PACE: PRECISION ALLOCATION WITH COST ESTIMATION FOR MIXED-PRECISION QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixed-precision quantization reduces model storage by assigning different bitwidths under a fixed budget. Its effectiveness depends on accurately estimating which precision decisions best preserve model quality. Existing methods often rely on indirect sensitivity measures that do not reflect the actual output damage caused by different quantization candidates. We introduce PACE (Precision Allocation with Cost Estimation), a second-order mixed-precision quantization method grounded in a stationary output-fidelity objective. PACE efficiently predicts the output damage of concrete precision candidates and globally allocates the available bit budget according to their estimated benefits. Experiments validate the consistency between predicted and measured quantization damage and compare PACE across three 7–8B model families, two perplexity benchmarks, six zero-shot tasks, and average 6-bit and 3-bit settings. Under the average 6-bit setting, PACE ranks first or second across all nine evaluations, with WikiText2 perplexity increases of at most 0.095, while remaining robust under the more challenging average 3-bit setting. On Qwen3-8B, PACE is approximately $490\times$ faster than SFMP, the strongest comparable-quality baseline. Its packed representations provide approximately $2.6\times$ compression under W6 and more than $4.6\times$ compression under W3.

1 INTRODUCTION

Deploying large language models (LLMs) is constrained by the storage and memory traffic of billions of parameters. Post-training quantization reduces these costs with a small calibration set, and methods such as GPTQ, AWQ, and SpQR preserve model quality at low precision (Frantar et al., 2022; Lin et al., 2024; Dettmers et al., 2024). Improved codebooks and learned rotations further expand the quality–compression trade-off (Tseng et al., 2024; Liu et al., 2025). Mixed-precision quantization introduces a complementary allocation problem: under a fixed budget, additional bits should be assigned where they best preserve model quality. Its effectiveness therefore depends on estimating the benefit of each precision decision.

Existing mixed-precision methods make allocation tractable through proxy signals. HAQ uses hardware-aware rewards (Wang et al., 2019), while OMPQ and CMPQ rely on feature or activation statistics (Ma et al., 2023; Chen et al., 2026). EdMIPS and SDQ optimize differentiable precision choices (Cai & Vasconcelos, 2020; Huang et al., 2022), and LLM-MQ and SignRoundV2 combine gradients with quantization error (Li et al., 2023; Cheng et al., 2025). Although effective for navigating the allocation space, these scores do not necessarily represent the output damage of independently optimized quantization candidates. They therefore cannot directly determine how much a specific group benefits from additional precision. Measuring every assignment through full-model evaluation would be more direct but becomes intractable as the allocation space grows.

Second-order methods account for nonuniform curvature and provide a closer connection to perturbation damage. HAWQ and HAWQ-V2 use Hessian spectra or traces to guide mixed precision (Dong et al., 2019; 2020). BRECQ and GPTQ develop curvature-informed reconstruction, while OWQ and SliM-LLM protect sensitive weights or groups (Li et al., 2021; Frantar et al., 2022; Lee et al., 2024; Huang et al., 2025). SqueezeLLM, FIT, and SFMP obtain tractable sensitivity estimates from gradient statistics (Kim et al., 2024; Zandonati et al., 2023; Nie et al., 2026), and KronQ retains richer structure through a Kronecker-factored Hessian approximation (Lee et al., 2026). These

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.

approaches expose a central trade-off. Richer loss approximations can improve allocation, but they are expensive to construct and apply at full-model scale. Local reconstruction curvature also need not measure how a quantization perturbation affects the final model output.

Recent methods increasingly combine sensitivity estimates with global precision allocation. MixLLM uses Taylor-based scores to compare the importance of output features (Zheng et al., 2026), while constrained optimization and GAMMA allocate precision under global budgets (Chen et al., 2021; Yao et al., 2026). ResQ and ScaleBITS further consider representation and hardware constraints (Saxena et al., 2025; Li et al., 2026). However, reliable allocation calls for estimates that reflect the output damage of each quantization choice, remain comparable across the model, and can be obtained efficiently at full-model scale. Sensitivity alone cannot determine the benefit of additional precision because it does not account for the quantization error produced at each bitwidth.

We introduce PACE (Precision Allocation with Cost Estimation), a second-order method for mixed-precision quantization. PACE defines a stationary output-fidelity objective and uses random projections to estimate shared output sensitivity with one full-model backward pass per calibration sequence. It evaluates each independently optimized quantization choice using both this sensitivity and the weight perturbation produced by quantization. The resulting estimates reflect the predicted output damage of each choice and remain comparable across layers and bitwidths.

PACE uses these costs to allocate precision globally while exactly satisfying a budget on the total number of bits assigned to model weights. Local candidate construction captures the error remaining at each bitwidth, while the global optimizer selects one candidate for every allocation group. This separates reusable model-wide sensitivity estimation from local candidate optimization and global budget assignment. For two candidate bitwidths, allocation is governed by the predicted reduction in output damage obtained from choosing the higher-bit candidate. When allocation groups have equal size, the exact solution reduces to ranking these predicted benefits.

We evaluate PACE on Qwen3-8B, Mistral-7B-v0.3, and Llama-3.1-8B under average 6-bit W4/W8 and average 3-bit W2/W4 budgets. Predicted costs reproduce measured quantization-damage patterns across layers, groups, and candidate scales. Under W6, PACE ranks first or second across all nine evaluations, with WikiText2 perplexity increases of at most 0.095. It remains robust under the more challenging W3 budget, where several competing allocation strategies diverge. On Qwen3-8B, PACE is approximately $490\times$ faster than SFMP, the strongest comparable-quality baseline. Its packed representations provide approximately $2.6\times$ target-weight compression under W6 and more than $4.6\times$ under W3.

