MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

# LatentLLM: Activation-Aware Transform to Multi-Head Latent Attention

Koike-Akino, Toshiaki; Chen, Xiangyu; Liu, Jing; Wang, Ye; Wang, Pu; Brand, Matthew

TR2026-018 January 22, 2026

**Abstract**

Modern foundation models such as large language models (LLMs) require a massive amount of computational and memory resources. We propose a new framework to convert such LLMs into a reduced-dimension latent structure. Our method extends a local activation-aware tensor decomposition to a global attention-aware joint tensor decomposition. Our framework can significantly improve the model accuracy over the existing model compression methods when reducing the latent dimension to realize computationally/memory- efficient LLMs. We show the benefit on several benchmark including multi-modal reasoning tasks.

*AAAI Conference on Artificial Intelligence 2026*



Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

# LatentLLM: Activation-Aware Transform to Multi-Head Latent Attention

**Toshiaki Koike-Akino, Xiangyu Chen, Jing Liu, Ye Wang, Pu (Perry) Wang, Mathew Brand**

Mitsubishi Electric Research Laboratories (MERL), 201 Broadway, Cambridge, MA 02319 USA
{koike, jiliu, yewang, pwang, brand}@merl.com

## Abstract

Modern foundation models such as large language models (LLMs) require a massive amount of computational and memory resources. We propose a new framework to convert such LLMs into a reduced-dimension latent structure. Our method extends a local activation-aware tensor decomposition to a global attention-aware joint tensor decomposition. Our framework can significantly improve the model accuracy over the existing model compression methods when reducing the latent dimension to realize computationally/memory-efficient LLMs. We show the benefit on several benchmark including multi-modal reasoning tasks.

## Introduction

Large language models (LLMs) (Touvron et al. 2023; Achiam et al. 2023) and large multi-modal models (LMMs) (Liu et al. 2023) have shown excellent performance across a variety of general tasks (Wei et al. 2022; Katz et al. 2024; Bubeck et al. 2023). Nonetheless, these models having billions of parameters demand significant computational resources (Schwartz et al. 2020). Towards increasing the accessibility and sustainability of LLMs/LMMs, extensive efforts have been devoted to model compression (Xu and McAuley 2023; Zhu et al. 2024; Bai et al. 2024a): e.g., partial activation (Jiang et al. 2024; Lin et al. 2024a), pruning (Frantar and Alistarh 2023; Sun et al. 2023; Bai et al. 2024b; Hassibi, Stork, and Wolff 1993), quantization (Frantar et al. 2022; Lin et al. 2024b; Wang et al. 2024a), knowledge distillation (Hsieh et al. 2023; DeepSeek-AI 2025; Hwang et al. 2024), and low-rank factorization (Yuan et al. 2023; Liu et al. 2024; Hwang et al. 2024; Saxena et al. 2024).

More recently, the reduced-dimension LLM DeepSeek-V3 (Liu et al. 2024) has attracted much attention for its high efficiency and performance. It employs a low-rank architecture called multi-head latent attention (MLA) to compress the standard multi-head attention (MHA), realizing an efficient KV cache (Chang et al. 2024; Saxena et al. 2024), accelerated training, and high-performance inference. In this paper, we provide a novel solution to convert a pretrained LLM/LMM built with MHA into a compressed LLM/LMM with a type of MLA. Our approach is motivated by a global compression framework introduced in SparseLLM (Bai et al. 2024b) and Q-VLM (Wang et al. 2024a). Although the original method was designed for pruning/quantization, we adopt it for tensor rank reduction. We further extended it to the joint compression of MHA, while the original SparseLLM was for compressing the multi-layer perceptron (MLP) part. Our derived solution is based on a high-order tensor-rank decomposition to jointly factorize multiple linear layers.

The contributions of our paper are summarized below.

- We propose a novel low-rank decomposition approach called LatentLLM to compress LLMs/LMMs.
- We discuss an optimal pre-conditioning for activation-aware SVD.
- We reveal that a choice of junction matrix can significantly reduce the model size.
- We then introduce an attention-aware joint SVD framework to compress multiple weights at the same time.
- Several experiments validate that our LatentLLM approach can improve the performance of LLM/LMM compression over existing methods.
- The LLaVA/Qwen-VL compressed with LatentLLM offer a significant advantage in multi-modal reasoning.

## Related Work

**Model Compression** The field of model compression for LLMs/LMMs has seen a surge of innovative techniques aimed at mitigating the substantial computation and memory requirements (Zhu et al. 2024; Yuan et al. 2024). Various methods have emerged to address this challenge, each taking a unique approach to reduce the memory footprint of LLMs. These methods primarily fall into four categories: weight quantization (Lin et al. 2024b; Frantar et al. 2022; Wang et al. 2024a), network pruning (LeCun, Denker, and Solla 1989; Hassibi, Stork, and Wolff 1993; Frantar and Alistarh 2023; Bai et al. 2024b), knowledge distillation (Hsieh et al. 2023; DeepSeek-AI 2025; Hwang et al. 2024), and low-rank factorization (Yuan et al. 2023; Liu et al. 2024; Hwang et al. 2024; Saxena et al. 2024; Saha et al. 2024).

Among them, weight quantization has gained significant traction in the context of large foundation models due to its effectiveness. However, all four compression techniques are orthogonal and can be applied together. We hence introduce a novel low-rank decomposition method which jointly compresses multiple layers of an LLM in a training-free manner.

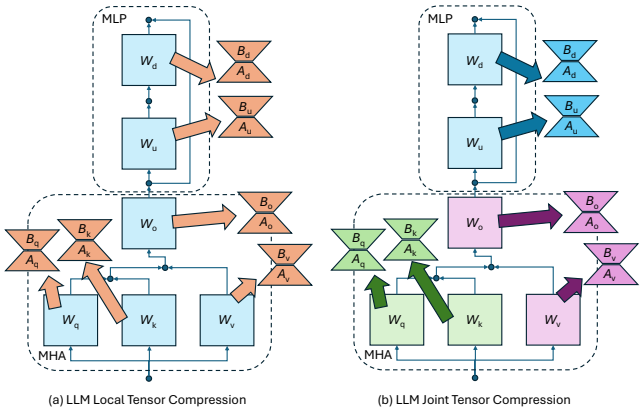


Figure 1: Reduced-dimension LLM/LMM with low-rank tensor decomposition. (a) each linear modules are locally compressed by activation-aware tensor decomposition. (b) multiple linear modules are globally compressed by attention-aware tensor decomposition.

**Low-Rank Decomposition** In the realm of low-rank decomposition (Schotthöfer et al. 2022) for neural network compression, existing methods typically involve decomposing weight matrices of pre-trained networks using techniques like Singular Value Decomposition (SVD) or tensor decomposition, followed by fine-tuning the factorized network (Denton et al. 2014; Sainath et al. 2013). LoSparse (Li et al. 2023) uses low-rank approximation plus a sparse matrix to compress the weight matrix in transformers. Similarly, CALDERA (Saha et al. 2024) uses low-rank approximation plus a quantized matrix. ASVD (Yuan et al. 2023) significantly improves the low-rank decomposition by dealing with activation statistics. It was applied to SVD-LLM (Wang et al. 2024b) and Palu (Chang et al. 2024). DeepSeek-V3 (Liu et al. 2024) employs the similar latent reduction via MLA to make MHA efficient and capable. Eigen attention (Saxena et al. 2024) is highly related to MLA.

## LatentLLM: Tensor Compression

### Reduced-Dimension LLM/LMM

Figure 1 illustrates the basic transformer architecture consisting of MHA and MLP, used in some LLMs/LMMs. For MLP, there are up and down projections, whereas MHA has query/key/value/output projections. By transforming those dense weight matrices into low-rank decompositions, we can realize an efficient latent LLM/LMM having potential benefits: (i) fewer-parameter model size; (ii) KV cache reduction; (iii) accelerated processing; (iv) lower-power consumption. In fact, some recent LLM models such as DeepSeek-V3 (Liu et al. 2024) demonstrated efficiency and high-performance with MLA. We focus on compressing a pre-trained LLM/LMM by converting MHA into a type of MLA in a zero-shot fashion, i.e., without any fine-tuning.

Most compression methods are based on a local loss minimization to approximate each weight individually. Motivated by recent work towards global optimization (Bai et al. 2024b; Wang et al. 2024a), we propose a joint tensor compression framework that we call "LatentLLM." Specifically, we derive a mathematical solution to jointly decompose a pair of query and value projections, a pair of value and output projections, and a pair of up and down projections to compress LLMs. We first address activation-aware compression to provide some new insights on the choice of pre-conditioner and junction matrix below.

### Activation-Aware SVD: Pre-Conditioning

A pioneering work by ASVD (Yuan et al. 2023) introduced a way to compress a layer depending on the activation statistics. Consider a pretrained-weight $W \in \mathbb{R}^{d' \times d}$ to compress with a lower-rank decomposition $\hat{W} = BA$ for compression matrix $A \in \mathbb{R}^{r \times d}$ and decompression matrix $B \in \mathbb{R}^{d' \times r}$. Using the input activation $X \in \mathbb{R}^{d \times l}$ ($l$ is the calibration sample length), ASVD aims to minimize the activation loss:

$$\mathcal{L}_1 = \mathbb{E}_X \big\| WX - \hat{W}X \big\|^2 = \mathbb{E}_X \big\| WX - BAX \big\|^2, \quad (1)$$

instead of the naïve weight-based loss:

$$\mathcal{L}_0 = \big\| W - \hat{W} \big\|^2 = \big\| W - BA \big\|^2. \quad (2)$$

It is well-known that the optimal solution to minimize $\mathcal{L}_0$ can be given by the plain SVD of $W$. To minimize $\mathcal{L}_1$, ASVD introduced a pre-conditioner $P \in \mathbb{R}^{d \times d}$ to whiten the statistical impact of the activation $X$. Specifically, ASVD uses the low-rank matrices given by whitened SVD:

$$BAP = \mathsf{svd}_r[WP], \quad (3)$$

where $\mathsf{svd}_r[\cdot]$ denotes the rank-$r$ truncated SVD.

Although ASVD originally suggested a diagonal $\ell_1$-norm pre-conditioning, the optimal pre-conditioning matrix $P$ can be given by reformulating $\mathcal{L}_1$ as follows:

$$\mathcal{L}_1 = \mathrm{tr}\big[ (W - BA) \mathbb{E}_X[XX^\top] (W - BA)^\top \big] \quad (4)$$

$$= \big\| (W - BA) C^{\frac{1}{2}} \big\|^2 = \big\| WC^{\frac{1}{2}} - BAC^{\frac{1}{2}} \big\|^2, \quad (5)$$

where $C = \mathbb{E}_X[XX^\top] \in \mathbb{R}^{d \times d}$ is a covariance (precisely, auto-correlation) of input activation. Hence, the above loss can be minimized by the SVD: $BAC^{\frac{1}{2}} = \mathsf{svd}_r[WC^{\frac{1}{2}}]$. Accordingly, it is found that the optimal pre-conditioner is the square-root covariance: $P = C^{\frac{1}{2}}$. Given the finite calibration data $X$, we can estimate the covariance as $C = XX^\top + \lambda I$, where the damping factor $\lambda \in \mathbb{R}_+$ corresponds to the shrunk estimator (Ledoit and Wolf 2004).

**Remark 1** *Different pre-conditioning methods were introduced in several techniques including pruning and quantization, as listed in Table 1. As those variants are sub-optimal, we use the optimal root covariance: $P = C^{\frac{1}{2}}$. See more discussion in Appendix.*

### Junction Matrix for Model Compression

In fact, the solution of (3) does not have a unique decomposition into low-rank matrices $B$ and $A$. The truncated SVD is written as

$$USV = \mathsf{svd}_r[WP], \quad (6)$$

| Conditioning $P$ | Expression | Reference |
|---|---|---|
| Identity | $I$ | Plain SVD (Sainath et al. 2013; Denton et al. 2014) |
| Hessian | $\mathsf{diag}[(XX^\top + \lambda I)^{-1}]^{\frac{-1}{2}}$ | OBS (Hassibi et al.); GPTQ (Frantar et al.); SparseGPT (Frantar and Alistarh) |
| $\ell_1$-norm | $\mathsf{diag}\left[\sum_j \lvert X_{1,j}\rvert, \ldots, \sum_j \lvert X_{d,j}\rvert\right]^\alpha$ | ASVD (Yuan et al. 2023); AWQ (Lin et al. 2024b) |
| $\ell_2$-norm | $\mathsf{diag}[XX^\top]^{\frac{1}{2}}$ | WandA (Sun et al. 2023) |
| Covariance | $XX^\top + \lambda I$ | CorDA (Yang et al. 2024) |
| Root-Covariance | $(XX^\top + \lambda I)^{\frac{1}{2}}$ | LatentLLM (Ours) |

Table 1: Variants of pre-conditioning matrices $P$ for activation-aware distillation.

where $U \in \mathbb{R}^{d' \times r}$, $S \in \mathbb{R}^{r \times r}$, and $V \in \mathbb{R}^{r \times d}$ are the left singular unitary matrix, singular-value diagonal matrix, and right singular unitary matrix, respectively. The decompression and compression matrices $B$ and $A$ can be expressed:

$$B = USJ, \qquad A = J^+ VP^+, \tag{7}$$

where $J \in \mathbb{R}^{r \times r}$ is a junction matrix and $[\cdot]^+$ denotes the pseudo inverse. Choosing any junction matrix that satisfies $SJJ^+ = S$ has no impact on the loss. Hence, there is few literature discussing the choice of $J$. Typically, one may use $J = I$ to put singular-values into the decompression matrix; $J = S^+$ to put it into the compression matrix; or $J = [S^{\frac{1}{2}}]^+$ to split it across both matrices equally.

However, a certain choice of $J$ has a noticeable advantage to reduce the number of parameters and floating-point operations (FLOPs). We can write the whitened right-singular matrix $VP^+$ as two sub-blocks:

$$VP^+ = [V_1 \quad V_2], \tag{8}$$

for $V_1 \in \mathbb{R}^{r \times r}$ and $V_2 \in \mathbb{R}^{r \times (d-r)}$. When we use $J = V_1$, the compression matrix $A$ will contain an identity block as long as $V_1$ is non-singular:

$$A = J^+ VP^+ = V_1^+ [V_1 \quad V_2] = \left[I \quad V_1^+ V_2\right]. \tag{9}$$

This can greatly reduce the number of parameters from $r(d' + d)$ to $r(d' + d) - r^2$, as well as the FLOPs, because no computation is needed for the identity projection.

For example, when the hidden size is $d = d'$, even if we compress it by $25\%$, i.e., the latent size is $r = 0.75d$, the total number of parameters will be $r(d'+d) = 1.5d^2$, which is $50\%$ more than the original $d^2$. This increased FLOPs hinders the low-rank compression of LLMs, even with the KV cache benefit (Liu et al. 2024; Yuan et al. 2023; Chang et al. 2024). Nevertheless, with our identity block form, we can always reduce the number of parameters regardless of the latent size, i.e., $r(d' + d) - r^2 < d'd$ for $r < \min(d', d)$. For the above example of $25\%$ latent compression, we can achieve $r(d' + d) - r^2 = (15/16)d^2 < d^2$. Figure 2 depicts the role of the pre-conditionning and junction matrices for the activation-aware compression. We also illustrate the tensor diagrams to understand the flexibility of tensor mapping.

**Remark 2** *Pivoting columns can solve the case when the left-most sub-block $V_1$ is singular. The pivoting does not require any FLOPs in inference while additional memory is required to record the permutation index.*

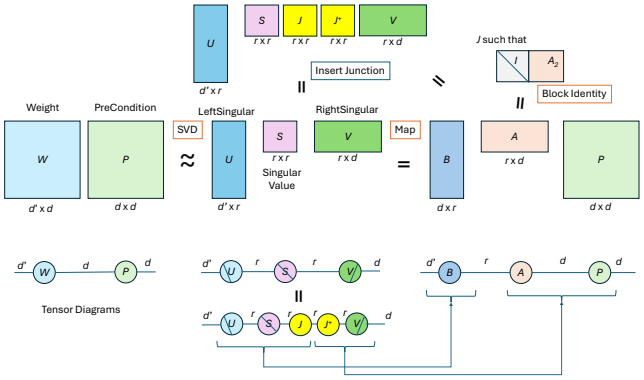


Figure 2: Activation-aware compression with pre-conditioning and junction matrix. The junction matrix $J$ can be adjusted such that $A$ or $B$ is block identity to save the number of parameters and inference computation.

## LatentLLM: Joint Tensor Compression

The SVD described above is optimal in the sense that the local error is minimized for the single tensor compression, whereas it does not guarantee global optimality. Motivated by the global loss minimization framework introduced by SparseLLM (Bai et al. 2024b), we propose a joint tensor compression technique which factorizes multiple tensors in adjacent modules concurrently.

### Multi-Head Latent Attention: Joint QK SVD

First, we consider a joint compression of query (Q) and key (K) projections in MHA to convert into MLA. The attention map is the dot product of query and key features:

$$M_i = X^\top W_{\mathrm{q},i}^\top W_{\mathrm{k},i} X, \tag{10}$$

where $M_i \in \mathbb{R}^{l \times l}$ is the $i$th head attention map before softmax operation, $W_{\mathrm{q},i} \in \mathbb{R}^{d_\mathrm{h} \times d}$ is the $i$th head query projection matrix, and $W_{\mathrm{k},i} \in \mathbb{R}^{d_\mathrm{h} \times d}$ is the $i$th head key projection matrix. Here, $d_\mathrm{h}$ is the head dimension, which is often $d_\mathrm{h} = d/h$ for the number of heads $h$.

Rather than individually compressing Q and K projections, we consider minimizing the attention map error:

$$\mathcal{L}_2 = \sum_{i=1}^{h} \left\| M_i - \hat{M}_i \right\|^2, \tag{11}$$

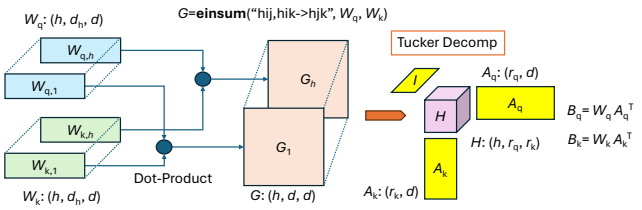


Figure 3: Tucker decomposition for joint QK compression. The compression matrices $A_{\mathrm{q}}$ and $A_{\mathrm{k}}$ correspond to the Tucker tensor planes, while $H = A_{\mathrm{q}} G A_{\mathrm{k}}^\top$ is the Tucker tensor core. For simplicity, we omit junction matrices and pre-conditioning matrix.

where $\hat{M}_i$ is the $i$th head latent attention with the low-rank compression:

$$\hat{M}_i = X^\top A_{\mathrm{q}}^\top B_{\mathrm{q},i}^\top B_{\mathrm{k},i} A_{\mathrm{k}} X, \tag{12}$$

where $A_{\mathrm{q}} \in \mathbb{R}^{r_{\mathrm{q}} \times d}$ is the Q compression matrix, $A_{\mathrm{k}} \in \mathbb{R}^{r_{\mathrm{k}} \times d}$ is the K compression matrix, $B_{\mathrm{q},i} \in \mathbb{R}^{d_{\mathrm{h}} \times r_{\mathrm{q}}}$ is the $i$th head Q decompression matrix, and $B_{\mathrm{k},i} \in \mathbb{R}^{d_{\mathrm{h}} \times r_{\mathrm{k}}}$ is the $i$th head K decompression matrix, respectively. Here, $r_{\mathrm{q}}$ and $r_{\mathrm{k}}$ are the latent dimensions for Q and K.

Similar to (5), we can write

$$\mathcal{L}_2 = \sum_{i=1}^{h} \Big\| \underbrace{C^{\frac{1}{2}} W_{\mathrm{q},i}^\top W_{\mathrm{k},i} C^{\frac{1}{2}}}_{G_i \in \mathbb{R}^{d \times d}} - \underbrace{C^{\frac{1}{2}} A_{\mathrm{q}}^\top}_{A_{\mathrm{q}}'^\top} \underbrace{B_{\mathrm{q},i}^\top B_{\mathrm{k},i}}_{H_i \in \mathbb{R}^{r_{\mathrm{q}} \times r_{\mathrm{k}}}} \underbrace{A_{\mathrm{k}} C^{\frac{1}{2}}}_{A_{\mathrm{k}}'} \Big\|^2. \tag{13}$$

This is known as a high-order SVD (HOSVD) problem to decompose for the 3-mode tensor $G \in \mathbb{R}^{h \times d \times d}$, whose $i$th slice is $G_i$. $A_{\mathrm{q}}'$ corresponds to the 2nd tensor plane, $A_{\mathrm{k}}'$ is the 3rd tensor plane, and $H \in \mathbb{R}^{h \times r_{\mathrm{q}} \times r_{\mathrm{k}}}$, whose $i$th slice is $H_i$, is the tensor core. This is illustrated in Figure 3.

This Tucker tensor decomposition is typically solved by alternating SVD over each slice sequentially. Algorithm 1 shows the pseudo-code of the joint SVD compression for QK latent projections. See the detailed derivations of the joint SVD algorithm in Appendix . Here, we generalize the pre-conditioning matrix $P$, as not necessarily the optimal $C^{\frac{1}{2}}$. In addition, we explicitly denoted any arbitrary junction matrices that do not change the error. Note that there are additional junction matrices per heads $J_i \in \mathbb{R}^{d_{\mathrm{h}} \times d_{\mathrm{h}}}$ as well as individual Q/K junctions $J_{\mathrm{q}} \in \mathbb{R}^{r_{\mathrm{k}} \times r_{\mathrm{k}}}$ and $J_{\mathrm{k}} \in \mathbb{R}^{r_{\mathrm{k}} \times r_{\mathrm{k}}}$. This suggests that we can further reduce the number of parameters by transforming into the block identity form per head. The total number of parameters will be $(r_{\mathrm{q}}+r_{\mathrm{k}})(d+d_{\mathrm{h}}h) - r_{\mathrm{q}}^2 - r_{\mathrm{k}}^2 - d_{\mathrm{h}}^2 h$, reduced from the original weights $2dd_{\mathrm{h}}h$.

