \mathrm{softmax}(a_t) - p_t = 0, \qquad \nabla^2 \phi_t(0) = F_t := \mathrm{diag}(p_t) - p_t p_t^\top.$$

Taylor's theorem therefore yields

$$\phi_t(h_t) = \frac{1}{2} h_t^\top F_t h_t + O(\|h_t\|^3) \quad \text{as } h_t \to 0. \tag{35}$$

Let $h(\delta)$ be the stacked change in all $T-1$ evaluated logit vectors when z_c is replaced by $z_c + \delta$. The differentiability assumption and Taylor's theorem for the downstream map imply

$$h(\delta) = J_c \delta + q(\delta), \qquad \|q(\delta)\| = O(\|\delta\|^2).$$

Averaging Eq. 35 over positions is equivalent to using the block-diagonal matrix F_{out}. Substitution of the preceding equation then gives

$$\frac{1}{T-1} \sum_{t=1}^{T-1} \phi_t(h_t(\delta)) = \frac{1}{2} \delta^\top J_c^\top F_{\mathrm{out}} J_c \delta + O(\|\delta\|^3).$$

Indeed, the cross term involving $J_c\delta$ and $q(\delta)$ is $O(\|\delta\|^3)$, the quadratic term in $q(\delta)$ is $O(\|\delta\|^4)$, and the Taylor remainder is $O(\|\delta\|^3)$.

Finally set $\delta := \rho \xi_c$ and take expectations. Almost-sure boundedness of ξ_c makes the remainder uniform and integrable for sufficiently small ρ, so the remainder's expectation is $O(\rho^3)$. For every fixed matrix H,

$$\mathbb{E}[\xi_c^\top H \xi_c] = \mathrm{Tr}(H \mathbb{E}[\xi_c \xi_c^\top]) = \mathrm{Tr}(H \Sigma_{c,0}(x)),$$

where the last equality uses $\mathbb{E}[\xi_c] = 0$. Taking $H := J_c^\top F_{\mathrm{out}} J_c$ proves Eq. 9. If the perturbation is not centered, the same calculation stops at $\mathbb{E}[\xi_c \xi_c^\top]$, proving the stated second-moment variant as well. □

H.2 Proofs for the Joint Local Risk Decomposition

Proof of Theorem C.2. Stack the intervention variables and write

$$J := [J_{c_1} \quad \cdots \quad J_{c_n}], \qquad J\xi = \sum_{c \in \mathcal{C}} J_c \xi_c.$$

Let $h(u) := g_x(u) - g_x(0)$ be the stacked logit increment. Taylor's theorem for g_x at zero gives

$$h(\rho\xi) = \rho \sum_{c \in \mathcal{C}} J_c \xi_c + r_\rho(\xi), \qquad \|r_\rho(\xi)\| = O(\rho^2), \tag{36}$$

uniformly over the support of ξ. The uniformity follows from the assumed almost-sure boundedness and the continuity of the second derivatives on a compact neighborhood of zero.

Applying the positionwise softmax expansion in Eq. 35 and using the definition of F_{out} yields

$$\begin{aligned} \mathcal{R}_{\mathcal{C},\rho}(x) &= \frac{1}{2} \mathbb{E}[h(\rho\xi)^\top F_{\mathrm{out}} h(\rho\xi)] + O(\rho^3) \\ &= \frac{\rho^2}{2} \mathbb{E}\left[\left(\sum_c J_c \xi_c \right)^\top F_{\mathrm{out}} \left(\sum_d J_d \xi_d \right) \right] + O(\rho^3). \end{aligned} \tag{37}$$

The first remainder is $O(\rho^3)$ because $\|h(\rho\xi)\| = O(\rho)$ uniformly. In the second line, the cross term between the linear part of Eq. 36 and r_ρ is $O(\rho^3)$ and the quadratic term in r_ρ is $O(\rho^4)$. All remainder bounds remain valid after expectation.

