Budget-Aware LLM Quantization and Low-Rank Correction via Information-Guided Subspace Matrices

Sinuo Fan , Yingjie Lao
Tufts University
sinuo.fan@tufts.edu, yingjie.lao@tufts.edu

Abstract

Compression techniques such as quantization and low-rank approximation enable large language models (LLMs) to run on current edge hardware with limited computing power, but the key challenge lies in balancing the allocation of precision and low rank within a fixed memory limit. We propose a training-free framework, BALANCER, which achieves global budget allocation for mixed-precision quantization and low-rank correction through information-guided subspace matrices. BALANCER collects a small set of calibration signals—activation statistics, reference-gradient magnitudes, and a diagonal curvature proxy—and converts them into information-guided subspace matrices. Instead of relying on coarse-grained layer-wise importance allocation, BALANCER transforms these signals into gain curves for each weight in the matrix singular subspace, allowing a principled greedy allocator to distribute compression bits and ranks across the entire model. On the LLaMA-3.1-8B model, with an average of 3.6 bits per parameter, BALANCER only increases the perplexity from 6.63 to 6.78 (+0.15), while reducing memory from 16 GB to 3.9 GB, and enabling the model to be deployed on edge GPUs such as the Jetson Orin Nano (8 GB).

1 Introduction

Current large language models (LLMs) demonstrate exceptional performance in language and various tasks, but their inference process typically requires ample memory bandwidth, sufficient storage space, and optimized kernels. In contrast, edge devices and some general-purpose hardware have very strict limitations on memory and resources, making model compression after training a necessary step for practical deployment [Jaiswal *et al.*, 2024]. Research on system deployment further emphasizes that memory and scheduling are major bottlenecks in the deployment of current large models [Dao *et al.*, 2022], leading researchers to investigate whether high-accuracy compression methods can be achieved under specific resource constraints [Clements and Lao, 2024b].

At the moment, there has been a lot of advancement in model compression methods, including traditional ones like pruning, quantization, and low-rank decomposition. Pruning can remove redundancy [Frantar and Alistarh, 2023]; quantization can achieve relatively low-precision inference through calibration and optimization [Frantar *et al.*, 2022; Dettmers *et al.*, 2022b; Lin *et al.*, 2024; Esser *et al.*, 2020]; and low-rank methods reduce the number of parameters in the matrix structures of neural networks and Transformer models to lower hardware computation requirements [Aghajanyan *et al.*, 2021; Trockman and Kolter, 2021; Hu *et al.*, 2022]. But when there are billions of parameters, compression is naturally heterogeneous: some layers and subspaces stay strong even at low precision, while others lose a lot of performance when compressed [Frantar and Alistarh, 2023; Dettmers *et al.*, 2022a; Xiao *et al.*, 2023]. This lack of uniformity makes it impossible to compress layers and models evenly when there are stringent hardware budget limits, especially when trying to keep the quality of the text (e.g., perplexity) at a certain compression or quantization bit rate.

This paper adopts a global planning perspective: with a fixed compression space, where can adding one bit (or one additional low-rank capacity unit) bring the greatest improvement in model quality? To make this perspective workable, the following two points need to be addressed: (i) the importance needs to be expressed in a way that clearly conveys "what needs to be retained"; and (ii) the rating representation needs to be consistent with the quantification and low-rank correction methods to be applied. For matrix weights, singular value vectors provide a natural coordinate system for model weights, since degradation or recovery is usually concentrated in a small set of vulnerable directions in the singular subspace of the current matrix [Meng *et al.*, 2025].

In this paper, we introduce BALANCER[1], a post-training compression framework that treats compression as a global allocation problem under an explicit storage budget. Figure 1 demonstrates the high-level overview. Starting from transformer projection matrices, BALANCER collects calibration signals from activation statistics, reference gradients, and a diagonal curvature surrogate. These signals are then converted into *information-guided subspace matrices* that define

[1]Code and experiment scripts are available at https://github.com/sinuofan/IJCAI2026-BALANCER.

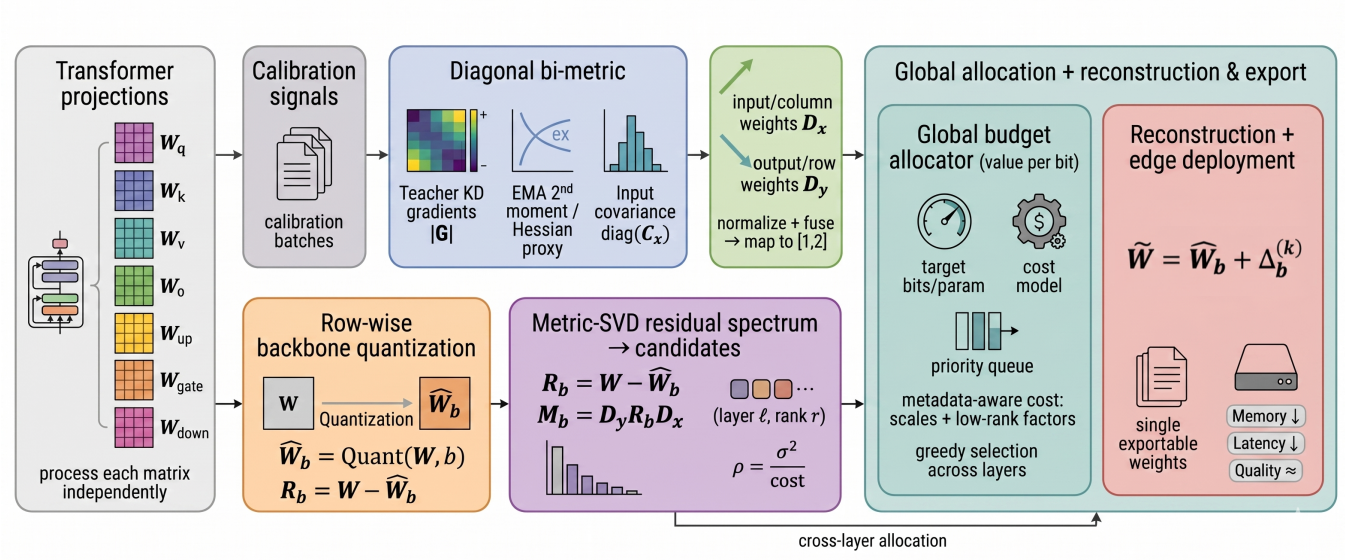

Figure 1: Overview of BALANCER. Calibration signals collected on a small calibration set are converted into information-guided subspace matrices for weighting residual directions in each projection. BALANCER then builds gain curves over singular directions and performs global budget allocation for mixed-precision quantization and low-rank correction under a target bits-per-parameter budget, before exporting a single set of reconstructed weights for edge deployment.

a consistent coordinate-wise weighting for analyzing each projection in its singular subspace. In this subspace, BALANCER converts the signal matrix into *gain curves* over singular directions, which quantify the marginal quality recovery expected from preserving additional directions. A principled greedy allocator then distributes bits and ranks *globally* across the model under an explicit bits-per-parameter budget, accounting for both weights and associated metadata. Finally, BALANCER exports a single set of reconstructed weights, enabling standard inference without fine-tuning or custom kernels.