Our main contributions are summarized as follows:

- We formulate mixed-precision quantization as the allocation of predicted output damage. Unlike sensitivity alone, our criterion evaluates the perturbation produced by each quantization choice and measures the benefit of assigning additional precision.
- We propose PACE, a mixed-precision quantization framework that efficiently estimates shared output sensitivity, evaluates independently optimized quantization choices, and allocates bits globally while exactly satisfying the total weight bit budget.
- We validate PACE on three LLM families under both W6 and W3 settings. PACE consistently predicts measured quantization damage, ranks first or second across all nine W6 evaluations, runs approximately $490\times$ faster than the strongest baseline with comparable quality, and provides approximately $2.6\times$ and more than $4.6\times$ storage compression under W6 and W3.

2 METHOD

2.1 SHARED OUTPUT SENSITIVITY

As shown in Figure 1(a), we pass calibration sequences $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_S\}$ through the pretrained model $f_{\boldsymbol{\theta}}$, where $\boldsymbol{\theta}$ collects the target attention and feed-forward weights. At evenly spaced non-final token positions, we select the top-k logits and collect them into $\mathbf{y}_{\mathbf{x}}(\boldsymbol{\theta}) \in \mathbb{R}^{K_{\mathbf{x}}}$. Their coordinates remain fixed throughout sensitivity estimation. With $\mathbb{E}_{\mathcal{D}}$ denoting the empirical mean, a perturbation $\boldsymbol{\delta}$ has output-fidelity loss

$$\mathcal{L}(\boldsymbol{\delta}) = \frac{1}{2}\mathbb{E}_{\mathcal{D}}\left[\frac{\|\mathbf{y}_{\mathbf{x}}(\boldsymbol{\theta}+\boldsymbol{\delta}) - \mathbf{y}_{\mathbf{x}}(\boldsymbol{\theta})\|_2^2}{K_{\mathbf{x}}}\right], \qquad \mathbf{H} = \nabla^2\mathcal{L}(\mathbf{0}) = \mathbb{E}_{\mathcal{D}}\left[\frac{\mathbf{J}_{\mathbf{x}}^\top \mathbf{J}_{\mathbf{x}}}{K_{\mathbf{x}}}\right]. \tag{1}$$

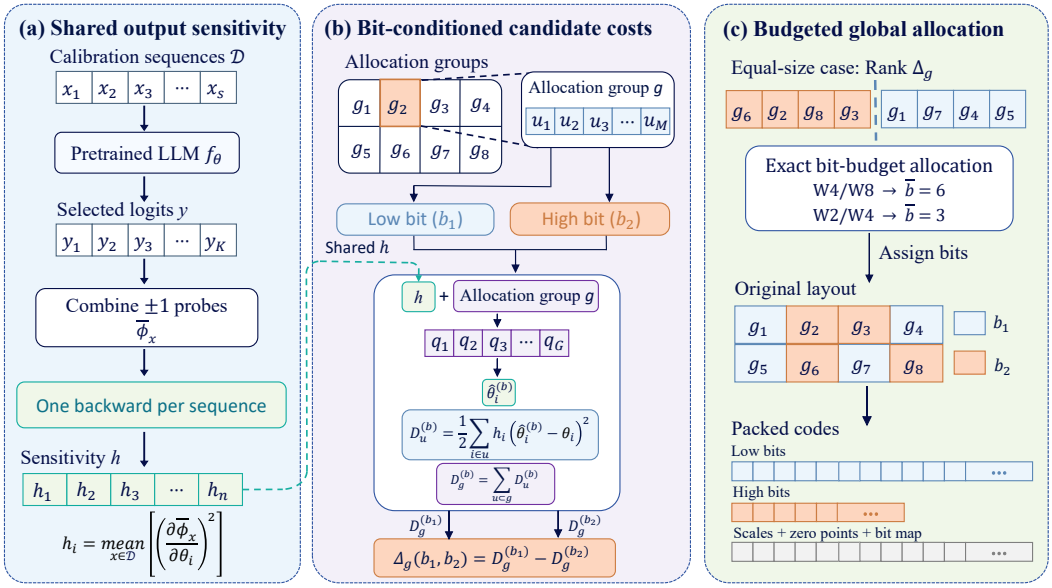

Figure 1: Overview of PACE. (a) Selected logits estimate output sensitivity shared across the model. (b) Independently optimized bit-conditioned candidates are ranked by predicted damage. (c) Global allocation meets the weight-code budget and packs codes with scales, zero points, and bit maps.

Here $\mathbf{J}_\mathbf{x}$ is the Jacobian of the selected logits with respect to the target weights, evaluated at $\boldsymbol{\theta}$. The unquantized output is treated as a fixed reference. At $\boldsymbol{\delta} = \mathbf{0}$, the residual and gradient are therefore zero, and the leading local cost is $\frac{1}{2}\boldsymbol{\delta}^\top\mathbf{H}\boldsymbol{\delta}$.

To estimate its diagonal, we draw R independent Rademacher vectors $\boldsymbol{\epsilon}_{\mathbf{x},r} \in \{-1,+1\}^{K_\mathbf{x}}$ with independent entries, combine their projections, and differentiate once:

$$\bar{\phi}_\mathbf{x} = \frac{1}{\sqrt{RK_\mathbf{x}}}\sum_{r=1}^{R}\boldsymbol{\epsilon}_{\mathbf{x},r}^\top\mathbf{y}_\mathbf{x}(\boldsymbol{\theta}), \qquad h_i = \mathbb{E}_\mathcal{D}\left[\left(\frac{\partial\bar{\phi}_\mathbf{x}}{\partial\theta_i}\right)^2\right]. \tag{2}$$

All target weights are differentiated together in one backward pass for each calibration sequence. The empirical estimator is unbiased over the probe randomness: $\mathbb{E}_\boldsymbol{\epsilon}[h_i] = H_{ii}$, since $R^{-1/2}\sum_r \boldsymbol{\epsilon}_{\mathbf{x},r}$ has identity covariance. Replacing $\mathbf{H}$ by $\mathrm{diag}(\mathbf{h})$ gives the additive approximation $\mathcal{L}(\boldsymbol{\delta}) \approx \frac{1}{2}\sum_i h_i\delta_i^2$. Thus, S backward passes produce a sensitivity shared by every group and bitwidth, as indicated by the dashed connection from Figure 1(a) to (b).