For every ordered pair (c, d), the corresponding term in the expectation is

$$\mathbb{E}[\xi_c^\top J_c^\top F_{\text{out}} J_d \xi_d] = \text{Tr}(J_c^\top F_{\text{out}} J_d \mathbb{E}[\xi_d \xi_c^\top]) = b_{cd}.$$

Substitution into Eq. 37 proves Eq. 10. On the diagonal,

$$b_{cc} = q_c = \mathbb{E}[(J_c\xi_c)^\top F_{\text{out}}(J_c\xi_c)] \geq 0$$

because F_{out} is positive semidefinite. For an off-diagonal pair, symmetry of F_{out} yields

$$b_{cd} = \mathbb{E}[\xi_c^\top J_c^\top F_{\text{out}} J_d \xi_d] = \mathbb{E}[\xi_d^\top J_d^\top F_{\text{out}} J_c \xi_c] = b_{dc}.$$

□

Proof of Corollary C.3. Applying the same local expansion, including the noncentered second-moment variant stated after Proposition C.1, to each marginal perturbation gives

$$\sum_{c\in\mathcal{C}} \mathcal{R}_{c,\rho}(x) = \frac{\rho^2}{2} \sum_{c\in\mathcal{C}} q_c + O(\rho^3),$$

where the finite number of sitewise remainders can be absorbed into one $O(\rho^3)$ term. Separating the diagonal and off-diagonal terms in Eq. 10 now proves Eq. 11. □

Proof of Corollary C.4. If every off-diagonal b_{cd} is zero, Eq. 12 follows directly from Eq. 11. The condition $\Omega_{cd,0} = 0$ makes every such trace zero. For centered perturbations, $\Omega_{cd,0}$ is the corresponding cross-covariance block, so vanishing of the cross-covariance block is precisely block uncorrelatedness. Finally, independence together with $\mathbb{E}[\xi_c] = 0$ implies $\mathbb{E}[\xi_c \xi_d^\top] = \mathbb{E}[\xi_c]\mathbb{E}[\xi_d]^\top = 0$ for $c \neq d$, proving the last assertion. □

H.3 Proofs for the Finite-Bit Output KL Rate and Top-1 Barrier

Proof of Theorem C.5. First consider the quantization error. For a nonzero-range group g, nearest rounding on the normalized grid of spacing u_g/q_m gives, for every $j \in g$,

$$\left| \frac{Q_{m,\sigma}(z_c)_j}{\sigma_j} - \frac{z_{c,j}}{\sigma_j} \right| \leq \frac{u_g}{2q_m}.$$

No clipping enlarges the quantization error because the definition of u_g places every normalized coordinate in the represented interval $[-u_g, u_g]$. Multiplying by σ_j, squaring, and summing over the groups gives

$$\|Q_{m,\sigma}(z_c) - z_c\|_2^2 \leq \frac{1}{4q_m^2} \sum_{g\in\mathcal{G}_c} u_g^2 \sum_{j\in g} \sigma_j^2 = \frac{\mathcal{E}_\sigma(z_c)}{4q_m^2}. \tag{38}$$

When $u_g = 0$, the group is represented exactly and contributes zero to both sides. If a positive numerical floor is used for an identically zero group, the zero-range group's left-hand contribution is still zero and the stated upper bound remains valid.

We next establish a global inequality for the output KL. For arbitrary logits a and increment h, let $p \coloneqq \text{softmax}(a)$ and $\phi(h) \coloneqq D_{\text{KL}}(p\|\,\text{softmax}(a+h))$. As in the proof of Proposition C.1, ϕ is the Bregman divergence of log-sum-exp; the value and gradient of ϕ vanish at zero, and the Hessian at h is

$$F(\text{softmax}(a+h)) = \text{diag}(\text{softmax}(a+h)) - \text{softmax}(a+h)\,\text{softmax}(a+h)^\top.$$

For any probability vector r, the matrix $F(r) \coloneqq \text{diag}(r) - rr^\top$ is positive semidefinite. In row i, the row's diagonal entry and the sum of the absolute off-diagonal entries are both $r_i(1-r_i)$. Gershgorin's theorem therefore gives

$$\lambda_{\max}(F(r)) \leq \max_i 2r_i(1-r_i) \leq \frac{1}{2}.$$

The integral form of Taylor's theorem consequently yields the global bound

$$\phi(h) = \int_0^1 (1-s)[h^\top F(\mathrm{softmax}(a+sh))h]\mathrm{d}s \le \frac{1}{4}\|h\|_2^2. \tag{39}$$

Let $\delta \coloneqq Q_{m,\sigma}(z_c) - z_c$ and $h_t \coloneqq G_{c,t}(z_c+\delta) - G_{c,t}(z_c)$. The Lipschitz assumption, Eq. 39, and Eq. 38 imply

$$D_t^{c,m} \le \frac{1}{4}\|h_t\|^2 \le \frac{L_{c,t}^2}{4}\|\delta\|^2 \le \frac{L_{c,t}^2 \mathcal{E}_\sigma(z_c)}{16 q_m^2}.$$

For $m \ge 2$, $q_m = 2^{m-1} - 1 \ge 2^{m-2}$, so $1/(16q_m^2) \le 2^{-2m}$. The inequality for q_m proves Eq. 14. □