Our main contributions are summarized below:

- *Budget-aware post-training formulation.* We cast compression as a global allocation problem under a unified storage accounting that includes quantized integers and metadata, enabling direct control of average bits-per-parameter.
- *Information-guided subspace matrices.* We construct a signal matrix from activation statistics, gradient significance, and a Hessian diagonal curvature surrogate, and transform it into information-guided subspace matrices for direction ranking in the matrix singular subspace.
- *Global allocation of bits and ranks.* We introduce a greedy allocator that distributes compression bits and low-rank capacity across the entire model to maximize marginal quality recovery under a target budget.
- *Evaluation and edge deployment.* We evaluate BALANCER on standard benchmarks and demonstrate deployability on Jetson Orin Nano (8 GB), achieving 0.78–0.92 s first-token latency and 1.1–1.3 tok/s steady-state throughput with 3.5–3.8 GB resident memory.

2 Related Work

Post-training quantization (PTQ) and calibration strategies. Transformer PTQ has progressed from calibration heuristics to structure-aware, low-bit regimes. Representative advances include INT8 with outlier handling [Dettmers *et al.*, 2022a], second-order layerwise optimization [Frantar *et al.*, 2022], activation-aware channel protection [Lin *et al.*, 2024], activation smoothing for joint W + A quantization [Xiao *et al.*, 2023], and efficient group-wise pipelines [Yao *et al.*, 2022]. Recent works further push sub-4-bit operation via codebooks, orthogonalization, or lattice formulations [Tseng *et al.*, 2024; Egiazarian *et al.*, 2024; Eliseev and Mazur, 2023; Ashkboos *et al.*, 2024; Zhang *et al.*, 2024]. Beyond improving a single fixed bit-width, several concurrent directions target *budget-adaptive* or *variable-memory* deployment: BitStack enables any-size compression under changing memory envelopes [Wang *et al.*, 2025], Any-Precision LLM targets low-cost deployment of multiple precision variants [Park *et al.*, 2024], and Matryoshka Quantization trains nested quantizers that support multiple bit-widths from shared parameters [Ahemad, 2025]. These works motivate treating PTQ as a *budgeted allocation* problem rather than a per-layer independent choice [Chu *et al.*, 2025; Hoang *et al.*, 2026; Yao *et al.*, 2024].

Hybrid quantization with structured residuals. To tighten the quality–compression tradeoff under stringent budgets, hybrid schemes keep a quantized backbone while adding small structured corrections where they are beneficial. A representative example is CALDERA, which couples low-bit quantization with low-rank residual structure for post-training compression [Saha *et al.*, 2024; Lee *et al.*, 2025]. Related lines also study the co-design of quantization and

adapter/low-rank components [Li *et al.*, 2024; Lawton *et al.*, 2024], serving-oriented low-bit designs [Zhao *et al.*, 2024], and elastic precision scheduling [Lee *et al.*, 2023; Kwon *et al.*, 2022]. Compared with pure PTQ, these methods explicitly reserve extra capacity for *structured residuals*; compared with full mixed-precision, they preserve a simple backbone and concentrate extra parameters into compact correction paths. In particular, LoftQ also leverages low-rank structure around quantization, but is primarily designed to be LoRA-fine-tuning-aware, whereas our setting focuses on training-free post-hoc allocation under an explicit storage budget [Li *et al.*, 2024].

Direction-aware protection and global allocation. A growing body of evidence indicates that compression sensitivity is *anisotropic within layers*. Hessian-guided sparsification [Frantar and Alistarh, 2023], magnitude/activation proxies [Sun *et al.*, 2024], and block-wise second-order reconstruction [Li *et al.*, 2021] all reveal direction-dependent sensitivity. At a coarser granularity, mixed-precision schedulers formalize bit-width assignment as a budgeted optimization so that precision is redistributed toward components with higher predicted gain per additional bit [Lee *et al.*, 2023]. Hybrid schemes further combine a low-bit backbone with structured residuals, using spectral cues to decide where limited corrective capacity is applied [Saha *et al.*, 2024]. Recent budget-adaptive methods (e.g., BitStack [Wang *et al.*, 2025], Any-Precision LLM [Park *et al.*, 2024], Matryoshka Quantization [Ahemad, 2025]) reinforce the same theme: *global allocation* under an explicit memory/bit budget is central for practical deployment.

3 Method

Algorithm 1 summarizes the end-to-end pipeline. All decisions are computed deterministically on a small calibration set, the subspace matrices, and a single bpp constraint (Eq. 3), with no hyperparameter tuning.

3.1 Setup

Notation. We compress a pretrained LLM with linear projection weights $\{W_\ell\}_{\ell=1}^{L}$. For module ℓ, $W_\ell \in \mathbb{R}^{m_\ell \times n_\ell}$ maps an input activation $x \in \mathbb{R}^{n_\ell}$ to an output $y \in \mathbb{R}^{m_\ell}$ via $y = W_\ell x$. Given a chosen bit-width $b_\ell \in \mathcal{B}$ and rank $r_\ell \in \{0, 1, \dots, r_{\max}\}$, the exported weight is

$$\widetilde{W}_\ell \triangleq \widehat{W}_{\ell,b_\ell} + \Delta_\ell, \tag{1}$$

where $\widehat{W}_{\ell,b_\ell}$ is the row-wise quantized reconstruction defined in Sec. 3.2.

