LLAX: Adjacent Layer-based Low-rank Approximation for Parameter Efficient Inference

Anonymous ACL submission

Abstract

Deploying large transformer models under tight memory budgets remains a major challenge for parameter-efficient inference. We observe that adjacent transformer layers exhibit strong representational similarity, suggesting that one can be approximated from another via a low-rank approximation. Building on this observation, we propose LLAX, which models the transition between adjacent layers as a low-rank affine transformation. To estimate this transformation, we develop three algorithms based on singular value decomposition, alternating least squares, and gradient descent, which can be applied interchangeably. In addition, the residual errors are compressed via quantization to further improve performance. Our experimental results demonstrate that LLAX provides a hardware-agnostic and training-free solution that is orthogonal to existing compression methods, and characterize the memory-accuracy trade-off across a range of practically deployable configurations.

1 Introduction

Large language models (LLMs) have grown to enormous scale, and deploying them under real-world memory constraints remains a persistent burden for both researchers and practitioners. To reduce this cost without retraining, post-training compression methods have been widely studied, including quantization (Frantar et al., 2022; Xiao et al., 2024; Lin et al., 2024), pruning (Frantar and Alistarh, 2023; Sun et al., 2024; Ma et al., 2023), and low-rank weight compression (Wang et al., 2024b; Hsu et al., 2022; Saha et al., 2024). While effective, these methods share common limitations: (1) many depend on calibration data or activation statistics that may be unavailable in data-restricted settings; (2) they typically follow an intra-layer perspective, compressing each weight matrix in isolation without exploiting structure across layers.

A complementary direction considers the inter-layer structure of the network itself. Since transformer layers share the same architectural template and are stacked repeatedly, adjacent layers often exhibit similar weight structure. This assumption is supported by several structural properties of the transformer architecture. The residual stream contains no nonlinear activation functions, making it a purely linear space where each attention layer reads and writes via linear projections (Elhage et al., 2021). Empirical analysis further shows that token trajectories across layers exhibit high linearity and that block Jacobians of adjacent layers display strong singular vector alignment (Aubry et al., 2025). This alignment is reinforced by weight decay and skip connections, which drive consecutive layers to share dominant singular vector directions (Yunis et al., 2024; beren, 2022; Staats et al., 2025; Dong et al., 2023).

Building on the theoretical and empirical evidence for inter-layer linearity, we further observe that the alignment matrix, which linearly maps a weight matrix to its homogeneous counterpart in the adjacent layer, exhibits substantially lower effective rank (Roy and Vetterli, 2007) than the original weight matrix itself. These observations suggest that, rather than applying low-rank factorization directly to each weight matrix as in standard weight compression (Sharma et al., 2023; Yuan et al., 2025), it is more effective to factorize the alignment matrix, which preserves a larger proportion of structural variance at the same parameter budget.

Our contributions are summarized as follows:

- We propose LLAX, a one-shot, training-free, and hardware-agnostic compression method that exploits inter-layer linear alignment to reduce LLM memory footprint to 66% of the original model without any fine-tuning or calibration data.

- LLAX is compatible with existing compression techniques such as quantization and pruning. Crucially, it can be applied to models that have already been compressed by these methods, allowing it to be integrated into existing compression frameworks.
- We introduce three mathematically grounded approximation schemes for estimating the inter-layer alignment matrix–LLAX-SVD, LLAX-ALS, and LLAX-GD–and further propose a residual recovery mechanism to enhance performance under aggressive compression.

2 Related Work

2.1 Quantization and Pruning

Post-training quantization reduces the memory footprint of pretrained language models without full retraining. GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2024) reduce memory via weight quantization, while SparseGPT (Frantar and Alistarh, 2023) and Wanda (Sun et al., 2024) achieve compression through structured pruning. AWP (Liu et al., 2025) unifies both paradigms under a single activation-aware framework, and SERQ (Park et al., 2026) takes a hybrid approach by compensating for quantization error with a low-rank matrix. These methods operate on individual weight matrices in isolation, whereas LLAX reduces stored parameters by modeling the alignment between adjacent homogeneous layers.

2.2 Low-Rank Weight Approximation

Low-rank decomposition replaces a weight matrix with smaller factor matrices. Classical SVD-based compression truncates the singular spectrum of each layer independently, and recent methods refine this with activation-aware criteria (Sharma et al., 2023; Yuan et al., 2025; Wang et al., 2024b). Dobi-SVD (Wang et al., 2025b) improves truncation via differentiable rank selection and SVD remapping. MoDeGPT (Lin et al., 2025) jointly decomposes consecutive sub-layer matrix pairs within each Transformer block. SVD-LLM v2 (Wang et al., 2025c) tightens the reconstruction bound via improved truncation-aware whitening. OBD-LLM (Li et al., 2026) incorporates second-order Hessian information to derive a closed-form bi-directional whitening solution. These methods approximate each target matrix independently, while LLAX instead learns a low-rank alignment map between adjacent layers, leveraging the structural similarity between consecutive homogeneous weights.