Proof of Lemma C.6. Let p be any strictly positive categorical distribution with unique winner w, and fix $j \neq w$. Consider

$$\inf_{q\in\Delta: q_j \ge q_w} D_{\mathrm{KL}}(p\|q), \tag{40}$$

where Δ is the probability simplex. Because p is strictly positive, the objective is infinite when any coordinate of q vanishes. On the positive simplex, the objective is strictly convex in q, so the minimizer is unique. A strictly positive distribution with $q_j > q_w$ is feasible, so Slater's condition holds and the KKT conditions are necessary and sufficient. Introduce a multiplier λ for $\sum_k q_k = 1$ and a multiplier $\nu \ge 0$ for $q_w - q_j \le 0$. The inequality must be active. Otherwise stationarity would give the unconstrained minimizer $q = p$, which is infeasible because $p_w > p_j$.

The stationarity equations at $q_w = q_j = s$ are

$$-\frac{p_w}{s} + \lambda + \nu = 0, \qquad -\frac{p_j}{s} + \lambda - \nu = 0, \qquad -\frac{p_k}{q_k} + \lambda = 0 \quad \text{for } k \notin \{w, j\}.$$

The stationarity equations imply

$$s = \frac{p_w + p_j}{2\lambda}, \qquad q_k = \frac{p_k}{\lambda} \quad \text{for } k \notin \{w, j\}.$$

Normalization gives $\lambda = 1$, and subtracting the two active-coordinate stationarity equations gives $\nu = (p_w - p_j)/(2s) > 0$, verifying dual feasibility. Hence the unique minimizer leaves every other coordinate unchanged and replaces (p_w, p_j) by the arithmetic mean of the two active coordinates. The minimum in Eq. 40 is therefore

$$\psi(p_w, p_j) \coloneqq p_w \log \frac{2p_w}{p_w + p_j} + p_j \log \frac{2p_j}{p_w + p_j}.$$

For fixed $a > 0$ and $0 < b < a$, direct differentiation gives

$$\frac{\partial}{\partial b}\psi(a,b) = \log \frac{2b}{a+b} < 0.$$

Thus the smallest cost over all $j \neq w$ is attained by the runner-up v, and the smallest cost equals χ_t in Eq. 15. Any distribution for which the clean winner is tied or beaten belongs to at least one constraint set in Eq. 40, so the distribution's forward KL from p is at least χ_t. The lower bound proves the lemma. □

Derivation of Eq. 16. Softmax ratios satisfy $p_{t,v_t}/p_{t,w_t} = e^{-\gamma_t}$; substituting the softmax-ratio identity into Eq. 15 gives Eq. 16. □

H.4 Proofs for the Nonasymptotic Simultaneous-Policy Certificate

Proof of Theorem C.7. Fix a policy and the associated sequence of hybrid prefix passes. At hook c_i, define the realized quantization error $\delta_i \coloneqq Q_{m_i,\sigma_i}(z_i^{(i-1)}) - z_i^{(i-1)}$. The nearest-rounding argument

in Eq. 38 applies to the realized prefix activation, including the prefix activation's realized dynamic ranges and any numerical floor; consequently,

$$\|\delta_i\|_2^2 \le \frac{E_i}{4q_{m_i}^2}. \tag{41}$$

The prefix-error bound does not compare δ_i with a clean-pass error. Earlier prefix quantizers have already been incorporated into $z_i^{(i-1)}$.

The $(i-1)$- and i-prefix passes have identical computation up to c_i. The two prefix passes differ there by precisely δ_i, and in both passes all later hooks are left unquantized. The assumed segmentwise Lipschitz property consequently implies

$$\|a_t^{(i)} - a_t^{(i-1)}\|_2 \le L_{i,t}\|\delta_i\|_2. \tag{42}$$

Telescoping the hybrid logits, applying the triangle inequality, and then using Eqs. 41 and 42 yield

$$\|a_t^{(n)} - a_t^{(0)}\|_2 \le \sum_{i=1}^{n}\|a_t^{(i)} - a_t^{(i-1)}\|_2 \le \frac{1}{2}\sum_{i=1}^{n}\frac{L_{i,t}\sqrt{E_i}}{q_{m_i}}. \tag{43}$$