**Remark 3** *Our joint QK SVD can be extended with most positional encoding methods. See Appendix.*

**Remark 4** *Attention-aware pruning (Liang et al. 2024) is related to our method, while our derivation provides an optimal tensor rank decomposition and only requires preconditioning matrices.*

Algorithm 1: Joint SVD to Transform MHA to MLA

**Input:** Pre-conditioning $P \in \mathbb{R}^{d \times d}$, query projection heads $W_{\mathrm{q},i} \in \mathbb{R}^{d_{\mathrm{h}} \times d}$, key projection heads $W_{\mathrm{k},i} \in \mathbb{R}^{d_{\mathrm{h}} \times d}$, number of heads $h$, rank $r_{\mathrm{q}}, r_{\mathrm{k}}$, iteration $N$
**Initialize:**
$W_{\mathrm{q},i} = W_{\mathrm{q},i} P$ for $i \in \{1, \dots, h\}$
$W_{\mathrm{k},i} = W_{\mathrm{k},i} P$ for $i \in \{1, \dots, h\}$
$G_i = W_{\mathrm{q},i}^\top W_{\mathrm{k},i}$ for $i \in \{1, \dots, h\}$
$A_{\mathrm{q}} = \mathsf{RightSingular}_{r_{\mathrm{q}}} \big[ \sum_{i=1}^{h} G_i G_i^\top \big]$
**for** $n = 1$ **to** $N$ **do**
  $A_{\mathrm{k}} = \mathsf{RightSingular}_{r_{\mathrm{k}}} \big[ \sum_{i=1}^{h} G_i^\top A_{\mathrm{q}}^\top A_{\mathrm{q}} G_i \big]$
  $A_{\mathrm{q}} = \mathsf{RightSingular}_{r_{\mathrm{q}}} \big[ \sum_{i=1}^{h} G_i A_{\mathrm{k}}^\top A_{\mathrm{k}} G_i^\top \big]$
**end for**
**Output:**
$B_{\mathrm{q},i} = J_i^\top W_{\mathrm{q},i} A_{\mathrm{q}}^\top J_{\mathrm{q}}$ for $i \in \{1, \dots, h\}$
$B_{\mathrm{k},i} = J_i^+ W_{\mathrm{k},i} A_{\mathrm{k}}^\top J_{\mathrm{k}}$ for $i \in \{1, \dots, h\}$
$A_{\mathrm{q}} = J_{\mathrm{q}}^+ A_{\mathrm{q}} P^+$
$A_{\mathrm{k}} = J_{\mathrm{k}}^+ A_{\mathrm{k}} P^+$

## Multi-Head Latent Attention: Joint VO SVD

Next, we discuss the joint SVD for value (V) and output (O) projections in MHA. The MHA output can be written as

$$Y' = \sum_{i=1}^{h} W_{\mathrm{o},i} W_{\mathrm{v},i} X \, \mathsf{softmax}[M_i^\top], \tag{14}$$

where $W_{\mathrm{o},i} \in \mathbb{R}^{d' \times d_{\mathrm{h}}}$ is the $i$th head output projection, and $W_{\mathrm{v},i} \in \mathbb{R}^{d_{\mathrm{h}} \times d}$ is the $i$th head value value projection. We may consider minimizing the loss:

$$\mathcal{L}_3 = \sum_{i=1}^{h} \big\| W_{\mathrm{o},i} W_{\mathrm{v},i} X - \hat{W}_{\mathrm{o},i} \hat{W}_{\mathrm{v},i} X \big\|^2, \tag{15}$$

for the low-rank compression: $\hat{W}_{\mathrm{o},i} = B_{\mathrm{o}} A_{\mathrm{o},i} \in \mathbb{R}^{d' \times d_{\mathrm{h}}}$ and $\hat{W}_{\mathrm{v},i} = B_{\mathrm{v},i} A_{\mathrm{v}} \in \mathbb{R}^{d_{\mathrm{h}} \times d}$ with $B_{\mathrm{o}} \in \mathbb{R}^{d' \times r_{\mathrm{o}}}$, $A_{\mathrm{o},i} \in \mathbb{R}^{r_{\mathrm{o}} \times d_{\mathrm{h}}}$, $B_{\mathrm{v},i} \in \mathbb{R}^{d_{\mathrm{h}} \times r_{\mathrm{v}}}$, and $A_{\mathrm{v}} \in \mathbb{R}^{r_{\mathrm{v}} \times d}$. The MLA output is thus given as

$$\hat{Y}' = \sum_{i=1}^{h} B_{\mathrm{o}} A_{\mathrm{o},i} B_{\mathrm{v},i} A_{\mathrm{v}} X \, \mathsf{softmax}[M_i]. \tag{16}$$

Interestingly, this is also formulated in a similar manner of (13), and it can be solved by the joint SVD algorithm.

## Latent MLP: Joint UD SVD

Finally, we address the joint compression of MLP layers which consists of up (U) projection and down (D) projection in typical LLMs/LMMs. Although the global optimization is generally difficult due to the nonlinear activations in the MLP layer, SparseLLM (Bai et al. 2024b) provides an elegant way to approximate MLP layer. The key idea is to minimize the MLP loss in a decoupled manner by introducing auxiliary variables. Our LatentLLM exploits the same philosophy to compress MLP layers.

| Compression | FLOPs | MACs | Parameters (byte) | Speed (token/sec) | KV Cache (byte) |
|---|---|---|---|---|---|
| 0% | 109.0T | 54.5T | 13.32G | 6.72k | 5.37G |
| 20% | 87.2T | 43.6T | 11.06G | 7.11k | 2.97G |
| 40% | 65.4T | 32.7T | 8.40G | 8.35k | 1.98G |
| 60% | 43.6T | 21.8T | 5.74G | 11.48k | 1.21G |
| 80% | 21.8T | 10.9T | 3.08G | 16.02k | 0.57G |

Table 2: Computational complexity and memory requirements of OPT-6.7B models compressed by LatentLLM.

| Compression | 10% | | | 20% | | | 30% | | | 40% | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | WT2 | PTB | C4 | WT2 | PTB | C4 | WT2 | PTB | C4 | WT2 | PTB | C4 |
| OPT-1.3B (WT2: 14.6, PTB: 20.3, C4: 16.1) | | | | | | | | | | | | |
| Plain SVD (Identity) | 9428.1 | 10670.8 | 4865.4 | 16461.2 | 20589.0 | 11039.8 | 18105.3 | 17360.8 | 12565.2 | 22155.9 | 15820.3 | 16566.2 |
| ASVD (Hessian) | 23.8 | 40.6 | 24.9 | 63.0 | 173.7 | 52.8 | 825.8 | 927.9 | 385.0 | 4912.3 | 3086.3 | 2138.9 |
| ASVD ($\ell_2$-norm) | 20.3 | 32.3 | 21.6 | 28.7 | 60.2 | 27.7 | 74.5 | 217.4 | 58.5 | 592.4 | 1072.0 | 336.7 |
| ASVD (Cov) | 29750.9 | 31499.1 | 18646.3 | 19716.9 | 21757.2 | 14967.2 | 21738.3 | 24300.2 | 16428.7 | 22776.5 | 23591.7 | 14922.1 |
| ASVD (RootCov) | 17.7 | 27.9 | 18.9 | 21.9 | 35.3 | 22.2 | 33.9 | 55.8 | 29.7 | 75.0 | 107.9 | 51.1 |
| **LatentLLM** (RootCov) | ***14.5** | **21.5** | **16.6** | **15.8** | **24.3** | **17.8** | **20.2** | **31.6** | **21.3** | **34.1** | **58.1** | **30.6** |
| Qwen3-1.7B (WT2: 16.7, PTB: 33.8, C4: 22.4) | | | | | | | | | | | | |
| Plain SVD (Identity) | 1.8e7 | 1.6e7 | 1.1e7 | 1.3e7 | 1.1e7 | 1.1e7 | 1.0e7 | 1.0e7 | 6.5e6 | 1.9e7 | 1.7e7 | 1.5e7 |
| ASVD (Hessian) | 1.1e5 | 6.8e5 | 3.4e5 | 5.2e6 | 8.0e6 | 4.6e6 | 4.4e6 | 6.7e6 | 4.0e6 | 3.0e6 | 1.6e7 | 3.2e6 |
| ASVD ($\ell_2$-norm) | 72.5 | 138.4 | 102.8 | 1679.1 | 2719.6 | 1639.8 | 4842.6 | 1.2e4 | 3960.3 | 2.8e5 | 2.8e5 | 9.8e4 |
| ASVD (Cov) | 860.6 | 3378.6 | 338.3 | 1989.8 | 1.0e4 | 516.1 | 6645.8 | 2.4e4 | 906.1 | 1.2e4 | 4.6e4 | 1796.1 |
| ASVD (RootCov) | 37.5 | 63.2 | 43.9 | 66.3 | 114.8 | 62.5 | 147.8 | 287.6 | 100.0 | 387.2 | 1066.1 | 193.7 |
| **LatentLLM** (RootCov) | **22.3** | **47.8** | **28.4** | **27.9** | **51.5** | **35.3** | **48.8** | **81.4** | **53.3** | **137.5** | **264.9** | **98.6** |
| Qwen3-8B (WT2: 9.7, PTB: 17.2, C4: 15.4) | | | | | | | | | | | | |
| Plain SVD (Identity) | 2.4e5 | 8.7e4 | 4.5e4 | 9.0e6 | 1.9e6 | 8.8e5 | 2.8e7 | 3.8e7 | 1.8e7 | 5.3e7 | 1.0e8 | 5.2e8 |
| ASVD (Hessian) | 33.6 | 78.3 | 40.7 | 90.8 | 573.7 | 115.2 | 1250.8 | 1.3e4 | 854.4 | 5324.6 | 3.1e4 | 4872.0 |
| ASVD ($\ell_2$-norm) | 18.8 | 32.2 | 25.1 | 26.0 | 43.9 | 32.0 | 40.6 | 71.8 | 47.7 | 98.6 | 171.1 | 92.1 |
| ASVD (Cov) | 1.3e5 | 4.7e5 | 4.4e4 | 1.2e5 | 3.1e5 | 4.4e4 | 8.3e4 | 2.3e5 | 3.4e4 | 6.1e4 | 1.2e5 | 2.7e4 |
| ASVD (RootCov) | 16.7 | 25.0 | 21.8 | 26.0 | 32.6 | 26.3 | 49.3 | 60.5 | 38.6 | 119.2 | 136.9 | 71.1 |
| **LatentLLM** (RootCov) | **11.8** | **21.2** | **17.9** | **14.2** | **23.1** | **19.9** | **22.4** | **29.5** | **24.8** | **53.9** | **68.5** | **40.8** |

Table 3: Perplexity (↓) of OPT/Qwen3 models with different SVD compression methods for 10–40% size reduction. Asterisk "*" indicates the better performance than the original un-compressed LLM.

Consider a 2-layer MLP:

$$Z = W_{\rm u} X, \quad Z' = \sigma(Z), \quad Y = W_{\rm d} Z', \tag{17}$$

where $W_{\rm u} \in \mathbb{R}^{d_{\rm i} \times d}$ is the up projection matrix, $W_{\rm d} \in \mathbb{R}^{d \times d_{\rm i}}$ is the down projection matrix, and $d_{\rm i}$ is the intermediate size which is typically four-fold of hidden size: $d_{\rm i} = 4d$. The intermediate variables $Z, Z' \in \mathbb{R}^{d_{\rm i} \times l}$ are auxiliary factors to be optimized.

Specifically, we consider minimizing the decoupled loss:

$$\mathcal{L}_4 = \alpha \|W_{\rm u} X - Z\|^2 + \beta \|Z' - \sigma(Z)\|^2 + \gamma \|W_{\rm d} Z' - Y\|^2, \tag{18}$$

for auxiliary variables $Z$ and $Z'$, given calibration input $X$ and output $Y$.

Following SparseLLM (Bai et al. 2024b), the optimal $Z'$ can be obtained given the other parameters fixed:

$$Z' = \left(\gamma W_{\rm d}^\top W_{\rm d} + \beta I\right)^+ \left(\beta \sigma(Z) + \gamma W_{\rm d}^\top Y\right). \tag{19}$$

The optimal closed-form $Z$ can be obtained for ReLU:

$$Z_- = W_{\rm u} X, \quad Z_+ = \frac{1}{\alpha + \beta}(\alpha Z_- + \beta Z'), \tag{20}$$

depending on $Z$'s element-wise sign.

This approach can be used for low-rank approximation. Given $Z$, we can optimize low-rank matrix $\hat{W}_{\rm u} = B_{\rm u} A_{\rm u}$ by SVD of $ZX^+ C^{\frac{1}{2}}$, where $ZX^+$ corresponds to the effective weight matrix to map $X$ onto $Z$. Given $Z'$, we approximate $\hat{W}_{\rm d} = B_{\rm d} A_{\rm d}$ by SVD of $YZ'^+ C_{\rm d}^{\frac{1}{2}}$, given correlation $C_{\rm d} = Z' Z'^\top$. This alternating solution is iterated over a few rounds. For detail, see Appendix.

## Experiments

**Experiments Setup** We conduct experiments for LLM and LMM benchmarks to evaluate the effectiveness of our method. Our experiments are based on the same setting of

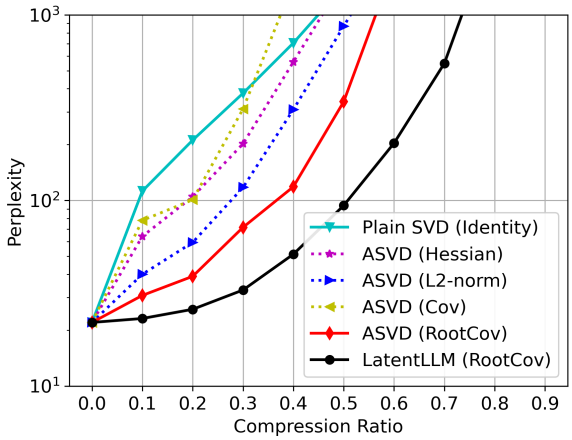


Figure 4: WT2 perplexity over compression ratio for OPT-350M model.

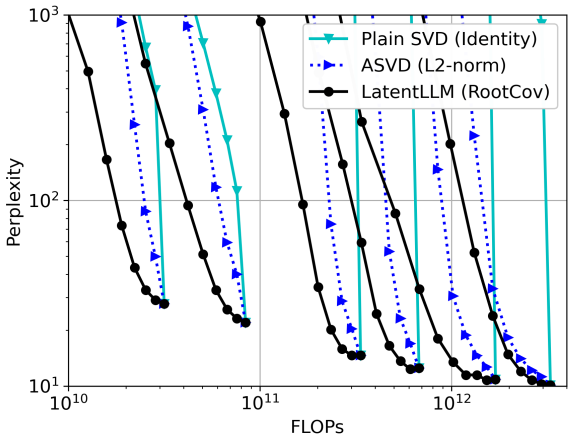


Figure 5: WT2 perplexity vs. FLOPs of six OPT models from 125M to 13B scales with varying compression ratios.

SparseLLM (Bai et al. 2024b) and their code base[1]. We implemented LatentLLM in PyTorch and used the HuggingFace transformers library for handling models and datasets. All experiments are conducted on NVIDIA A40 GPUs.

For LLM benchmark, we follow the same setup of (Frantar and Alistarh 2023) and use 64 samples of 2048-token segments, randomly chosen from the first shard of the C4 (Raffel et al. 2020) dataset. This dataset represents generic text data crawled from the internet and ensures our experiments are zero-shot as no task-specific data is seen during compression. We followed existing work (Sun et al. 2023) and compressed all linear layers in MLP and MHA in LLMs to the target compression ratio. For LMM benchmark, we use 64 samples, randomly chosen from the train split of the multi-modal question answering dataset. We consider two benchmarks: ScienceQA (Lu et al. 2022); and TextVQA (Singh et al. 2019).

For LLM experiments, we consider the OPT (Zhang et al. 2022) and Qwen3 (Yang et al. 2025) models as they provide a wide range of model sizes. We show results on different sizes of models to provide a broader picture for the performance of LatentLLM. We mainly focus on perplexity, which is known to be a stable metric for evaluating the accuracy of compression methods (Yao et al. 2022; Dettmers and Zettlemoyer 2023). We consider the test sets of raw-WikiText2 (WT2) (Merity et al. 2016) and the Penn Treebank (PTB) (Marcus et al. 1994) as well as a subset of the C4 validation data, all popular benchmarks in the LLM compression literature (Frantar and Alistarh 2023; Frantar et al. 2022; Sun et al. 2023).

For LMM experiments, we use LLaVA (Liu et al. 2023) and Qwen2.5-VL (Bai et al. 2025). We evaluate the capability of the multi-modal answer reasoning based on the ScienceQA dataset, which contains 21K vision-language multi-choice questions for three subjects: natural, social, and language science. Some fractions of questions have image and/or text contexts, and the problem levels range from grade 1 to 12. In addition, we also evaluate TextVQA, which makes LMMs to read and reason about text in images to answer visual reasoning questions for 28K images.

**Computational Complexity** When all linear modules are compressed with LatentLLM, the inference complexity is expected to be reduced with the compression ratio almost linearly. Nevertheless LLMs/VLMs have extra complexity other than linear affine transforms, the inference complexity is not precisely proportional to the compression factor. We show the complexity analysis in Table 2 for the compressed OPT-6.7B models, based on the `calflops` library. We assume the token length of 2048 at 4 batches. We found that the FLOPs, multiply-accumulation operations (MACs), and parameters are almost linearly reduced with the compression factor. The inference throughput on A40 GPU can be also monotonically increased by compressing LLMs. While we used `torch.compile(mode="max-autotune")`, it was not a perfectly linear speedup due to the sub-optimal GPU kernel. The reduction of KV cache memory is significant because the latent dimension has quadratic relation to the sparsity: $r(d' + d) - r^2 = \rho dd'$.

**Compression over Model Size** We first look into the compression capabilities of our LatentLLM across various model sizes in comparison to baseline methods. Some results are shown for a size reduction over 10–40% in Table 3. The perplexity results of the original un-compressed LLM models are reported next to the names of the models in the table.

We can see that the conventional plain SVD has a poor performance, and that ASVD with a proper pre-conditioning can significantly improve the perplexity. It was found that the diagonal Hessian is worse than the diagonal $\ell_2$-norm, whereas covariance pre-conditioning can be terrible in low compression regimes for larger LLMs. In contrast, the superiority of root covariance is remarkable. In addition, the joint SVD used for LatentLLM offers an additional improvement consistently across different model sizes and families. Notice that our methods can also achieve slightly better performance than the original un-compressed LLMs for some

[1]https://github.com/BaiTheBest/SparseLLM

| Method | Compression | Subject | | | Context Modality | | | Grades | | Avg |
|---|---|---|---|---|---|---|---|---|---|---|
| | | NAT | SOC | LAN | TXT | IMG | NO | G1-6 | G7-12 | |
| Original un-compressed | 0% | 72.47 | 69.18 | 65.73 | 73.51 | 68.82 | 65.99 | 72.72 | 65.19 | 70.03 |
| Plain SVD (Identity) | 10% | 5.33 | 1.35 | 0.27 | 5.77 | 6.69 | 0.00 | 3.30 | 2.97 | 3.18 |
| ASVD (Hessian) | 10% | 17.23 | 24.97 | 3.18 | 18.43 | 29.55 | 2.16 | 17.40 | 11.27 | 15.21 |
| ASVD ($\ell_2$-norm) | 10% | 16.70 | 18.34 | 2.55 | 17.89 | 24.34 | 2.23 | 16.04 | 8.57 | 13.37 |
| ASVD (Cov) | 10% | 41.21 | 27.22 | 37.91 | 41.30 | 35.15 | 38.33 | 38.62 | 35.27 | 37.42 |
| ASVD (RootCov) | 10% | 64.08 | 56.13 | 57.36 | 64.03 | 60.98 | 57.35 | 62.70 | 57.02 | 60.67 |
| **LatentLLM** (RootCov) | 10% | **68.52** | **64.23** | **61.36** | **69.06** | **65.20** | **61.53** | **68.72** | **60.45** | **65.76** |
| Plain SVD (Identity) | 30% | 0.13 | 0.00 | 0.00 | 0.10 | 0.00 | 0.07 | 0.11 | 0.00 | 0.07 |
| ASVD (Hessian) | 30% | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| ASVD ($\ell_2$-norm) | 30% | 0.09 | 0.00 | 0.00 | 0.10 | 0.10 | 0.00 | 0.04 | 0.07 | 0.05 |
| ASVD (Cov) | 30% | 41.25 | 27.33 | 37.36 | 41.40 | 35.25 | 37.84 | 38.47 | 35.27 | 37.33 |
| ASVD (RootCov) | 30% | 56.66 | 51.18 | 52.27 | 56.74 | 56.27 | 51.99 | 55.73 | 51.94 | 54.37 |
| **LatentLLM** (RootCov) | 30% | **64.03** | **56.24** | **55.27** | **64.47** | **61.77** | **55.40** | **62.78** | **55.37** | **60.13** |
| Plain SVD (Identity) | 50% | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| ASVD (Hessian) | 50% | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| ASVD ($\ell_2$-norm) | 50% | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| ASVD (Cov) | 50% | 40.94 | 26.88 | 36.91 | 41.20 | 35.10 | 37.28 | 38.18 | 34.74 | 36.95 |
| ASVD (RootCov) | 50% | 52.58 | 45.11 | 46.00 | 52.93 | 50.07 | 45.99 | 51.28 | 45.75 | 49.30 |
| **LatentLLM** (RootCov) | 50% | **55.55** | **47.24** | **49.55** | **56.01** | **54.09** | **48.78** | **54.55** | **48.12** | **52.25** |

Table 4: Accuracy in percent (↑) on ScienceQA dataset of LLaVA-7B model with different compression methods for $10\%$–$50\%$ size reduction. Question subjects: natural science (NAT); social science (SOC); language science (LAN). Context modality: text (TXT); image (IMG); or no context (NO). Grades: 1–6 (G1-6); 7–12 (G7-12).

| Compression | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|
| LLaVA-7B: Uncompressed Acc 61.32 | | | | | |
| Plain SVD (identity) | 2.36 | 0.48 | 0.35 | 0.34 | 0.36 |
| ASVD (Hessian) | 23.88 | 9.60 | 1.24 | 0.21 | 0.31 |
| ASVD ($\ell_2$-norm) | 24.41 | 9.53 | 2.77 | 0.82 | 0.75 |
| ASVD (Cov) | 0.38 | 0.36 | 0.40 | 0.33 | 0.35 |
| ASVD (RootCov) | 52.51 | 49.91 | 45.53 | 38.47 | 27.36 |
| **LatentLLM** (RootCov) | **60.06** | **57.65** | **52.63** | **46.90** | **35.94** |
| Qwen2.5-VL-7B-Instruct: Uncompressed Acc 82.11 | | | | | |
| Plain SVD (identity) | 0.02 | 0.47 | 0.32 | 0.05 | 0.11 |
| ASVD (Hessian) | 58.76 | 7.03 | 0.23 | 0.45 | 0.41 |
| ASVD ($\ell_2$-norm) | 77.84 | 73.92 | 57.13 | 18.79 | 0.41 |
| ASVD (Cov) | 0.41 | 0.41 | 0.41 | 0.41 | 0.41 |
| ASVD (RootCov) | 79.46 | 74.76 | 66.31 | 51.80 | 34.91 |
| **LatentLLM** (RootCov) | **80.85** | **79.30** | **73.90** | **62.11** | **42.53** |

Table 5: Accuracy in percent (↑) on TextVQA dataset for compressed LLaVA-7B and Qwen2.5-VL-7B.

cases. Figure 4 shows the plot at a wider range of compression ratios for OPT-350M model, and Figure 5 plots the performance of all six models in OPT family across FLOPs when varying the compression ratios.

**Multi-Modal Reasoning Capability** We show the accuracy of latent LLaVA models for ScienceQA multi-modal reasoning benchmark in Table 4. It is verified that our LatentLLM can significantly outperform other low-rank compression methods across diverse reasoning problems over different subjects/contexts/grades, approaching the performance of the original un-compressed LLaVA model. It is seen that ASVD without using proper pre-conditioning matrix degrades the performance quickly with higher compression ratios, while our LatentLLM keeps relatively higher performance across all cases. Another benchmark on TextVQA shown in Table 5 validates the clear superiority of our LatentLLM over baselines for both LLaVA and Qwen-VL models.

**Discussion** Our framework with optimal pre-conditioning and joint tensor distillations can be readily applied to pruning and quantization as well. See some results in Appendix. Further fine-tuning is expected to be able to compensate for the performance degradation by the latent reduction.