2.2 Bit-Conditioned Candidate Costs

As shown in Figure 1(b), PACE partitions each target matrix into consecutive row-wise groups u of 128 parameters along the input-channel dimension. Groups covering the same input-channel interval across all output rows form an allocation group $g \in \mathcal{G}$. All $u \subset g$ share a candidate bitwidth $b \in \mathcal{B}$, while retaining their own scale and zero point. PACE evaluates each (g, b) candidate by summing the costs of its row-wise groups.

For a scalar parameter θ_i, candidate bitwidth b, scale $s > 0$, and integer zero point z, quantization and reconstruction are

$$q_i^{(b)}(s,z) = \mathrm{clip}\big(\mathrm{round}(\theta_i/s) + z, 0, 2^b - 1\big), \qquad \widehat{\theta}_i^{(b)}(s,z) = s\left(q_i^{(b)}(s,z) - z\right). \tag{3}$$

Quantization is elementwise, with s and z shared within each row-wise group u.

For a row-wise group u contained in g, PACE initializes

$$s_0 = \frac{\max_{i\in u}\theta_i - \min_{i\in u}\theta_i}{2^b - 1}, \qquad z_0 = \mathrm{clip}\left(\mathrm{round}\left(-\frac{\min_{i\in u}\theta_i}{s_0}\right), 0, 2^b - 1\right). \tag{4}$$

A constant row-wise group uses a finite positive fallback scale. It evaluates $z = \mathrm{clip}(z_0 + r, 0, 2^b - 1)$ for $r \in \{-1, 0, 1\}$ and scales $s = \alpha s_0$, where $\alpha \in \{0.5, 0.625, 0.75, 0.875, 1, 1.125, 1.25, 1.5\}$. For each pair, the predicted output damage is

$$D_u^{(b)}(s, z) = \frac{1}{2} \sum_{i \in u} h_i \left(\widehat{\theta}_i^{(b)}(s, z) - \theta_i \right)^2 . \tag{5}$$

For every searched pair (α, z), PACE fixes the integer offsets $v_i = q_i^{(b)}(s, z) - z$ and evaluates the weighted least-squares refinement

$$s^* = \frac{\sum_{i \in u} h_i v_i \theta_i}{\sum_{i \in u} h_i v_i^2}. \tag{6}$$

When finite, positive, and lower-cost, s^* replaces the searched scale for the fixed integer codes. PACE retains the minimum-cost candidate over all searched and refined choices, with its scale and zero point denoted by $(s_u^{(b)}, z_u^{(b)})$.

The notation $u \subset g$ indicates that the sum ranges over all row-wise groups contained in allocation group g. The candidate cost for allocation group g is the sum of its selected row-wise costs:

$$D_g^{(b)} = \sum_{u \subset g} D_u^{(b)} \left(s_u^{(b)}, z_u^{(b)} \right). \tag{7}$$

Thus, $D_g^{(b)}$ measures the predicted loss of assigning bitwidth b to group g.

2.3 Budgeted Global Allocation

As shown in Figure 1(c), PACE selects one bitwidth for every allocation group under a global weight-code budget. Let $a_g^{(b)} \in \{0, 1\}$ indicate whether group g uses candidate bitwidth b, let n_g be the number of parameters in that group, and let $\bar{b}$ denote the target parameter-weighted average bitwidth. PACE solves

$$\min_{\{a_g^{(b)}\}} \sum_{g \in \mathcal{G},\, b \in \mathcal{B}} a_g^{(b)} D_g^{(b)} \quad \text{s.t.} \quad \sum_{b \in \mathcal{B}} a_g^{(b)} = 1 \ \ (\forall g \in \mathcal{G}), \qquad \sum_{g \in \mathcal{G},\, b \in \mathcal{B}} n_g b\, a_g^{(b)} = \bar{b} \sum_{g \in \mathcal{G}} n_g. \tag{8}$$

The first constraint selects exactly one candidate for each group, while the second enforces the target parameter-weighted average bitwidth $\bar{b}$.

For two candidate bitwidths $b_1 < b_2$ in $\mathcal{B}$, using the predicted upgrade benefit $\Delta_g(b_1, b_2) = D_g^{(b_1)} - D_g^{(b_2)}$, the problem reduces to

$$\max_{\{a_g^{(b_2)}\}} \sum_{g \in \mathcal{G}} a_g^{(b_2)} \Delta_g(b_1, b_2), \qquad \text{s.t.} \qquad \sum_{g \in \mathcal{G}} n_g a_g^{(b_2)} = \frac{\bar{b} - b_1}{b_2 - b_1} \sum_{g \in \mathcal{G}} n_g. \tag{9}$$

PACE solves this exact parameter-weighted allocation over all target groups. In the principal setting, $(b_1, b_2, \bar{b}) = (4, 8, 6)$, so exactly half of the target parameters use W8. When all groups have the same size, this reduces to selecting half of the groups with the largest $\Delta_g(4, 8)$. The same formulation gives W2/W4 allocation at an average of three bits.