Low-rank. We store the low-rank term in factored form

$$\Delta_\ell \triangleq L_\ell R_\ell^\top, \qquad L_\ell \in \mathbb{R}^{m_\ell \times r_\ell},\ R_\ell \in \mathbb{R}^{n_\ell \times r_\ell}. \tag{2}$$

Budget accounting. Let $N \triangleq \sum_{\ell=1}^{L} m_\ell n_\ell$ be the total number of weight entries across all compressed matrices. Let B be the target bits-per-parameter (bpp), so the total budget is BN bits. For module ℓ, the integer storage is $m_\ell n_\ell b_\ell$. We store one FP16 scale per row, so $C_{\text{meta}}(m_\ell, b_\ell) = m_\ell b_s$ with $b_s = 16$. The low-rank factors are stored at b_{lr} bits, so

Algorithm 1 BALANCER: global allocation with information-guided subspace matrices

Require: $\{W_\ell\}_{\ell=1}^{L}$, calibration set $\mathcal{D}$, budget B, bit set $\mathcal{B}$, rank limit $r_{\max}$; optional $\{G_\ell\}_{\ell=1}^{L}$, optional $\{h_\ell\}_{\ell=1}^{L}$
Ensure: $\{\widetilde{W}_\ell\}_{\ell=1}^{L}$
for $\ell = 1$ to L **do**
  $(S_\ell^{(x)}, S_\ell^{(y)}) \leftarrow \textsc{Subspace}(W_\ell, \mathcal{D}; G_\ell, h_\ell)$ (Alg. 2)
  for each $b \in \mathcal{B}$ **do**
    $\widehat{W}_{\ell,b} \leftarrow \textsc{Quant}(W_\ell, b)$ (Sec. 3.2)
    $R_{\ell,b} \leftarrow W_\ell - \widehat{W}_{\ell,b}$
    $e_\ell(b) \leftarrow \mathcal{E}(R_{\ell,b}; S_\ell^{(x)}, S_\ell^{(y)})$ (Eq. 13)
    $\{g_{\ell,b}(t)\}_{t=1}^{r_{\max}} \leftarrow \textsc{Gain}(R_{\ell,b}, r_{\max}; S_\ell^{(x)}, S_\ell^{(y)})$ (Sec. 3.4)
  end for
end for
$\{(b_\ell, r_\ell)\}_{\ell=1}^{L} \leftarrow \textsc{Allocate}(\{e_\ell\}, \{g_{\ell,\cdot}\}, B)$ (Alg. 3)
for $\ell = 1$ to L **do**
  $\Delta_\ell \leftarrow \mathrm{LR}(R_{\ell,b_\ell}, r_\ell; S_\ell^{(x)}, S_\ell^{(y)})$ (Sec. 3.4)
  $\widetilde{W}_\ell \leftarrow \widehat{W}_{\ell,b_\ell} + \Delta_\ell$
end for
return $\{\widetilde{W}_\ell\}$

$C_{\text{lr}}(m_\ell, n_\ell, r_\ell) = (m_\ell + n_\ell) r_\ell b_{\text{lr}}$. In all main experiments, $b_{\text{lr}} = 4$. We enforce

$$\sum_{\ell=1}^{L} (m_\ell n_\ell b_\ell + C_{\text{meta}}(m_\ell, b_\ell) + C_{\text{lr}}(m_\ell, n_\ell, r_\ell)) \le BN. \tag{3}$$

Thus C_{meta} accounts for per-row scales, and C_{lr} accounts for the stored factors L_ℓ and R_ℓ.

3.2 Row-wise symmetric quantization

For $W \in \mathbb{R}^{m \times n}$ and bit-width $b \in \mathcal{B}$, define the signed integer range

$$q_{\max}(b) \triangleq 2^{b-1} - 1, \qquad q_{\min}(b) \triangleq -q_{\max}(b). \tag{4}$$

For each row $i \in \{1, \dots, m\}$, define the scale

$$s_i \triangleq \frac{\max_{j \in \{1,\dots,n\}} |W_{ij}|}{q_{\max}(b) + \varepsilon_q}, \tag{5}$$

with a fixed $\varepsilon_q > 0$ to avoid division by zero for all-zero rows. The quantized integers and reconstructed weights are

$$Q_{ij} \triangleq \mathrm{clip}\left(\left\lfloor \frac{W_{ij}}{s_i} \right\rceil,\ q_{\min}(b), q_{\max}(b) \right), \qquad \widehat{W}_{ij} \triangleq s_i\, Q_{ij}, \tag{6}$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer and $\mathrm{clip}(\cdot)$ clips elementwise.

3.3 Information-guided subspace matrices

Let $\mathcal{D}$ be a calibration set and $x \in \mathbb{R}^n$ be the input activation observed on $\mathcal{D}$. We construct two nonnegative diagonal matrices D_x and D_y (equivalently, their diagonal entries $S^{(x)}$ and $S^{(y)}$), referred to as information-guided subspace matrices. When available, we denote by G_ℓ the reference-gradient

matrix associated with module W_ℓ, and by h_ℓ the corresponding nonnegative diagonal curvature surrogate. We construct *information-guided subspace matrices* by aggregating three complementary *subspace signals* that act on the input/output coordinates: (i) an *activation subspace* view from the second moment of x, (ii) a *reference-gradient subspace* view from G, and (iii) a *curvature subspace* view from the diagonal surrogate h.

(1) Activation RMS (root second moment).

$$c_j \triangleq \mathbb{E}_{x\sim\mathcal{D}}\left[x_j^2\right], \qquad j = 1, \ldots, n. \tag{7}$$

(2) Reference-gradient magnitude. When a reference model is available, we collect a gradient matrix $G \in \mathbb{R}^{m\times n}$ for the same projection w.r.t. a reference objective (e.g., a distillation-style loss), and define

$$g_j^{(x)} \triangleq \sum_{i=1}^{m} |G_{ij}|, \qquad g_i^{(y)} \triangleq \sum_{j=1}^{n} |G_{ij}|. \tag{8}$$

(3) Diagonal curvature surrogate. For projection W_ℓ, we use $h_{\ell,i} = \mathbb{E}_{(x,y)\sim\mathcal{D}} \sum_{j=1}^{n_\ell} (\partial\mathcal{L}_{\text{ref}}/\partial W_{\ell,ij})^2$ as the output-side curvature surrogate.