2.3 Layer Redundancy and Weight Sharing

Several lines of work exploit redundancy across Transformer layers. LayerDrop (Fan et al., 2019) demonstrates that entire layers can be removed with limited degradation, and prior analyses report that adjacent layers often encode similar representations (Dalvi et al., 2020). Basis Sharing (Wang et al., 2024a) represents cross-layer weights as linear combinations of shared SVD basis vectors. Relaxed Recursive Transformers (Bae et al., 2025) combine shared weights with layer-wise LoRA residuals initialized via truncated SVD. CoMe (Wang et al., 2025a) compresses depth by merging adjacent layers through channel concatenation and hierarchical distillation. xKV (Chang et al., 2025) consolidates KV-Cache across adjacent layers into a shared low-rank subspace via cross-layer SVD. Unlike these approaches, LLAX learns a dedicated low-rank linear map between each pair of adjacent layers, preserving layer-specific variation without assuming a common subspace.

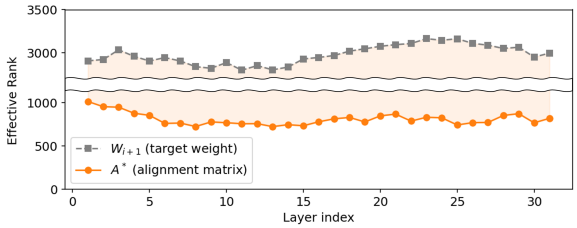

Figure 1: Effective rank of the alignment matrix A_i^* versus the target weight matrix W_{i+1} across all transformer layers of LLAMA-3.1-8B for output projection weight. The effective rank of A_i^* is consistently $3\times$ lower than that of W_{i+1} across all layers. This trend holds consistently across all weight types and layers.

3 Adjacent Layer-based Low-rank Weight Approximation

This section motivates alignment matrices as compact targets for model compression and outlines how they can be approximated under low-rank constraints.

3.1 Observation

We hypothesize that homogeneous weight matrices $W_i, W_{i+1} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ from adjacent Transformer layers exhibit structural similarity, such that W_{i+1} can be approximated from W_i through a simple linear transformation $W_{i+1} \approx A_i^* W_i$, where

$A_i^* \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$ is an alignment matrix for the i-th layer. To test this hypothesis, we examine the effective rank (Roy and Vetterli, 2007) of A_i^*: if the layers share substantial structure, the alignment between them should have a compact, low-rank representation.

We obtain A_i^* by solving for an unconstrained matrix that minimizes the Frobenius-norm reconstruction error between consecutive layers:

$$A_i^* = \arg\min_{A_i} \|A_i W_i - W_{i+1}\|_F^2.$$

By treating the columns of $W_i^\top$ as observations and those of $W_{i+1}^\top$ as targets, this standard linear least squares problem yields the closed-form solution:

$$A_i^* = W_{i+1} W_i^\top \left(W_i W_i^\top\right)^+, \tag{1}$$

where $(\cdot)^+$ denotes the Moore–Penrose pseudoinverse (Moore, 1920; Penrose, 1955).

We then quantify the effective rank of A_i^*, defined as the exponential of the Shannon entropy over its normalized singular value distribution and interpreted as the number of numerically significant singular values. As shown in Fig. 1, the effective rank of A_i^* is substantially lower than that of the original weight matrix W_{i+1} indicating that the alignment between adjacent layers possesses a compact low-rank structure and supporting our hypothesis of inter-layer similarity.

While W_{i+1} must retain the high-dimensional capacity required to represent complex features across the network, A_i^* merely captures the directional and scaling transformation between adjacent internal representations. Consequently, A_i^* exhibits a highly concentrated singular value spectrum. By capitalizing on this low-effective-rank property, a low-rank factorization of the alignment matrix can preserve a significantly larger proportion of the structural variance. This minimizes the overall reconstruction error at a strict parameter budget compared to direct weight matrix compression (Sharma et al., 2023; Yuan et al., 2025).

3.2 Problem Definition

Let $W_i \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote a weight matrix in the i-th Transformer block. We consider homogeneous weight pairs across consecutive blocks, where W_i and W_{i+1} have the same functional role and identical shape, such as two query projections or two FFN up-projection matrices. The approximation is applied to each pair of corresponding weight matrices from adjacent layers, grouping them consecutively.

Our goal is to approximate the target weight W_{i+1} from the preceding weight W_i using a low-rank alignment transformation $f^{(i)}$ which minimizes the residual sum of squares (RSS), defined as the squared Frobenius norm of the reconstruction error:

$$\begin{aligned} \min_{f^{(i)}} \quad & \|f^{(i)}(W_i) - W_{i+1}\|_F^2 \\ \text{s.t.} \quad & |f^{(i)}(W_i)| \ll |W_{i+1}|, \end{aligned}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and $|\cdot|$ denotes the number of stored parameters.

In this work, we instantiate $f^{(i)}$ as a low-rank linear transformation:

$$f^{(i)}(W_i) = L_A^{(i)} L_B^{(i)} W_i, \tag{2}$$

where $L_A^{(i)} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $L_B^{(i)} \in \mathbb{R}^{r \times d_{\text{out}}}$ are low-rank factor matrices with $r \leq d_{\text{out}}/2$. Equivalently, $L_A^{(i)} L_B^{(i)}$ forms a rank-r approximation to an alignment matrix $A_i \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$ that maps W_i toward W_{i+1}:

$$W_{i+1} \approx A_i W_i, \qquad \text{rank}(A_i) \leq r.$$

For ease of notation, we omit the layer superscript (i) and subscript i for L_A, L_B, and A in the rest of the paper, with the understanding that they are estimated independently for each adjacent layer pair. In the next section, we present three ways to estimate the low-rank factors: closed-form SVD, alternating least squares, and gradient-based optimization.