After allocation, PACE applies the selected bitwidth to every row-wise group in the corresponding allocation group and reruns the local search only at that bitwidth. The resulting quantized weights retain the original layout. The allocation budget covers weight codes only, while physical storage also includes scales, zero points, and bitwidth metadata.

3 Experiments

We evaluate PACE along four axes: the consistency of its damage estimator, model quality under average W6 and W3 budgets, the structure and effectiveness of the learned allocations, and quantization cost and storage. Experiments cover Qwen3-8B (Yang et al., 2025), Mistral-7B-v0.3 (Jiang et al., 2023), and Llama-3.1-8B (Grattafiori et al., 2024).

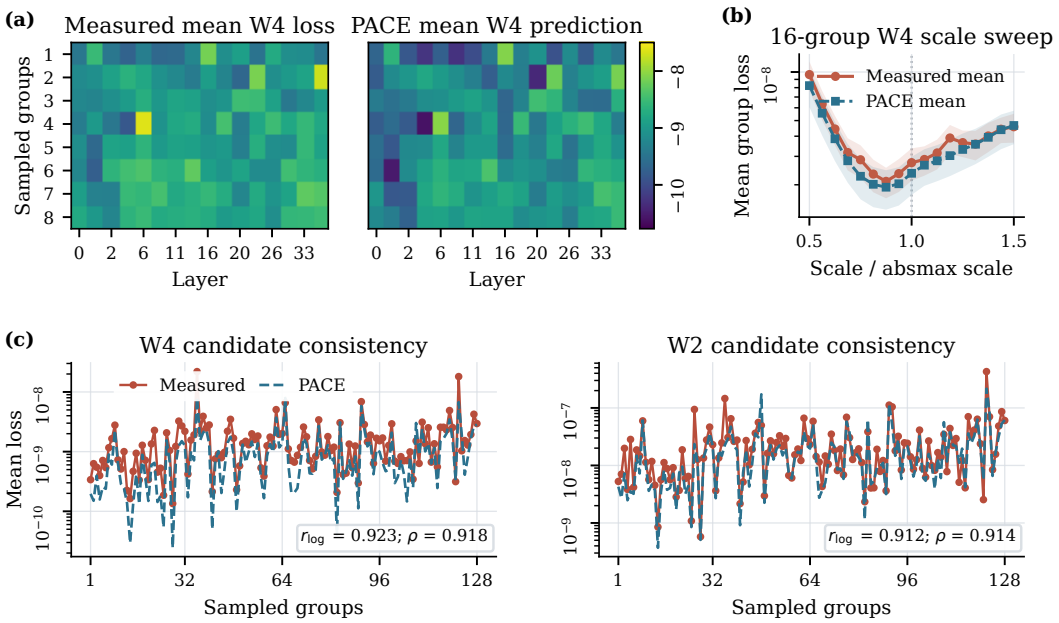

Figure 2: Estimator consistency on Qwen3-8B across sampled row-wise groups: (a) W4 damage maps on a shared log scale, (b) a 16-group W4 scale sweep with standard-error bands, and (c) group-wise W4 and W2 comparisons.

We compare PACE with SFMP (Nie et al., 2026), KronQ (Lee et al., 2026), AutoRound (Cheng et al., 2024), OWQ (Lee et al., 2024), SliM-LLM (Huang et al., 2025), AutoScheme (Cheng et al., 2025), MixLLM (Zheng et al., 2026), MixQuant (Misra et al., 2026), and AMQ (Lee et al., 2025). These baselines span second-order, reconstruction-based, sensitivity-based, and search-based quantization methods.

All methods use the same candidate bitwidths and average weight-code budgets, while retaining their original scoring and allocation principles. Unless stated otherwise, PACE uses asymmetric quantization with group size 128 (g128) and tests three zero-point candidates at each bitwidth. The following sections report estimator consistency, model quality, allocation behavior, and computational cost under this common protocol.

3.1 Estimator Consistency

We test whether PACE predicts the output damage caused by quantizing a sampled row-wise group. Using 32 C4 calibration sequences (Raffel et al., 2020), we sample eight row-wise groups from each of 16 layers. Measured damage is the output-fidelity loss over the same selected logits after replacing only the sampled group, averaged across the 32 sequences. Figure 2(a) shows that the predicted W4 damage reproduces the main high- and low-damage regions in the measured map. Figure 2(b) averages a controlled W4 scale sweep over 16 groups, with shading denoting the standard error. The predicted and measured curves follow similar trends and reach their minima near 0.875 times the absolute-maximum scale.

Across all 128 groups, Figure 2(c) gives log-space Pearson and Spearman correlations of 0.923 and 0.918 for W4, and 0.912 and 0.914 for W2. Correlations above 0.91 support using the predicted damage to rank precision choices. The controlled sweep identifies a useful scale region but not a universal optimum, so PACE searches scale separately for every group and bitwidth.

3.2 Main Results under the W4/W8 Budget

We evaluate PACE and all baselines on these three models under an average 6-bit budget with the $\{$W4, W8$\}$ candidate set and group size 128. Calibration draws the same 32 C4 training sequences

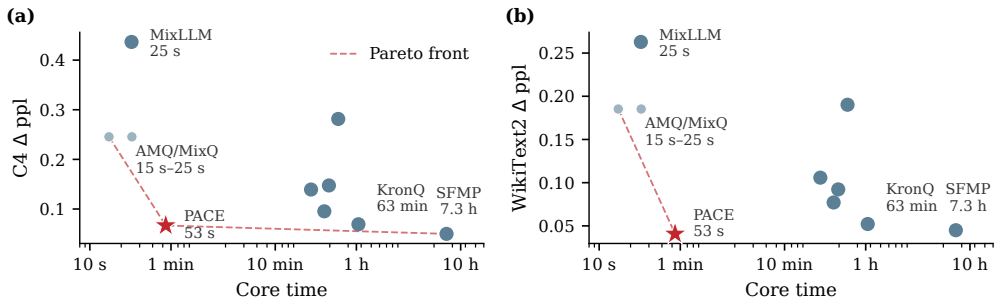

Figure 3: Cost-quality trade-off on Qwen3-8B at average W6: (a) C4 and (b) WikiText2. Core time covers calibration, scoring, and allocation. Dashed curves are Pareto fronts. AMQ/MixQ represents two equal-quality points on this model.