Information-guided subspace matrices. For any nonnegative vector $u \in \mathbb{R}^d_{\geq 0}$, define min–max normalization

$$\mathrm{Norm}(u) \triangleq \frac{u - \min(u)}{\max(u) - \min(u) + \varepsilon}, \tag{9}$$

where $\varepsilon > 0$ is a fixed numerical constant. We combine available components by *summation* followed by $\mathrm{Norm}(\cdot)$, hence no hyperparameters are introduced.

Let $\mathbb{I}_G \in \{0, 1\}$ indicate whether G is available, and $\mathbb{I}_h \in \{0, 1\}$ indicate whether h is available. We define

$$S^{(x)} \triangleq \mathbf{1}_n + \mathrm{Norm}\Big(\mathrm{Norm}(\sqrt{c}) + \mathbb{I}_G\ \mathrm{Norm}(g^{(x)})\Big), \tag{10}$$

$$S^{(y)} \triangleq \mathbf{1}_m + \mathrm{Norm}\Big(\mathbb{I}_G\ \mathrm{Norm}(g^{(y)}) + \mathbb{I}_h\ \mathrm{Norm}(h)\Big). \tag{11}$$

Min-max normalization aligns signal scales, summation avoids tunable fusion weights, and the $\mathbf{1}+$ shift makes $S^{(x)}$ and $S^{(y)}$ positive bounded weights. Algorithm 2 summarizes the construction of $(S^{(x)}, S^{(y)})$.

3.4 Subspace gain curves

Subspace weighting. Given $S^{(x)} \in \mathbb{R}^n_{>0}$ and $S^{(y)} \in \mathbb{R}^m_{>0}$, define diagonal matrices

$$D_x \triangleq \mathrm{Diag}(S^{(x)}) \in \mathbb{R}^{n\times n}, \qquad D_y \triangleq \mathrm{Diag}(S^{(y)}) \in \mathbb{R}^{m\times m}. \tag{12}$$

For any residual $R \in \mathbb{R}^{m\times n}$, define

$$\mathcal{E}(R;\, S^{(x)}, S^{(y)}) \triangleq \|D_y\, R\, D_x\|_F^2. \tag{13}$$

This weighting emphasizes residual components aligned with informative input/output directions encoded by $(S^{(x)}, S^{(y)})$. Intuitively, it penalizes distortion more heavily along directions that are deemed important by calibration-time signals, making the subsequent gain curves direction-aware under a unified criterion. Equivalently, $\mathcal{E}(R)$ measures reconstruction error after re-scaling columns and rows according to calibration signal strengths, so directions deemed more important incur a larger penalty when distorted.

Algorithm 2 SUBSPACE: information-guided subspace matrices construction

Require: $W \in \mathbb{R}^{m\times n}$, calibration set $\mathcal{D}$; optional $G \in \mathbb{R}^{m\times n}$; optional $h \in \mathbb{R}^m$
Ensure: $S^{(x)} \in \mathbb{R}^n$, $S^{(y)} \in \mathbb{R}^m$

```
Collect c on D                                                    (Eq. 7)
I_G ← 1 if G exists else 0;   I_h ← 1 if h exists else 0
if I_G = 1 then
    Compute g^(x), g^(y) from G                                   (Eq. 8)
end if
S^(x) ← 1_n + Norm(Norm(√c) + I_G Norm(g^(x)))
S^(y) ← 1_m + Norm(I_G Norm(g^(y)) + I_h Norm(h))
return S^(x), S^(y)
```

Scaled SVD. Fix $b \in \mathcal{B}$ and let $R_b \triangleq W - \widehat{W}_b$. Define the weighted residual

$$M_b \triangleq D_y\, R_b\, D_x. \tag{14}$$

Let $M_b = U\Sigma V^\top$ denote the (standard) singular value decomposition (SVD), where $U = [u_1, \ldots, u_m]$ and $V = [v_1, \ldots, v_n]$. Let the singular values satisfy $\sigma_b(1) \geq \sigma_b(2) \geq \cdots$. For rank r, define the truncated approximation

$$M_b^{(r)} \triangleq \sum_{t=1}^{r} \sigma_b(t)\, u_t v_t^\top. \tag{15}$$

We map it back to a low-rank correction for the original residual:

$$\Delta_b^{(r)} \triangleq D_y^{-1} M_b^{(r)} D_x^{-1}. \tag{16}$$

In practice we store $\Delta_b^{(r)}$ in factored form by letting

$$L_b^{(r)} \triangleq D_y^{-1} U_r \Sigma_r^{1/2}, \qquad R_b^{(r)} \triangleq D_x^{-1} V_r \Sigma_r^{1/2}, \tag{17}$$

where $U_r = [u_1, \ldots, u_r]$, $V_r = [v_1, \ldots, v_r]$, and $\Sigma_r = \mathrm{Diag}(\sigma_b(1), \ldots, \sigma_b(r))$. Then $\Delta_b^{(r)} = L_b^{(r)}(R_b^{(r)})^\top$, matching Eq. 2. Allocating the t-th singular component reduces the error in Eq. 13 by

$$g_b(t) \triangleq \sigma_b(t)^2, \qquad t = 1, \ldots, r_{\max}, \tag{18}$$

which yields the gain curve used by the allocator. No additional coefficients are introduced beyond $(S^{(x)}, S^{(y)})$.

3.5 Budget-aware global greedy allocation

We choose $\{(b_\ell, r_\ell)\}_{\ell=1}^L$ by a global greedy allocator that maximizes reconstruction value per additional bit.

Coupling bit-width and rank. The correction is always computed on the quantization residual $R_{\ell,b} = W_\ell - \widehat{W}_{\ell,b}$, hence the gain curve $g_{\ell,b}(t)$ is specific to the current bit-width b. To keep allocation well-defined and training-free, we restrict bit-width changes to be applied at $r_\ell = 0$ and reset $r_\ell \leftarrow 0$ whenever b_ℓ changes. After a bit-width update, subsequent rank actions for that module are taken from the precomputed gain curve of the new bit-width. A rank-one correction costs $(m_\ell + n_\ell) b_{\text{lr}}$ bits, whereas increasing the backbone by one bit costs $m_\ell n_\ell$ bits, so the allocator compares these actions by the same density $\rho = \Delta e / \Delta c$.

Value terms. For module ℓ and bit-width b, let

$$e_\ell(b) \triangleq \mathcal{E}(W_\ell - \widehat{W}_{\ell,b}; S_\ell^{(x)}, S_\ell^{(y)}). \tag{19}$$

For a rank increment at fixed b, the marginal value is $g_{\ell,b}(t)$ from Eq. 18.