4 Methodology

An overview of this architectural transformation and the alignment process is illustrated in Fig. 2. We propose three distinct optimization methods to obtain the factor matrices in Eq. 2. Additionally, a quantized residual recovery mechanism is integrated to capture the remaining approximation error for near-lossless reconstruction.

4.1 LLAX-SVD: Closed-Form Alignment

To obtain the low-rank factors of A, LLAX-SVD applies truncated SVD to the optimal alignment matrix A^*. Since A^* from Eq. 1 is generally full-rank, we enforce the rank budget r by applying rank-r truncated SVD. Let $A^* = U\Sigma V^\top$ be the full SVD;

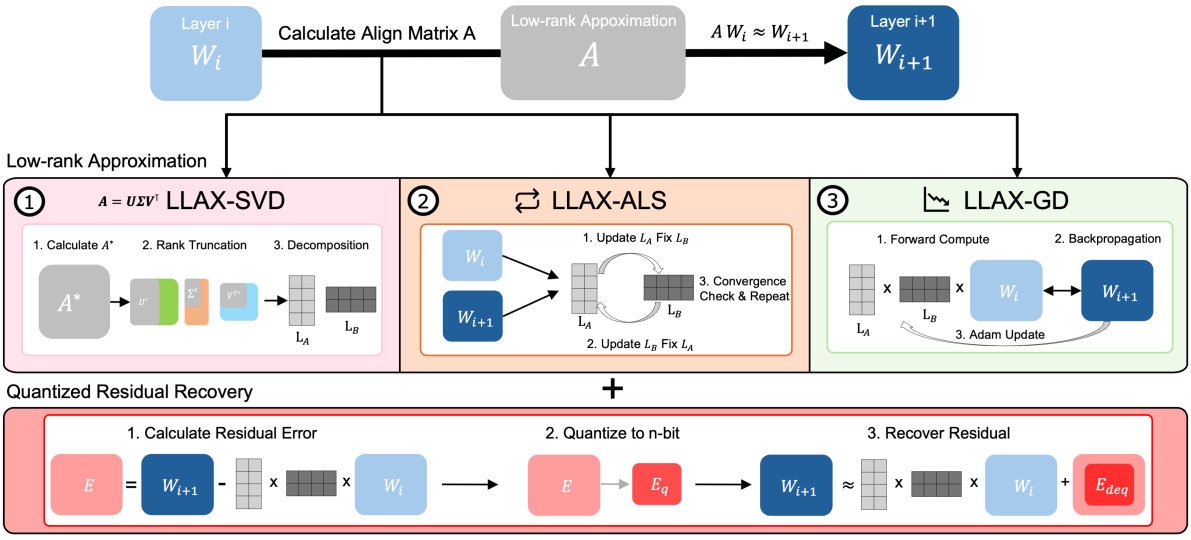

Figure 2: Overview of LLAX. Instead of compressing layers independently, LLAX captures adjacent layer alignment by an approximate linear transformation of matrix from W_i to W_{i+1}. To satisfy strict parameter budgets, this mapping is factorized into low-rank matrices L_A and L_B obtained using any of three alternative optimization algorithms: LLAX-SVD, LLAX-ALS and LLAX-GD. LLAX-SVD provides a closed-form and training-free algebraic solution, LLAX-ALS offers an iterative alternating update, and LLAX-GD leverages a gradient-based optimization. Furthermore, the residual error E compressed via min-max quantization is added during inference.

the truncated approximation $A_r^* = \sum_{j=1}^{r} \sigma_j u_j v_j^\top$ minimizes the reconstruction error

$$\|A^* - A_r^*\|_F^2 = \sum_{j=r+1}^{d_{\text{out}}} \sigma_j^2.$$

and, by the Eckart–Young–Mirsky theorem (Eckart and Young, 1936; Mirsky, 1960), such A^* is the unique optimal rank-r approximation under the Frobenius norm.

We then distribute the singular values equally between the two factors:

$$L_A = U_r \Sigma_r^{1/2} \in \mathbb{R}^{d_{\text{out}} \times r},$$
$$L_B = \Sigma_r^{1/2} V_r^\top \in \mathbb{R}^{r \times d_{\text{out}}},$$

such that $L_A L_B = A_r^* \approx A^*$. LLAX-SVD thus provides a principled, analytically optimal baseline for inter-layer alignment that requires no iterative optimization.

4.2 LLAX-ALS: Iterative Factorization

Unlike LLAX-SVD, which computes the alignment matrix in closed form, LLAX-ALS employs Alternating Least Squares (Zachariah et al., 2012) to iteratively optimize L_A and L_B by alternating between closed-form updates while keeping the other fixed. Given W_i and W_{i+1}, we solve the following ridge-regularized objective:

$$\min_{L_A, L_B} \|L_A L_B W_i - W_{i+1}\|_F^2 + \lambda \left(||L_A||^2 + ||L_B||^2\right).$$

As presented in Algorithm 1, the L_A update follows the exact ALS closed-form solution. For the L_B update, we adopt an approximate substitution $V_B = L_A^+ W_{i+1}$, which simplifies the subproblem to a standard least squares form for faster convergence, though it is an approximation to the exact ALS update. By adding the L_2 regularization term to stabilize convergence, both update rules take the form of a ridge-regularized closed-form solution. We run for a maximum of 500 iterations, with early stopping applied when the change in reconstruction error falls below a tolerance of `1e-6` between consecutive iterations.