for all methods. Language modeling is reported as perplexity on C4 (validation split) and WikiText2 (Merity et al., 2017). Because the calibration data is C4, the former measures in-domain quality while the latter tests generalization to a held-out distribution, and no calibration document appears in either evaluation set. Downstream capability is measured by the average zero-shot accuracy over WinoGrande (Sakaguchi et al., 2020), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), ARC-Easy and ARC-Challenge (Clark et al., 2018), and BoolQ (Clark et al., 2019). Table 1 reports the full results. Figure 3 relates language-modeling quality to quantization cost.

Table 1: Average-W6 W4/W8 results. PPL increases are relative to BF16. Avg. ZS averages six tasks. Bold and underline mark the top two results.

	Qwen3-8B			Mistral-7B			Llama-3.1-8B		
Method	ΔC4↓	ΔWiki↓	Avg. ZS↑	ΔC4↓	ΔWiki↓	Avg. ZS↑	ΔC4↓	ΔWiki↓	Avg. ZS↑
PACE	<u>0.067</u>	**0.041**	**73.2**	**0.030**	<u>0.018</u>	**71.6**	**0.147**	<u>0.095</u>	<u>72.5</u>
SFMP	**0.050**	<u>0.045</u>	**73.2**	0.032	**0.011**	<u>71.5</u>	<u>0.157</u>	**0.086**	**72.7**
KronQ	0.069	0.052	<u>73.1</u>	<u>0.031</u>	0.027	71.4	0.159	0.105	72.4
AutoRound	0.147	0.092	72.4	0.115	0.061	71.4	0.500	0.302	71.8
OWQ	0.095	0.077	72.9	0.333	0.215	70.9	0.387	0.426	71.9
SliM-LLM	0.139	0.106	72.1	0.062	0.047	71.5	0.237	0.159	72.0
AutoScheme	0.281	0.190	<u>73.1</u>	0.089	0.052	71.2	0.373	0.221	72.0
MixLLM	0.437	0.263	72.9	0.090	0.055	71.5	0.400	0.260	72.4
MixQuant	0.245	0.185	72.2	0.132	0.060	**71.6**	0.590	0.349	72.4
AMQ	0.245	0.185	72.2	0.131	0.062	71.5	0.597	0.376	<u>72.5</u>

Table 1 reports perplexity increases over the BF16 dense models on C4 and WikiText2. PACE leaves increases of 0.067 and 0.041 on Qwen3-8B, 0.030 and 0.018 on Mistral-7B, and 0.147 and 0.095 on Llama-3.1-8B. SFMP and KronQ are the strongest baselines, and PACE ranks first or second in all nine reported comparisons, a consistency not achieved by any baseline. The similar ordering on C4 and WikiText2 suggests that the learned allocation transfers beyond the calibration domain. PACE obtains average zero-shot accuracies of 73.2, 71.6, and 72.5. It ranks first or tied first on Qwen3-8B and Mistral-7B. Since all methods remain close to their dense references under the W4/W8 budget, perplexity provides a clearer distinction among methods.

Figure 3 compares quality and algorithm core time on Qwen3-8B under each method's native calibration protocol. PACE completes calibration, scoring, and allocation in 53.4 seconds. SFMP and KronQ attain comparable quality but require 7.3 hours and 63 minutes, making PACE approximately $490\times$ and $70\times$ faster, respectively. AMQ, MixLLM, and MixQuant finish in 15 to 25 seconds, but incur 3.7 to 6.5 times larger perplexity increases across C4 and WikiText2. Consequently, no evaluated method is both faster and more accurate than PACE, as indicated by the dashed Pareto front.

3.3 Low-Bit Results under the W2/W4 Budget

The average 3-bit W2/W4 budget changes the character of the problem. Perplexity increases are now measured in whole units rather than hundredths, and downstream accuracy becomes clearly

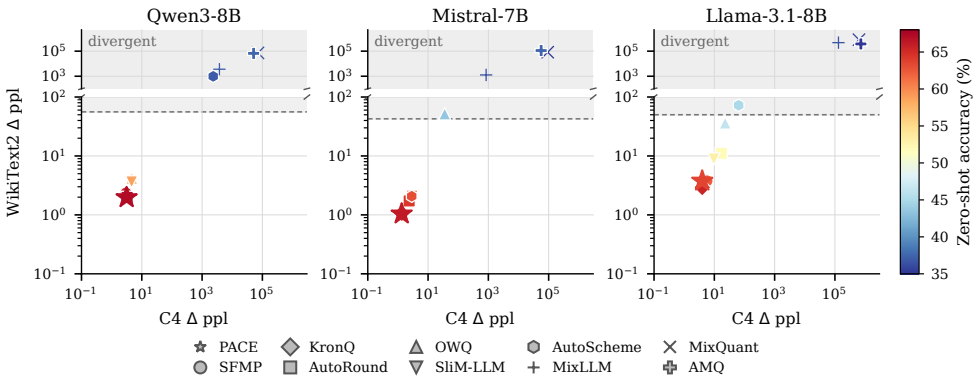

Figure 4: Average-W3 W2/W4 robustness on three models. Axes show PPL increases over BF16, colors encode six-task zero-shot accuracy, and shapes identify methods. The broken y-axis enlarges stable runs, while shading marks WikiText2 PPL above $10\times$ BF16. Failed runs are omitted.

discriminative. Figure 4 places every run in the plane of C4 and WikiText2 perplexity increases, with marker color encoding zero-shot accuracy. A run whose WikiText2 perplexity exceeds ten times the dense model is drawn in the shaded divergent region. MixLLM, MixQuant, and AMQ diverge on all three models, AutoScheme on two, and OWQ on one. The KronQ run on Qwen3-8B produces no artifact because its sequential pass aborts during Cholesky factorization even with increased damping. See the supplementary material for complete results.