Rank-r weighted residual error. For module ℓ at bit-width b and rank r, let

$$E_\ell(b,r) \triangleq \min_{\mathrm{rank}(\Delta)\le r} \mathcal{E}(R_{\ell,b} - \Delta;\, S_\ell^{(x)}, S_\ell^{(y)}). \tag{20}$$

With the weighted SVD in Sec. 3.4, this admits the closed form

$$E_\ell(b,r) = e_\ell(b) - \sum_{t=1}^{r} g_{\ell,b}(t), \tag{21}$$

where $e_\ell(b)$ is Eq. 19 and $g_{\ell,b}(t) = \sigma_{\ell,b}(t)^2$ is Eq. 18.

Bit costs. For a bit-width change $b \to b'$ on module ℓ, we use

$$\Delta\mathrm{bits}_{\mathrm{q}} \triangleq m_\ell n_\ell (b' - b) + \Big(C_{\mathrm{meta}}(m_\ell, b') - C_{\mathrm{meta}}(m_\ell, b)\Big). \tag{22}$$

For a rank increment $r \to r+1$ on module ℓ, we use

$$\Delta\mathrm{bits}_{\mathrm{lr}} \triangleq C_{\mathrm{lr}}(m_\ell, n_\ell, r+1) - C_{\mathrm{lr}}(m_\ell, n_\ell, r). \tag{23}$$

These costs are consistent with the single constraint in Eq. 3.

Actions and densities. We maintain a state (b_ℓ, r_ℓ) for each module and consider two action types.

Total module cost. For module ℓ, define the total bit cost under choice (b, r) as

$$c_\ell(b,r) \triangleq m_\ell n_\ell b + C_{\mathrm{meta}}(m_\ell, b) + C_{\mathrm{lr}}(m_\ell, n_\ell, r). \tag{24}$$

For brevity, write $c_\ell^{b,r} \triangleq c_\ell(b,r)$ and $E_\ell^{b,r} \triangleq E_\ell(b,r)$.

We define the error reduction Δe and bit cost Δc for each action as

$$\textbf{Rank:}\ (b_\ell, r_\ell) \to (b_\ell, r_\ell{+}1), \quad \Delta e = g_{\ell,b_\ell}(r_\ell{+}1), \qquad \Delta c = c_\ell^{b_\ell, r_\ell+1} - c_\ell^{b_\ell, r_\ell}. \tag{25a}$$

$$\textbf{Bit:}\ (b_\ell, r_\ell) \to (b'_\ell, 0),\ \ b'_\ell > b_\ell, \quad \Delta e = E_\ell^{b_\ell, r_\ell} - E_\ell^{b'_\ell, 0}, \qquad \Delta c = c_\ell^{b'_\ell, 0} - c_\ell^{b_\ell, r_\ell}. \tag{25b}$$

Each candidate action is scored by density $\rho = \Delta e / \Delta c$, and we greedily apply the best feasible action under the global budget. Algorithm 3 provides the full allocation routine.

Algorithm 3 ALLOCATE: budget-aware global greedy allocation

Require: Precomputed $\{e_\ell(b)\}$, $\{g_{\ell,b}(t)\}$; costs $C_{\mathrm{meta}}, C_{\mathrm{lr}}$; budget B; bit set $\mathcal{B}$; total parameters N
Ensure: Choices $\{(b_\ell, r_\ell)\}_{\ell=1}^{L}$

$b_\ell \leftarrow \min(\mathcal{B}),\ \ r_\ell \leftarrow 0\ \ \forall \ell$
bits $\leftarrow \sum_\ell \big(m_\ell n_\ell b_\ell + C_{\mathrm{meta}}(m_\ell, b_\ell)\big)$ {initial cost}
Initialize max-heap $\mathcal{H}$
{Each heap element is an action $a = (\ell, \mathsf{type})$ with density $\rho(a) = \Delta e / \Delta c$ computed from Eq. 25.}
{PUSHRANK/PUSHBIT insert the next feasible rank/bit action for module ℓ; REFRESH recomputes actions for the updated module.}
for $\ell = 1$ to L **do**
  PUSHRANK(ℓ); PUSHBIT(ℓ)
end for
while $\mathcal{H}$ not empty **do**
  Pop action a with largest density ρ
  if bits $+ \Delta c(a) > BN$ **then**
    continue
  end if
  Apply a; bits $\leftarrow$ bits $+ \Delta c(a)$
  REFRESH($\ell(a)$) {recompute next actions for affected module}
end while
return $\{(b_\ell, r_\ell)\}_{\ell=1}^{L}$

4 Experiment

4.1 Experimental Setup

We evaluate BALANCER to validate its effectiveness and practicality. We report (i) main results at matched budgets versus state-of-the-art baselines, (ii) ablation studies of key design choices, (iii) flexibility under mixed-precision schedules, and (iv) deployability on resource-constrained hardware. Unless otherwise stated, "mean±SD" reflects variability over different calibration-set samplings (and corresponding allocation/quantization outcomes) with the same evaluation protocol.

Models and Datasets. Our primary evaluations are conducted on the LLaMA model family, specifically LLaMA-2-7B and LLaMA-3.1-8B. To demonstrate the generalizability of our method across different architectural families, we also verify the same trend on Mistral-7B, where BALANCER reduces WikiText-2 PPL from 5.58 (RTN Q4) to 5.35 at 4.0 bits. Perplexity (PPL) is the primary metric for language modeling quality, reported on WikiText-2 and C4.

Evaluation Metrics. We report "Bits" as the left-hand side of Eq. 3 divided by N, and CR as $16/\mathrm{Bits}$ relative to FP16, and runtime performance including steady-state throughput ("Tok/s"), first-token latency ("Lat"), and resident VRAM.

Baselines. We compare to RTN (uniform round-to-nearest), GPTQ [Frantar *et al.*, 2022], AWQ [Lin *et al.*, 2024], CALDERA [Saha *et al.*, 2024], LoftQ [Li *et al.*, 2024], Atom [Zhao *et al.*, 2024], and FlexQuant [Liu *et al.*, 2025] under their recommended settings. The data-dependent signals for BALANCER are derived using a small calibration set (64 sequences from WikiText-2 and 1000 from C4).
Teacher-guided gradients. Unless otherwise stated, we use the *same FP16 base model* as the teacher (self-teaching) and compute reference gradients with respect to a reconstruction-style objective (matching the teacher logits on a small mixture of instruction corpora such as Alpaca [Taori *et al.*, 2023], Dolly [Conover *et al.*, 2023], and OpenAssistant [Köpf *et al.*, 2023]). This keeps profiling reproducible without requiring a larger external teacher; using a stronger teacher is compatible with the same profiling pipeline.