4.3 LLAX-GD: Gradient-Based Optimization

LLAX-GD treats the factors $L_A \in \mathbb{R}^{d_{\text{out}} \times r}$ and $L_B \in \mathbb{R}^{r \times d_{\text{out}}}$ as learnable parameters and optimizes them using stochastic gradient descent (Paterek, 2007). The objective function is the Mean Squared Error under the Frobenius norm:

$$\mathcal{L} = \|L_A L_B W_i - W_{i+1}\|_F^2.$$

By leveraging backpropagation, LLAX-GD directly minimizes the reconstruction error with re-

Algorithm 1 LLAX-ALS Optimization with Ridge Regularization

Initialize $L_A \in \mathbb{R}^{d \times r}, L_B \in \mathbb{R}^{r \times d}$
while not converged **do**
  $V_A \leftarrow L_B W_i$ ▷ Fix L_B, solve for L_A
  $L_A \leftarrow W_{i+1} V_A^\top (V_A V_A^\top + \lambda I)^{-1}$
  $V_B \leftarrow L_A^+ W_{i+1}$ ▷ Fix L_A, solve for L_B
  $L_B \leftarrow V_B W_i^\top (W_i W_i^\top + \lambda I)^{-1}$
end while
return L_A, L_B

spect to the low-rank factors, allowing the alignment matrix to capture more expressive structures beyond what closed-form solutions can represent. In contrast to LLAX-ALS, no regularization term is applied, as the objective prioritizes accurate reconstruction of the target weight over convergence stability. In our implementation, we minimize the objective using the Adam optimizer (Kingma and Ba, 2017) with a learning rate of `1e-3` and run for a maximum of 5,000 iterations, with early stopping applied when the loss change falls below a tolerance of `1e-6` between consecutive iterations.

4.4 QRR: Quantized Residual Recovery

Regardless of the optimization method used to determine the low-rank alignment factors, a residual error matrix $E \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ naturally remains:

$$E = W_{i+1} - L_A L_B W_i.$$

To recover this dropped information and achieve near-lossless performance, the QRR mechanism compresses E using a min-max quantization scheme. During inference, the dequantized residual is added back to the low-rank prediction, executing the layer computation as:

$$W_{i+1} x \approx (L_A L_B W_i + \text{dequant}(E))\, x.$$

5 Experiments

5.1 Experimental Setup

Settings. Experiments are conducted on Llama-3.1-8B (Grattafiori et al., 2024). All experiments are implemented using PyTorch (Paszke et al., 2019) and the Hugging Face Transformers library (Wolf et al., 2020), and run on a single NVIDIA A100 GPU. Across these evaluations, we compare the performance of each method using ranks $r \in 32, 64, 128$. For QRR, we evaluate bit-widths of 2 and 4; we exclude 1-bit configurations as they yield negligible recovery, and 8-bit and higher as they offer limited compression benefit relative to their memory cost.

Datasets. To evaluate our methods, we employ a comprehensive benchmark suite covering three key aspects: commonsense reasoning, language understanding, and generation quality. For commonsense reasoning, we measure zero-shot accuracy across eight tasks, including BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC (Easy and Challenge) (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), and COPA (Roemmele et al., 2011). For language understanding, we report five-shot accuracy on MMLU (Hendrycks et al., 2020). Additionally, we assess generation quality by measuring perplexity (PPL) on the WikiText-2 dataset (Merity et al., 2016).

Target Weight Matrices. We apply the alignment approximation to the attention weight matrices of adjacent transformer layers. Compared to FFN weights, attention matrices participate more directly in the linear residual stream and have been shown to compose across layers as near-linear transformations (Elhage et al., 2021; Aubry et al., 2025). We exclude the first and last transformer layers as they are known to perform qualitatively distinct roles in token prediction, and apply the approximation to all remaining intermediate layers. (Shibata et al., 2026)

Parameter Efficiency. We derive the theoretical memory reduction of LLAX applied to the attention projection matrices of a transformer layer, where d denotes the hidden dimension, d_{kv} the key/value projection dimension, and r the target rank. Each layer originally holds $2d^2 + 2\, d_{kv}\, d$ attention parameters, which under LLAX are replaced by low-rank factors with a total of $4d\, r_q + 2(d + d_{kv})\, r_{kv}$ parameters, where $r_q = r$ for W_Q, W_O, and r_{kv} is progressively reduced by a factor of 2 until the resulting factors are smaller than d_{kv} in GQA models (Ainslie et al., 2023). Retaining an n-bit quantized residual adds an additional $n/16$ of the original parameter count.

5.2 Main Results

Table 1 presents the benchmark results on LLaMA-3.1-8B across three alignment methods, ranks, and QRR bit-widths. MR denotes the percentage of

Table 1: Experiment results on LLaMA-3.1-8B across three alignment methods and QRR bit-widths. Among the three methods, GD consistently outperforms ALS and SVD at the same rank and bit-width, and higher QRR bit-widths generally improve performance across all methods. Avg. denotes the mean accuracy across all 9 benchmarks, combining commonsense reasoning tasks and MMLU. HS, OBQA, WG and MR refer to HellaSwag, OpenBookQA, WinoGrande, and Memory Ratio respectively.