## Summary

We introduced LatentLLM which jointly compresses multiple tensors through the use of high-order tensor-rank decomposition. We also provided some new perspectives for activation-aware compression when choosing the preconditioner and junction matrix. With a proper selection, we demonstrated that the model compression performance can be significantly improved. Our latent LMMs showed a significant advantage in multi-modal reasoning tasks compared to other baseline methods.

## References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Bai, G.; Chai, Z.; Ling, C.; Wang, S.; Lu, J.; Zhang, N.; Shi, T.; Yu, Z.; Zhu, M.; Zhang, Y.; et al. 2024a. Beyond efficiency: A systematic survey of resource-efficient large language models. *arXiv preprint arXiv:2401.00625*.

Bai, G.; Li, Y.; Ling, C.; Kim, K.; and Zhao, L. 2024b. SparseLLM: Towards global pruning of pre-trained language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*.

Bubeck, S.; Chandrasekaran, V.; Eldan, R.; Gehrke, J.; Horvitz, E.; Kamar, E.; Lee, P.; Lee, Y. T.; Li, Y.; Lundberg, S.; et al. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*.

Chang, C.-C.; Lin, W.-C.; Lin, C.-Y.; Chen, C.-Y.; Hu, Y.-F.; Wang, P.-S.; Huang, N.-C.; Ceze, L.; Abdelfattah, M. S.; and Wu, K.-C. 2024. Palu: Compressing KV-cache with low-rank projection. *arXiv preprint arXiv:2407.21118*.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q. V.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948.

Denton, E. L.; Zaremba, W.; Bruna, J.; LeCun, Y.; and Fergus, R. 2014. Exploiting linear structure within convolutional networks for efficient evaluation. *Advances in neural information processing systems*, 27.

Dettmers, T.; and Zettlemoyer, L. 2023. The case for 4-bit precision: k-bit inference scaling laws. In *International Conference on Machine Learning*, 7750–7774. PMLR.

Frantar, E.; and Alistarh, D. 2023. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 10323–10337. PMLR.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Hassibi, B.; Stork, D. G.; and Wolff, G. J. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, 293–299. IEEE.

Hsieh, C.-Y.; Li, C.-L.; Yeh, C.-K.; Nakhost, H.; Fujii, Y.; Ratner, A.; Krishna, R.; Lee, C.-Y.; and Pfister, T. 2023. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Hwang, I.; Park, H.; Lee, Y.; Yang, J.; and Maeng, S. 2024. PC-LoRA: Low-Rank Adaptation for Progressive Model Compression with Knowledge Distillation. *arXiv preprint arXiv:2406.09117*.

Jiang, A. Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Hanna, E. B.; Bressand, F.; et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Katz, D. M.; Bommarito, M. J.; Gao, S.; and Arredondo, P. 2024. GPT-4 passes the bar exam. *Philosophical Transactions of the Royal Society A*, 382(2270): 20230254.

Kenton, J. D. M.-W. C.; and Toutanova, L. K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, 2. Minneapolis, Minnesota.

LeCun, Y.; Denker, J.; and Solla, S. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Ledoit, O.; and Wolf, M. 2004. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2): 365–411.

Li, Y.; Yu, Y.; Zhang, Q.; Liang, C.; He, P.; Chen, W.; and Zhao, T. 2023. LoSparse: Structured compression of large language models based on low-rank and sparse approximation. In *International Conference on Machine Learning*, 20336–20350. PMLR.

Liang, Y.; Long, J.; Shi, Z.; Song, Z.; and Zhou, Y. 2024. Beyond linear approximations: A novel pruning approach for attention matrix. *arXiv preprint arXiv:2410.11261*.

Lin, B.; Tang, Z.; Ye, Y.; Cui, J.; Zhu, B.; Jin, P.; Zhang, J.; Ning, M.; and Yuan, L. 2024a. MoE-LlaVa: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Chen, W.-M.; Wang, W.-C.; Xiao, G.; Dang, X.; Gan, C.; and Han, S. 2024b. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. *Proceedings of Machine Learning and Systems*, 6: 87–100.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024. DeepSeek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual Instruction Tuning.

Lu, P.; Mishra, S.; Xia, T.; Qiu, L.; Chang, K.-W.; Zhu, S.-C.; Tafjord, O.; Clark, P.; and Kalyan, A. 2022. Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*.

Marcus, M.; Kim, G.; Marcinkiewicz, M. A.; MacIntyre, R.; Bies, A.; Ferguson, M.; Katz, K.; and Schasberger, B. 1994. The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Radford, A. 2018. Improving language understanding by generative pre-training. *Preprint*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140): 1–67.

Saha, R.; Sagan, N.; Srivastava, V.; Goldsmith, A.; and Pilanci, M. 2024. Compressing large language models using low rank and low precision decomposition. *Advances in Neural Information Processing Systems*, 37: 88981–89018.

Sainath, T. N.; Kingsbury, B.; Sindhwani, V.; Arisoy, E.; and Ramabhadran, B. 2013. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *2013 IEEE international conference on acoustics, speech and signal processing*, 6655–6659. IEEE.

Saxena, U.; Saha, G.; Choudhary, S.; and Roy, K. 2024. Eigen attention: Attention in low-rank space for KV cache compression. *arXiv preprint arXiv:2408.05646*.

Schotthöfer, S.; Zangrando, E.; Kusch, J.; Ceruti, G.; and Tudisco, F. 2022. Low-rank lottery tickets: finding efficient low-rank neural networks via matrix differential equations. *Advances in Neural Information Processing Systems*, 35: 20051–20063.

Schwartz, R.; Dodge, J.; Smith, N. A.; and Etzioni, O. 2020. Green AI. *Communications of the ACM*, 63(12): 54–63.

Singh, A.; Natarjan, V.; Shah, M.; Jiang, Y.; Chen, X.; Parikh, D.; and Rohrbach, M. 2019. Towards VQA Models That Can Read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8317–8326.

Su, J.; Ahmed, M.; Lu, Y.; Pan, S.; Bo, W.; and Liu, Y. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568: 127063.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, C.; Wang, Z.; Xu, X.; Tang, Y.; Zhou, J.; and Lu, J. 2024a. Q-VLM: Post-training Quantization for Large Vision-Language Models. *arXiv preprint arXiv:2410.08119*.

Wang, X.; Zheng, Y.; Wan, Z.; and Zhang, M. 2024b. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Xu, C.; and McAuley, J. 2023. A survey on model compression and acceleration for pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 10566–10575.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, Y.; Li, X.; Zhou, Z.; Song, S. L.; Wu, J.; Nie, L.; and Ghanem, B. 2024. CorDA: Context-Oriented Decomposition Adaptation of Large Language Models. *arXiv preprint arXiv:2406.05223*.

Yao, Z.; Yazdani Aminabadi, R.; Zhang, M.; Wu, X.; Li, C.; and He, Y. 2022. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35: 27168–27183.

Yuan, Z.; Shang, Y.; Song, Y.; Wu, Q.; Yan, Y.; and Sun, G. 2023. ASVD: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*.

Yuan, Z.; Shang, Y.; Zhou, Y.; Dong, Z.; Zhou, Z.; Xue, C.; Wu, B.; Li, Z.; Gu, Q.; Lee, Y. J.; et al. 2024. LLM inference unveiled: Survey and roofline model insights. *arXiv preprint arXiv:2402.16363*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhu, X.; Li, J.; Liu, Y.; Ma, C.; and Wang, W. 2024. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12: 1556–1577.

# Weight-Aware Compression

## Plain SVD

Given a pretrained weight matrix $W \in \mathbb{R}^{d' \times d}$, we wish to approximate it with a low-rank structure:

$$\hat{W} = B \times A, \tag{21}$$

where $\hat{W} \in \mathbb{R}^{d' \times d}$ is an approximated weight, $B \in \mathbb{R}^{d' \times r}$ and $A \in \mathbb{R}^{r \times d}$ are low-rank matrices with a rank $r \leq d, d'$. We assume $d' \leq d$ for simplicity, as modifying for $d' \geq d$ is straightforward.

Consider the approximation loss:

$$\mathcal{L} = \|W - \hat{W}\|^2 \tag{22}$$

$$= \|W - BA\|^2. \tag{23}$$

The best solution is given by SVD of $W$ as follows:

$$A = J^+ V, \tag{24}$$

$$B = USJ, \tag{25}$$

where $U \in \mathbb{R}^{d' \times r}$ is $r$ most-principal left-singular vectors, $S = \mathsf{diag}[\sigma_1, \ldots, \sigma_r] \in \mathbb{R}^{r \times r}$ is diagonal singular-values, and $V \in \mathbb{R}^{r \times d}$ is the most-principal right-singular vectors for $W$:

$$USV = \mathsf{svd}_r[W], \tag{26}$$

where we assume the singular values are sorted in the descending order: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$. The loss is the accumulation of all the squared singular-values outside the rank $r$: $\mathcal{L}_{\min} = \sum_{i>r} \sigma_i^2$.

## Junction Matrix

There is no literature discussing the choice of a junction matrix $J \in \mathbb{R}^{r \times r}$, which has no impact on performance, provided that $SJJ^+ = S$ is satisfied. There are many suitable choices for this matrix $J$, such as:

- Left singular: $J = I$;
- Right singular: $J = S^+$;
- Symmetry singular: $J = [S^{\frac{1}{2}}]^+$.
- Left block identity: $J = [US]^+_{:r}$
- Right block identity: $J = [V]_{:r}$

Although there is no performance impact by the choice of $J$, it is notable that the block identity which is based on block LU factorization can significantly reduce the number of parameters and FLOPs by $r^2$. This parameter reduction is particularly significant in high-dimensional (high-rank) latent cases. For example, when the weight is a size of $d = d' = 2048$, even with the half-rank latent $r = d/2 = 1024$, there is no parameter reduction as the dense compression and decompression matrices $B$ takes $2dr = d^2$ parameters. Hence, the $50\%$ latent has no benefit in complexity but only for KV cache memory reduction as discussed in DeepSeek-V3 (Liu et al. 2024). It is even worse for $r > d/2$: e.g., if we use $75\%$ latent of $r = 0.75d = 1536$, the total parameter and FLOPs increases by 50% of the original weight (i.e., $2rd - d^2 = d^2/2$). However, using the block identity form, we can save $r^2$, and the total FLOPs will be always less than the original weight: $2rd - r^2 < d^2$ for any $r < d$.

# Activation-Aware Compression

Consider an input token $X \in \mathbb{R}^{d \times l}$ for a context length $l \gg d$, the linear projection output $Y \in \mathbb{R}^{d' \times l}$ is:

$$Y = WX. \tag{27}$$

We assume that no bias is used for simplicity.

We wish to minimize the expected approximation error of output activation vectors between the true $Y$ and the approximation $\hat{Y} \in \mathbb{R}^{d' \times l}$:

$$\hat{Y} = \hat{W}X, \tag{28}$$

projected by a low-rank weight $\hat{W} = BA$. Consider the loss function:

$$\mathcal{L} = \mathbb{E}_X \|Y - \hat{Y}\|^2 \tag{29}$$

$$= \mathbb{E}_X \|(W - \hat{W})X\|^2 \tag{30}$$

$$= \mathbb{E}_X \|(W - BA)X\|^2 \tag{31}$$

$$= \mathrm{tr}[(W - BA)\mathbb{E}_X[XX^\top](W - BA)^\top] \tag{32}$$

$$= \mathrm{tr}[(W - BA)C(W - BA)^\top] \tag{33}$$

$$= \|WC^{\frac{1}{2}} - BAC^{\frac{1}{2}}\|^2, \tag{34}$$

where $C = \mathbb{E}_X[XX^\top] \in \mathbb{R}^{d\times d}$ is a (positive semidefinite) correlation matrix of the input vector.

The loss function of the activation-aware distillation in (34) is identical to the weight-aware distillation in (23) by transforming as:

$$W \to W' = WC^{\frac{1}{2}}, \tag{35}$$

$$A \to A' = AC^{\frac{1}{2}}. \tag{36}$$

Hence, the optimal solution is obtained similarly.

Specifically, from (24), (25), and (26), we have

$$A' = AC^{\frac{1}{2}} = J^+V', \tag{37}$$

$$B = U'S'J, \tag{38}$$

where $U' \in \mathbb{R}^{d'\times d'}$, $S' \in \mathbb{R}^{d'\times d}$, and $V' \in \mathbb{R}^{d\times d}$ are SVD of $W' = WC^{\frac{1}{2}}$:

$$U'S'V' = \mathsf{svd}[WC^{\frac{1}{2}}]. \tag{39}$$

From (37), we finally obtain optimal $A$ as:

$$A = J^+S'V'[C^{\frac{1}{2}}]^+. \tag{40}$$

## Pre-Conditioning Matrix

ASVD (Yuan et al. 2023) proposed to use a projection matrix $P \in \mathbb{R}^{d\times d}$ on weights before SVD: $\mathsf{svd}[WP]$. The optimal projection $P$ is apparently the square-root covariance $P = C^{\frac{1}{2}}$, while there are many other approximated projections that were considered in literature:

- Root-covariance: $P = (XX^\top + \lambda I)^{\frac{1}{2}}$
- Covariance (e.g., CorDA (Yang et al. 2024)): $P = XX^\top$
- Diagonal L2-norm (e.g., WandA (Sun et al. 2023)): $P = \mathrm{diag}[XX^\top]^{\frac{1}{2}}$
- Diagonal L1-norm (e.g., AWQ (Lin et al. 2024b), ASVD (Yuan et al. 2023)): $P = \mathrm{diag}[\|[X]_{1,:}\|_1, \ldots, \|[X]_{d,:}\|_1]$
- Diagonal Hessian (e.g., OBS (Hassibi, Stork, and Wolff 1993), GPTQ (Frantar et al. 2022), SparseGPT (Frantar and Alistarh 2023)): $P = \mathrm{diag}[(XX^\top + \lambda I)^{-1}]^{\frac{-1}{2}}$
- Identity (Plain SVD, e.g., (Sainath et al. 2013)): $P = I$

In the context of fine-tuning initialization, CorDA (Yang et al. 2024) uses covariance matrix $C$ without square root, which should be worse than the square-root covariance. Fig. 6 demonstrates the benefit for random weights approximation with covariance drawn from the Wishart distribution. GPTQ (Frantar et al. 2022) and SparseGPT (Frantar and Alistarh 2023) use another preconditioning matrix based on optimal brain surgeon (OBS) (Hassibi, Stork, and Wolff 1993) using Hessian, in the context of quantization and pruning. Similarly, we can use it for low-rank compression as we have evaluated. In the context of pruning, WandA (Sun et al. 2023) proposed a simpler diagonal projection based on $\ell_2$-norm, while it achieves an excellent performance. AWQ and ASVD used the diagonal $\ell_1$-norm, while the theoretical justification is missing. They introduced another exponent factor to adjust.

## Bias Update

When the bias is there, we have

$$\mathcal{L} = \|(WX + b1^\top) - (BAX + \hat{b}1^\top)\|^2. \tag{41}$$

The gradient with respect to $\hat{b}$ is

$$\nabla\hat{b} = -2((WX + b1^\top) - (BAX + \hat{b}1^\top))1. \tag{42}$$

Hence, the optimal bias modification is:

$$\hat{b} = b + (W - BA)\mu, \tag{43}$$

where $\mu = X1/1^\top 1 \in \mathbb{R}^{d\times 1}$ is a mean bias of input activation. Plugging into the loss, we have

$$\mathcal{L} = \|(W - BA)(X - \mu 1^\top)\|^2 \tag{44}$$

$$= \mathrm{tr}[(W - BA)\underbrace{(X - \mu 1^\top)(X - \mu 1^\top)^\top}_{C_0 \in \mathbb{R}^{d\times d}}(W - BA)^\top] \tag{45}$$

$$= \|(W - BA)C_0^{\frac{1}{2}}\|^2. \tag{46}$$

Hence, the optimal solution is the SVD of weight multiplied with square root of covariance $C_0$ not auto-correlation ($XX^\top$):

$$C_0 = (C - \mu\mu^\top)l. \tag{47}$$

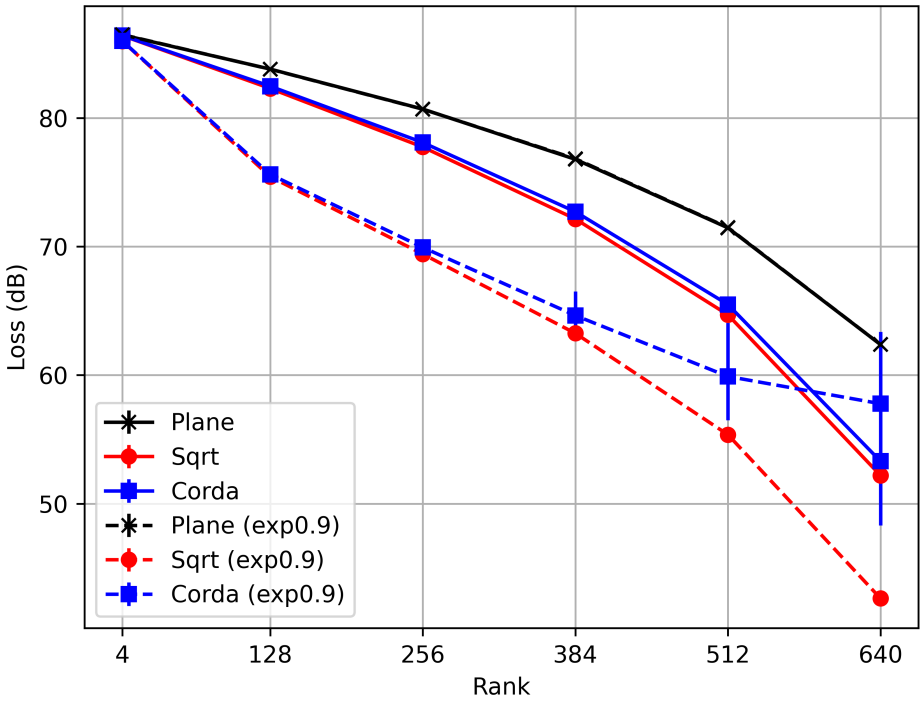


Figure 6: Comparison of SVD, CorDA, and RootCorDA.

## Activation-Aware Joint QKV Compression

Consider minimizing QKV activation:

$$\mathcal{L} = \left\| \underbrace{\begin{bmatrix} W_{\mathrm{q}} \\ W_{\mathrm{k}} \\ W_{\mathrm{v}} \end{bmatrix}}_{W \in \mathbb{R}^{3d' \times d}} X - \underbrace{\begin{bmatrix} B_{\mathrm{q}} \\ B_{\mathrm{k}} \\ B_{\mathrm{v}} \end{bmatrix}}_{B \in \mathbb{R}^{3d' \times r}} AX \right\|^2. \tag{48}$$

In this case, the optimal solution is an SVD of $WC^{\frac{1}{2}}$.

Note that this is different from QKV individual optimization:

$$\mathcal{L}' = \sum_{i \in [\mathrm{q},\mathrm{k},\mathrm{v}]} \|W_i X - B_i A_i X\|^2 \tag{49}$$

$$= \left\| \underbrace{\begin{bmatrix} W_{\mathrm{q}} \\ W_{\mathrm{k}} \\ W_{\mathrm{v}} \end{bmatrix}}_{A \in \mathbb{R}^{3d' \times d}} X - \underbrace{\begin{bmatrix} B_{\mathrm{q}} & O & O \\ O & B_{\mathrm{k}} & O \\ O & O & B_{\mathrm{v}} \end{bmatrix}}_{B \in \mathbb{R}^{3d' \times 3r}} \underbrace{\begin{bmatrix} A_{\mathrm{q}} \\ A_{\mathrm{k}} \\ A_{\mathrm{v}} \end{bmatrix}}_{W \in \mathbb{R}^{3r \times d}} X \right\|^2. \tag{50}$$

The solution is 3 SVDs: $W_i C^{\frac{1}{2}}$.

The key difference: (i) the shared vs. non-shared compression matrix $A$; (ii) block-diagonal vs. dense decompression matrix $B$. The number of parameters will be $r(3d' + d)$ from $3r(d' + d)$, allowing 50% more rank for joint QKV when $d' = d$. When we use LU factorizatin, the number of parameters will be $r(3d' + d - r)$ from $3r(d' + d - r)$. We show the benefit of joint-QKV activation-aware distillation in Fig. 7.

Nevertheless, the relative magnitudes over Q/K/V are not well-treated for joint case, and it could be worse in the global performance in the end.

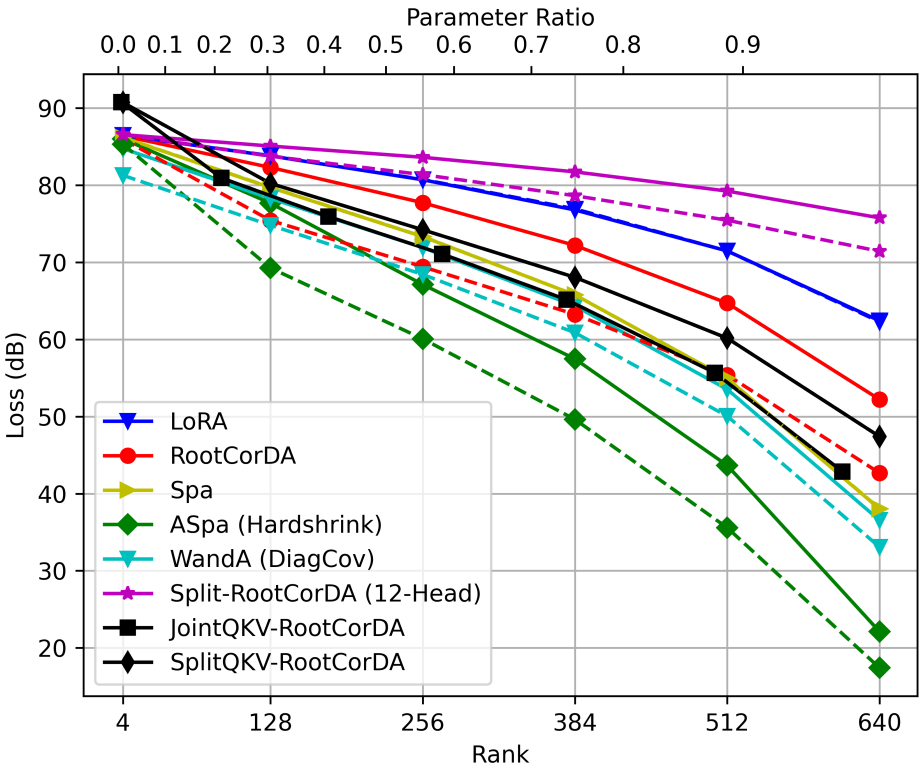


Figure 7: Joint-QKV vs split-QKV approximation.