Method	Bits↓	CR↑	PPL-W↓	PPL-C4↓	Tok/s↑	Lat (ms)↓	VRAM (GB)↓
LLaMA-3.1-8B							
FP16	16.0	1.00	6.63	10.42	38.1	78	16.2
Q4 (RTN)	4.0	4.00	9.92±0.55	15.42±0.65	38.3±1.2	55±0.8	3.9
GPTQ	4.0	4.00	7.15±0.68	10.89±0.98	38.2±1.4	56±0.9	3.9
AWQ	4.0	4.00	7.08±0.79	10.75±1.01	38.1±1.5	57±0.8	3.9
CALDERA	4.0	4.00	6.68±0.77	10.25±1.03	38.2±1.3	53±0.9	3.9
BALANCER (Q-Mix)	3.6	4.44	6.78±0.43	10.69±0.99	38.0±2.3	55±1.0	3.9
BALANCER (Q4+LR)	4.0	4.00	6.72±0.54	10.58±1.06	38.5±2.2	54±0.8	3.9
LLaMA-2-7B							
FP16	16.0	1.00	5.47	6.97	42.9	84	13.1
Q4 (RTN)	4.0	4.00	7.05±0.64	10.43±1.02	42.9±1.3	59±1.0	3.4
GPTQ	4.0	4.00	6.98±0.75	8.35±1.06	42.7±1.2	56±1.2	3.4
AWQ	4.0	4.00	6.89±0.86	8.21±1.08	42.8±1.6	57±0.7	3.4
CALDERA	4.0	4.00	6.45±0.74	7.88±1.00	42.8±1.4	52±1.0	3.4
BALANCER (Q-Mix)	3.6	4.44	6.95±0.97	8.31±1.07	43.0±2.6	48±1.0	3.4
BALANCER (Q4+LR)	4.0	4.00	5.95±0.96	7.65±1.09	43.0±2.4	48±0.9	3.4

Table 1: Compression under budgets. For compressed methods, we report mean±SD over 5 independent runs with different calibration-set samplings (each run recomputes allocation/quantization and evaluates the resulting checkpoint). FP16 perplexity is deterministic under our protocol and is reported as a single value. Lower PPL/Lat is better; higher CR/Tok/s is better.

Method	HS↑	WG↑	PIQA↑	ARC-E↑	ARC-C↑	BoolQ↑	RTE↑
LLaMA-3.1-8B							
FP16	59.2	70.1	79.5	77.8	44.5	79.2	64.8
Q4 (RTN)	52.1	66.3	74.2	71.5	38.2	72.1	58.3
GPTQ	57.5	69.2	77.8	75.1	43.1	75.8	61.2
AWQ	57.8	69.5	78.1	75.4	43.3	76.1	61.5
CALDERA	58.2	69.8	78.5	75.8	43.8	76.5	62.1
BALANCER (Q-Mix)	58.5	70.0	78.8	76.0	44.0	77.0	62.5
BALANCER (Q4+LR)	58.8	70.2	79.1	76.3	44.2	77.3	62.8
LLaMA-2-7B							
FP16	57.1	68.8	77.8	76.3	43.3	77.9	63.5
Q4 (RTN)	52.8	65.2	74.5	70.8	38.5	72.5	58.1
GPTQ	56.2	68.1	77.2	74.8	42.5	75.2	60.8
AWQ	56.5	68.3	77.4	75.0	42.7	75.5	61.1
CALDERA	57.0	68.6	77.7	75.5	43.0	76.2	61.8
BALANCER (Q-Mix)	55.8	68.2	77.0	74.2	42.0	74.5	60.5
BALANCER (Q4+LR)	55.5	68.6	76.7	73.5	42.2	73.9	56.0

Table 2: Downstream accuracies under budgeted compression. Higher is better. Entries are averages over 5 runs.

4.2 Main Results

BALANCER Configurations. We evaluate two configurations: (i) BALANCER (Q4+LR), our main setting at the 4-bit operating point with a uniform 4-bit backbone and globally allocated low-rank corrections; and (ii) BALANCER (Q-Mix), a high-compression variant that uses a mixed-precision backbone (e.g., 2–3 bits for less sensitive MLP projections) to reach tighter budgets (e.g., 3.6 bits). We report the main operating points in Tables 1–5.

At 4-bit average budgets, BALANCER consistently attains lower perplexity than uniform quantization (RTN) and remains competitive with or superior to other strong post-training baselines. We highlight two key configurations from our results in Table 1. Our primary configuration, BALANCER (Q4+LR), is designed for a direct and fair comparison at the standard 4-bit operating point. To verify that BALANCER's quality improvements do not increase runtime cost, we compare BALANCER (Q4+LR) against the RTN Q4 baseline under identical kernels and settings. Table 3 shows matched throughput and VRAM, and latency that is comparable or lower. On LLaMA-2-7B, for instance, it successfully recovers over 50% of the perplexity gap between the uncompressed FP16 model and the naive round-to-nearest (RTN) baseline. On the other hand, the BALANCER Q-Mix variant demonstrates the flexibility of our framework, pushing compression further to 3.6 bits with only a mild PPL increase.

Downstream evaluation. In addition to perplexity and runtime, we report downstream accuracies to verify that budget allocation preserves task-level quality. We separate downstream results from the compression table for readability: Ta-

Method	Tok/s↑	Lat↓	VRAM↓
LLaMA-3.1-8B			
Q4 (RTN)	38.3±1.5	55.0±1.1	3.9
BALANCER (Q4+LR)	38.5±2.4	54.0±0.9	3.9
LLaMA-2-7B			
Q4 (RTN)	42.9±1.6	59.0±1.1	3.4
BALANCER (Q4+LR)	43.0±2.5	48.0±0.8	3.4

Table 3: Runtime under 4-bit budgets. Entries are mean±SD over 5 runs. Lat is in ms; VRAM is in GB.