Method	**Rank**	**QRR**	**Benchmark Scores**										**PPL↓**	**MR(%)↓**
		n-bit	ARC-C↑	ARC-E↑	BoolQ↑	COPA↑	HS↑	OBQA↑	PIQA↑	WG↑	MMLU↑	Avg.↑		
Baseline	–	–	0.52	0.82	0.83	0.92	0.61	0.42	0.79	0.74	0.65	0.70	6.48	100
LLAX-SVD	32	–	0.29	0.60	0.43	0.76	0.39	0.23	0.71	0.55	0.24	0.47	52.66	54
		2	0.27	0.60	0.42	0.79	0.38	0.24	0.71	0.54	0.24	0.47	67.95	60
		4	0.32	0.66	0.50	0.84	0.41	0.26	0.74	0.62	0.31	0.52	64.83	66
	64	–	0.29	0.61	0.43	0.78	0.39	0.23	0.71	0.54	0.24	0.47	55.19	55
		2	0.26	0.59	0.44	0.79	0.36	0.23	0.68	0.54	0.24	0.46	74.63	61
		4	0.34	0.67	0.57	0.89	0.43	0.29	0.75	0.64	0.39	0.55	48.70	67
	128	–	0.27	0.61	0.43	0.78	0.38	0.24	0.71	0.55	0.24	0.47	61.51	57
		2	0.24	0.59	0.44	0.76	0.34	0.22	0.68	0.54	0.24	0.45	91.79	63
		4	0.41	0.73	0.69	0.87	0.51	0.30	0.75	0.67	0.47	0.60	21.82	69
LLAX-ALS	32	–	0.26	0.60	0.47	0.75	0.38	0.25	0.70	0.53	0.24	0.46	69.02	54
		2	0.22	0.31	0.42	0.70	0.33	0.17	0.56	0.50	0.23	0.38	1.20K	60
		4	0.33	0.65	0.63	0.79	0.46	0.27	0.72	0.63	0.36	0.54	76.32	66
	64	–	0.24	0.55	0.48	0.75	0.37	0.24	0.68	0.53	0.24	0.45	108.58	55
		2	0.19	0.31	0.43	0.54	0.31	0.17	0.56	0.50	0.24	0.36	617.56	61
		4	0.33	0.62	0.55	0.79	0.45	0.25	0.69	0.60	0.33	0.51	55.46	67
	128	–	0.31	0.62	0.41	0.77	0.42	0.23	0.72	0.54	0.25	0.47	67.22	57
		2	0.23	0.26	0.44	0.50	0.25	0.18	0.50	0.49	0.24	0.34	49.25K	63
		4	0.38	0.66	0.80	0.80	0.53	0.30	0.76	0.69	0.57	0.61	13.42	69
LLAX-GD	32	–	0.29	0.61	0.44	0.78	0.39	0.23	0.72	0.56	0.24	0.47	56.00	54
		2	0.28	0.60	0.42	0.76	0.40	0.23	0.71	0.56	0.24	0.47	61.03	60
		4	**0.47**	0.76	0.71	**0.91**	0.55	**0.32**	0.77	0.69	0.53	0.63	9.00	66
	64	–	0.29	0.62	0.45	0.74	0.41	0.24	0.73	0.54	0.24	0.47	53.75	55
		2	0.28	0.63	0.45	0.79	0.41	0.26	0.72	0.54	0.24	0.48	65.54	61
		4	0.44	0.77	0.72	0.90	0.55	0.32	**0.78**	**0.70**	0.55	0.64	8.43	67
	128	–	0.31	0.63	0.45	0.78	0.42	0.24	0.73	0.54	0.25	0.48	43.82	57
		2	0.32	0.64	0.46	0.81	0.43	0.25	0.74	0.57	0.57	0.53	59.79	63
		4	0.45	**0.78**	**0.80**	0.89	**0.57**	0.32	0.78	0.70	**0.58**	**0.65**	**7.85**	69

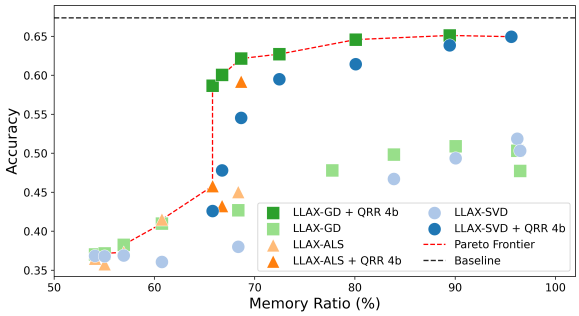

Figure 3: Pareto frontier of accuracy versus memory ratio, with each point corresponding to a combination of method, rank (32–1024), and QRR configuration (no-QRR and 4-bit QRR). The Pareto frontier is shown as a red dashed line.

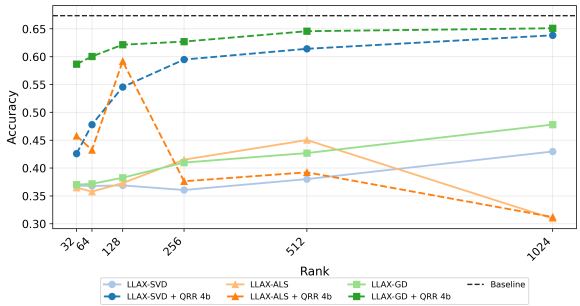

Figure 4: Average accuracy at different ranks from 32 to 1024 under no-QRR and 4-bit QRR.

attention weight memory retained relative to the baseline, where lower values indicate greater compression. Across all configurations, performance improves in the order of ALS, SVD, and GD, with GD achieving the best results in nearly every setting.