## Split-Head Compression

Typically, the weight matrix is split into multiple heads, what happens if we use split-head activation loss? Consider

$$\mathcal{L} = \sum_{i=1}^{h} \|W_i X - B_i A_i X\|^2 \tag{51}$$

$$= \left\| \underbrace{\begin{bmatrix} W_1 \\ \ddots \\ W_h \end{bmatrix}}_{W \in \mathbb{R}^{d' \times d}} X - \underbrace{\operatorname{diag}[B_1, \ldots, B_h]}_{B \in \mathbb{R}^{d' \times r}} \underbrace{\begin{bmatrix} A_1 \\ \ddots \\ A_h \end{bmatrix}}_{A \in \mathbb{R}^{r \times d}} X \right\|^2. \tag{52}$$

where $W_i \in \mathbb{R}^{d'/h \times d}$, $B_i \in \mathbb{R}^{d'/h \times r/h}$, $A_i \in \mathbb{R}^{r/h \times d}$ are the $i$th head approximation with $h$ being the number of heads. The solution is individual SVD of $W_i C^{\frac{1}{2}}$. However, as decompression matrix $B$ is sparse block diagonal, it is not efficient than joint head approximation. It is shown in Fig. 8.

## Multi-Head Attention (MHA)

Typically, the attention projection uses a square weight $d' = d$, but it is divided into multiple heads such that:

$$W_{\mathrm{q}} = \begin{bmatrix} W_{\mathrm{q},1} \\ W_{\mathrm{q},2} \\ \vdots \\ W_{\mathrm{q},h} \end{bmatrix} \in \mathbb{R}^{d \times d}, \qquad W_{\mathrm{k}} = \begin{bmatrix} W_{\mathrm{k},1} \\ W_{\mathrm{k},2} \\ \vdots \\ W_{\mathrm{k},h} \end{bmatrix} \in \mathbb{R}^{d \times d}, \tag{53}$$

where $W_{\mathrm{q},i} \in \mathbb{R}^{d/h \times d}$ and $W_{\mathrm{k},i} \in \mathbb{R}^{d/h \times d}$ are the $i$th head projection weights, and $h$ is number of heads. The analysis so far is still valid for per-head low-rank approximation to regard $d' = d/h$.

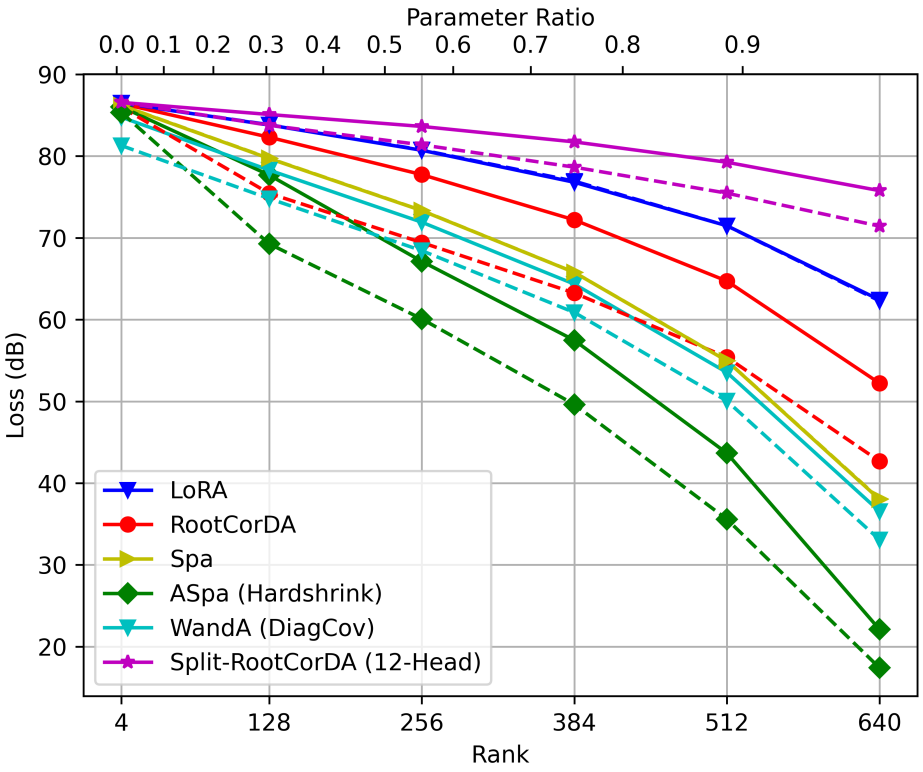


Figure 8: Split-head activation-aware approximation had terrible performance.

However, joint-head low-rank approximation may have a benefit over head-wise low-rank approximation. The $i$th head attention map is given as:

$$M_i = X^\top W_{\mathrm{q},i}^\top W_{\mathrm{k},i} X. \tag{54}$$

When we jointly decompose the low-rank with independent rank $r_\mathrm{q}$ and $r_\mathrm{k}$: $\hat{W}_\mathrm{q} = B_\mathrm{q} A_\mathrm{q}$ and $\hat{W}_\mathrm{k} = B_\mathrm{k} A_\mathrm{k}$ for $B_\mathrm{q}, A_\mathrm{q}^\top \in \mathbb{R}^{d \times r_\mathrm{q}}$, $B_\mathrm{k}, A_\mathrm{k}^\top \in \mathbb{R}^{d \times r_\mathrm{k}}$, we can write $\hat{W}_{\mathrm{q},i} = B_{\mathrm{q},i} A_\mathrm{q}$ and $\hat{W}_{\mathrm{k},i} = B_{\mathrm{k},i} A_\mathrm{k}$ for $B_{\mathrm{q},i} \in \mathbb{R}^{d/h \times r_\mathrm{q}}$ and $B_{\mathrm{k},i} \in \mathbb{R}^{d/h \times r_\mathrm{k}}$, i.e., the compression matrix $A$ is shared, and decompression matrix $B$ is individual over multiple heads. It suggests that the rank $r_\mathrm{q}$ and $r_\mathrm{k}$ should not be lower than $d/h$, otherwise some heads can be redundant.

For arbitrary $X$ (worst-case), we may consider minimizing

$$\mathcal{L} = \sum_{i=1}^{h} \| \underbrace{W_{\mathrm{q},i}^\top W_{\mathrm{k},i}}_{G_i \in \mathbb{R}^{d \times d}} - A_\mathrm{q}^\top \underbrace{B_{\mathrm{q},i}^\top B_{\mathrm{k},i}}_{H_i \in \mathbb{R}^{r_\mathrm{q} \times r_\mathrm{k}}} A_\mathrm{k} \|^2 \tag{55}$$

$$= \sum_{i=1}^{h} \| G_i - A_\mathrm{q}^\top H_i A_\mathrm{k} \|^2. \tag{56}$$

Note that $G_i = W_{\mathrm{q},i}^\top W_{\mathrm{k},i} \in \mathbb{R}^{d \times d}$ is at most of rank $d/h$. And the rank of $H_i = B_{\mathrm{q},i}^\top B_{\mathrm{k},i} \in \mathbb{R}^{r_\mathrm{q} \times r_\mathrm{k}}$ is not greater than $r = \min(r_\mathrm{q}, r_\mathrm{k}, d/h)$.

Note that there is no loss in generality to restrict that $A_\mathrm{q}$ and $A_\mathrm{k}$ are ortho-normal, i.e., $A_\mathrm{q} A_\mathrm{q}^\top = I_{r_\mathrm{q}}$ and $A_\mathrm{k} A_\mathrm{k}^\top = I_{r_\mathrm{k}}$, as a

full matrix $H_i$ can absorb any non-ortho-normal impact. Then, we have

$$\mathcal{L} = \sum_{i=1}^{h} \mathrm{tr}[G_i G_i^\top + A_\mathrm{q}^\top H_i \underbrace{A_\mathrm{k} A_\mathrm{k}^\top}_{I_{r_\mathrm{k}}} H_i^\top A_\mathrm{q} - A_\mathrm{q}^\top H_i A_\mathrm{k} G_i^\top - G_i A_\mathrm{k}^\top H_i^\top A_\mathrm{q}] \tag{57}$$

$$= \sum_{i=1}^{h} \|G_i\|^2 + \mathrm{tr}[H_i H_i^\top \underbrace{A_\mathrm{q} A_\mathrm{q}^\top}_{I_{r_\mathrm{q}}}] - \mathrm{tr}[H_i A_\mathrm{k} G_i^\top A_\mathrm{q}^\top + A_\mathrm{q} G_i A_\mathrm{k}^\top H_i^\top] \tag{58}$$

$$= \sum_{i=1}^{h} \|G_i\|^2 + \|H_i\|^2 - 2\mathrm{tr}[H_i A_\mathrm{k} G_i^\top A_\mathrm{q}^\top]. \tag{59}$$

The gradients:

$$\nabla_{H_i} \mathcal{L} = 2H_i - 2A_\mathrm{q} G_i A_\mathrm{k}^\top. \tag{60}$$

The KKT condition for $H_i$ given $A_\mathrm{q}$ and $A_\mathrm{k}$ is thus:

$$H_i = A_\mathrm{q} G_i A_\mathrm{k}^\top \tag{61}$$

$$= A_\mathrm{q} W_{\mathrm{q},i}^\top W_{\mathrm{k},i} A_\mathrm{k}^\top. \tag{62}$$

Putting it back to the loss, we obtain:

$$\mathcal{L} = \sum_{i=1}^{h} \|G_i\|^2 + \|H_i\|^2 - 2\mathrm{tr}[H_i A_\mathrm{k} G_i^\top A_\mathrm{q}^\top] \tag{63}$$

$$= \sum_{i} \|G_i\|^2 + \|A_\mathrm{q} G_i A_\mathrm{k}^\top\|^2 - 2\mathrm{tr}[A_\mathrm{q} G_i A_\mathrm{k}^\top A_\mathrm{k} G_i^\top A_\mathrm{q}^\top] \tag{64}$$

$$= \sum_{i=1}^{h} \|G_i\|^2 + \|A_\mathrm{q} G_i A_\mathrm{k}^\top\|^2 - 2\|A_\mathrm{q} G_i A_\mathrm{k}^\top\|^2 \tag{65}$$

$$= \sum_{i=1}^{h} \|G_i\|^2 - \|A_\mathrm{q} G_i A_\mathrm{k}^\top\|^2. \tag{66}$$

Let's rewrite as

$$\mathcal{L} = \sum_{i} \|G_i\|^2 - \|(A_\mathrm{k} \otimes A_\mathrm{q})\mathrm{vec}[G_i]\|^2 \tag{67}$$

$$= \sum_{i} \|G_i\|^2 - \sum_{i} \|(A_\mathrm{k} \otimes A_\mathrm{q})\mathrm{vec}[G_i]\|^2 \tag{68}$$

$$= \sum_{i} \|G_i\|^2 - \|(A_\mathrm{k} \otimes A_\mathrm{q}) \underbrace{[\mathrm{vec}[G_1], \mathrm{vec}[G_2], \ldots, \mathrm{vec}[G_h]]}_{G \in \mathbb{R}^{dd \times h}} \|^2 \tag{69}$$

$$= \|G\|^2 - \|(A_\mathrm{k} \otimes A_\mathrm{q})G\|^2 \tag{70}$$

$$= \|G\|^2 - \|(I_h \otimes A_\mathrm{k} \otimes A_\mathrm{q})\mathrm{vec}[G]\|^2. \tag{71}$$

This is the 3-mode tensor-rank decomposition problem involving the high-order SVD (HOSVD) for folding $G$ into the size of $d \times d \times h$, but with a restriction that the first mode plane is identity (it may suggest that we may be able to improve by relaxing this constraint).

HOSVD has no simple solution, while alternating method works well in practice. Specifically, each tensor plane is alternatingly obtained by left singular of the unfolded tensor in different axis. Given $A_\mathrm{k}$, the best $A_\mathrm{q}$ is the $r_\mathrm{q}$ left singular:

$$A_\mathrm{q}^\top \leftarrow \mathrm{LeftSingular}_{r_\mathrm{q}}(\underbrace{[G_1 A_\mathrm{k}^\top, G_2 A_\mathrm{k}^\top, \ldots, G_h A_\mathrm{k}^\top]}_{\mathbb{R}^{d \times r_\mathrm{k} h}}) \tag{72}$$

$$= \mathrm{LeftSingular}_{r_\mathrm{q}}(\sum_{i} G_i A_\mathrm{k}^\top A_\mathrm{k} G_i^\top). \tag{73}$$

The loss will be the residual accumulation of the eigenvalues beyond the rank $r_\mathrm{q}$. Then given $A_\mathrm{q}$, the best $A_\mathrm{k}$ is the $r_\mathrm{k}$ left singular:

$$A_\mathrm{k}^\top \leftarrow \mathrm{LeftSingular}_{r_\mathrm{k}}(\underbrace{[G_1^\top A_\mathrm{q}^\top, G_2^\top A_\mathrm{q}^\top, \dots, G_h^\top A_\mathrm{q}^\top]}_{\mathbb{R}^{d \times r_\mathrm{q} h}}) \tag{74}$$

$$= \mathrm{LeftSingular}_{r_\mathrm{k}}(\sum_i G_i^\top A_\mathrm{q}^\top A_\mathrm{q} G_i). \tag{75}$$

The loss will be the residual accumulation of the eigenvalues beyond the rank $r_\mathrm{k}$. Iterating the above often achieves a good solution. NOTE: singular vectors of $\sum_i G_i G_i^\top$ and $\sum_i G_i^\top G_i$ can be a good initialization of $A_\mathrm{q}$ and $A_\mathrm{k}$.

NOTE: the non-uniform choice of ranks $r_\mathrm{q}$ and $r_\mathrm{k}$ can be optimized to minimize the loss, rather than using the same rank. It can be adaptively adjusted from the eigenvalue distributions.

Once we obtained the HOSVD solution for tensor planes $A_\mathrm{q}$ and $A_\mathrm{k}$, the tensor core $H_i \in \mathbb{R}^{r_\mathrm{q} \times r_\mathrm{k}}$ is generated by (62) as $H_i = A_\mathrm{q} G_i A_\mathrm{k}^\top = A_\mathrm{q} W_{\mathrm{q},i}^\top W_{\mathrm{k},i} A_\mathrm{k}^\top$. Given optimized $H_i$, any arbitrary $B_{\mathrm{q},i}$ and $B_{\mathrm{k},i}$ provides the same error as long as it holds:

$$H_i = B_{\mathrm{q},i}^\top B_{\mathrm{k},i}. \tag{76}$$

The solution is

$$B_{\mathrm{q},i} = J_i^\top W_{\mathrm{q},i} A_\mathrm{q}^\top, \tag{77}$$

$$B_{\mathrm{k},i} = J_i^+ W_{\mathrm{k},i} A_\mathrm{k}^\top, \tag{78}$$

where $J_i \in \mathbb{R}^{d/h \times d/h}$ is any arbitrary full-rank matrix of our choice. The most natural choice is identity: $J_i = I_{d/h}$.

Nevertheless, another simple solution will be

$$B_{\mathrm{q},i} = I_{d/h \times r_\mathrm{q}}, \tag{79}$$

$$B_{\mathrm{k},i} = \begin{bmatrix} H_i \\ O_{(d/h - r_\mathrm{q}) \times r_\mathrm{k}} \end{bmatrix}, \tag{80}$$

when $r_\mathrm{q} \leq d/h$. This has a benefit that query decompression does not require any memory and key decompression is a block sparse.

Another solution will be

$$B_{\mathrm{q},i} = \begin{bmatrix} H_i^\top \\ O_{(d/h - r_\mathrm{k}) \times r_\mathrm{q}} \end{bmatrix}, \tag{81}$$

$$B_{\mathrm{k},i} = I_{d/h \times r_\mathrm{k}}, \tag{82}$$

when $r_\mathrm{k} \leq d/h$. Similar benefit, but probably removing the requirement of query decompression is more beneficial than key decompression in practice.

When $r_\mathrm{k}, r_\mathrm{q} \geq d/h$ (in most case?), we can select $J_i$ such that $B_{\mathrm{q},i}$ or $B_{\mathrm{k},i}$ is block matrix to save $(d/h)^2$ parameters from $(r_\mathrm{q} + r_\mathrm{k})d/h$. For this case, fine-tuning two decompression $B_{\mathrm{q},i}$ and $B_{\mathrm{k},i}$ rather than a product $H_i$ will be more parameter-efficient.

### Activation-Aware Multi-Head Latent Attention

Consider the loss:

$$\mathcal{L} = \sum_i \|M_i - \hat{M}_i\|^2 \tag{83}$$

$$= \sum_i \|X^\top \underbrace{(G_i - A_{\mathrm{q}}^\top H_i A_{\mathrm{k}})}_{\Delta_i \in \mathbb{R}^{d\times d}} X\|^2 \tag{84}$$

$$= \sum_i \mathrm{tr}[X^\top \Delta_i X X^\top \Delta_i^\top X] \tag{85}$$

$$= \sum_i \mathrm{tr}[\Delta_i \underbrace{XX^\top}_{C\in\mathbb{R}^{d\times d}} \Delta_i^\top XX^\top] \tag{86}$$

$$= \sum_i \mathrm{tr}[\Delta_i C \Delta_i^\top C] \tag{87}$$

$$= \sum_i \mathrm{tr}[\Delta_i C^{\frac{1}{2}} C^{\frac{1}{2}} \Delta_i^\top C^{\frac{1}{2}} C^{\frac{1}{2}}] \tag{88}$$

$$= \sum_i \|C^{\frac{1}{2}} \Delta_i C^{\frac{1}{2}}\|^2 \tag{89}$$

$$= \sum_i \| \underbrace{C^{\frac{1}{2}} G_i C^{\frac{1}{2}}}_{G_i'} - \underbrace{C^{\frac{1}{2}} A_{\mathrm{q}}^\top}_{A_{\mathrm{q}}'^\top} H_i \underbrace{A_{\mathrm{k}} C^{\frac{1}{2}}}_{A_{\mathrm{k}}'} \|^2, \tag{90}$$

where $C$ is a positive semi-definite of rank no greater than $\min(d, l)$.

In fact, the solution is same as the case without $X$ comparing (56) and (90), where we can regard as

$$G_i \to G_i' = C^{\frac{1}{2}} G_i C^{\frac{1}{2}}, \tag{91}$$

$$A_{\mathrm{q}} \to A_{\mathrm{q}}' = A_{\mathrm{q}} C^{\frac{1}{2}}, \tag{92}$$

$$A_{\mathrm{k}} \to A_{\mathrm{k}}' = A_{\mathrm{k}} C^{\frac{1}{2}}. \tag{93}$$

Here, we can consider $A_{\mathrm{q}} C^{\frac{1}{2}}$ and $A_{\mathrm{k}} C^{\frac{1}{2}}$ are instead orthogonal, and thus the solution can be given by HOSVD likewise.

Fig. 9 shows the comparison between adaptive and non-adaptive distillation with activation/attention-aware methods.

### Bias Update

Some LLMs use bias for QKV. For this case, we need to modify the bias term as well. We have

$$\mathcal{L} = \sum_i \|(W_{\mathrm{q},i} X + b_{\mathrm{q},i} 1^\top)^\top (W_{\mathrm{k},i} X + b_{\mathrm{k},i} 1^\top) - (\hat{W}_{\mathrm{q},i} X + \hat{b}_{\mathrm{q},i} 1^\top)^\top (\hat{W}_{\mathrm{k},i} X + \hat{b}_{\mathrm{k},i} 1^\top)\|^2 \tag{94}$$

$$= \sum_i \Big\| \Big( \underbrace{[W_{\mathrm{q},i} \quad b_{\mathrm{q},i}]}_{W_{\mathrm{q},i}' \in \mathbb{R}^{d/h\times(d+1)}} \underbrace{\begin{bmatrix} X \\ 1^\top \end{bmatrix}}_{X' \in \mathbb{R}^{(d+1)\times l}} \Big)^\top \Big( \underbrace{[W_{\mathrm{k},i} \quad b_{\mathrm{k},i}]}_{W_{\mathrm{k},i}' \in \mathbb{R}^{d/h\times(d+1}} \begin{bmatrix} X \\ 1^\top \end{bmatrix} \Big) - \Big( \underbrace{[\hat{W}_{\mathrm{q},i} \quad \hat{b}_{\mathrm{q},i}]}_{\hat{W}_{\mathrm{q},i}' \in \mathbb{R}^{d/h\times(d+1)}} \begin{bmatrix} X \\ 1^\top \end{bmatrix} \Big)^\top \Big( \underbrace{[\hat{W}_{\mathrm{k},i} \quad \hat{b}_{\mathrm{k},i}]}_{\hat{W}_{\mathrm{k},i}' \in \mathbb{R}^{d/h\times(d+1)}} \begin{bmatrix} X \\ 1^\top \end{bmatrix} \Big) \Big\|^2 \tag{95}$$

$$= \sum_i \|X'^\top (W_{\mathrm{q},i}'^\top W_{\mathrm{k},i}' - \hat{W}_{\mathrm{q},i}'^\top \hat{W}_{\mathrm{k},i}') X'\|^2 \tag{96}$$

$$= \sum_i \|\tilde{C}^{\frac{1}{2}} (W_{\mathrm{q},i}'^\top W_{\mathrm{k},i}' - \hat{W}_{\mathrm{q},i}'^\top \hat{W}_{\mathrm{k},i}') \tilde{C}^{\frac{1}{2}}\|^2 \tag{97}$$

where we have a modified covariance:

$$\tilde{C} = X' X'^\top \in \mathbb{R}^{(d+1)\times(d+1)} \tag{98}$$

$$= \begin{bmatrix} X \\ 1^\top \end{bmatrix} \begin{bmatrix} X^\top & 1 \end{bmatrix} \tag{99}$$

$$= \begin{bmatrix} lC & l\mu \\ l\mu^\top & l \end{bmatrix} = l \begin{bmatrix} C & \mu \\ \mu^\top & 1 \end{bmatrix} \tag{100}$$

$$= l \begin{bmatrix} C^{\frac{1}{2}} & O \\ \mu^\top C^{\frac{-1}{2}} & (1 - \mu^\top C^+ \mu)^{\frac{1}{2}} \end{bmatrix} \begin{bmatrix} C^{\frac{1}{2}} & C^{\frac{-1}{2}} \mu \\ O & (1 - \mu^\top C^+ \mu)^{\frac{1}{2}} \end{bmatrix} \tag{101}$$

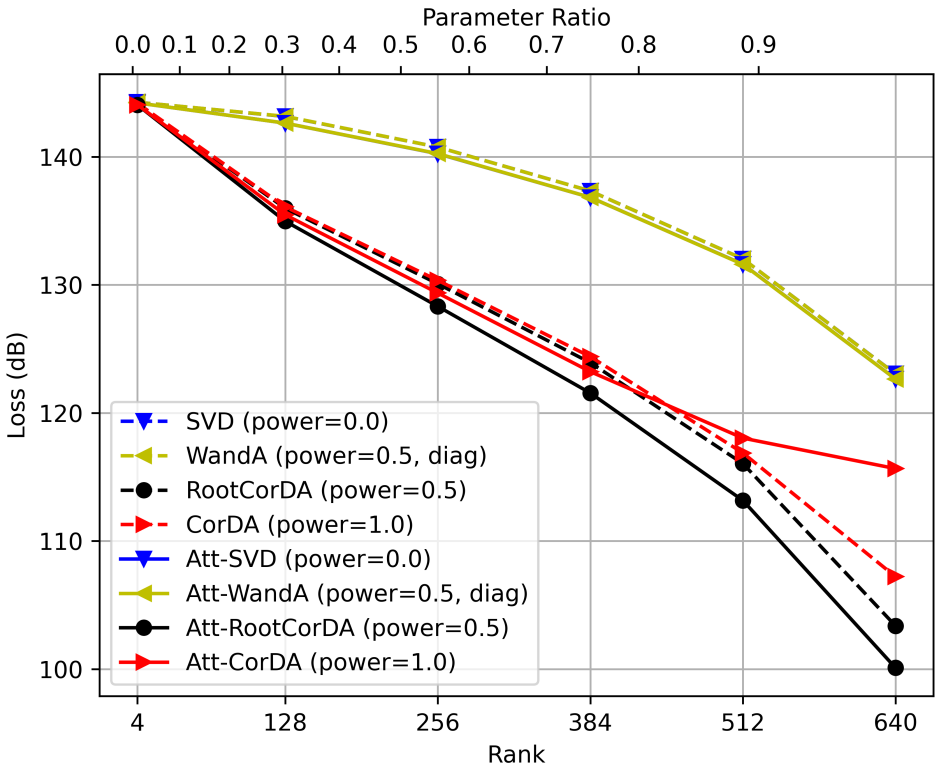


Figure 9: Attention-Aware vs. Activation-Aware Approximation. Loss is attention map error. Random query/key projections with Wishart sample correlation (0.9 decaying). WandA uses diagonal correlation.