The global softmax inequality in Eq. 39, applied with $h \coloneqq a_t^{(n)} - a_t^{(0)}$, gives

$$D_t^m \le \frac{1}{4}\|a_t^{(n)} - a_t^{(0)}\|_2^2.$$

Combining the global softmax inequality with Eq. 43 proves Eq. 17. □

H.5 Proofs for the Multi-Token Choice-Likelihood Certificate

Proof of Theorem C.8. For a vocabulary token y, define the token's log-probability as a function of the logits, $f_y(a) \coloneqq \log \mathrm{softmax}(a)_y$. At any a, with $p \coloneqq \mathrm{softmax}(a)$ and e_y the yth standard basis vector, $\nabla f_y(a) = e_y - p$. The log-probability gradient's Euclidean norm is globally bounded because

$$\|e_y - p\|_2^2 = (1-p_y)^2 + \sum_{j\neq y} p_j^2 \le (1-p_y)^2 + \left(\sum_{j\neq y} p_j\right)^2 = 2(1-p_y)^2 \le 2.$$

The fundamental theorem of calculus along the line segment from a to $a+h$ therefore gives the global Lipschitz inequality

$$|f_y(a+h) - f_y(a)| = \left|\int_0^1 [\nabla f_y(a+sh)^\top h]\mathrm{d}s\right| \le \sqrt{2}\|h\|_2. \tag{44}$$

Fix candidate k. Apply the prefix-pass construction from the proof of Theorem C.7 to candidate k's forward pass. In particular, Eq. 43, with the candidate-specific energies and Lipschitz constants, gives at every scored position

$$\|a_{k,t}^{(n)} - a_{k,t}^{(0)}\|_2 \le \frac{1}{2}\sum_{i=1}^{n}\frac{L_{k,i,t}\sqrt{E_{k,i}}}{q_{m_i}} = H_{k,t}(m). \tag{45}$$

The candidate-specific prefix construction uses the realized activation presented to each hook after the earlier prefix quantizers, and hence does not require the errors of different hooks to be independent or centered.

Use Eq. 44 with $y \coloneqq y_{k,t}$ and the logit difference in Eq. 45. Nonnegativity of the weights and the triangle inequality yield

$$\begin{aligned}|S_k^{(n)} - S_k^{(0)}| &\le \sum_{t\in\mathcal{T}_k} w_{k,t}|f_{y_{k,t}}(a_{k,t}^{(n)}) - f_{y_{k,t}}(a_{k,t}^{(0)})| \\ &\le \sqrt{2}\sum_{t\in\mathcal{T}_k} w_{k,t}H_{k,t}(m) = B_k(m),\end{aligned}$$

which proves Eq. 20. Since weighted target-token NLL is $-S_k$, the same absolute bound applies to the NLL change. □

H.6 Proof of the Continuous RAR-log Optimum

Proof of Proposition 1. Let $f(m) \coloneqq \sum_c A_c 2^{-2(m_c - m_0)}$. For each site,

$$\frac{d^2}{dm_c^2}(A_c 2^{-2(m_c-m_0)}) = (2\ln 2)^2 A_c 2^{-2(m_c-m_0)} > 0.$$

Thus f is strictly convex. The feasible set in Eq. 3 is nonempty, compact, and convex, so a minimizer exists and strict convexity makes the minimizer unique.

The budget constraint must be active. Otherwise there is positive budget slack. Because $B < m_{\max} \sum_c \kappa_c$, not every coordinate can equal $m_{\max}$; increasing a nonsaturated coordinate by a sufficiently small positive amount would remain feasible and would strictly decrease f, a contradiction.

The strict inequalities on B also provide a strictly feasible point, so the KKT conditions are necessary and sufficient. Let $\lambda \geq 0$ be the multiplier of the budget constraint. For a fixed λ, minimization of the Lagrangian over the box separates by site:

$$\min_{m_c \in [m_{\min}, m_{\max}]} \{A_c 2^{-2(m_c-m_0)} + \lambda \kappa_c m_c\}.$$

The multiplier is positive. If $\lambda = 0$, every separated problem is minimized at $m_{\max}$, which violates the active budget because $B < m_{\max} \sum_c \kappa_c$. For $\lambda > 0$, the unconstrained stationary point satisfies

$$-2A_c 2^{-2(m_c-m_0)} \ln 2 + \lambda \kappa_c = 0,$$

and hence

$$m_c = m_0 + \frac{1}{2}\log_2 \frac{A_c}{\kappa_c} + \frac{1}{2}\log_2 \frac{2\ln 2}{\lambda}.$$