Low-bit quantization amplifies errors in candidate construction and precision allocation. Layer-local proxies become less robust under large W2 perturbations, whereas PACE compares the predicted output damage of each candidate before global allocation.

Among methods that complete all three models, PACE ranks first or second in all nine comparisons. Its WikiText2 increases are 1.97, 1.04, and 3.77, with zero-shot averages of 67.1, 66.2, and 63.6. KronQ performs best on Llama-3.1-8B WikiText2 and zero-shot evaluation, while SFMP exceeds PACE by 0.5 points on Mistral-7B zero-shot accuracy. On Llama-3.1-8B, the completed methods span 19.1 zero-shot points under W2/W4, compared with only 0.9 points under W4/W8.

3.4 Structure of the Learned Allocation

Figure 5 examines the final W4/W8 allocations. Each cell in panel (a) is the parameter-weighted W8 fraction within one layer–projection pair, and panel (b) aggregates the same decisions over depth. A common projection-wise structure emerges across all three model families. PACE assigns 7.91, 7.99, and 8.00 average bits to v_{proj} on Qwen3-8B, Mistral-7B, and Llama-3.1-8B, respectively, but only 4.43, 4.29, and 4.45 bits to q_{proj}. This ordering has a functional reading. Perturbations to queries mildly distort the attention distribution, whereas the value pathway directly determines the content that attention writes into the residual stream. The gate projection is also consistently compressed, whereas the remaining attention and MLP projections receive intermediate precision. The global optimizer recovers this shared ordering without imposing projection-specific budgets.

The depth-wise pattern is less universal. Qwen3-8B and Mistral-7B allocate more W8 capacity toward later layers, whereas Llama-3.1-8B exhibits additional early- and middle-layer peaks. This contrast argues against a fixed rule that protects the same depths in every architecture: the projection-level tendency transfers, but its realization over depth remains model dependent.

Panel (c) tests whether this structure is an artifact of calibration noise on Qwen3-8B. Holding the 32 calibration sequences fixed and changing only the estimator seed yields pairwise W8-stripe Jaccard scores of 0.93–0.95 and layer–projection Spearman correlations of 0.97–0.99. Across three C4 subsets, the corresponding ranges are 0.84–0.87 and 0.92–0.95. Across three WikiText subsets, they are 0.82–0.91 and 0.90–0.97. Individual W8 locations therefore vary more than the aggregate profile, but the learned macrostructure remains stable. The plotted ranges are observed minima and maxima over three pairwise comparisons, not confidence intervals.

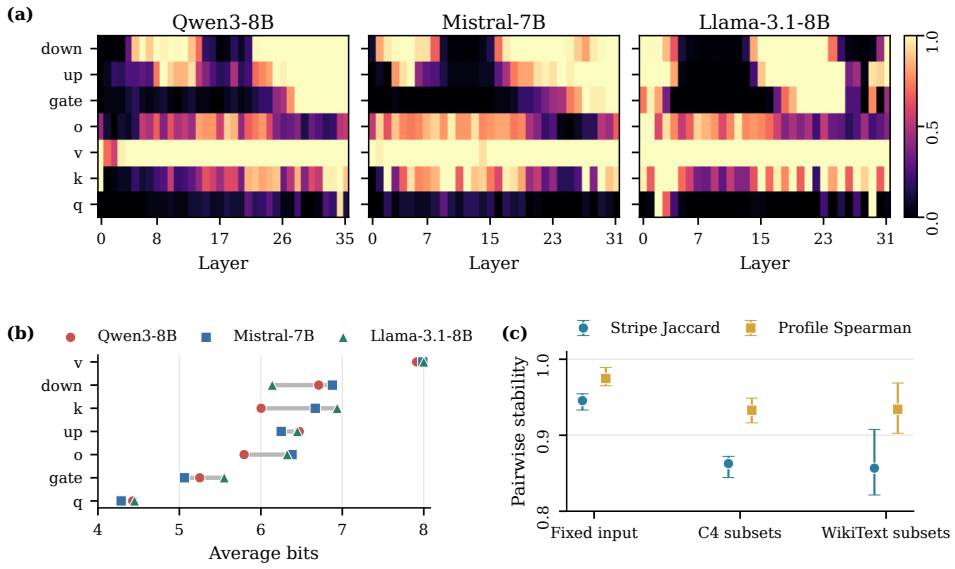

Figure 5: PACE allocation structure and stability. Panel (a) shows layer–projection W8 fractions, panel (b) compares projection-level average bitwidths across models, and panel (c) reports stability under different calibration conditions. Ranges denote observed minima and maxima.