where we assume $C$ is normalized as $C = XX^\top/l$, and $\mu \in \mathbb{R}^{d\times 1}$ is a mean of input tokens: $\mu = X1/l$. Then, we can omit $l$. Similar format but it cannot be solved by the same way as we have a structured low-rank expression:

$$\hat{W}'^\top_{\mathrm{q},i}\hat{W}'_{\mathrm{k},i} = \begin{bmatrix} A_\mathrm{q}^\top B_{\mathrm{q},i}^\top \\ \hat{b}_{\mathrm{q},i}^\top \end{bmatrix} \begin{bmatrix} B_{\mathrm{k},i}A_\mathrm{k} & \hat{b}_{\mathrm{k},i} \end{bmatrix} \tag{102}$$

$$= \underbrace{\begin{bmatrix} A_\mathrm{q}^\top & O_{d\times 1} \\ O_{1\times r_\mathrm{q}} & 1 \end{bmatrix}}_{A'^\top_\mathrm{q}\in\mathbb{R}^{(d+1)\times(r_\mathrm{q}+1)}} \overbrace{\underbrace{\begin{bmatrix} B_{\mathrm{q},i}^\top \\ \hat{b}_{\mathrm{q},i}^\top \end{bmatrix}}_{B'^\top_{\mathrm{q},i}\in\mathbb{R}^{(r_\mathrm{q}+1)\times d}} \underbrace{\begin{bmatrix} B_{\mathrm{k},i} & \hat{b}_{\mathrm{k},i} \end{bmatrix}}_{B'_{\mathrm{k},i}\in\mathbb{R}^{d\times(r_\mathrm{k}+1)}}}^{H'_i\in\mathbb{R}^{(r_\mathrm{q}+1)\times(r_\mathrm{k}+1)}} \underbrace{\begin{bmatrix} A_\mathrm{k} & O_{r_\mathrm{k}\times 1} \\ O_{1\times d} & 1 \end{bmatrix}}_{A'_\mathrm{k}\in\mathbb{R}^{(r_\mathrm{k}+1)\times(d+1)}}. \tag{103}$$

We may use the HOSVD to decompose with one more rank for bias, while the compression matrix $A'_\mathrm{q}$ and $A'_\mathrm{k}$ needs to be a particular format. Nonetheless, we can modify the bias by the KKT condition:

$$A'_\mathrm{q}\tilde{C}G_i\tilde{C}A'^\top_\mathrm{k} = A'_\mathrm{q}\tilde{C}A'^\top_\mathrm{q}H'_iA'_\mathrm{k}\tilde{C}A'^\top_\mathrm{k}. \tag{104}$$

Hence we have

$$H'_i = (A'_\mathrm{q}\tilde{C}A'^\top_\mathrm{q})^+A'_\mathrm{q}\tilde{C}G_i\tilde{C}A'^\top_\mathrm{k}(A'_\mathrm{k}\tilde{C}A'^\top_\mathrm{k})^+ \tag{105}$$

$$= \underbrace{(A'_\mathrm{q}\tilde{C}A'^\top_\mathrm{q})^+A'_\mathrm{q}\tilde{C}W'_{\mathrm{q},i}}_{B_{\mathrm{q},i}^\top}\underbrace{W'_{\mathrm{k},i}\tilde{C}A'^\top_\mathrm{k}(A'_\mathrm{k}\tilde{C}A'^\top_\mathrm{k})^+}_{B_{\mathrm{k},i}}. \tag{106}$$

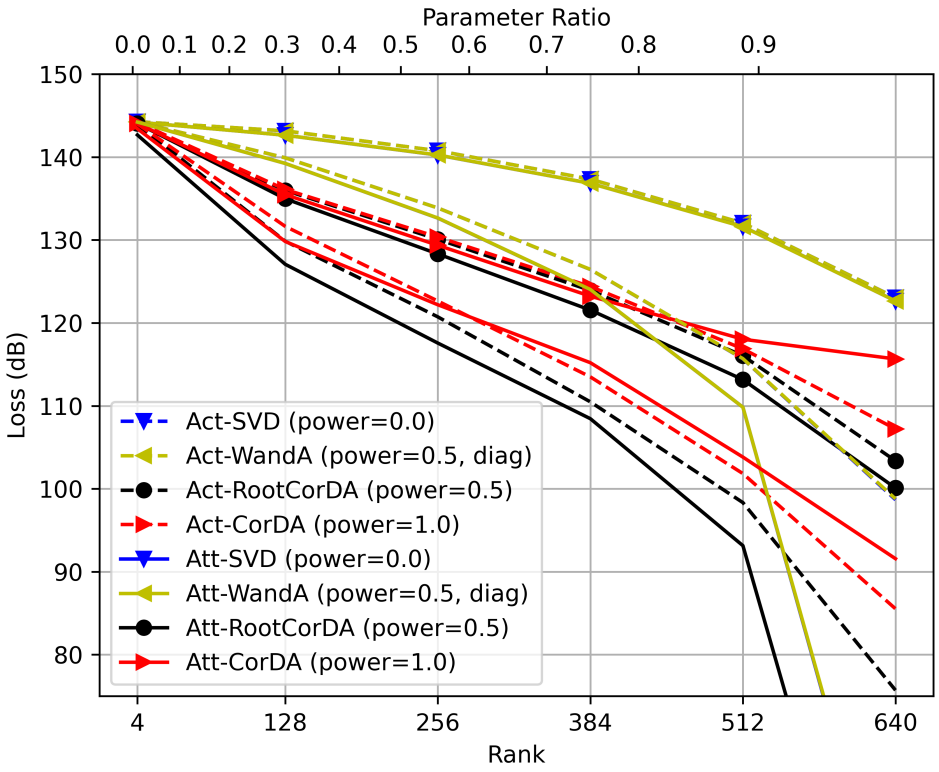


Figure 10: Sparse approximation for Attention-Aware vs. Activation-Aware distillation. No markers are sparse approximation. Sparse is better than low-rank.

Thus, given optimized $A_{\mathrm{q}}$ and $A_{\mathrm{k}}$, we have optimized decompression matrix with updated bias:

$$
\begin{aligned}
B'_{\mathrm{q},i} &= \begin{bmatrix} B_{\mathrm{q},i} & \hat{b}_{\mathrm{q},i} \end{bmatrix} && (107)\\
&= J_i^\top W'_{\mathrm{q},i} \tilde{C} A_{\mathrm{q}}'^\top (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ && (108)\\
&= J_i^\top \begin{bmatrix} W_{\mathrm{q},i} & b_{\mathrm{q},i} \end{bmatrix} \tilde{C} A_{\mathrm{q}}'^\top (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ && (109)\\
&= J_i^\top \begin{bmatrix} W_{\mathrm{q},i} & b_{\mathrm{q},i} \end{bmatrix} \begin{bmatrix} CA_{\mathrm{q}}^\top & \mu \\ \mu^\top A_{\mathrm{q}}^\top & 1 \end{bmatrix} \begin{bmatrix} A_{\mathrm{q}} C A_{\mathrm{q}}^\top & A_{\mathrm{q}}\mu \\ \mu^\top A_{\mathrm{q}}^\top & 1 \end{bmatrix}^+ && (110)\\
&= J_i^\top \begin{bmatrix} W_{\mathrm{q},i} & b_{\mathrm{q},i} \end{bmatrix} \begin{bmatrix} CA_{\mathrm{q}}^\top & \mu \\ \mu^\top A_{\mathrm{q}}^\top & 1 \end{bmatrix} \begin{bmatrix} I & O \\ -\mu^\top A_{\mathrm{q}}^\top & 1 \end{bmatrix} \begin{bmatrix} (A_{\mathrm{q}} C A_{\mathrm{q}}^\top - A_{\mathrm{q}}\mu\mu^\top A_{\mathrm{q}}^\top)^+ & O \\ O & 1 \end{bmatrix} \begin{bmatrix} I & -A_{\mathrm{q}}\mu \\ O & 1 \end{bmatrix} && (111)\\
&= J_i^\top \begin{bmatrix} W_{\mathrm{q},i} & b_{\mathrm{q},i} \end{bmatrix} \begin{bmatrix} (C-\mu\mu^\top)A_{\mathrm{q}}^\top (A_{\mathrm{q}} C A_{\mathrm{q}}^\top - A_{\mathrm{q}}\mu\mu^\top A_{\mathrm{q}}^\top)^+ & -(C-\mu\mu^\top)A_{\mathrm{q}}^\top (A_{\mathrm{q}} C A_{\mathrm{q}}^\top - A_{\mathrm{q}}\mu\mu^\top A_{\mathrm{q}}^\top)^+ A_{\mathrm{q}}\mu + \mu \\ O & 1 \end{bmatrix} && (112)\\
&= J_i^\top \begin{bmatrix} W_{\mathrm{q},i}(C-\mu\mu^\top)A_{\mathrm{q}}^\top (A_{\mathrm{q}} C A_{\mathrm{q}}^\top - A_{\mathrm{q}}\mu\mu^\top A_{\mathrm{q}}^\top)^+ & -W_{\mathrm{q},i}(C-\mu\mu^\top)A_{\mathrm{q}}^\top (A_{\mathrm{q}} C A_{\mathrm{q}}^\top - A_{\mathrm{q}}\mu\mu^\top A_{\mathrm{q}}^\top)^+ A_{\mathrm{q}}\mu + W_{\mathrm{q},i}\mu + b_{\mathrm{q},i} \end{bmatrix}. && (113)
\end{aligned}
$$

It gives the bias modifications:

$$
\begin{aligned}
\hat{b}_{\mathrm{q}} &= \mathrm{diag}[J_i]\big(b_{\mathrm{q}} + W_{\mathrm{q}}\mu - W_{\mathrm{q}}(C-\mu\mu^\top)A_{\mathrm{q}}^\top (A_{\mathrm{q}} C A_{\mathrm{q}}^\top - A_{\mathrm{q}}\mu\mu^\top A_{\mathrm{q}}^\top)^+ A_{\mathrm{q}}\mu\big), && (114)\\
\hat{b}_{\mathrm{k}} &= \mathrm{diag}[J_i^+]\big(b_{\mathrm{k}} + W_{\mathrm{k}}\mu - W_{\mathrm{k}}(C-\mu\mu^\top)A_{\mathrm{k}}^\top (A_{\mathrm{k}} C A_{\mathrm{k}}^\top - A_{\mathrm{k}}\mu\mu^\top A_{\mathrm{k}}^\top)^+ A_{\mathrm{k}}\mu\big). && (115)
\end{aligned}
$$

We define the centered auto-correlation:

$$
C_0 = C - \mu\mu^\top. \tag{116}
$$

Then, we assume that the optimal compression matrices $A_{\mathrm{q}}$ and $A_{\mathrm{k}}$ are orthogonal on $C_0^{\frac{1}{2}}$:

$$A_{\mathrm{q}} C_0 A_{\mathrm{q}}^\top = I_{r_{\mathrm{q}}}, \tag{117}$$

$$A_{\mathrm{k}} C_0 A_{\mathrm{k}}^\top = I_{r_{\mathrm{k}}}. \tag{118}$$

In this case, the bias modification can reduce to

$$\hat{b}_{\mathrm{q}} = \mathrm{diag}[J_i]\big(b_{\mathrm{q}} + W_{\mathrm{q}}\mu - W_{\mathrm{q}} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}}\mu\big), \tag{119}$$

$$\hat{b}_{\mathrm{k}} = \mathrm{diag}[J_i^+]\big(b_{\mathrm{k}} + W_{\mathrm{k}}\mu - W_{\mathrm{k}} C_0 A_{\mathrm{k}}^\top A_{\mathrm{k}}\mu\big). \tag{120}$$

For this case, we have

$$A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}^\top = \begin{bmatrix} A_{\mathrm{q}} C A_{\mathrm{q}}^\top & A_{\mathrm{q}}\mu \\ \mu^\top A_{\mathrm{q}}^\top & 1 \end{bmatrix}, \tag{121}$$

$$(A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}^\top)^+ = \begin{bmatrix} I & -A_{\mathrm{q}}\mu \\ -\mu^\top A_{\mathrm{q}}^\top & 1 + \mu^\top A_{\mathrm{q}}^\top A_{\mathrm{q}}\mu \end{bmatrix}, \tag{122}$$

$$A'_{\mathrm{q}} (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A_{\mathrm{q}}'^\top = \begin{bmatrix} A_{\mathrm{q}}^\top A_{\mathrm{q}} & -A_{\mathrm{q}}^\top A_{\mathrm{q}}\mu \\ -\mu^\top A_{\mathrm{q}}^\top A_{\mathrm{q}} & 1 + \mu^\top A_{\mathrm{q}}^\top A_{\mathrm{q}}\mu \end{bmatrix}, \tag{123}$$

$$A'_{\mathrm{q}} (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A_{\mathrm{q}}'^\top \tilde{C} = \begin{bmatrix} A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 & O \\ \mu^\top - \mu^\top A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 & 1 \end{bmatrix}, \tag{124}$$

$$\tilde{C} A'_{\mathrm{q}} (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A_{\mathrm{q}}'^\top = \begin{bmatrix} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} & \mu - C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}}\mu \\ O & 1 \end{bmatrix}, \tag{125}$$

$$\tilde{C} A'_{\mathrm{q}} (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A_{\mathrm{q}}'^\top \tilde{C} = \begin{bmatrix} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 + \mu\mu^\top & \mu \\ \mu^\top & 1 \end{bmatrix}. \tag{126}$$

Plugging the optimized $H_i$, the loss is expressed as

$$\mathcal{L} = \sum_i \big\| \tilde{C}^{\frac{1}{2}} W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C}^{\frac{1}{2}} - \tilde{C}^{\frac{1}{2}} A_{\mathrm{q}}'^\top (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A'_{\mathrm{q}} \tilde{C} W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C} A_{\mathrm{k}}'^\top (A'_{\mathrm{k}} \tilde{C} A_{\mathrm{k}}'^\top)^+ A'_{\mathrm{k}} \tilde{C}^{\frac{1}{2}} \big\|^2 \tag{127}$$

$$= \sum_i \|\tilde{C}^{\frac{1}{2}} W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C}^{\frac{1}{2}}|^2 - \|\tilde{C}^{\frac{1}{2}} A_{\mathrm{q}}^\top H_i A_{\mathrm{k}} \tilde{C}^{\frac{1}{2}}\|^2 \tag{128}$$

$$= \sum_i \|\tilde{C}^{\frac{1}{2}} W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C}^{\frac{1}{2}}|^2 - \|\tilde{C}^{\frac{1}{2}} A_{\mathrm{q}}'^\top (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A'_{\mathrm{q}} \tilde{C} W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C} A_{\mathrm{k}}'^\top (A'_{\mathrm{k}} \tilde{C} A_{\mathrm{k}}'^\top)^+ A'_{\mathrm{k}} \tilde{C}^{\frac{1}{2}} \|^2 \tag{129}$$

$$\tag{130}$$

$$= \sum_i \|\tilde{C}^{\frac{1}{2}} W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C}^{\frac{1}{2}}|^2 - \mathrm{tr}[\tilde{C} A_{\mathrm{q}}'^\top (A'_{\mathrm{q}} \tilde{C} A_{\mathrm{q}}'^\top)^+ A'_{\mathrm{q}} \tilde{C} \underbrace{W_{\mathrm{q},i}'^\top W'_{\mathrm{k},i} \tilde{C} A_{\mathrm{k}}'^\top (A'_{\mathrm{k}} \tilde{C} A_{\mathrm{k}}'^\top)^+ A'_{\mathrm{k}} \tilde{C} W_{\mathrm{k},i}'^\top W'_{\mathrm{q},i}}_{G_{\mathrm{q},i} \in \mathbb{R}^{(d+1)\times(d+1)}}]. \tag{131}$$

Focusing on optimizing $A_{\mathrm{q}}$, the second term will be

$$\sum_i \mathrm{tr}\Big[\Big( \begin{bmatrix} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 & O \\ O & 0 \end{bmatrix} + \begin{bmatrix} \mu \\ 1 \end{bmatrix} \begin{bmatrix} \mu \\ 1 \end{bmatrix}^\top \Big) G_{\mathrm{q},i} \Big] = \sum_i \mathrm{tr}\Big[ \begin{bmatrix} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 & O \\ O & 0 \end{bmatrix} G_{\mathrm{q},i} \Big] + \text{c.c.} \tag{132}$$

$$= \mathrm{tr}\Big[ C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 I_{d\times(d+1)} \sum_i G_{\mathrm{q},i} I_{(d+1)\times d} \Big] + \text{c.c.} \tag{133}$$

$$= \|A_{\mathrm{q}} C_0 (I_{d\times(d+1)} \sum G_{\mathrm{q},i} I_{(d+1)\times d})^{\frac{1}{2}}\|^2 + \text{c.c.}. \tag{134}$$

Hence the optimal $A_{\mathrm{q}}$ is the right-singular vectors:

$$A_{\mathrm{q}} C_0^{\frac{1}{2}} = \mathsf{RightSingular}_{r_{\mathrm{q}}}\big[ C_0^{\frac{1}{2}} I_{d\times(d+1)} (\sum_i G_{\mathrm{q},i}) I_{(d+1)\times d} C_0^{\frac{1}{2}} \big]. \tag{135}$$

In fact, we can re-write $G_{\mathrm{q},i}$ as

$$G_{\mathrm{q},i} = \begin{bmatrix} W_{\mathrm{q},i}^\top \\ b_{\mathrm{q},i}^\top \end{bmatrix} [W_{\mathrm{k},i} \quad b_{\mathrm{k},i}] \Big( \begin{bmatrix} C_0 A_{\mathrm{k}}^\top A_{\mathrm{k}} C_0 & O \\ O & 0 \end{bmatrix} + \begin{bmatrix} \mu \\ 1 \end{bmatrix} \begin{bmatrix} \mu \\ 1 \end{bmatrix}^\top \Big) \begin{bmatrix} W_{\mathrm{k},i}^\top \\ b_{\mathrm{k},i}^\top \end{bmatrix} [W_{\mathrm{q},i} \quad b_{\mathrm{q},i}] \tag{136}$$

$$= \begin{bmatrix} W_{\mathrm{q},i}^\top \\ b_{\mathrm{q},i}^\top \end{bmatrix} \Big( W_{\mathrm{k},i} C_0 A_{\mathrm{k}}^\top A_{\mathrm{k}} C_0 W_{\mathrm{k},i}^\top + (W_{\mathrm{k},i}\mu + b_{\mathrm{k},i})(W_{\mathrm{k},i}\mu + b_{\mathrm{k},i})^\top \Big) [W_{\mathrm{q},i} \quad b_{\mathrm{q},i}]. \tag{137}$$

Hence, we have

$$A_{\mathrm{q}}C_0^{\frac{1}{2}} = \mathsf{RightSingular}_{r_{\mathrm{k}}}\big[\sum_i C_0^{\frac{1}{2}} W_{\mathrm{q},i}^\top W_{\mathrm{k},i} C_0 A_{\mathrm{k}}^\top A_{\mathrm{k}} C_0 W_{\mathrm{k},i}^\top W_{\mathrm{q},i} C_0^{\frac{1}{2}} + \sum_i C_0^{\frac{1}{2}} W_{\mathrm{q},i}^\top (W_{\mathrm{k},i}\mu + b_{\mathrm{k},i})(W_{\mathrm{k},i}\mu + b_{\mathrm{k},i})^\top W_{\mathrm{q},i} C_0^{\frac{1}{2}}\big]. \tag{138}$$

The first term is the solution if no bias and mean are present.

Similarly the solution for $A_{\mathrm{k}}$ is given by

$$A_{\mathrm{k}}C_0^{\frac{1}{2}} = \mathsf{RightSingular}_{r_{\mathrm{k}}}\big[C_0^{\frac{1}{2}} I_{d\times(d+1)}(\sum_i G_{\mathrm{k},i}) I_{(d+1)\times d} C_0^{\frac{1}{2}}\big] \tag{139}$$

$$= \mathsf{RightSingular}_{r_{\mathrm{q}}}\big[\sum_i C_0^{\frac{1}{2}} W_{\mathrm{k},i}^\top W_{\mathrm{q},i} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 W_{\mathrm{q},i}^\top W_{\mathrm{k},i} C_0^{\frac{1}{2}} + \sum_i C_0^{\frac{1}{2}} W_{\mathrm{k},i}^\top (W_{\mathrm{q},i}\mu + b_{\mathrm{q},i})(W_{\mathrm{q},i}\mu + b_{\mathrm{q},i})^\top W_{\mathrm{k},i} C_0^{\frac{1}{2}}\big]. \tag{140}$$

where

$$G_{\mathrm{k},i} = W_{\mathrm{k},i}'^\top W_{\mathrm{q},i}' \tilde{C} A_{\mathrm{q}}'^\top (A_{\mathrm{q}}' \tilde{C} A_{\mathrm{q}}'^\top)^+ A_{\mathrm{q}}' \tilde{C} W_{\mathrm{q},i}'^\top W_{\mathrm{k},i}' \tag{141}$$

$$= \begin{bmatrix} W_{\mathrm{k},i}^\top \\ b_{\mathrm{k},i}^\top \end{bmatrix} W_{\mathrm{q},i} C_0 A_{\mathrm{q}}^\top A_{\mathrm{q}} C_0 W_{\mathrm{q},i}^\top \begin{bmatrix} W_{\mathrm{k},i} & b_{\mathrm{k},i} \end{bmatrix} + \begin{bmatrix} W_{\mathrm{k},i}^\top \\ b_{\mathrm{k},i}^\top \end{bmatrix} (W_{\mathrm{q},i}\mu + b_{\mathrm{q},i})(W_{\mathrm{q},i}\mu + b_{\mathrm{q},i})^\top \begin{bmatrix} W_{\mathrm{k},i} & b_{\mathrm{k},i} \end{bmatrix}. \tag{142}$$

## Grouped Query Attention (GQA)

MHA (e.g., for Llama-2) uses $h$-heads for query, key, and value. However, Llama-3 uses grouped query attention (GQA), where the number of heads for key and value are smaller than the number of heads for query. Let $n_{\mathrm{q}}$ be the query group size. Then, the number of query heads is $n_{\mathrm{q}}h$, whereas $h$ is the number of KV heads. Suppose $n_{\mathrm{q}}$ is the integer so that simple repetition can be used. Q and K projections:

$$W_{\mathrm{q}} = \begin{bmatrix} W_{\mathrm{q},1} \\ W_{\mathrm{q},2} \\ \vdots \\ W_{\mathrm{q},n_{\mathrm{q}}h} \end{bmatrix} \in \mathbb{R}^{n_{\mathrm{q}}hd'\times d}, \qquad W_{\mathrm{k}} = \begin{bmatrix} W_{\mathrm{k},1} \\ W_{\mathrm{k},2} \\ \vdots \\ W_{\mathrm{k},h} \end{bmatrix} \in \mathbb{R}^{hd'\times d}, \tag{143}$$

for $W_{\mathrm{q},i} \in \mathbb{R}^{d'\times d}$, $W_{\mathrm{k},i} \in \mathbb{R}^{d'\times d}$ with the head dimension $d'$. Llama-3 uses repeat-interleave to match the number of heads by repeating the KV projections $n_{\mathrm{q}}$-times:

$$W_{\mathrm{k}}' = \begin{bmatrix} W_{\mathrm{k},1} \\ W_{\mathrm{k},1} \\ \vdots \\ W_{\mathrm{k},1} \\ \vdots \\ W_{\mathrm{k},h} \end{bmatrix} \in \mathbb{R}^{n_{\mathrm{q}}hd'\times d}. \tag{144}$$

For such GQA, we have attention map for the $j$th head in the $i$the group ($j \in \mathbb{Z}_h^+$, $i \in \mathbb{Z}_{n_{\mathrm{q}}}^+$):

$$M_{i,j} = X^\top W_{\mathrm{q},i,j}^\top W_{\mathrm{k},i} X, \tag{145}$$

where we use an index convention: $W_{\mathrm{q},i,j} = W_{\mathrm{q},in_{\mathrm{q}}+j}$.