Although ALS and SVD both compute the alignment matrix in closed form, SVD consistently outperforms ALS. This is likely because SVD finds the globally optimal low-rank approximation in the least-squares sense, whereas ALS is an iterative alternating procedure that is more sensitive to initialization and may converge to a suboptimal solution. GD, by directly minimizing the reconstruction error through gradient-based optimization, achieves the best alignment quality at the cost of additional compute.

The effect of QRR varies notably by bit-width. Adding a 2-bit quantized residual tends to degrade performance relative to compression without QRR, as the quantization noise at 2 bits outweighs the

recovery benefit. At 4 bits, however, QRR consistently improves results across all methods and ranks, often recovering a substantial portion of the accuracy lost from low-rank compression alone.

Among all configurations, GD with rank 32 and 4-bit QRR stands out as a particularly well-balanced setting. It achieves a meaningful MR of 66% while maintaining competitive benchmark scores and perplexity, making it a strong candidate for deployment scenarios where both memory efficiency and model quality are important.

5.3 Ablation Studies

We conduct ablation studies to examine how rank, QRR bit-width, and target weight selection jointly affect the accuracy-memory trade-off, and further assess inference latency and generalization across model architectures.

5.3.1 Performance-Memory Trade-off

Fig. 3 plots accuracy against MR for all method and rank combinations ranging from 32 to 1024 on LLaMA-3.1-8B, with the Pareto frontier highlighted. Each point represents a specific method configuration at a given memory budget, allowing comparison of how much accuracy each method achieves at equivalent memory costs rather than at a fixed rank. The frontier is dominated entirely by LLAX-GD, with 4-bit QRR configurations occupying the upper portion and no-QRR configurations the lower portion.

Without QRR, the Pareto frontier advances gradually from an MR of 54% with accuracy 0.37 to 61% with accuracy 0.42, with only modest accuracy gains relative to memory cost. At an MR of around 66%, adding 4-bit QRR causes a sharp jump, where GD + QRR 4-bit at rank 32 reaches 0.59, a significant gain over any no-QRR configuration at similar memory cost. From there, the frontier continues to improve steadily up to an MR of 89% with accuracy 0.65 at rank 1024, approaching the baseline of 0.67.

These results suggest a practical guideline for deployment under memory constraints. When aggressive compression is required, reducing rank alone is the most memory-efficient path, as QRR adds a fixed memory cost of $n/16$ per matrix regardless of rank. Once a moderate memory budget is available, 4-bit QRR at a low rank offers a better accuracy-memory trade-off than increasing rank alone, making it the preferred configuration in most practical settings.

Table 2: TTFT of LLAX-SVD on LLaMA-3.1-8B across ranks 32–128 with and without 4-bit QRR.

Config	**No QRR**	**QRR 4b**
Baseline	32.4	
LLAX-SVD $r = 32$	57.5	202.7
LLAX-SVD $r = 64$	57.7	202.7
LLAX-SVD $r = 128$	58.5	203.2

5.3.2 Inference Latency

We measure the time to first token (TTFT) for LLAX-SVD on LLaMA-3.1-8B across configurations spanning ranks $\{32, 64, 128\}$ with and without 4-bit QRR, as reported in Table 2. Without QRR, TTFT increases modestly from the baseline of 32.4ms to around 57–58ms across all ranks, remaining stable regardless of rank. With 4-bit QRR, TTFT rises to approximately 203ms, about a 4× increase over the no-QRR setting, due to the additional dequantization and matrix addition operations required to recover the residual at inference time. When near-baseline accuracy is required, the dequantization and matrix addition operations of QRR are unavoidable, while rank-only compression offers a lighter alternative with lower latency. In latency-sensitive deployments, rank-only LLAX provides a practical middle ground between compression and response time.

5.3.3 Scaling Behavior Across Ranks

We analyze how the choice of rank affects performance across different methods by varying the rank from 32 to 1024 on LLaMA-3.1-8B, with results shown in Fig. 4. The baseline achieves an average accuracy of 0.67

GD with 4-bit QRR reaches 0.59 at rank 32 and

Table 3: Average accuracy comparing Attention-only and Attention+FFN compression at ranks 32–256 under no-QRR and 4-bit QRR.

		Attention Only		**Attention + FFN**	
Method	r	No QRR	QRR 4b	No QRR	QRR 4b
LLAX-SVD	32	0.37	0.43	0.32	0.31
	64	0.37	0.48	0.32	0.31
	128	0.37	0.55	0.32	0.32
	256	0.36	0.59	0.32	0.31
LLAX-ALS	32	0.36	0.46	0.31	0.31
	64	0.36	0.43	0.31	0.31
	128	0.37	0.59	0.32	0.31
	256	0.42	0.38	0.33	0.30
LLAX-GD	32	0.37	0.59	0.32	0.32
	64	0.37	0.60	0.31	0.32
	128	0.38	0.62	0.33	0.31
	256	0.41	0.63	0.32	0.32

climbs steadily to 0.65 at rank 1024, approaching the baseline most closely among all configurations. SVD with 4-bit QRR follows a similar trajectory, rising from 0.43 at rank 32 to 0.64 at rank 1024, closely tracking GD across the full rank range. However, both GD and SVD without QRR remain below the baseline across all ranks, plateauing around 0.43–0.48 even at rank 1024. ALS shows inconsistent performance across ranks. Without QRR, scores peak around rank 512 before dropping to 0.31 at rank 1024, and 4-bit QRR does not stabilize this trend.