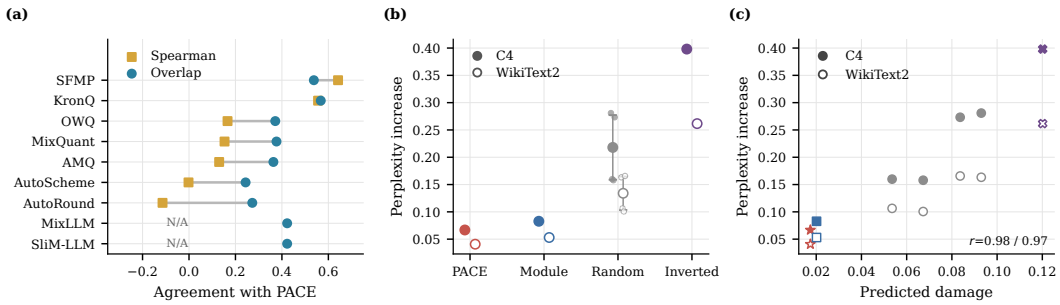

Figure 6: Allocation agreement and equal-budget interventions. Panel (a) compares allocation profiles across methods, while panels (b) and (c) measure the perplexity effects and predicted damage of controlled allocations on Qwen3-8B.

3.5 What Makes the Learned Allocation Effective?

Figure 6 separates structural agreement from causal evidence. Because the compared methods allocate precision at different native granularities, panel (a) first aggregates every realized allocation to a parameter-weighted layer–projection W8 profile. At this common resolution, PACE agrees most strongly with SFMP and KronQ: their mean Spearman correlations with PACE across the three models are 0.64 and 0.56, and their continuous allocation overlaps are 0.54 and 0.57. Other criteria agree less strongly. SliM-LLM and MixLLM produce a constant profile at this resolution, for which rank correlation is undefined. These results indicate partial agreement among independent sensitivity criteria, rather than a universal allocation shared by all methods.

We next intervene on the allocation while holding the model, calibration set, quantizer, and average six-bit W4/W8 budget fixed. The Module control optimizes the same predicted damage but restricts each complete projection to one bitwidth. Random permutes PACE's bit labels within equal-parameter stripe buckets over four seeds, preserving both the exact global budget and each bucket's W8 count. Inverted exchanges every W4 and W8 decision. As shown in panel (b), PACE obtains C4/WikiText2 perplexity increases of $0.067/0.041$, compared with $0.083/0.053$ for Module, $0.218/0.134$ for the four-run Random mean, and $0.398/0.262$ for Inverted. Hence, quality

depends on where the high-precision budget is placed, and the finer allocation resolution provides a measurable gain beyond module-level selection.

Finally, panel (c) connects the optimization objective to end-model quality. Across PACE, Module, four Random allocations, and Inverted, predicted damage has Pearson correlation 0.976 with C4 perplexity degradation and 0.967 with WikiText2 degradation. The matching monotonic trend explains why the learned ordering outperforms the equal-budget controls. This intervention is currently performed on Qwen3-8B. The cross-model allocation plots are descriptive, so we do not claim cross-model causality from this experiment.

3.6 Quantization Cost and Storage

Table 2 reports the mean and standard deviation of per-stage cost over the six mainline experiments. Saliency computation requires one backward pass per calibration sequence and finishes in 6.4 seconds over the 32 sequences. Candidate scoring, which refines a scale and evaluates the quantization loss for every group–bit pair, dominates at 47.2 seconds. Exact allocation reduces to a single global sort and completes in under one second. The algorithmic core therefore finishes in about 54 seconds, or 59 seconds including model loading. Dense QDQ export takes another 68.5 seconds, bringing the complete evaluation pipeline to about 128 seconds. It materializes a dense evaluation overlay consisting of BF16 dequantized weights and per-group scale and zero-point maps, and is not part of the allocation algorithm. Peak GPU memory of 53.9 GB occurs during saliency estimation, where the backward pass holds the BF16 model together with its activations. Candidate scoring, allocation, and export remain near 41 GB. Costs vary little across models and budgets, as both budgets enumerate the same number of candidate bitwidths per group.

Table 2: Time and peak GPU memory of PACE on an A100 80GB GPU. Values are reported as mean (standard deviation) across the six main experiments.

	Model load	Saliency (32 seq.)	Candidate scoring	Exact allocation	Dense QDQ export
Time (s)	4.5 (0.7)	6.4 (0.4)	47.2 (1.0)	0.9 (0.9)	68.5 (1.7)
Memory (GB)	14.6 (0.8)	53.9 (0.8)	40.9 (0.8)	40.5 (0.8)	40.9 (0.7)

On the storage side, the dense evaluation overlay has an effective width of 16.00 bits and provides no physical compression. Every adapted baseline writes the same overlay. The packed PACE representation stores quantized weights with scales, masks, and offsets, including all metadata. At the 6.0-bit budget, it reduces target-weight storage by $2.60\times$ relative to BF16, with an effective width of 6.15 bits per target weight. All 252 packed modules exactly recover their evaluated QDQ weights.

4 Conclusion

We presented PACE, a mixed-precision post-training quantization method that allocates precision according to the predicted output damage of concrete quantization candidates. PACE estimates reusable output sensitivity through shared calibration, independently optimizes each group for every candidate width, and performs exact global allocation under the prescribed precision budget.

Across Qwen3-8B, Mistral-7B, and Llama-3.1-8B, PACE preserves near-dense language-modeling quality under average-W6 W4/W8 quantization and remains effective under average-W3 W2/W4. Its allocations outperform equal-budget random, inverted, and module-level controls and remain stable across calibration conditions. Calibration, scoring, and allocation finish in about 54 seconds, while the packed representations provide approximately $2.6\times$ and more than $4.6\times$ target-weight storage compression under W6 and W3.

Future work will extend PACE to larger models, broader candidate precision sets, and hardware-aware objectives that jointly account for quality, memory traffic, and latency. Fused kernels for group-wise mixed W4/W8 execution can translate the current physical storage reduction into end-to-end inference acceleration. Richer structured curvature approximations may further improve allocation in aggressive low-bit regimes while retaining the efficiency of shared calibration.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, code development, and figure preparation. All AI-assisted text, code, analyses, and artifacts were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Section 2 specifies the PACE objective, candidate construction, and global allocation procedure. Section 3 reports the models, datasets, calibration settings, evaluation metrics, and computational costs used in our experiments. Additional low-bit results are provided in the supplementary material.