Consider the loss:

$$\mathcal{L} = \sum_{i,j} \|M_{i,j} - \hat{M}_{i,j}\|^2 \tag{146}$$

$$= \sum_{i,j} \|X^\top \underbrace{(\overbrace{W_{\mathrm{q},i,j}^\top W_{\mathrm{k},i}}^{G_{i,j}\in\mathbb{R}^{d\times d}} - A_{\mathrm{q}}^\top \overbrace{B_{\mathrm{q},i,j}^\top B_{\mathrm{k},i}}^{H_{i,j}\in\mathbb{R}^{r_{\mathrm{q}}\times r_{\mathrm{k}}}} A_{\mathrm{k}})}_{\Delta_{i,j}\in\mathbb{R}^{d\times d}} X\|^2 \tag{147}$$

$$= \sum_{i,j} \|C^{\frac{1}{2}} G_{i,j} C^{\frac{1}{2}} - C^{\frac{1}{2}} A_{\mathrm{q}}^\top H_{i,j} A_{\mathrm{k}} C^{\frac{1}{2}}\|^2. \tag{148}$$

Hence the solution can be obtained with HOSVD likewise MHA in Sec. .

# Positional Encoding

## Additive PE

Consider additive PE for a token $X \in \mathbb{R}^{d \times l}$:

$$X' = X + E, \tag{149}$$

where $E \in \mathbb{R}^{d \times l}$ is a positional embedding matrix. Often it is sinusoidal like

$$E_{i,j} = \exp(\jmath 2\pi f_i j / l), \tag{150}$$

with a predefined frequency $f_i$ for $i \in \mathbb{Z}_d$. Note that complex rotation is not used in typical case, and instead split into $\cos$ and $\sin$. Many work also considered trainable PE (Radford 2018; Kenton and Toutanova 2019).

In this additive PE case, the solution is same by replacing the correlation matrix $C$ with

$$C' = \mathbb{E}_X[X'X'^\top] \tag{151}$$

$$= \mathbb{E}_X[(X+E)(X+E)^\top] \tag{152}$$

$$= C + EE^\top + \mathbb{E}_X[XE^\top + EX^\top]. \tag{153}$$

For zero-mean token case, is reduces to $C + EE^\top$. For static token case, we may use $(X+E)(X+E)^\top$ directly.

Nevertheless, some PE methods (Dai et al. 2019) use different additive PE for query and key individually:

$$X_\mathrm{q} = X + E_\mathrm{q}, \tag{154}$$

$$X_\mathrm{k} = X + E_\mathrm{k}. \tag{155}$$

In this case, the attention map will have bias terms:

$$M_i = X_\mathrm{q}^\top W_{\mathrm{q},i}^\top W_{\mathrm{k},i} X_\mathrm{k} \tag{156}$$

$$= X^\top G_i X + X^\top G_i E_\mathrm{k} + E_\mathrm{q}^\top G_i X + E_\mathrm{q}^\top G_i E_\mathrm{k}. \tag{157}$$

There are many variants to relax them or generalize them.

Consider loss:

$$\mathcal{L} = \sum_i \|X_\mathrm{q}^\top \Delta_i X_\mathrm{k}\|^2 \tag{158}$$

$$= \sum_i \mathrm{tr}[\Delta_i \underbrace{X_\mathrm{k} X_\mathrm{k}^\top}_{C_\mathrm{k} \in \mathbb{R}^{d \times d}} \Delta_i^\top \underbrace{X_\mathrm{q} X_\mathrm{q}^\top}_{C_\mathrm{q} \in \mathbb{R}^{d \times d}}] \tag{159}$$

$$= \sum_i \mathrm{tr}[\Delta_i C_\mathrm{k} \Delta_i^\top C_\mathrm{q}] \tag{160}$$

$$= \sum_i \mathrm{tr}[\Delta_i C_\mathrm{k}^{\frac{1}{2}} C_\mathrm{k}^{\frac{1}{2}} \Delta_i^\top C_\mathrm{q}^{\frac{1}{2}} C_\mathrm{q}^{\frac{1}{2}}] \tag{161}$$

$$= \sum_i \mathrm{tr}[C_\mathrm{q}^{\frac{1}{2}} \Delta_i C_\mathrm{k}^{\frac{1}{2}} C_\mathrm{k}^{\frac{1}{2}} \Delta_i^\top C_\mathrm{q}^{\frac{1}{2}}] \tag{162}$$

$$= \sum_i \|C_\mathrm{q}^{\frac{1}{2}} \Delta_i C_\mathrm{k}^{\frac{1}{2}}\|^2 \tag{163}$$

$$= \sum_i \| \underbrace{C_\mathrm{q}^{\frac{1}{2}} G_i C_\mathrm{k}^{\frac{1}{2}}}_{G_i'} - \underbrace{C_\mathrm{q}^{\frac{1}{2}} A_\mathrm{q}^\top}_{A_\mathrm{q}'^\top} H_i \underbrace{A_\mathrm{k} C_\mathrm{k}^{\frac{1}{2}}}_{A_\mathrm{k}'} \|^2. \tag{164}$$

Hence, we can still solve it with HOSVD.

## Concatenative PE

Another PE uses concatenation:

$$Q_i = \begin{bmatrix} W_{\mathrm{q},i} X \\ E_{\mathrm{q},i} \end{bmatrix}, \tag{165}$$

$$K_i = \begin{bmatrix} W_{\mathrm{k},i} X \\ E_{\mathrm{k},i} \end{bmatrix}. \tag{166}$$

Then, the attention map will be

$$M_i = Q_i^\top K_i \tag{167}$$

$$= X^\top G_i X + E_{\mathrm{q},i}^\top E_{\mathrm{k},i}, \tag{168}$$

which has just a bias term $E_\mathrm{q}^\top E_\mathrm{k}$ and there is no impact in loss function with low-rank approximation.

### Multiplicative PE

Consider a multiplicative PE for token $X$:

$$X' = X \odot E, \tag{169}$$

where $\odot$ denotes Hadamard product. We just need to replace the correlation with $C' = X'X'^\top$ to solve in a straightforward manner.

However, rotary PE (RoPE) (Su et al. 2024) uses multiplicative PE on query and key, not token $X$. More precisely, we can represent per token:

$$q_{i,m} = \Theta_{i,m} W_{\mathrm{q},i} x_m, \tag{170}$$
$$k_{i,m} = \Theta_{i,m} W_{\mathrm{k},i} x_m, \tag{171}$$

where $\Theta_{i,m}$ is a block diagonal rotation matrix for $i$th head and $m$th token, such that $\Theta_{i,m}^\top \Theta_{i,n} = \Theta_{i,n-m}$.

For example, Llama-2 uses the same RoPE for all heads with block rotation:

$$\Theta_{i,m} = \begin{bmatrix} \cos(m\Phi) & -\sin(m\Phi) \\ \sin(m\Phi) & \cos(m\Phi) \end{bmatrix} \in \mathbb{R}^{d/h \times d/h}, \tag{172}$$
$$\Phi = \mathrm{diag}\big[\{\theta^{-2ih/d}\}_{i=0}^{d/2h-1}\big] \in \mathbb{R}^{d/2h \times d/2h}, \tag{173}$$

with a base rope theta of $\theta = 10^4$.

We have the loss:

$$\mathcal{L} = \mathbb{E}_X \sum_{i,m,n} \|q_{i,m}^\top k_{i,m} - \hat{q}_{i,m}^\top \hat{k}_{i,m}\|^2 \tag{174}$$
$$= \mathbb{E}_X \sum_{i,m,n} \|x_m^\top \underbrace{(W_{\mathrm{q},i}^\top \Theta_{i,n-m} W_{\mathrm{k},i} - A_\mathrm{q}^\top B_{\mathrm{q},i}^\top \Theta_{i,n-m} B_{\mathrm{k},i} A_\mathrm{k})}_{\Delta_{i,n-m} \in \mathbb{R}^{d\times d}} x_n\|^2 \tag{175}$$
$$= \mathbb{E}_X \sum_{i,m,n} \mathrm{tr}[\Delta_{i,n-m} x_n x_n^\top \Delta_{i,n-m}^\top x_m x_m^\top] \tag{176}$$
$$= \sum_{i,m,n} \mathrm{tr}[\Delta_{i,n-m} C \Delta_{i,n-m}^\top C] \tag{177}$$
$$= \sum_{i,m,n} \|C^{\frac{1}{2}} \Delta_{i,n-m} C^{\frac{1}{2}}\|^2 \tag{178}$$
$$= \sum_{i,m,n} \| \underbrace{C^{\frac{1}{2}} W_{\mathrm{q},i}^\top \Theta_{i,n-m} W_{\mathrm{k},i} C^{\frac{1}{2}}}_{W'_{i,n-m}} - \underbrace{C^{\frac{1}{2}} A_\mathrm{q}^\top}_{A_\mathrm{q}'^\top} \underbrace{B_{\mathrm{q},i}^\top \Theta_{i,n-m} B_{\mathrm{k},i}}_{H_{i,n-m}} \underbrace{A_\mathrm{k} C^{\frac{1}{2}}}_{A'_\mathrm{k}} \|^2 \tag{179}$$

where we assumed $x_m$ and $x_n$ are independent. Then, we can solve it with HOSVD. However, considering all token lengths over $m$ and $n$ is not practical, and we may need to consider attention windows such as $|n-m| \le 5$ to optimize. When a causal mask is used, we do not need to sum over $m > n$ but only $m \ge n$.

NOTE: we can generalize RoPE with other unitary rotations.

Fig. 11 shows the result of HOSVD with/without RoPE consideration. The loss is calculated over 10-token window, with RoPE base theta of $10^4$, used in Llama-2, while the hiiden size is still 768. HOSVD without RoPE consideration was already a good approximation as it is optimal at diagonal token. RoPE-aware HOSVD offers additional 1–2 dB gain.

## Joint Value-Output Compression

Many LLMs have output projection after QKV attention. The attention output will be

$$Y = \sum_i W_{\mathrm{o},i} W_{\mathrm{v},i} X \sigma(M_i^\top), \tag{180}$$

where $W_{\mathrm{o},i} \in \mathbb{R}^{d \times d/h}$ is the $i$th head output projection, and $Y \in \mathbb{R}^{d \times l}$ is the attention output. This motivates us to optimize value projection and output projection jointly.

NOTE: $W_{\mathrm{o},i} W_{\mathrm{v},i}$ can be triangularized by LU factorization to save the number of parameters from $2d^2$ to $2d^2 - d^2/h$ without any performance loss.

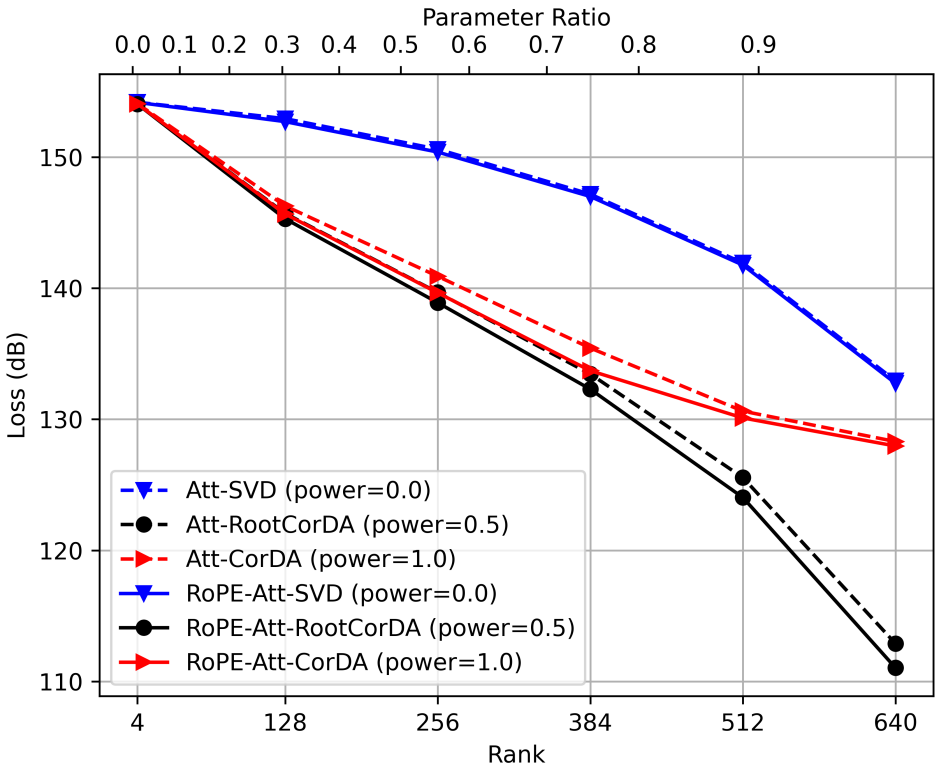


Figure 11: RoPE-Attention-Aware Distillation: 10-token window.

As $\sigma(M_i)$ is just weighting $X$, we may assume that the statistics still holds as $\mathbb{E}[X\sigma(M_i)\sigma(M_i)^\top X^\top] = C$ for uncorrelated tokens. Hence, we end up with optimizing

$$\mathcal{L} = \|W_{\rm o}W_{\rm v}C^{\frac{1}{2}} - B_{\rm o}\underbrace{A_{\rm o}B_{\rm v}}_{H}A_{\rm v}C^{\frac{1}{2}}\|^2. \tag{181}$$

Hence, both projections can be combined together.

Nevertheless, when we consider minimizing individual head projection loss for arbitrary attention weights:

$$\mathcal{L} = \sum_i \|\underbrace{W_{{\rm o},i}W_{{\rm v},i}C^{\frac{1}{2}}}_{G_i\in\mathbb{R}^{d\times d}} - B_{\rm o}\underbrace{A_{{\rm o},i}B_{{\rm v},i}}_{H_i\in\mathbb{R}^{r_{\rm o}\times r_{\rm v}}}\underbrace{A_{\rm v}C^{\frac{1}{2}}}_{A'_{\rm v}}\|^2. \tag{182}$$

Then the solution is HOSVD:

$$B_{\rm o} = \text{RightSingular}_{r_{\rm o}}[\sum_i G_i A_{\rm v}'^\top A'_{\rm v} G_i^\top], \tag{183}$$

$$A'_{\rm v} = \text{RightSingular}_{r_{\rm v}}[\sum_i G_i^\top B_{\rm o} B_{\rm o}^\top G_i], \tag{184}$$

$$A_{{\rm o},i} = B_{\rm o}^\top W_{{\rm o},i}J_i \tag{185}$$

$$B_{{\rm v},i} = J_i^+ W_{{\rm v},i}A_{\rm v}'^\top, \tag{186}$$

for arbitrary full-rank matrix $J_i \in \mathbb{R}^{d/h\times d/h}$. Selecting $J_i$ can save the number of parameters by up to $d/h \times d/h$.

## Bias Update

Some LLMs such as OPT uses bias in QKVO. Let's consider bias impact. The attention output will be:

$$Y = \sum_i W_{{\rm o},i}(W_{{\rm v},i}X + b_{{\rm v},i}1^\top)\sigma(M_i) + b_{{\rm o},i}1^\top \tag{187}$$

$$= \sum_i W_{{\rm o},i}W_{{\rm v},i}X\sigma(M_i) + W_{{\rm o},i}b_{{\rm v},i}1^\top\sigma(M_i) + b_{{\rm o},i}1^\top. \tag{188}$$

Considering any arbitrary attention map $M_i$, we may want to optimize:

$$\mathcal{L} = \sum_i \|W_{\mathrm{o},i}(W_{\mathrm{v},i}X + b_{\mathrm{v},i}1^\top) + b_{\mathrm{o},i}1^\top - \hat{W}_{\mathrm{o},i}(\hat{W}_{\mathrm{v},i}X + \hat{b}_{\mathrm{v},i}1^\top) - \hat{b}_{\mathrm{o},i}1^\top\|^2. \tag{189}$$

The gradient with respective to $\hat{b}_o$ is given

$$-\big(W_{\mathrm{o},i}(W_{\mathrm{v},i}X + b_{\mathrm{v},i}1^\top) + b_{\mathrm{o},i}1^\top - \hat{W}_{\mathrm{o},i}(\hat{W}_{\mathrm{v},i}X + \hat{b}_{\mathrm{v},i}1^\top) - \hat{b}_{\mathrm{o},i}1^\top\big)1. \tag{190}$$

Thus the KKT condition gives:

$$\hat{b}_{\mathrm{o},i} = b_{\mathrm{o},i} + W_{\mathrm{o},i}(W_{\mathrm{v},i}\mu + b_{\mathrm{v},i}) - \hat{W}_{\mathrm{o},i}(\hat{W}_{\mathrm{v},i}\mu + \hat{b}_{\mathrm{v},i}). \tag{191}$$

Plugging into the loss gives:

$$\mathcal{L} = \sum_i \|W_{\mathrm{o},i}W_{\mathrm{v},i}(X - \mu 1^\top) - \hat{W}_{\mathrm{o},i}\hat{W}_{\mathrm{v},i}(X - \mu 1^\top)\|^2 \tag{192}$$

$$= \sum_i \| \underbrace{W_{\mathrm{o},i}W_{\mathrm{v},i}}_{G_i \in \mathbb{R}^{d\times d}} C_0^{\frac{1}{2}} - B_{\mathrm{o}} \underbrace{A_{\mathrm{o},i}B_{\mathrm{v},i}}_{H_i \in \mathbb{R}^{r_{\mathrm{o}}\times r_{\mathrm{v}}}} A_{\mathrm{v}} C_0^{\frac{1}{2}} \|^2. \tag{193}$$

Here, $C_0 = (X - \mu 1^\top)(X - \mu 1^\top)^\top$ is centered covariance (it can be normalized). Hence, this is solved by HOSVD:

$$B_{\mathrm{o}} = \mathsf{RightSingular}_{r_{\mathrm{o}}}\big[\sum_i G_i C_0 A_{\mathrm{v}}^\top A_{\mathrm{v}} C_0 G_i^\top\big], \tag{194}$$

$$A_{\mathrm{v}} C_0^{\frac{1}{2}} = \mathsf{RightSingular}_{r_{\mathrm{o}}}\big[\sum_i C_0^{\frac{1}{2}} G_i^\top B_{\mathrm{o}} B_{\mathrm{o}}^\top G_i C_0^{\frac{1}{2}}\big]. \tag{195}$$

Note that $\hat{b}_{\mathrm{v}}$ has no impact as it can be absorbed by $\hat{b}_{\mathrm{o}}$. Hence, we can keep the original bias or changed to zero bias.

### Attention-Aware Joint VO Compression

The output projection module takes the input token:

$$X_{\mathrm{o},i} = W_{\mathrm{v},i}X\sigma(M_i^\top). \tag{196}$$

The covariance of the token is

$$C_{\mathrm{o},i} = X_{\mathrm{o},i}X_{\mathrm{o},i}^\top \tag{197}$$

$$= W_{\mathrm{v},i}X\sigma(M_i^\top)\sigma(M_i)W_{\mathrm{v},i}^\top. \tag{198}$$

Over the heads, we have cross-correation terms:

$$X_{\mathrm{o}} = \begin{bmatrix} W_{\mathrm{v},1}X\sigma(M_1^\top) \\ W_{\mathrm{v},2}X\sigma(M_2^\top) \\ \vdots \\ W_{\mathrm{v},h}X\sigma(M_h^\top) \end{bmatrix} \tag{199}$$

$$= \underbrace{\mathsf{diag}\big[W_{\mathrm{v},1}, W_{\mathrm{v},2}, \ldots, W_{\mathrm{v},h}\big]}_{W'_{\mathrm{v}} \in \mathbb{R}^{hd_{\mathrm{h}}\times hd}} \underbrace{(I_h \otimes X) \begin{bmatrix} \sigma(M_1^\top) \\ \sigma(M_2^\top) \\ \vdots \\ \sigma(M_h^\top) \end{bmatrix}}_{X' \in \mathbb{R}^{hd\times l}}. \tag{200}$$

We consider using the covariance of output projection module not value projection module instead. The covariance of the output projection $C_{\mathrm{o}} \in \mathbb{R}^{hd_{\mathrm{h}}\times hd_{\mathrm{h}}}$ is given as

$$C_{\mathrm{o}} = X_{\mathrm{o}}X_{\mathrm{o}}^\top \tag{201}$$

$$= W'_{\mathrm{v}} \underbrace{X'X'^\top}_{C_{\mathrm{v}} \in \mathbb{R}^{hd\times hd}} W'^\top_{\mathrm{v}}. \tag{202}$$

Using this attention-aware token statistics $C_{\mathrm{v}}$ can be more accurate to optimize, rather than simple token statistics $C$.

The value projection module takes the input token $X$ typically. However, there is no impact when we instead take the attention-weighted token for each head before value projection: $X'$. Even though we have no statistics on this, we can predict it from $C_{\mathrm{o}}$ as $C_o = W'_{\mathrm{v}} C_{\mathrm{v}} W'^{\top}_{\mathrm{v}}$:

$$C_{\mathrm{v}} = W'^{+}_{\mathrm{v}} C_{\mathrm{o}} [W'^{+}_{\mathrm{v}}]^{\top}. \tag{203}$$

Note that this is at most the rank of $hd_{\mathrm{h}}$.