5.3.4 Full Weight Compression

Table 3 presents a full weight comparison results on LLaMA-3.1-8B, where LLAX is applied to all weight matrices, including the FFN layers, rather than to attention matrices alone. Under attention-only compression with QRR 4b, all three methods benefit substantially from rank scaling. LLAX-SVD improves from 0.43 at $r = 32$ to 0.59 at $r = 256$, LLAX-GD monotonically increases from 0.59 to 0.63 over the same range, and LLAX-ALS attains its peak accuracy of 0.59 at $r = 128$. Extending compression to FFN weights, however, leads to a notable drop in performance compared to the attention-only setting, as FFN weights are less suited to inter-layer linear approximation. Notably, QRR, which provides large improvements in the attention-only setting, also yields no benefit over the No QRR baseline when FFN compression is enabled. This is consistent with our earlier motivation: unlike attention matrices, FFN weights interact with nonlinear activations between layers, which disrupts the near-linear structure.

5.3.5 Generalization to Diverse Models

To assess whether LLAX generalizes beyond LLaMA-3.1-8B, we evaluate all three methods on Gemma-2-9B (Team et al., 2024) and Qwen3-8B (Yang et al., 2025) across ranks of 32, 64, and 128, as reported in Table 4.

On Gemma-2-9B, SVD and ALS with 4-bit QRR approach the baseline of 0.71 across all ranks, while GD shows a notable performance drop at ranks 32 and 128 with QRR, suggesting that gradient-based optimization is less stable on this architecture. Gemma-2-9B interleaves local sliding window attention (Beltagy et al., 2020) and global attention across layers, which may introduce structural heterogeneity between adjacent layers that makes iterative optimization harder to converge reliably.

Table 4: Average accuracy on Gemma-2-9B and Qwen3-8B at ranks $r \in \{32, 64, 128\}$ with and without 4-bit QRR.

		Gemma-2-9B		Qwen3-8B	
Method	r	W/O QRR	QRR 4b	W/O QRR	QRR 4b
Baseline	–	0.71		0.73	
LLAX-SVD	32	0.32	0.65	0.40	0.57
	64	0.35	0.67	0.39	0.60
	128	0.32	0.68	0.39	0.61
LLAX-ALS	32	0.36	0.67	0.31	0.31
	64	0.35	0.68	0.30	0.31
	128	0.32	0.68	0.31	0.61
LLAX-GD	32	0.36	0.34	0.31	0.31
	64	0.36	0.68	0.42	0.30
	128	0.38	0.35	0.46	0.66

On Qwen3-8B, SVD with 4-bit QRR delivers consistent performance across ranks, while GD recovers well only at rank 128. ALS fails to recover meaningful accuracy in most configurations. Qwen3-8B uses a higher GQA ratio (32Q/8KV) compared to LLaMA-3.1-8B, which affects the effective rank constraints on key/value matrices and may influence how well each method captures the inter-layer alignment.

These results show that the relative strengths of SVD, ALS, and GD vary across model families, likely due to differences in attention architecture and weight characteristics. This underscores the importance of evaluating all three methods per model, as no single optimization strategy dominates across architectures.

6 Conclusion

In this work, we proposed LLAX, a training-free compression method that approximates adjacent transformer layer weight matrices via low-rank alignment. This design is motivated by our observation that the alignment matrix between adjacent layers exhibits substantially lower effective rank than the weight matrix itself, which allows more efficient use of a fixed parameter budget.

To estimate the low-rank factors, we proposed three optimization strategies: SVD, ALS, and GD, each with different trade-offs between computational cost and reconstruction accuracy. An optional QRR mechanism further improves accuracy at a modest memory cost. Experiments show that LLAX achieves competitive accuracy-memory trade-offs across a range of ranks and bit-widths, requires no calibration data, and is orthogonal to existing compression methods.

Limitations

While LLAX achieves promising compression performance through inter-layer low-rank approximation, several limitations remain.

Pairwise-only Alignment. Our method aligns adjacent layers in pairs in its current form. Extending it to longer chains or to adaptive layer clustering based on weight similarity poses additional difficulties, as approximation errors are expected to accumulate across successive alignments. Mitigating such error propagation toward broader alignment schemes is an active line of our ongoing work.

Downstream Adaptation. We focus on the compression stage and do not investigate task-specific fine-tuning of LLAX-based compressed models, where the resulting inter-layer dependency may introduce additional considerations. Because adjacent layer weights are reconstructed rather than stored independently, standard fine-tuning procedures that treat each weight matrix as an independent parameter no longer apply directly, and updating one weight alters both layers at once. A natural starting point is to update only the low-rank alignment matrices, though this restricts per-layer updates to a rank-r subspace and may limit expressiveness. Designing a fine-tuning recipe that overcomes this restriction while preserving the memory benefits of compression remains an open problem.

Applicability beyond Transformers. The core idea of LLAX, approximating one weight matrix as a low-rank linear transformation of another, is not inherently tied to a specific architecture and could in principle be applied to other neural networks that exhibit similar inter-layer weight regularities. However, we have not empirically verified whether the weight similarity patterns that LLAX relies on are present in architectures beyond the ones studied in this work. Extending LLAX to broader model families therefore requires a careful empirical analysis of inter-layer weight structures before any claim of general applicability can be made, which we leave to future work.

References

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901.