REFERENCES

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020.

Zhaowei Cai and Nuno Vasconcelos. Rethinking differentiable search for mixed-precision neural networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2346–2355. IEEE, 2020.

Weihan Chen, Peisong Wang, and Jian Cheng. Towards mixed-precision quantization of neural networks via constrained optimization. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5330–5339. IEEE, 2021.

Zihan Chen, Bike Xie, Jundong Li, and Cong Shen. Adapting to any bit-width: Channel-wise mixed-precision quantization for LLMs. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=1t6sEhdLxf.

Wenhua Cheng, Weiwei Zhang, Haihao Shen, Yiyang Cai, Xin He, Kaokao Lv, and Yi Liu. Optimize weight rounding via signed gradient descent for the quantization of llms. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 11332–11350, 2024.

Wenhua Cheng, Weiwei Zhang, Heng Guo, Haihao Shen, and Zaner Ma. SignRoundV2: Toward closing the performance gap in extremely low-bit post-training quantization for LLMs. *arXiv preprint arXiv:2512.04746*, 2025.

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 *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 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. *arXiv preprint arXiv:1803.05457*, 2018.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 5733–5761, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/1787533e171dcc8549cc2eb5a4840eec-Paper-Conference.pdf.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. HAWQ: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 293–302, 2019.

Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. HAWQ-V2: Hessian aware trace-weighted quantization of neural networks. *Advances in Neural Information Processing Systems*, 33:18518–18529, 2020.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Qinshuo Liu, Xianglong Liu, Luca Benini, Michele Magno, Shiming Zhang, and Xiaojuan Qi. SliM-LLM: Salience-driven mixed-precision quantization for large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 25672–25692. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/huang25aa.html.

Xijie Huang, Zhiqiang Shen, Shichao Li, Zechun Liu, Xianghong Hu, Jeffry Wicaksana, Eric Xing, and Kwang-Ting Cheng. SDQ: Stochastic differentiable quantization with mixed precision. In *International Conference on Machine Learning*, pp. 9295–9309. PMLR, 2022.

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, 2023. URL https://arxiv.org/abs/2310.06825.

Sehoon Kim, Coleman Richard Charles Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. SqueezeLLM: Dense-and-sparse quantization. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 23901–23923. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/kim24f.html.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. OWQ: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13355–13364, 2024.

Donghyun Lee, Yuhang Li, Ruokai Yin, and Priyadarshini Panda. KronQ: LLM quantization via Kronecker-factored Hessian. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=A3YwGs0WWT.

Sangjun Lee, Seung-taek Woo, Jun-gyu Jin, Changhun Lee, and Eunhyeok Park. Amq: Enabling automl for mixed-precision weight-only quantization of large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35532–35550, 2025.

Shiyao Li, Xuefei Ning, Ke Hong, Tengxuan Liu, Luning Wang, Xiuhong Li, Kai Zhong, Guohao Dai, Huazhong Yang, and Yu Wang. LLM-MQ: Mixed-precision quantization for efficient LLM deployment. In *NeurIPS 2023 Efficient Natural Language and Speech Processing Workshop*, 2023. URL https://neurips2023-enlsp.github.io/papers/paper_4.pdf.

Xinlin Li, Timothy Chou, Josh Fromm, Zichang Liu, Yunjie Pan, and Christina Fragouli. ScaleBITS: Scalable bitwidth search for hardware-aligned mixed-precision LLMs. *arXiv preprint arXiv:2602.17698*, 2026.

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 *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=POWv6hDd9XH.

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. *Proceedings of Machine Learning and Systems*, 6:87–100, 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 *International Conference on Learning Representations*, volume 2025, pp. 92009–92032, 2025.

Yuexiao Ma, Taisong Jin, Xiawu Zheng, Yan Wang, Huixia Li, Yongjian Wu, Guannan Jiang, Wei Zhang, and Rongrong Ji. OMPQ: Orthogonal mixed precision quantization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 9029–9037, 2023.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Ashitabh Misra, Madhav Agrawal, Arham Jain, and Tarek Abdelzaher. Mixquant: Adaptive mixed-precision quantization for large language models. *arXiv preprint arXiv:2607.23047*, 2026.

Xin Nie, Haicheng Zhang, Liang Dong, Beining Feng, Jinhong Weng, and Guiling Sun. SFMP: Fine-grained, hardware-friendly and search-free mixed-precision quantization for large language models. *arXiv preprint arXiv:2602.01027*, 2026.

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. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Utkarsh Saxena, Sayeh Sharify, Kaushik Roy, and Xin Wang. ResQ: Mixed-precision quantization of large language models with low-rank residuals. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 53095–53114. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/saxena25b.html.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. QuIP#: Even better LLM quantization with hadamard incoherence and lattice codebooks. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48630–48656. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/tseng24a.html.

Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-aware automated quantization with mixed precision. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8604–8612. IEEE, 2019.

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

Zhangyang Yao, Haiyan Zhao, Haoyu Wang, and Xu Han. GAMMA: Global bit allocation for mixed-precision models under arbitrary budgets. *arXiv preprint arXiv:2605.18475v2*, 2026.

Ben Zandonati, Adrian Alan Pol, Maurizio Pierini, Olya Sirkin, and Tal Kopetz. FIT: A metric for model sensitivity. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PDG4-Y3aboN.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Zhen Zheng, Xiaonan Song, and Chuanjie Liu. MixLLM: LLM quantization with global mixed-precision between output-features and highly-efficient system design. *Proceedings of Machine Learning and Systems*, 8:1771–1785, 2026.