Strategy	PPL-W↓	PPL-C4↓	Attn	MLP
Uniform Q4	9.918	15.424	4.0	4.0
Attention-heavy	6.832	10.751	4.0	3.2
MLP-heavy	6.851	10.798	3.2	4.0
Extreme mix	6.892	10.834	2.8	4.4
BALANCER (Q-Mix)	6.781	10.692	3.6	3.6

Table 4: Mixed-precision trade-offs on LLaMA-3.1-8B at a 3.60-bit average budget. "Attn"/"MLP" report average bit-widths.

ble 1 summarizes budget, perplexity, and runtime, while Table 2 reports accuracies on HS, WG, PIQA, ARC-E, ARC-C, BoolQ, and RTE under the same compressed checkpoints. Overall, BALANCER remains competitive with strong post-training baselines at the standard 4-bit operating point, and the mixed-precision configuration (Q-Mix) provides an additional operating point under tighter budgets while maintaining comparable task-level quality.

4.3 Performance under Different Budgets

We next analyze performance under different budgets on LLaMA-3.1-8B. Table 1 shows that as the budget tightens from 4.0 to 3.2, BALANCER degrades smoothly, unlike typical uniform schemes. At 4.0 bits, BALANCER (Q4+LR) closes the gap to FP16 more effectively than RTN. At 3.6 bits, BALANCER (Q-Mix) remains competitive with 4-bit baselines while increasing CR to $4.44\times$. Importantly, the results confirm that the performance decline of BALANCER is gradual and predictable: each reduction in budget leads to a proportional increase in perplexity, rather than sudden degradation. This demonstrates that the allocator consistently invests capacity in the most fragile regions, ensuring that the model remains usable even at aggressive compression levels. Signal ablations show that the full activation-gradient-curvature construction gives the lowest PPL, indicating that curvature provides complementary output-side sensitivity.

4.4 Mixed-Precision Analysis

While BALANCER is strong with a uniform 4-bit backbone, its global allocation is particularly well-suited for mixed-precision schedules. Fixing the budget to 3.6 bits for LLaMA-3.1-8B, we vary precision between Attention and MLP. The results are shown in Table 4. A balanced allocation, our BALANCER (Q-Mix) configuration, performs best at this budget.

Method	Bits↓	Lat↓	Tok/s↑	Res.↓	Peak↓
LLaMA-2-7B					
Q4 (RTN)	4.00	910±18	1.1±0.1	4.1	4.6
BALANCER (Q4+LR)	4.00	780±15	1.3±0.1	3.5	3.9
LLaMA-3.1-8B					
Q4 (RTN)	4.00	1080±20	0.9±0.1	4.5	5.0
BALANCER (Q-Mix)	3.60	920±16	1.1±0.1	3.8	4.3

Table 5: On-device performance on Jetson Orin Nano (8GB). Lat is first-token latency at prompt length 512 (ms); Tok/s is steady-state decoding over the next 512 tokens. "Res." and "Peak" report resident and peak VRAM (GB). Values are mean±SD over repeated runs.

4.5 Deployment Results

A core objective of BALANCER is to enable compressed models that retain quality while being practical on constrained hardware [Clements and Lao, 2024a; 2024b]. We therefore evaluate on a real-world edge device, the Jetson Orin Nano (8GB), and compare directly to the standard Q4 (RTN) baseline. Unless otherwise stated, we benchmark Jetson Orin Nano in the default *15W* power mode and use *TensorRT-LLM* for inference with standard GEMM kernels. Importantly, BALANCER does not require custom CUDA kernels: it exports a single reconstructed checkpoint with the same runtime interface as uniform PTQ models, so deployment uses an unchanged serving stack.

On-Device Performance. As shown in Table 5, BALANCER achieves clear deployment gains. On LLaMA-2-7B, our Q4+LR variant improves throughput by 18% and reduces first-token latency by 14%, while also lowering memory usage by about 15%. On LLaMA-3.1-8B, the Q-Mix variant shows similar advantages, reaching 22% higher throughput and 15% lower latency. These improvements suggest that preserving fragile directions can improve both quality and practical deployability under the same budget.

5 Conclusion

We introduced **BALANCER**, a training-free framework that reframes post-training compression as a global, budget-aware allocation problem. The core innovation of BALANCER is converting calibration-time signals (activations, reference gradients, and a diagonal curvature proxy) into information-guided subspace matrices, which induce direction-wise gain curves and enable global budget allocation for mixed-precision quantization and low-rank correction under an explicit bpp budget. This map guides a global allocator that strategically preserves critical singular directions with low-rank corrections. Our experiments on LLaMA-2-7B and LLaMA-3.1-8B demonstrate that BALANCER achieves consistently strong performance among training-free post-training compression methods under matched budgets, while remaining deployable on resource-constrained edge GPUs.

Acknowledgments

This research was supported in part by the National Science Foundation (NSF) SaTC-2426299 and SaTC-2413046.

References

[Aghajanyan *et al.*, 2021] Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021*, pages 7319–7328. Association for Computational Linguistics, 2021.

[Ahemad, 2025] Faizan Ahemad. Quantization aware matryoshka adaptation: Leveraging matryoshka learning, quantization, and bitwise operations for reduced storage and improved retrieval speed. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025*, pages 25–34. ACM, 2025.

[Ashkboos *et al.*, 2024] 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 *Advances in Neural Information Processing Systems 37*, 2024.

[Chu *et al.*, 2025] Rui Chu, Bingyin Zhao, Hanling Jiang, Shuchin Aeron, and Yingjie Lao. Bam-icl: Causal hijacking in-context learning with budgeted adversarial manipulation. *Advances in Neural Information Processing Systems*, 38:14152–14183, 2025.

[Clements and Lao, 2024a] Joseph Clements and Yingjie Lao. Resource efficient deep learning hardware watermarks with signature alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11651–11659, 2024.

[Clements and Lao, 2024b] Joseph Franklin Clements and Yingjie Lao. Reliable hardware watermarks for deep learning systems. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 32(4):752–762, 2024.

[Conover *et al.*, 2023] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023.

[Dao *et al.*, 2022] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems 35*, 2022.

[Dettmers *et al.*, 2022a] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems 35*, 2022.

[Dettmers *et al.*, 2022b] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *CoRR*, abs/2208.07339, 2022.