Murdock Aubry, Haoming Meng, Anton Sugolov, and Vardan Papyan. 2025. Transformer block coupling and its correlation with generalization in llms. *Preprint*, arXiv:2407.07810.

Sangmin Bae, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Seungyeon Kim, and Tal Schuster. 2025. Relaxed recursive transformers: Effective parameter sharing with layer-wise lora. *Preprint*, arXiv:2410.20672.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Preprint*, arXiv:2004.05150.

Sid Black beren. 2022. The singular value decompositions of transformer weight matrices are highly interpretable.

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

Chi-Chih Chang, Chien-Yu Lin, Yash Akhauri, Wei-Cheng Lin, Kai-Chiang Wu, Luis Ceze, and Mohamed S. Abdelfattah. 2025. xkv: Cross-layer svd for kv-cache compression. *Preprint*, arXiv:2503.18893.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers)*, pages 2924–2936.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. 2020. Analyzing redundancy in pretrained transformer models. *Preprint*, arXiv:2004.04010.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. 2023. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. *Preprint*, arXiv:2103.03404.

Carl Eckart and Gale Young. 1936. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli,

Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, and 6 others. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *Preprint*, arXiv:1909.11556.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pages 10323–10337. PMLR.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization. *arXiv preprint arXiv:2207.00112*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Yuhang Li, Donghyun Lee, Ruokai Yin, and Priyadarshini Panda. 2026. Optimal brain decomposition for accurate llm low-rank approximation. *Preprint*, arXiv:2604.00821.

Chi-Heng Lin, Shangqian Gao, James Seale Smith, Abhishek Patel, Shikhar Tuli, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2025. Modegpt: Modular decomposition for large language model compression. *Preprint*, arXiv:2408.09632.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100.

Jing Liu, Toshiaki Koike-Akino, Ye Wang, Hassan Mansour, and Matthew Brand. 2025. Awp: Activation-aware weight pruning and quantization with projected gradient descent. *Preprint*, arXiv:2506.10205.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

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

Leon Mirsky. 1960. Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1):50–59.

Eliakim H Moore. 1920. On the reciprocal of the general algebraic matrix. *Bulletin of the american mathematical society*, 26:294–295.

Yeonsik Park, Hyeonseong Kim, and Seungkyu Choi. 2026. Serq: Saliency-aware low-rank error reconstruction for llm quantization. *Preprint*, arXiv:2603.08185.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. *Preprint*, arXiv:1912.01703.

Arkadiusz Paterek. 2007. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8.

Roger Penrose. 1955. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*.

Olivier Roy and Martin Vetterli. 2007. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pages 606–610. IEEE.

Rajarshi Saha, Naomi Sagan, Varun Srivastava, Andrea J Goldsmith, and Mert Pilanci. 2024. Compressing large language models using low rank and low precision decomposition. *Advances in Neural Information Processing Systems*, 37:88981–89018.

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

Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. 2023. The Truth is in There: Improving Reasoning in Language Models with Layer-Selective Rank Reduction. *Preprint*, arXiv:2312.13558.

Keigo Shibata, Kazuki Yano, Ryosuke Takahashi, Jaesung Lee, Wataru Ikeda, and Jun Suzuki. 2026. Suppressing final layer hidden state jumps in transformer pretraining. *Preprint*, arXiv:2601.18302.

Max Staats, Matthias Thamm, and Bernd Rosenow. 2025. Small singular values matter: A random matrix analysis of transformer models. *Preprint*, arXiv:2410.17770.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, volume 2024, pages 4942–4964.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Fei Wang, Li Shen, Liang Ding, Chao Xue, Ye Liu, and Changxing Ding. 2025a. Layer as puzzle pieces: Compressing large language models through layer concatenation. *Preprint*, arXiv:2510.15304.

Jingcun Wang, Yu-Guang Chen, Ing-Chao Lin, Bing Li, and Grace Li Zhang. 2024a. Basis sharing: Cross-layer parameter sharing for large language model compression. *Preprint*, arXiv:2410.03765.

Qinsi Wang, Jinghan Ke, Masayoshi Tomizuka, Yiran Chen, Kurt Keutzer, and Chenfeng Xu. 2025b. Dobi-svd: Differentiable svd for llm compression and some new perspectives. *Preprint*, arXiv:2502.02723.

Xin Wang, Samiul Alam, Zhongwei Wan, Hui Shen, and Mi Zhang. 2025c. Svd-llm v2: Optimizing singular value truncation for large language model compression. *Preprint*, arXiv:2503.12340.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2024b. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2024. Smoothquant: Accurate and efficient post-training quantization for large language models. *Preprint*, arXiv:2211.10438.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zhihang Yuan, Yuzhang Shang, Yue Song, Dawei Yang, Qiang Wu, Yan Yan, and Guangyu Sun. 2025. ASVD: Activation-aware Singular Value Decomposition for Compressing Large Language Models. *Preprint*, arXiv:2312.05821.

David Yunis, Kumar Kshitij Patel, Samuel Wheeler, Pedro Savarese, Gal Vardi, Karen Livescu, Michael Maire, and Matthew R. Walter. 2024. Approaching deep learning through the spectral dynamics of weights. *Preprint*, arXiv:2408.11804.

Dave Zachariah, Martin Sundin, Magnus Jansson, and Saikat Chatterjee. 2012. Alternating least-squares for low-rank matrix reconstruction. *IEEE Signal Processing Letters*, 19(4):231–234.

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