The loss will be

$$\begin{aligned}
\mathcal{L} &= \| \sum_i W_{\mathrm{o},i} W_{\mathrm{v},i} X \sigma(M_i^{\top}) - \hat{W}_{\mathrm{o},i} \hat{W}_{\mathrm{v},i} X \sigma(M_i^{\top}) \|^2 && (204)\\
&= \| W_{\mathrm{o}} W'_{\mathrm{v}} X' - \hat{W}_{\mathrm{o}} \hat{W}'_{\mathrm{v}} X' \|^2 && (205)\\
&= \| W_{\mathrm{o}} W'_{\mathrm{v}} C_{\mathrm{v}}^{\frac{1}{2}} - \hat{W}_{\mathrm{o}} \hat{W}'_{\mathrm{v}} C_{\mathrm{v}}^{\frac{1}{2}} \|^2 && (206)\\
&= \| W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - \hat{W}_{\mathrm{o}} \hat{W}'_{\mathrm{v}} W'^{+}_{\mathrm{v}} C_{\mathrm{o}}^{\frac{1}{2}} \|^2. && (207)
\end{aligned}$$

Here we have

$$\hat{W}'_{\mathrm{v}} W'^{+}_{\mathrm{v}} = \mathsf{diag}[\hat{W}_{\mathrm{v},1} W_{\mathrm{v},1}^{\top} (W_{\mathrm{v},1} W_{\mathrm{v},1}^{\top})^{+}, \hat{W}_{\mathrm{v},2} W_{\mathrm{v},2}^{\top} (W_{\mathrm{v},2} W_{\mathrm{v},2}^{\top})^{+} \ldots, \hat{W}_{\mathrm{v},h} W_{\mathrm{v},h}^{\top} (W_{\mathrm{v},h} W_{\mathrm{v},h}^{\top})^{+}] \in \mathbb{R}^{hd_{\mathrm{h}} \times hd_{\mathrm{h}}}. \tag{208}$$

We write:

$$\begin{aligned}
\mathcal{L} &= \Big\| \sum_i W_{\mathrm{o},i} [C_{\mathrm{o}}^{\frac{1}{2}}]_i - B_{\mathrm{o}} \underbrace{A_{\mathrm{o},i} B_{\mathrm{v},i}}_{H_i \in \mathbb{R}^{r_{\mathrm{o}} \times r_{\mathrm{v}}}} A_{\mathrm{v}} W^{+}_{\mathrm{v},i} [C_0^{\frac{1}{2}}]_i \Big\|^2 && (209)\\
&= \Big\| W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} \sum_i H_i A_{\mathrm{v}} W^{+}_{\mathrm{v},i} [C_0^{\frac{1}{2}}]_i \Big\|^2 && (210)\\
&= \Big\| W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} \underbrace{[H_1 \quad \cdots \quad H_h]}_{H \in \mathbb{R}^{r_{\mathrm{o}} \times h r_{\mathrm{v}}}} (I_h \otimes A_{\mathrm{v}}) \mathsf{diag}[W^{+}_{\mathrm{v},1}, \ldots W^{+}_{\mathrm{v},h}] C_0^{\frac{1}{2}} \Big\|^2. && (211)
\end{aligned}$$

Note that $H_i$ is of rank up to $\min(r_{\mathrm{o}}, r_{\mathrm{v}}, d_{\mathrm{h}})$.

Gradient:

$$\begin{aligned}
\nabla_H \mathcal{L} &= -\Big(B_{\mathrm{o}}\Big)^{\top} \Big(W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} H (I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0^{\frac{1}{2}}\Big) \Big((I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0^{\frac{1}{2}}\Big)^{\top}, && (212)\\
\nabla_{A_{\mathrm{o},j}} \mathcal{L} &= -\Big(B_{\mathrm{o}}\Big)^{\top} \Big(W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} H (I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0^{\frac{1}{2}}\Big) \Big(B_{\mathrm{v},j} A_{\mathrm{v}} W^{+}_{\mathrm{v},j} [C_0^{\frac{1}{2}}]_j\Big)^{\top}, && (213)\\
\nabla_{B_{\mathrm{v},j}} \mathcal{L} &= -\Big(B_{\mathrm{o}} A_{\mathrm{o},j}\Big)^{\top} \Big(W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} H (I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0^{\frac{1}{2}}\Big) \Big(A_{\mathrm{v}} W^{+}_{\mathrm{v},j} [C_0^{\frac{1}{2}}]_j\Big)^{\top}, && (214)\\
\nabla_{B_{\mathrm{o}}} \mathcal{L} &= -\Big(W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} H (I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0^{\frac{1}{2}}\Big) \Big(H (I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0^{\frac{1}{2}}\Big)^{\top}, && (215)\\
\nabla_{A_{\mathrm{v}}} \mathcal{L} &= -\sum_j \Big(B_{\mathrm{o}} H_j\Big)^{\top} \Big(W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} - B_{\mathrm{o}} \sum_i H_i A_{\mathrm{v}} W^{+}_{\mathrm{v},i} [C_0^{\frac{1}{2}}]_i\Big) \Big(W^{+}_{\mathrm{v},j} [C_0^{\frac{1}{2}}]_j\Big)^{\top}. && (216)
\end{aligned}$$

The optimal $B_{\mathrm{o}}$ is the left-singular of $W_{\mathrm{o}} C_0^{\frac{1}{2}}$, having unitary condition: $B_{\mathrm{o}}^{\top} B_{\mathrm{o}} = I_{r_{\mathrm{o}}}$.

From the first KKT, we have a linear system to solve for $H$:

$$H \begin{bmatrix} A_{\mathrm{v}} W^{+}_{\mathrm{v},1} [C_0^{\frac{1}{2}}]_1 \\ \vdots \\ A_{\mathrm{v}} W^{+}_{\mathrm{v},h} [C_0^{\frac{1}{2}}]_h \end{bmatrix} \begin{bmatrix} A_{\mathrm{v}} W^{+}_{\mathrm{v},1} [C_0^{\frac{1}{2}}]_1 \\ \vdots \\ A_{\mathrm{v}} W^{+}_{\mathrm{v},h} [C_0^{\frac{1}{2}}]_h \end{bmatrix}^{\top} = B_{\mathrm{o}}^{\top} W_{\mathrm{o}} C_{\mathrm{o}}^{\frac{1}{2}} \begin{bmatrix} A_{\mathrm{v}} W^{+}_{\mathrm{v},1} [C_0^{\frac{1}{2}}]_1 \\ \vdots \\ A_{\mathrm{v}} W^{+}_{\mathrm{v},h} [C_0^{\frac{1}{2}}]_h \end{bmatrix}^{\top}. \tag{217}$$

Hence, we have

$$H = B_{\mathrm{o}}^{\top} W_{\mathrm{o}} C_0 [W'^{+}_{\mathrm{v}}]^{\top} (I_h \otimes A_{\mathrm{v}}^{\top}) \Big((I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0 [W'^{+}_{\mathrm{v}}]^{\top} (I_h \otimes A_{\mathrm{v}}^{\top})\Big)^{+}. \tag{218}$$

Plugging into the loss, we have

$$\mathcal{L} = \| W_{\mathrm{o}} C_0^{\frac{1}{2}} \|^2 - \Big\| B_{\mathrm{o}}^{\top} W_{\mathrm{o}} C_0 [W'^{+}_{\mathrm{v}}]^{\top} (I_h \otimes A_{\mathrm{v}})^{\top} \Big((I_h \otimes A_{\mathrm{v}}) W'^{+}_{\mathrm{v}} C_0 [W'^{+}_{\mathrm{v}}]^{\top} (I_h \otimes A_{\mathrm{v}})^{\top}\Big)^{-\frac{1}{2}} \Big\|^2 \tag{219}$$

The last KKT condition requires solving in vectorization:

$$\sum_{i,j}(G_jG_i^\top \otimes H_j^\top H_i)\mathsf{vec}[A_{\mathrm{v}}] = \sum_j \mathsf{vec}[H_jB_{\mathrm{o}}^\top W_{\mathrm{o}}C^{\frac{1}{2}}G_j^\top], \tag{220}$$

where $G_i \in \mathbb{R}^{d\times hd_{\mathrm{h}}}$ is defined:

$$G_i = W_{\mathrm{v},i}^+[C_0^{\frac{1}{2}}]_i. \tag{221}$$

## MLP-Aware Joint Compression

SparseLLM (Bai et al. 2024b) proposed the way to sparsify MLP layer in LLM models as it consumes two thirds of trainable parameters. The key idea is to minimize the MLP loss, not local loss. LLM uses typically 2-layer MLP:

$$z = W_1x + b_1, \tag{222}$$
$$a = \sigma(z), \tag{223}$$
$$y = W_2a + b_2. \tag{224}$$

The first linear layer typically upsamples by a factor of four, and then the second linear layer downsamples to the same dimension. Activation-aware low-rank approximation can minimize loss individually for $z$ given $x$ and $y$ given $a$, but not the MLP output $y$ given $x$.

SparseLLM uses the closed-form solution to minimize:

$$\mathcal{L} = \alpha\|W_1x + b_1 - z\|^2 + \beta\|a - \sigma(z)\|^2 + \gamma\|W_2a + b_2 - y\|^2, \tag{225}$$

for auxiliary variables $a$ and $z$, given pre-trained input $x$ and output $y$.

Optimizing $a$ can be obtained by ridge regression:

$$a^\star = (\gamma W_2^\top W_2 + \beta I)^+(\beta\sigma(z) + \gamma W_2^\top(y - b_2)). \tag{226}$$

Optimal $z$ can be also obtained closed-form way with case for ReLU:

$$z_- = W_1x + b_1, \tag{227}$$
$$z_+ = \frac{1}{\alpha+\beta}(\alpha z_- + \beta a), \tag{228}$$

depending on $[z]_i$'s sign.

The same approach can be used for low-rank approximation. Given $z$, we can optimize low-rank matrix $\hat{W}_1 = B_1A_1$ by SVD of $(z-b_1)x^+C_x^{\frac{1}{2}}$, where $(z-b_1)x^+ = (z-b_1)x^\top C_x^+$ corresponds to the effective weight matrix to map $x$ onto $z$. Given $a$, we approximate $\hat{W}_2 = B_2A_2$ by SVD of $(y-b_2)a^+C_a^{\frac{1}{2}} = (y-b_2)a^\top C_a^{\frac{-1}{2}}$, given correlation $C_a = aa^\top$.

## Sparse Matrix

Consider low-rank plus sparse decomposition:

$$\hat{W} = BA + D, \tag{229}$$

where $D \in \mathbb{R}^{d'\times d}$ is a sparse matrix such that $\|D\|_0 \le \kappa$. As discussed so far, given a $D$ matrix, the best low-rank matrices are SVD of $(W-D)C^{\frac{1}{2}}$. Given $BA$, finding sparse $D$ is an NP-hard problem, and often it is solved by greedy or relaxed methods such as matching pursuit and proximal gradient. Considering the $\ell_1$ relaxation, we have

$$\mathcal{L}' = \|(D + BA - W)C^{\frac{1}{2}}\|^2 + \lambda(\|D\|_1 - \kappa). \tag{230}$$

Fast iterative shrinkage-threshold algorithm (FISTA) uses iterations with Nesterov's accelerating technique:

$$D_k = \mathcal{T}_{\lambda\mu_k}[D_{k-1} - 2\mu_k(D_{k-1} + BA - W)C], \tag{231}$$
$$\mu_{k+1} = \frac{1}{2}(1 + \sqrt{1 + 4\mu_k^2}), \tag{232}$$
$$D_k \leftarrow D_k + \frac{\mu_k - 1}{\mu_{k+1}}(D_k - D_{k-1}), \tag{233}$$

for iterations $k = 1, 2, \ldots$ with a stepsize $\mu_1 = 1$. $\mathcal{T}_\alpha$ is a soft shrinkage operator:

$$\mathcal{T}_\alpha[x] = \mathrm{sign}[x](x - \alpha)_+. \tag{234}$$

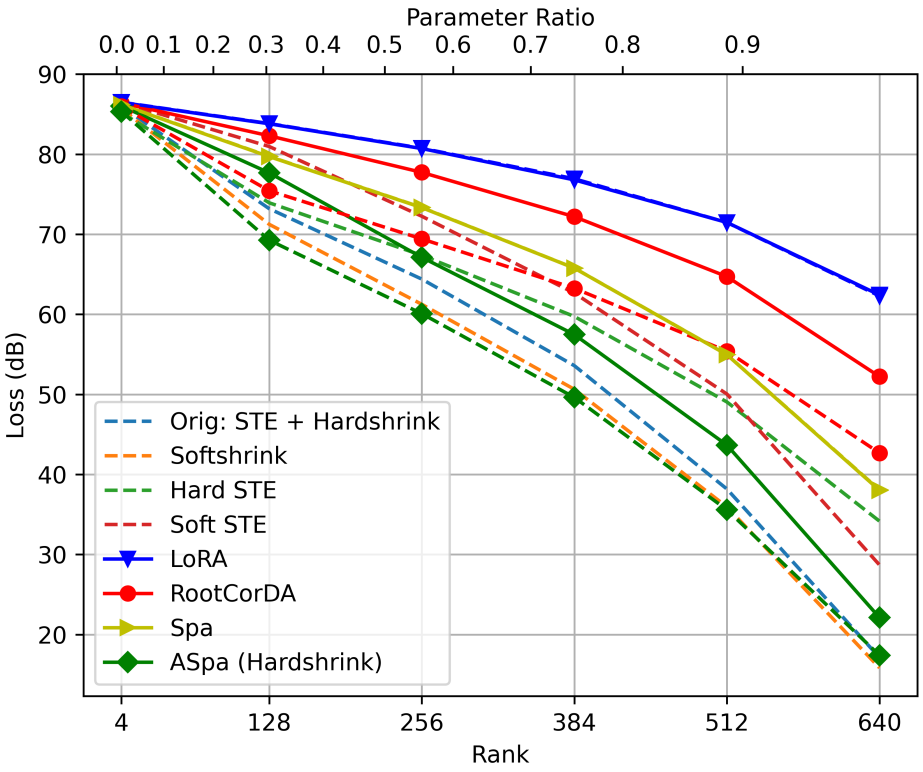


Figure 12: Random weight approximation with/without correlation. Correlation is sampled from Wishart distribution with covariance of identity or off-diagonal decaying of $0.9$ factor. Weight is normal distributed.

We may iterate SVD and FISTA. The choice of $\lambda$ is crucial to have a target sparsity. It is not easy to adjust $\lambda$ such that the target sparsity is achieved beforehand.

Alternatively, we use a regular gradient method with straight-through estimator (STE):

$$D = \underbrace{D - D.\text{detach}}_{\text{STE Trick}} + \mathcal{S}_\kappa[D.\text{detach}], \tag{235}$$

where $\mathcal{S}_\kappa[\cdot]$ is a hard shrinkage operator, i.e., sparcification operator passing only $\kappa$ elements having largest magnitude. This STE method has a benefit over FISTA: i) the sparsity can be specified; ii) any other loss function including the final downstream task loss can be incorporated; and iii) the quantization-aware training can be readily integrated in the STE projection. Nevertheless, soft shrink and hard shrink are actually differentiable, and we may not need to use STE. Fig. 12 shows the comparison of STE and Hard/Softshrink. In this experiment, Hardshrink works best.

We also notice that sparse approximation can be better than low-rank approximation. And, also joint low-rank plus sparse approximation did not work well as shown in Fig. 13.

However, unstructured sparse matrix may require index storage to memorize the non-zero entry locations. When we use a mask, it requires $d'd$ binary memory as well as non-zero values in $D$. When the sparsity is small, keeping index will be more efficient, i.e., keeping $\log_2(dd')\tau$. Fig. 14 shows the case with sparsification for low-rank adapter $B, A$, starting RootCorDA of rank 640 and 512. Although sparsified low-rank approximation has a benefit, it does not outperform sparse approximation alone.

Another possibility using sparse approximation is to sparsify LoRA matrices $B$ and $A$. However, doing so may be poor because the product of two sparse matrices can be much more sparse: e.g., 50% sparse $B$ and $A$ will result in 25% sparse $BA$. Hence, using sparse matrices for $B$ and $A$ may be a bad solution. Similarly, using sparse $W_\text{q}$ and $W_\text{k}$ may be a poor combination for attention map approximation.

WandA (Sun et al. 2023), SparseGPT (Frantar and Alistarh 2023) and SparseLLM (Bai et al. 2024b) use non-iterative solutions by considering only diagonal covariance:

$$C \to C \odot I_d. \tag{236}$$

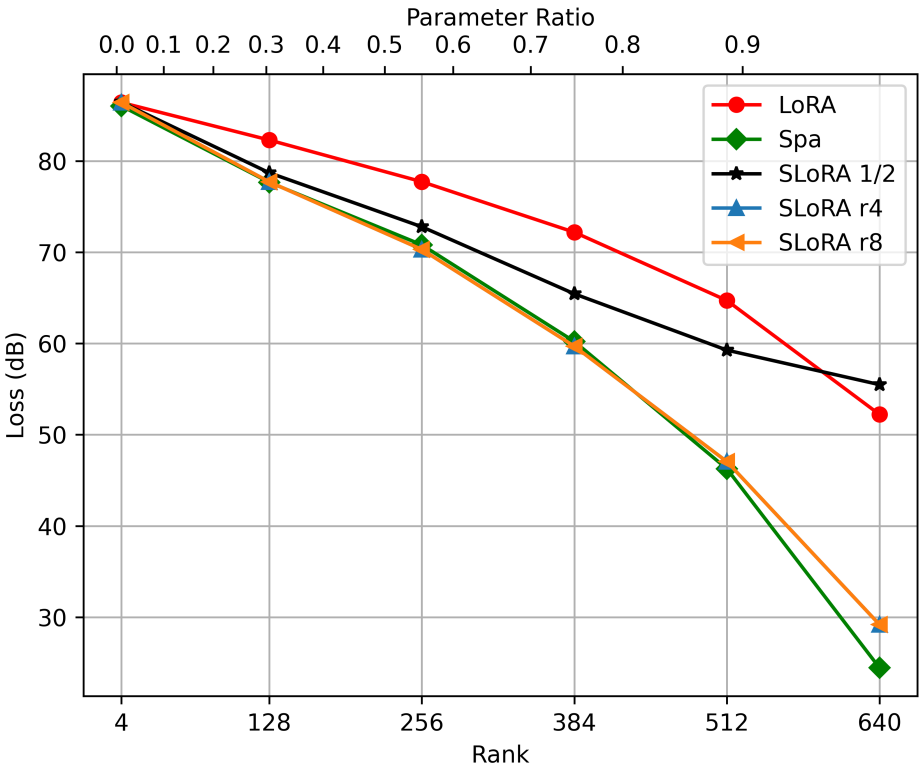


Figure 13: Low-rank plus sparse approximation does not outperform sparse-alone approximation.

This does not require iterative compressed sensing. However, the diagonal approximation has a degraded performance as in Fig. 15.

### Quantization-Aware Distillation

We can use STE for quantization-aware distillation in a straightforward manner. Whatever the loss, we can use STE for the trainable parameters, e.g., for $B$ and $A$ low-rank matrices:

$$B \leftarrow B - B.\text{detach} + \mathcal{Q}[B.\text{detach}], \tag{237}$$

$$A \leftarrow A - A.\text{detach} + \mathcal{Q}[A.\text{detach}], \tag{238}$$

where we may consider a simple chunk-wise $q$-bit uniform quantization:

$$x' = \mathcal{Q}[x] \tag{239}$$

$$= \text{round}\big[(x - x_{\min}) \cdot \frac{2^q - 1}{x_{\max} - x_{\min}}\big] \cdot \frac{x_{\max} - x_{\min}}{2^q - 1} + x_{\min}, \tag{240}$$

where $x_{\min}$ and $x_{\max}$ are determined from a chunk of $x$.

## LLM Models

Parameters for some major transformer models are listed in Table 11. OPT model variants are listed in Table 6. Table 7 shows parameters of Qwen3 models.

For LMM models, we used LLaVa: `liuhaotian/llava-v1.6-vicuna-7b`. It has Vicuna-7B model for LLM and ViT based on CLIP for the vision encoder. The Vicuna is an instruction-tuned version of LLaMa, having 32 transformer layers. CLIP ViT has 24 transformer layers.

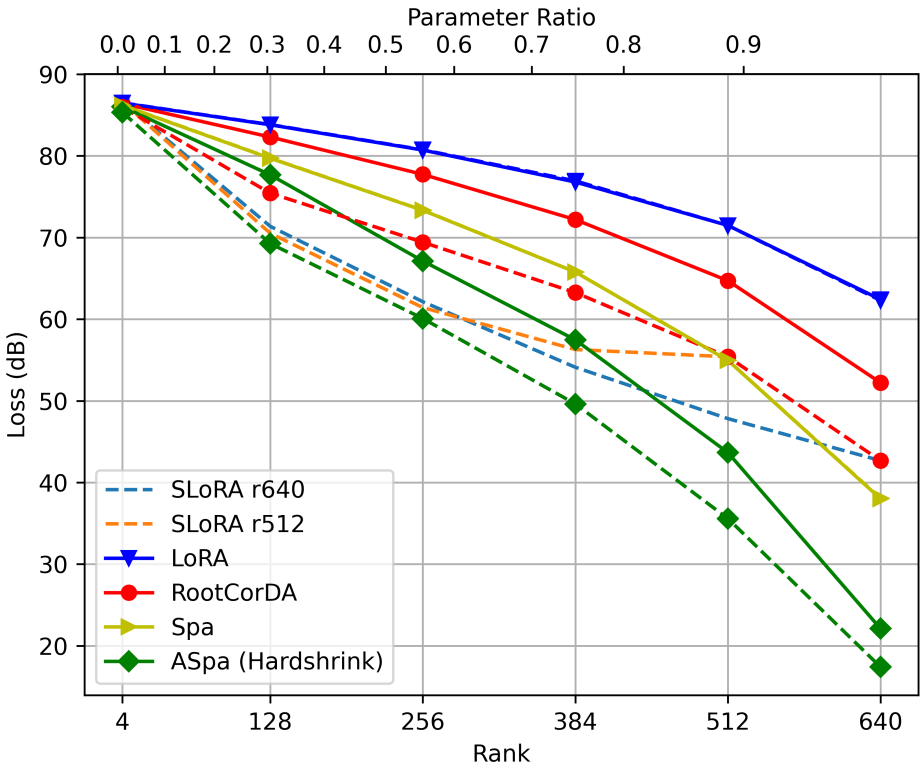


Figure 14: Sparsification of $B$ and $A$ low-rank matrices.