[Egiazarian *et al.*, 2024] Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization. In *Forty-first International Conference on Machine Learning, ICML 2024*. OpenReview.net, 2024.

[Eliseev and Mazur, 2023] Artyom Eliseev and Denis Mazur. Fast inference of mixture-of-experts language models with offloading. *CoRR*, abs/2312.17238, 2023.

[Esser *et al.*, 2020] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

[Frantar and Alistarh, 2023] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning, ICML 2023*, pages 10323–10337. PMLR, 2023.

[Frantar *et al.*, 2022] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: accurate post-training quantization for generative pre-trained transformers. *CoRR*, abs/2210.17323, 2022.

[Hoang *et al.*, 2026] Duy Cao Hoang, Thanh Quoc Hung Le, Rui Chu, Ping Li, Weijie Zhao, Yingjie Lao, and Khoa D Doan. Spark: An embarrassingly simple sparse watermarking in llms with enhanced text quality. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 4603–4626, 2026.

[Hu *et al.*, 2022] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022*. OpenReview.net, 2022.

[Jaiswal *et al.*, 2024] Ajay Kumar Jaiswal, Zhe Gan, Xianzhi Du, Bowen Zhang, Zhangyang Wang, and Yinfei Yang. Compressing llms: The truth is rarely pure and never simple. In *The Twelfth International Conference on Learning Representations, ICLR 2024*. OpenReview.net, 2024.

[Köpf *et al.*, 2023] Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. Openassistant conversations - democratizing large language model alignment. In *Advances in Neural Information Processing Systems 36*, 2023.

[Kwon *et al.*, 2022] Se Jung Kwon, Jeonghoon Kim, Jeongin Bae, Kang Min Yoo, Jin-Hwa Kim, Baeseong Park, Byeongwook Kim, Jung-Woo Ha, Nako Sung, and Dongsoo Lee. Alphatuning: Quantization-aware parameter-efficient adaptation of large-scale pre-trained language models. In *Findings of the Association for Computational*

Linguistics: EMNLP 2022, pages 3288–3305. Association for Computational Linguistics, 2022.

[Lawton *et al.*, 2024] Neal Lawton, Aishwarya Padmakumar, Judith Gaspers, Jack FitzGerald, Anoop Kumar, Greg Ver Steeg, and Aram Galstyan. Quailora: Quantization-aware initialization for lora. In *NeurIPS Efficient Natural Language and Speech Processing Workshop*, pages 22–33. PMLR, 2024.

[Lee *et al.*, 2023] Jung Hyun Lee, Jeonghoon Kim, Se Jung Kwon, and Dongsoo Lee. Flexround: Learnable rounding based on element-wise division for post-training quantization. In *International Conference on Machine Learning, ICML 2023*, pages 18913–18939. PMLR, 2023.

[Lee *et al.*, 2025] Jung Hyun Lee, Jeonghoon Kim, June Yong Yang, Se Jung Kwon, Eunho Yang, Kang Min Yoo, and Dongsoo Lee. Lrq: Optimizing post-training quantization for large language models by learning low-rank weight-scaling matrices. In *Proceedings of NAACL-HLT 2025 (Long Papers)*, pages 7708–7743. Association for Computational Linguistics, 2025.

[Li *et al.*, 2021] 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 *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net, 2021.

[Li *et al.*, 2024] Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024*. OpenReview.net, 2024.

[Lin *et al.*, 2024] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Guangxuan Xiao, and Song Han. AWQ: activation-aware weight quantization for on-device LLM compression and acceleration. *GetMobile Mob. Comput. Commun.*, 28(4):12–17, 2024.

[Liu *et al.*, 2025] Fangxin Liu, Zongwu Wang, JinHong Xia, Junping Zhao, Jian Liu, Haibing Guan, and Li Jiang. Flexquant: A flexible and efficient dynamic precision switching framework for LLM quantization. *CoRR*, abs/2506.12024, 2025.

[Meng *et al.*, 2025] Nicole Meng, Caleb Manicke, Ronak Sahu, Caiwen Ding, and Yingjie Lao. Advancing adversarial robustness in gnerfs: The il2-nerf attack. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16388–16397, 2025.

[Park *et al.*, 2024] Yeonhong Park, Jake Hyun, SangLyul Cho, Bonggeun Sim, and Jae W. Lee. Any-precision LLM: low-cost deployment of multiple, different-sized llms. In *Forty-first International Conference on Machine Learning, ICML 2024*. OpenReview.net, 2024.

[Saha *et al.*, 2024] Rajarshi Saha, Naomi Sagan, Varun Srivastava, Andrea Goldsmith, and Mert Pilanci. Compressing large language models using low rank and low precision decomposition. In *Advances in Neural Information Processing Systems 37*, 2024.

[Sun *et al.*, 2024] Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024*. OpenReview.net, 2024.

[Taori *et al.*, 2023] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[Trockman and Kolter, 2021] Asher Trockman and J. Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net, 2021.

[Tseng *et al.*, 2024] Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better LLM quantization with hadamard incoherence and lattice codebooks. In *Forty-first International Conference on Machine Learning, ICML 2024*. OpenReview.net, 2024.

[Wang *et al.*, 2025] Xinghao Wang, Pengyu Wang, Bo Wang, Dong Zhang, Yunhua Zhou, and Xipeng Qiu. Bitstack: Any-size compression of large language models in variable memory environments. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*. OpenReview.net, 2025.

[Xiao *et al.*, 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 *International Conference on Machine Learning, ICML 2023*, pages 38087–38099. PMLR, 2023.

[Yao *et al.*, 2022] 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 *Advances in Neural Information Processing Systems 35*, 2022.

[Yao *et al.*, 2024] Zhewei Yao, Xiaoxia Wu, Cheng Li, Stephen Youn, and Yuxiong He. Exploring post-training quantization in llms from comprehensive study to low rank compensation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2024)*, pages 19377–19385, 2024.

[Zhang *et al.*, 2024] Ying Zhang, Peng Zhang, Mincong Huang, Jingyang Xiang, Yujie Wang, Chao Wang, Yineng Zhang, Lei Yu, Chuan Liu, and Wei Lin. QQQ: quality quattuor-bit quantization for large language models. *CoRR*, abs/2406.09904, 2024.

[Zhao *et al.*, 2024] Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate LLM serving. In *Proceedings of the Seventh Annual Conference on Machine Learning and Systems, MLSys 2024*. mlsys.org, 2024.