The separated objective is strictly convex, so the interval-constrained minimizer is the stationary point clipped to $[m_{\min}, m_{\max}]$. Setting $\tau \coloneqq m_0 + (1/2)\log_2(2\ln 2/\lambda)$ gives Eq. 4. Complementary slackness and the already proved activity of the budget give $\sum_c \kappa_c m_c^\star = B$. Since the KKT conditions are sufficient for the convex program, the clipped allocation is the unique global minimizer. At either endpoint budget, feasibility and monotonicity of every objective term force the corresponding uniform endpoint solution. □

H.7 Proofs for the Heterogeneity Dividend

Proof of Theorem C.9. First verify that Eq. 23 uses exactly the available budget. By the definition of the weighted geometric mean, $\log_2 G_r = \sum_c \pi_c \log_2 r_c$. Therefore

$$\sum_c \kappa_c m_c^{\mathrm{het}} = K_\kappa \sum_c \pi_c \left(\overline{m} + \frac{1}{2}\log_2 \frac{r_c}{G_r}\right) = K_\kappa \overline{m} = B.$$

For any real-valued allocation m satisfying the same equality budget, weighted AM–GM applied to the positive numbers $r_c 2^{-2(m_c-m_0)}$ with weights π_c gives

$$\begin{aligned}\frac{\Phi_A(m)}{K_\kappa} &= \sum_c \pi_c r_c 2^{-2(m_c-m_0)} \\ &\geq \prod_c (r_c 2^{-2(m_c-m_0)})^{\pi_c} \\ &= G_r 2^{-2(\sum_c \pi_c m_c - m_0)} = G_r 2^{-2(\overline{m}-m_0)}. \end{aligned} \tag{46}$$

At the allocation in Eq. 23, every term in the weighted arithmetic mean has the common value

$$r_c 2^{-2(m_c^{\mathrm{het}}-m_0)} = r_c 2^{-2(\overline{m}-m_0)} 2^{-\log_2(r_c/G_r)} = G_r 2^{-2(\overline{m}-m_0)}.$$

Thus equality holds in Eq. 46, proving global optimality and Eq. 24. Equality in weighted AM–GM, whose weights are all positive, requires all $r_c 2^{-2(m_c-m_0)}$ to be equal. Together with the equality budget, the equality condition uniquely determines Eq. 23; hence the relaxed optimum is unique. □

Proof of Corollary C.10. For the uniform policy, direct substitution yields

$$\Phi_A(m^{\mathrm{uni}}) = \sum_c \kappa_c r_c 2^{-2(\overline{m}-m_0)} = K_\kappa M_r 2^{-2(\overline{m}-m_0)},$$

which proves Eq. 25. Dividing the uniform-policy identity by Eq. 24 proves Eq. 26. Weighted AM–GM gives $M_r \geq G_r$, with equality if and only if all r_c are equal, which proves the equality characterization. □

Proof of Proposition C.11. If m^{het} lies inside the bit box, consider any policy feasible for the original inequality budget. Repeating the weighted AM–GM calculation with $\sum_c \pi_c m_c \leq \overline{m}$ gives

$$\frac{\Phi_A(m)}{K_\kappa} \geq G_r 2^{-2(\sum_c \pi_c m_c - m_0)} \geq G_r 2^{-2(\overline{m}-m_0)}.$$

The feasible vector m^{het} attains the lower bound and is the optimum of the boxed inequality-budget problem.

For the general statement when clipping is active, if $m_{\min} < \overline{m} < m_{\max}$, the uniform policy is feasible and is an interior point of the bit box. Hence the boxed optimum cannot have a larger objective. If all r_c are equal, Eq. 23 defines a uniform allocation, so equality holds. Conversely, suppose r_c are not all equal and choose indices c, d with $r_c > r_d$. Define a budget-neutral direction v by

$$v_c \coloneqq \kappa_c^{-1}, \qquad v_d \coloneqq -\kappa_d^{-1}, \qquad v_j \coloneqq 0 \quad \text{for } j \notin \{c, d\}.$$

Then $\sum_j \kappa_j v_j = 0$, and the directional derivative of Φ_A at the uniform policy is

$$\begin{aligned}\nabla\Phi_A(m^{\mathrm{uni}})^\top v &= -2 \cdot 2^{-2(\overline{m}-m_0)}(\ln 2)\left(\frac{A_c}{\kappa_c} - \frac{A_d}{\kappa_d}\right) \\ &= -2 \cdot 2^{-2(\overline{m}-m_0)}(\ln 2)(r_c - r_d) < 0.\end{aligned}$$

Because the uniform point is strictly inside the box, a sufficiently small positive step along v remains feasible and strictly decreases the objective. Uniform therefore cannot be optimal, proving the strict boxed improvement even when the unconstrained allocation is clipped. □