Table 6: OPT Models (Zhang et al. 2022)

| Models | # layers $L$ | # heads $h$ | hidden size $d$ | head dim $d_{\mathrm{h}}$ | $d_{\mathrm{i}} = 4d$ | Huggingface ID |
|---|---|---|---|---|---|---|
| 125M | 12 | 12 | 768 | 64 | 3072 | facebook/opt-125m |
| 350M | 24 | 16 | 1024 | 64 | 4096 | facebook/opt-350m |
| 1.3B | 24 | 32 | 2048 | 64 | 8192 | facebook/opt-1.3b |
| 2.7B | 32 | 32 | 2560 | 80 | 10240 | facebook/opt-2.7b |
| 6.7B | 32 | 32 | 4096 | 128 | 16384 | facebook/opt-6.7b |
| 13B | 40 | 40 | 5120 | 128 | 20480 | facebook/opt-13b |
| 30B | 48 | 56 | 7168 | 128 | 28672 | facebook/opt-30b |
| 66B | 64 | 72 | 9216 | 128 | 36864 | facebook/opt-66b |
| 175B | 96 | 96 | 12288 | 128 | 49152 | |

Table 7: Qwen3 Models

| Models | # layers $L$ | # heads $h$ | # KV heads $h_{\mathrm{kv}}$ | hidden size $d$ | head dim $d_{\mathrm{h}}$ | $d_{\mathrm{i}}$ | Huggingface ID |
|---|---|---|---|---|---|---|---|
| 0.6B | 28 | 16 | 8 | 1024 | 128 | 3072 | Qwen/Qwen3-0.6B |
| 1.7B | 28 | 16 | 8 | 2048 | 128 | 6144 | Qwen/Qwen3-1.7B |
| 4B | 36 | 32 | 8 | 2560 | 128 | 9728 | Qwen/Qwen3-4B |
| 8B | 36 | 32 | 8 | 4096 | 128 | 12288 | Qwen/Qwen3-8B |
| 14B | 40 | 40 | 8 | 5120 | 128 | 17408 | Qwen/Qwen3-14B |
| 32B | 64 | 64 | 8 | 5120 | 128 | 25600 | Qwen/Qwen3-32B |

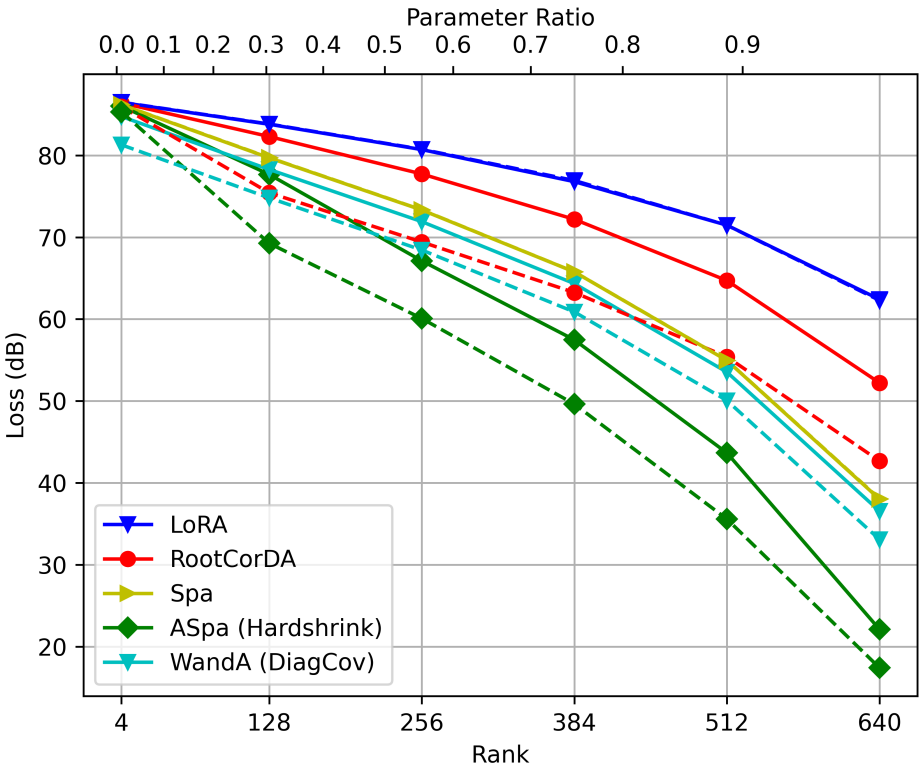


Figure 15: Comparison with WandA.

| Compression | 10% | | | 20% | | | 30% | | | 40% | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | WT2 | PTB | C4 | WT2 | PTB | C4 | WT2 | PTB | C4 | WT2 | PTB | C4 |
| OPT-125M (WT2: 27.7, PTB: 39.0, C4: 26.6) | | | | | | | | | | | | |
| Plain SVD (Identity) | 393.8 | 608.8 | 274.6 | 668.9 | 1098.0 | 559.0 | 1298.3 | 1888.7 | 806.5 | 3306.5 | 2985.9 | 1637.0 |
| ASVD (Hessian) | 57.8 | 92.8 | 45.0 | 106.9 | 169.8 | 79.9 | 288.1 | 530.4 | 215.0 | 838.9 | 1581.9 | 608.2 |
| ASVD ($\ell_2$-norm) | 49.7 | 74.7 | 42.2 | 87.3 | 126.8 | 72.0 | 256.0 | 282.1 | 188.3 | 906.9 | 864.3 | 528.4 |
| ASVD (Cov) | 87.5 | 121.5 | 67.6 | 115.7 | 157.0 | 83.1 | 163.1 | 242.8 | 109.9 | 248.3 | 390.6 | 158.4 |
| ASVD (RootCov) | 40.5 | 64.4 | 34.5 | 54.8 | 86.8 | 42.7 | 88.8 | 148.9 | 61.5 | 177.5 | 306.7 | 116.8 |
| **LatentLLM** (RootCov) | **29.0** | **42.3** | **27.6** | **32.9** | **50.9** | **30.4** | **43.4** | **68.7** | **37.4** | **73.3** | **116.9** | **55.7** |
| OPT-350M (WT2: 22.0, PTB: 31.1, C4: 22.6) | | | | | | | | | | | | |
| Plain SVD (Identity) | 112.3 | 130.8 | 82.8 | 211.3 | 226.8 | 151.5 | 378.1 | 392.0 | 258.7 | 705.5 | 635.5 | 509.8 |
| ASVD (Hessian) | 64.0 | 89.1 | 50.9 | 104.6 | 134.4 | 80.3 | 202.1 | 212.0 | 145.9 | 557.3 | 558.6 | 371.6 |
| ASVD ($\ell_2$-norm) | 40.0 | 59.9 | 36.6 | 59.4 | 78.0 | 49.8 | 117.5 | 134.2 | 86.9 | 308.7 | 283.9 | 201.1 |
| ASVD (Cov) | 78.0 | 90.6 | 61.7 | 100.8 | 111.0 | 72.7 | 311.2 | 356.8 | 129.4 | 1485.3 | 922.7 | 548.2 |
| ASVD (RootCov) | 30.8 | 42.2 | 28.5 | 39.0 | 51.4 | 33.6 | 71.6 | 86.1 | 49.5 | 118.5 | 132.1 | 73.0 |
| **LatentLLM** (RootCov) | **23.1** | **33.3** | **23.6** | **25.9** | **37.0** | **25.8** | **32.9** | **45.0** | **30.6** | **51.3** | **63.4** | **42.5** |
| OPT-1.3B (WT2: 14.6, PTB: 20.3, C4: 16.1) | | | | | | | | | | | | |
| Plain SVD (Identity) | 9428.1 | 10670.8 | 4865.4 | 16461.2 | 20589.0 | 11039.8 | 18105.3 | 17360.8 | 12565.2 | 22155.9 | 15820.3 | 16566.2 |
| ASVD (Hessian) | 23.8 | 40.6 | 24.9 | 63.0 | 173.7 | 52.8 | 825.8 | 927.9 | 385.0 | 4912.3 | 3086.3 | 2138.9 |
| ASVD ($\ell_2$-norm) | 20.3 | 32.3 | 21.6 | 28.7 | 60.2 | 27.7 | 74.5 | 217.4 | 58.5 | 592.4 | 1072.0 | 336.7 |
| ASVD (Cov) | 29750.9 | 31499.1 | 18646.3 | 19716.9 | 21757.2 | 14967.2 | 21738.3 | 24300.2 | 16428.7 | 22776.5 | 23591.7 | 14922.1 |
| ASVD (RootCov) | 17.7 | 27.9 | 18.9 | 21.9 | 35.3 | 22.2 | 33.9 | 55.8 | 29.7 | 75.0 | 107.9 | 51.1 |
| **LatentLLM** (RootCov) | ***14.5** | **21.5** | **16.6** | **15.8** | **24.3** | **17.8** | **20.2** | **31.6** | **21.3** | **34.1** | **58.1** | **30.6** |
| OPT-2.7B (WT2: 12.5, PTB: 18.0, C4: 14.3) | | | | | | | | | | | | |
| Plain SVD (Identity) | 1922.0 | 2250.3 | 900.7 | 7446.2 | 7042.4 | 5113.6 | 11253.8 | 10109.6 | 7742.6 | 26177.5 | 29321.3 | 17035.3 |
| ASVD (Hessian) | 18.2 | 31.9 | 20.0 | 31.6 | 96.9 | 28.0 | 216.2 | 852.3 | 74.8 | 2714.9 | 2894.0 | 626.0 |
| ASVD ($\ell_2$-norm) | 16.9 | 27.1 | 18.7 | 23.1 | 44.6 | 23.4 | 53.0 | 190.7 | 43.2 | 524.3 | 981.5 | 229.3 |
| ASVD (Cov) | 16419.9 | 15136.0 | 10680.6 | 15495.8 | 14896.4 | 10891.6 | 17392.3 | 15994.8 | 11926.0 | 17976.5 | 16298.1 | 11566.8 |
| ASVD (RootCov) | 14.5 | 22.1 | 16.5 | 17.1 | 26.7 | 18.8 | 24.1 | 36.3 | 23.7 | 48.4 | 66.5 | 37.1 |
| **LatentLLM** (RootCov) | ***12.3** | **18.8** | **14.7** | **13.6** | **20.6** | **15.7** | **16.5** | **24.3** | **18.1** | **24.5** | **36.0** | **24.2** |
| OPT-6.7B (WT2: 10.9, PTB: 15.8, C4: 12.7) | | | | | | | | | | | | |
| Plain SVD (Identity) | 14839.0 | 28665.9 | 22936.1 | 67517.7 | 116974.8 | 110860.5 | 123286.4 | 213333.5 | 190378.4 | 27304.0 | 31719.7 | 24071.3 |
| ASVD (Hessian) | 14.3 | 22.0 | 16.6 | 17.3 | 27.3 | 20.1 | 26.0 | 51.0 | 28.8 | 73.3 | 252.2 | 67.6 |
| ASVD ($\ell_2$-norm) | 12.6 | 19.6 | 15.1 | 14.6 | 23.0 | 17.2 | 18.7 | 32.1 | 21.4 | 30.6 | 73.2 | 33.7 |
| ASVD (Cov) | 9111.6 | 9171.3 | 7220.2 | 9842.6 | 9465.6 | 7175.0 | 11848.0 | 10046.0 | 6973.6 | 8514.7 | 7931.2 | 6660.3 |
| ASVD (RootCov) | 11.8 | 17.7 | 14.2 | 13.5 | 19.5 | 15.4 | 17.0 | 23.9 | 17.8 | 27.2 | 36.1 | 24.0 |
| **LatentLLM** (RootCov) | ***10.7** | **16.1** | **13.0** | **11.5** | **17.4** | **13.7** | **13.5** | **19.2** | **15.3** | **18.0** | **24.2** | **18.4** |
| OPT-13B (WT2: 10.1, PTB: 14.5, C4: 12.1) | | | | | | | | | | | | |
| Plain SVD (Identity) | 892.2 | 1003.5 | 789.3 | 2157.4 | 2068.3 | 1716.1 | 3612.9 | 3381.8 | 2806.9 | 5838.7 | 5069.1 | 4292.5 |
| ASVD (Hessian) | 12.5 | 18.6 | 14.3 | 14.6 | 22.0 | 15.8 | 19.1 | 29.7 | 18.7 | 29.5 | 48.9 | 25.1 |
| ASVD ($\ell_2$-norm) | 11.2 | 16.8 | 13.4 | 12.2 | 18.6 | 14.4 | 14.0 | 21.9 | 16.3 | 18.2 | 29.0 | 20.3 |
| ASVD (Cov) | 13999.3 | 11053.5 | 8991.2 | 10250.7 | 8883.2 | 6556.4 | 12885.3 | 11756.7 | 7658.0 | 12625.5 | 10709.9 | 7972.3 |
| ASVD (RootCov) | 10.9 | 15.9 | 13.1 | 11.9 | 17.0 | 13.9 | 14.3 | 20.0 | 15.3 | 20.2 | 24.1 | 18.3 |
| **LatentLLM** (RootCov) | **10.2** | **14.8** | **12.4** | **10.7** | **15.4** | **13.0** | **12.0** | **16.7** | **13.9** | **14.8** | **19.2** | **15.8** |

Table 8: Perplexity (↓) of OPT models with different SVD compression methods for 10–40% size reduction. Asterisk "*" indicates the better performance than the original un-compressed LLM.

| Compression | 10% | | | 20% | | | 30% | | | 40% | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | WT2 | PTB | C4 | WT2 | PTB | C4 | WT2 | PTB | C4 | WT2 | PTB | C4 |
| Qwen3-0.6B (WT2: 21.0, PTB: 43.8, C4: 30.3) | | | | | | | | | | | | |
| Plain SVD (Identity) | 2.7e6 | 6.1e6 | 2.6e6 | 1.7e7 | 4.9e7 | 1.8e7 | 2.8e7 | 2.4e7 | 2.5e7 | 4.3e7 | 5.4e7 | 3.4e7 |
| ASVD (Hessian) | 1.6e6 | 6.3e6 | 1.5e6 | 2.3e6 | 4.2e6 | 3.0e6 | 4.2e6 | 3.4e6 | 4.9e6 | 6.8e6 | 1.3e7 | 6.4e6 |
| ASVD ($\ell_2$-norm) | 9593.9 | 1.4e4 | 3297.9 | 3.1e5 | 8.0e5 | 1.4e5 | 4.0e5 | 4.5e5 | 2.2e5 | 8.0e4 | 1.0e5 | 3.1e4 |
| ASVD (Cov) | 1811.5 | 4.7e4 | 606.8 | 7388.4 | 1.3e5 | 1288.4 | 1.1e4 | 7.0e4 | 2398.6 | 1.7e4 | 2.6e4 | 5406.7 |
| ASVD (RootCov) | 145.6 | 379.2 | 130.2 | 484.1 | 1054.6 | 250.6 | 3531.1 | 1.4e4 | 996.7 | 2.4e4 | 5.8e4 | 2496.1 |
| **LatentLLM** (RootCov) | **30.4** | **60.3** | **44.2** | **59.9** | **118.3** | **77.6** | **232.6** | **510.3** | **161.2** | **1951.6** | **6794.9** | **688.9** |
| Qwen3-1.7B (WT2: 16.7, PTB: 33.8, C4: 22.4) | | | | | | | | | | | | |
| Plain SVD (Identity) | 1.8e7 | 1.6e7 | 1.1e7 | 1.3e7 | 1.1e7 | 1.1e7 | 1.0e7 | 1.0e7 | 6.5e6 | 1.9e7 | 1.7e7 | 1.5e7 |
| ASVD (Hessian) | 1.1e5 | 6.8e5 | 3.4e5 | 5.2e6 | 8.0e6 | 4.6e6 | 4.4e6 | 6.7e6 | 4.0e6 | 3.0e6 | 1.6e7 | 3.2e6 |
| ASVD ($\ell_2$-norm) | 72.5 | 138.4 | 102.8 | 1679.1 | 2719.6 | 1639.8 | 4842.6 | 1.2e4 | 3960.3 | 2.8e5 | 2.8e5 | 9.8e4 |
| ASVD (Cov) | 860.6 | 3378.6 | 338.3 | 1989.8 | 1.0e4 | 516.1 | 6645.8 | 2.4e4 | 906.1 | 1.2e4 | 4.6e4 | 1796.1 |
| ASVD (RootCov) | 37.5 | 63.2 | 43.9 | 66.3 | 114.8 | 62.5 | 147.8 | 287.6 | 100.0 | 387.2 | 1066.1 | 193.7 |
| **LatentLLM** (RootCov) | **22.3** | **47.8** | **28.4** | **27.9** | **51.5** | **35.3** | **48.8** | **81.4** | **53.3** | **137.5** | **264.9** | **98.6** |
| Qwen3-4B (WT2: 13.7, PTB: 24.7, C4: 19.9) | | | | | | | | | | | | |
| Plain SVD (Identity) | 5.0e4 | 5.2e4 | 3.7e4 | 6.5e5 | 1.7e6 | 4.0e5 | 3.1e6 | 5.0e6 | 1.8e6 | 3.8e7 | 3.3e7 | 4.8e7 |
| ASVD (Hessian) | 682.9 | 1372.1 | 598.5 | 2782.5 | 4110.1 | 1377.6 | 4.5e4 | 2.9e4 | 2.3e4 | 4.1e5 | 4.5e5 | 3.1e5 |
| ASVD ($\ell_2$-norm) | 29.3 | 46.7 | 34.8 | 46.1 | 73.2 | 52.8 | 80.6 | 153.6 | 101.2 | 229.8 | 451.1 | 245.2 |
| ASVD (Cov) | 1.9e5 | 1.1e6 | 2.5e4 | 1.5e5 | 4.9e5 | 2.4e4 | 1.5e6 | 3.3e6 | 3.0e5 | 9.2e5 | 1.7e6 | 5.1e5 |
| ASVD (RootCov) | 23.1 | 36.0 | 27.0 | 40.0 | 55.5 | 35.4 | 84.1 | 108.7 | 58.4 | 195.3 | 225.9 | 115.0 |
| **LatentLLM** (RootCov) | **15.8** | **32.9** | **21.6** | **18.7** | **37.1** | **24.5** | **35.1** | **47.0** | **32.1** | **90.8** | **122.9** | **63.5** |
| Qwen3-8B (WT2: 9.7, PTB: 17.2, C4: 15.4) | | | | | | | | | | | | |
| Plain SVD (Identity) | 2.4e5 | 8.7e4 | 4.5e4 | 9.0e6 | 1.9e6 | 8.8e5 | 2.8e7 | 3.8e7 | 1.8e7 | 5.3e7 | 1.0e8 | 5.2e8 |
| ASVD (Hessian) | 33.6 | 78.3 | 40.7 | 90.8 | 573.7 | 115.2 | 1250.8 | 1.3e4 | 854.4 | 5324.6 | 3.1e4 | 4872.0 |
| ASVD ($\ell_2$-norm) | 18.8 | 32.2 | 25.1 | 26.0 | 43.9 | 32.0 | 40.6 | 71.8 | 47.7 | 98.6 | 171.1 | 92.1 |
| ASVD (Cov) | 1.3e5 | 4.7e5 | 4.4e4 | 1.2e5 | 3.1e5 | 4.4e4 | 8.3e4 | 2.3e5 | 3.4e4 | 6.1e4 | 1.2e5 | 2.7e4 |
| ASVD (RootCov) | 16.7 | 25.0 | 21.8 | 26.0 | 32.6 | 26.3 | 49.3 | 60.5 | 38.6 | 119.2 | 136.9 | 71.1 |
| **LatentLLM** (RootCov) | **11.8** | **21.2** | **17.9** | **14.2** | **23.1** | **19.9** | **22.4** | **29.5** | **24.8** | **53.9** | **68.5** | **40.8** |

Table 9: Perplexity (↓) of Qwen3 models with different SVD compression methods for 10–40% size reduction.

| Compression | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|
| LLaVA-7B: Uncompressed Acc 61.32 | | | | | |
| Plain SVD (identity) | 2.36 | 0.48 | 0.35 | 0.34 | 0.36 |
| ASVD (Hessian) | 23.88 | 9.60 | 1.24 | 0.21 | 0.31 |
| ASVD ($\ell_2$-norm) | 24.41 | 9.53 | 2.77 | 0.82 | 0.75 |
| ASVD (Cov) | 0.38 | 0.36 | 0.40 | 0.33 | 0.35 |
| ASVD (RootCov) | 52.51 | 49.91 | 45.53 | 38.47 | 27.36 |
| **LatentLLM** (RootCov) | **60.06** | **57.65** | **52.63** | **46.90** | **35.94** |
| Qwen2.5-VL-7B-Instruct: Uncompressed Acc 82.11 | | | | | |
| Plain SVD (identity) | 0.02 | 0.47 | 0.32 | 0.05 | 0.11 |
| ASVD (Hessian) | 58.76 | 7.03 | 0.23 | 0.45 | 0.41 |
| ASVD ($\ell_2$-norm) | 77.84 | 73.92 | 57.13 | 18.79 | 0.41 |
| ASVD (Cov) | 0.41 | 0.41 | 0.41 | 0.41 | 0.41 |
| ASVD (RootCov) | 79.46 | 74.76 | 66.31 | 51.80 | 34.91 |
| **LatentLLM** (RootCov) | **80.85** | **79.30** | **73.90** | **62.11** | **42.53** |
| Qwen2.5-VL-3B-Instruct: Uncompressed Acc 78.17 | | | | | |
| Plain SVD (identity) | 0.01 | 0.08 | 0.09 | 0.09 | 0.01 |
| ASVD (Hessian) | 0.14 | 0.31 | 0.31 | 0.31 | 0.34 |
| ASVD ($\ell_2$-norm) | 44.23 | 0.14 | 0.00 | 0.41 | 0.37 |
| ASVD (Cov) | 0.41 | 0.41 | 0.41 | 0.41 | 0.41 |
| ASVD (RootCov) | 73.78 | 67.30 | 54.20 | 33.93 | 13.99 |
| **LatentLLM** (RootCov) | **76.44** | **74.29** | **64.28** | **45.80** | **19.67** |

Table 10: Accuracy in percent (↑) on TextVQA dataset for compressed LLaVA-7B and Qwen2.5-VL-7/3B.

Table 11: Transformer Models

| | ViT-16/B | Llama-2-7B | Llama-3.2-1B |
|---|---|---|---|
| ID | google/vit-base-patch16-224 | meta-llama/Llama-2-7b-hf | meta-llama/Llama-3.2-1B-Instruct |
| hidden size $d$ | 768 | 4096 | 2048 |
| hidden act | gelu | silu | silu |
| intermediate size $d_{\text{i}}$ | 3072 ($4d$) | 11008 ($2.68d$) | 8192 ($4d$) |
| head dim $d_{\text{h}} = d/h$ | 64 | 128 | 64 |
| num attention heads $h$ | 12 | 32 | 32 |
| num key value heads $h_{\text{kv}}$ | 12 | 32 | 8 |
| num hidden layers $L$ | 12 | 32 | 16 |
| qkv bias | True | False | False |
| mlp bias | True | False | False |
| rope theta $\theta$ | — | 1e4 | 5e5 |
| max position embeddings | 197 | 4096 | 131072 |
| | OPT-350M | BLOOM-560M | Qwen2-0.5B |
| ID | facebook/opt-350m | bigscience/bloom-560m | Qwen/Qwen2-0.5B |
| hidden size | 1024 | 1024 | 896 |
| hidden act | relu | gelu | silu |
| intermediate size | 4096 | 4096 | 4864 |
| head dim | 64 | 64 | 64 |
| num attention heads | 16 | 16 | 14 |
| num key value heads | 16 | 16 | 2 |
| num hidden layers | 24 | 24 | 24 |
| qkv bias | True | True | True |
| mlp bias | True | True | False |
| rope theta | — | — | 1e6 |
| max position embeddings | 2048 | 2048 | 131072 |
| | RoBERTa-350M | Phi-3.5 mini | Gemma-2B |
| ID | FacebookAI/roberta-base | microsoft/Phi-3.5-mini-instruct | google/gemma-2b |
| hidden size | 768 | 3072 | 2048 |
| hidden act | gelu | silu | gelu |
| intermediate size | 3072 ($4d$) | 8192 | 16384 |
| head dim | 64 | 96 | 256 |
| num attention heads | 12 | 32 | 8 |
| num key value heads | 12 | 32 | 1 |
| num hidden layers | 12 | 32 | 18 |
| qkv bias | True | False | False |
| mlp bias | True | False | False |
| rope theta | — | 1e4 | 1e4 |
| max position embeddings | 514 | 131072 | 8192 |